SCHNEIDER-CPC-WELT

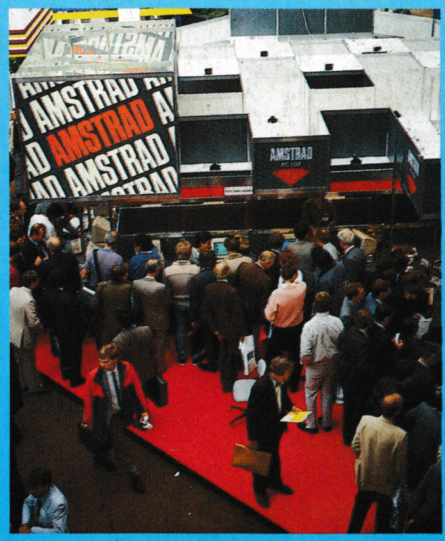
CPC EXTRA

Nr. 4/88

DM 14,80/ÖS 124/SFR 14,80

Neu: Mit grossem Spiele-Sonderteil!

Scheidung Amstrad/ Schneider: Geht Schneider K.O.?





Computer im Einsatz: Film & Forschung

Der CPC als Grafiker

5214428 cav c

stop stop stop stop stop stop stop

trennung schneider/amstrad ist offiziell. schneider tuerkheim praesentiert eine neue produktpalette. mittelpunkt ist der euro-pc auf der basis einer 8088 cpu mit 3.5 zoll flopppy und monochrom-monitor. preis: dm 1.298,-. lieferbar ab april/mai. weiterhin ein neuer at und ein telekopierer. mehr ueber alle geraete in pc-populaer, der neuen, jungen computerzeitschrift.

stop stop stop stop stop stop stop

5214428 cav d

DIALOG

Clubnachrichten
Die Public Domain User Group bietet . . .

Fremde Colormonitore am CPC
Ergänzung eines Lesers

Komprimierter Ausdruck beim DMP 2000
Karteiprogramm SPECIAL 4/87

Modelleisenbahner benötigt Hilfe
Initialisierung der Schnittstelle
Seite **17**

HARDWARE

Hilfe, mein Laufwerk!
Am falschen Platz gespart
Seite **41**

Vortex-ROM geknackt
Einblick ins DOS
Seite **66**

Selbstbau einer CPC-Uhr
Die Uhr wird fertig
Seite **153**

SOFTWARE

Im Test: XBC-BASIC-Compiler
Bedingt einsatzbereit
Seite **158**

REPORT

Kollege Chip kommt gleich
Computer in der Forschung
Seite **4**

Scheidung zwischen Amstrad und Schneider
Geht Schneider K.O.?
Seite **12**

Rund um den Computer
Neuigkeiten
Seite **89**

Bastard — echt kriminell
Zu Besuch bei Dreharbeiten
Seite **156**

BASIC-GRUNDLAGEN

Rund um die Diskette
Von Spuren und Sektoren
Seite **8**

Knacken Sie den Safe
Directory in BASIC
Seite **21**

Einfach und effektvoll
Grafikkurs, Teil 1
Seite **24**

Dem Fehler auf der Spur
Was steckt hinter den Fehlermeldungen?
Seite **28**

PROBLEM & LÖSUNG

Gregors Rache
Kalenderberechnung in BASIC
Seite **32**

Fenster zum Hof
Windows ohne WINDOW
Seite **34**

TIPS—TRICKS—KNIFFE

Auch in BASIC können Sie Bäume pflanzen
Binärbäume nicht nur in Pascal
Seite **44**

Print macht Power
Sinnvolle Steuerzeichen
Seite **47**

Die CPC-Trickkiste
Besser editieren
Seite **62**

BASIC an die Kette gelegt
Trickreiche Strings
Seite **64**

SPRACHEN UND BETRIEBS-SYSTEME

HiSoft-Pascal wird erweitert
Vektorgrafik
Seite **37**

Sprechen Sie C?
Die ersten Befehle
Seite **38**

Rekursive Grafiken
Auch unter Turbo-Pascal
Seite **40**

Von PIP bis TYPE
Dateien unter CP/M
Seite **42**

Sprachengewirr wie zu Babylon
Programmiersprachen zur Auswahl
Seite **50**

Wie mit MS-DOS
Komfort mit CP/M Plus
Seite **68**

SERVICE

Clubnachrichten
Info für Gesellige
Seite **119**

Börse
Gesucht und gefunden
Seite **150**

LISTINGS

Mini-Datei
Dateiverwaltung für alle CPCs
Seite **90**

Motorway
Vollgas ist angesagt
Seite **103**

DOS 208
Mehr Speicherplatz auf der Diskette
Seite **120**

Wordprocessor
Einfache Textverarbeitung
Seite **123**

Risiko
Blackjack-Simulation
Seite **128**

Greifer
Spannendes Reaktionsspiel
Seite **136**

Sounddemo
Zwei Nationalhymnen
Seite **146**

HIGHSCORE
Das Spiele-Magazin

Super Sprint
Autorennspiel aus der Vogelperspektive
Seite **II**

Thundercats
Abenteuer eines Katzenvolkes
Seite **II**

Championship Water Skiing
Wasserspiele mit dem CPC
Seite **IV**

Nebulus
Abenteuer in einer Wasserwelt
Seite **V**

Masters of the Universe
Das Imperum schlägt zurück
Seite **VII**

Asterix und Blueberry
Comic-Helden auf dem Vormarsch
Seite **VII**

Vengeance
Ballerspiel im Weltraumschiff
Seite **IX**

Airborne Ranger
Ein militärisches Kommandospiel
Seite **X**

Lucky Luke
Westernabenteuer wie im Comic
Seite **XI**

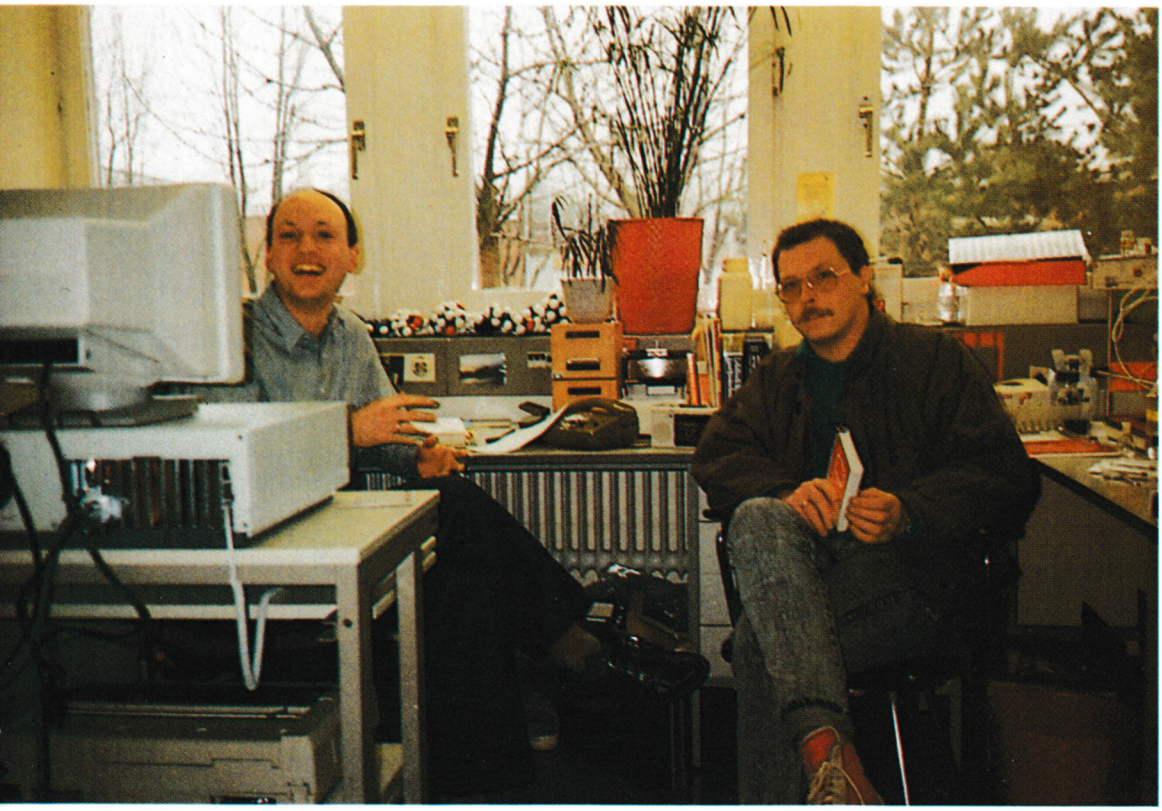
Kurzberichte
Neue Games für Homecomputer-systeme
Seite **XIII**

Ein Poke für alle Fälle
Tips, Pokes und Lösungen für Spiele
Seite **XIV**

COMPUTER IN DER FORSCHUNG

Kollege Chip kommt gleich

Computer greifen in der Arbeitswelt immer mehr um sich. Die Rede ist vom Jobkiller. Das Elektronenhirn kann auch hilfreicher Kollege sein. Auch wenn man auf Schritt und Tritt die Rechner sieht, im Supermarkt oder in der Arztpraxis: Was genau damit gemacht wird, weiß nur der Eingeweihte. Das Bild von der Chip-Revolution, vor der sich jeder so fürchtet, stimmt nicht mehr.



Günter Strobl (links) und Gert Seidel im Gespräch

Für Computeranwender, die wissen wollen, ob sie ihre Computer-Erfahrung auch im Arbeitsleben gebrauchen können, ist ein Blick über den Zaun der eigenen Umgebung sinnvoll. Steht es dafür, seine Zeit mit Pascal und C zu vertun, oder werden die Rechner in den Betrieben nicht ohnehin mit einer Software ausgestattet, die den Benutzer dazu degradiert, stupide auf Knöpfe zu drücken?

Bleibt vom einmal Erlernten nur das Wissen, was ein Cursor und eine Return-Taste ist? Dies waren Fragen, die ich mit einem Besuch im Toxikologie-Labor der GSF (Gesellschaft für Strahlenforschung und Umwelt) in München klären wollte.

UMWELT— GRUND GENUG ZUR FORSCHUNG

Die GSF gilt als eine der größten und angesehensten Forschungsinstitute der Bundesrepublik. Die Sorge um die Umwelt steht im Vordergrund, egal, ob es aktuelle Probleme sind wie etwa der Reaktorunfall von Tschernobyl oder das AIDS-Virus, oder ob es längerfristige Aufgaben zu lösen gilt.
Die zunehmende Verschmutzung von Luft, Wasser und Boden stehen an erster Stelle. Da diese Umweltprobleme vorwiegend von Giften hervorgerufen werden, galt unser Besuch dem Labor für Toxikologie. Hier steht die Erforschung von Substanzen und der ihnen eigenen Wirkungen im Vordergrund.
Dabei ist es zunächst gar nicht so wichtig, ein Mittel gegen das Übel zu finden, manchmal ist dies ohnehin nicht möglich. Vorrang hat die Grundlagenforschung über den Stoff und die Festlegung von vertretbaren Höchstwerten. Mit anderen Worten: Wie weit ist letztlich der Mensch damit belastbar, oder verbietet sich der Umgang mit derartigen Substanzen von vornherein?
Erst dann kommt die Pharmakologie, die Arzneimittelkunde, zum Zuge. Kein finanzieller Ertrag also, sondern Erforschung und Kontrolle der Umwelt.

EDV — START MIT DER TEXTVERARBEITUNG

Welche Rolle spielt der Computer in der Forschung der GSF? Günter Strobl, Chemiker, gab Auskünft über das Labor und sein Forschungsprojekt. Die erste Überraschung war, daß das Rechenzentrum der GSF kaum eine Rolle spielt. Nicht etwa, weil die Anwendungen zu unwichtig wären, ganz im Gegenteil.
Der Trend hat sich innerhalb weniger Jahre gewandelt. Bislang war es ja so, daß jede Forschungsgemeinschaft auf ein gemeinsam benutztes Rechenzentrum zugriff. War es einmal eine allgemeine Vorstellung, daß eine Zentraleinheit mehrere Terminals versorgt, so haben sich die Ansprüche gewandelt. Glücklicherweise kam es noch nicht zur Endausbaustufe eines solchen Netzwerkes. Heute sind,nicht nur in der GSF, Individualrechner angesagt, eben das, was man mit der Wortschöpfung PC meinte.

Unser Blick auf die Ausstattung in den Labors fiel zuerst auf einige Wang-Rechner, die speziell für die Textverarbeitung vernetzt wurden. Arbeitsberichte, Doktorarbeiten und jede Korrespondenz wird mit diesem Verbund erledigt. Daß man trotzdem nicht in Saus und Braus lebt, beweist die Tatsache, daß für die fünf Terminals nur ein Drucker vorhanden ist.
Überhaupt sind Computer trotz ihrer Bedeutung relativ neue Kollegen in der Forschungsgemeinschaft. Erst vor einem Jahr haben die Wang-Geräte Gesellschaft bekommen. IBM heißt nun die Devise und als Superstar darf neuerdings ein Wyse AT mit 80386-Prozessor seinen Dienst verrichten. So glaubte ich jedenfalls, bis mich Günter Strobl auf den Denkfeher hinwies.

GETARNTE COMPUTER

Wie fast alle Außenstehenden habe auch ich mich von Äußerlichkeiten beeindrucken lassen. Ich saß lange Zeit neben einem gerade beschäftigten Rechner, ohne etwas davon zu ahnen. Es handelte sich um ein HPLC-Gerät und wurde von



Wenig Tasten und aufwendige Technik

mir deshalb nicht als Rechner erkannt, weil die typische alphanumerische Tastatur fehlte. Zweckgebunden sorgt hier ein Standardprozessor für Steuerung und Auswertung. Ein fest installiertes Programm läßt nur die Eingabe einiger Parameter, etwa die Versuchszeit, zu. HPLC heißt dabei nichts anderes als Hochdruck-Flüssigkeits-Chromatograph (High Performance Li-Quid Chromatograph).

Grob gesagt geht es darum, Substanzen in geringster Menge sowohl quantitativ oder auch qualitativ zu bestimmen.
Soweit mein Laienverstand begriff, werden hierzu die Moleküle über eine bestimmte Wegstrekke in einem flüssigen Medium transportiert. Für die Auswertung gibt eine gezackte Linie Auskunft, die sich wie eine Gebirgssilhouette darstellt.
Jeder einzelne Berg, richtig als Peak bezeichnet, läßt sich in Höhe und Fläche berechnen. Entweder gibt das Diagramm bei bekannten Stoffen Hinweise auf die Konzentration der Substanz in Lösung, oder es läßt sich im Vergleich mit bekannten Standardsubstanzen ermitteln, um welchen Stoff es sich handelt.
Ein recht komplexes Aufgabenfeld für einen Rechner, auch wenn das Äußere der Tastatur täuscht.
So mancher Homecomputer sieht leistungsfähiger aus als das, was sich da unter einer billigen Folientastatur verbirgt.

Wer nun aber glaubt, das Ganze sei ebenso einfach zu bedienen wie ein Videorekorder, der sollte sich das Handbuch anschauen oder es auch nur abwiegen. Auch wenige Tasten können für viel Verwirrung sorgen, und die Liste dessen, was falsch gemacht werden kann, ist ellenlang.
Trotz meiner Anerkennung für derartige Arbeiten galt mein Interesse dem Wyse-AT, an dem Chemiker Strobl schon bei meiner Ankunft arbeitete. Da auch in den Forschungslabors das Geld nicht reichlich fließt, mußte für den leistungsfähigen Rechner zunächst ein konkreter Bedarf, eine Anwendung, nachgewiesen werden. Daran mangelt es allerdings nicht.

DREIDIMENSIONALES MOLEKÜLWUNDER IM AT

Der Chemiker brütet über molekulare Wechselwirkungen. Das Problem ist – wie immer – schnell beschrieben: Jedes Molekül hat ein bestimmtes Aussehen, eine plastische, dreidimensionale Symmetrie. Dabei sagen gleiche Bruttoformeln und gleiches Molekulargewicht noch lange nicht aus, daß auch die Strukturformel gleich ist.
Isomere nennt man derartige Täuschungen der Natur, die auf den ersten Blick zwar identisch scheinen, trotzdem aber andere chemische und physikalische Eigenschaften haben.
Es kommt also auf die Symmetrie an, und hier setzt die Arbeit von Günter Strobl ein. Sein Werkzeug: Ein Wyse-AT mit großer Rechenleistung und – eine Stereobrille.
Sie haben richtig gelesen, es handelt sich um das altbekannte Verfahren, mit dem ein dreidimensionaler Effekt erzeugt wird. Die Farben Rot und Grün werden getrennt auf den Bildschirm ausgegeben und knapp nebeneinander plaziert.
Eine entsprechend eingefärbte Brille kann das Auge, in erster Linie aber

Bei der Computeranwendung in der Forschung geht der Trend weg vom zentralen Terminal, hin zur individuellen Anwendung. Nicht vorhanden oder praktisch unbezahlbar ist die Software. Programmierer in der Forschung – ein Beruf mit Zukunft.

das Gehirn, täuschen und erzielt einen dreidimensionalen Eindruck.
Auf dem Sektor der Computerspiele gibt es einige derartige Versuche, und im Kino oder Männermagazinen haben Sie sicher schon Ähnliches gesehen. Falls es sich dabei um Bilder mit Jayne Mansfield handelte, wissen Sie, wie leicht sich das menschliche Gehirn von der Plastizität überzeugen läßt.
Ganz so vergnüglich geht es in der GSF allerdings nicht zu. Trotz guter Hardware lassen sich immer noch keine sehr großen Moleküle verarbeiten, aber vorerst gibt es auch mit kleineren Bildern genug zu tun. Aus der Erscheinungsform lassen sich Rückschlüsse auf die Wirkung der Substanzen ziehen.

LIGHTPEN IN DER KREBSFORSCHUNG

Recht unscheinbar gibt sich eine Computereinheit namens Videoplan. Hierbei geht es immerhin um Krebsforschung, eine der vordringlichsten wissenschaftlichen Aufgaben. Wichtigstes Werkzeug ist dabei ein Lightpen, der Computerzusatz, der bislang bei Homeanwendern eher mitleidig belächelt wurde.
Behandelte tierische Zellen werden in einem äußerst dünnen Querschnitt unter das Mikroskop gelegt und vom Computer abgetastet. Das auf dem Monitor entstehende Bild kann geprüft und bearbeitet werden.
Stößt der Wissenschaftler dabei auf anomale Zellen im Gewebe, so markiert er sie mit dem Lightpen und erhält über ein Rechenprogramm die exakten Werte. Zusätzlich werden sämtliche Zellen, die eine ähnliche Anomalie aufweisen, gezählt. Eine Statistik gibt Aufschluß, inwieweit das injizierte Material krebserregend war. Dazu muß vielleicht gesagt werden, daß eine einzige deformierte Zelle innerhalb eines Gewebes noch lange kein Alarmzeichen ist. Erst der Rechner, beziehungsweise das Programm gibt Aufschluß über den Grad der Krebsgefährdung.

GUTE AUSSICHTEN FÜR TIERFREUNDE

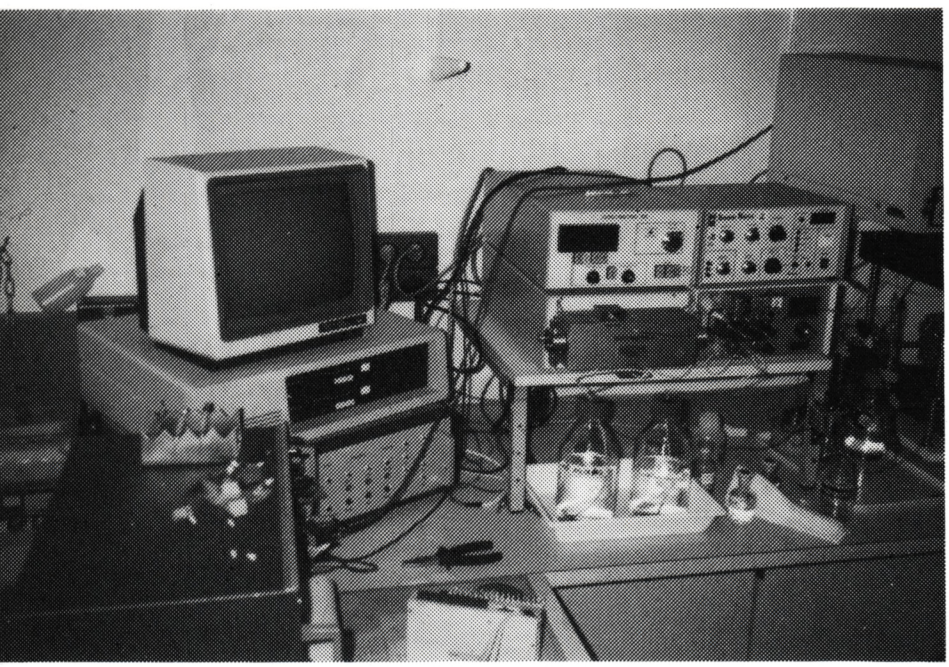
Dabei geht es natürlich nicht ohne Tierversuche. Kein angenehmes Thema. Es ist aber Tatsache, daß Tierversuche, abgesehen von denen der kosmetischen Industrie, notwendig sind. Deshalb finden sie auch im Bereich der Toxikologie der GSF statt, es geht letztlich um Menschenleben. Aber hier wirkt sich der Computer bereits segensreich aus.
Es gibt Datenbanken, in denen Basisversuche abgespeichert werden. Daß dieses Archiv genutzt wird, ist ein simples Rechenexempel, ist ist einfach billiger und schneller; Doktoranden müssen nicht immer wieder aufs neue toxische Werte im Tierversuch ermitteln, sondern können die Daten abfragen.
Auch rein theoretisch wird auf diesem Gebiet gearbeitet. Aufgrund vorhandener Ergebnisse simulieren Rechner den Ausgang eines neuen Versuches, ohne daß er wirklich durchgeführt wurde.

SOFTWARE – WOHER NEHMEN UND NICHT STEHLEN?

Zumindest ich kannte die beschriebene Anwendung nur vom Hörensagen, und die naheliegende Frage war: Woher kommt die Software? Günter Strobl hat zwei Programme für diese Arbeit, wobei ein amerikanisches, relativ preiswertes, Angebot bereits in der Schublade verschwunden ist: nicht geeignet, kein effektiver Nutzen, bestenfalls einem Studenten zu empfehlen. Dem Mangel konnte mit einigen Beziehungen zur Technischen Universität und sehr viel Sachkenntnis abgeholfen werden.

UNBEZAHLBARE SOFTWARE

Ein persönlicher Bekannter stellte ein noch nicht ganz fertiges Programm zur Verfügung, mit dessen Leistungsfähigkeit man zufrieden ist. Hier ist Strobl plötzlich in der Rolle eines Pilotanwenders, eine Arbeit, die auch Informatikkenntnisse erfordert. Natürlich gibt es auch kommerzielle Programme. Der Kostenaufwand liegt allerdings bei einer Größenordnung von 60000 Mark, die Hardware in Form einer Workstation eingeschlossen. Zu viel.
Selbst dann, so befürchtet Günter Strob, sind die Anwendungen in der Wissenschaft zu speziell, und es dürfte schwierig werden, die geeignete Software zu finden. Abhilfe ist dort zu suchen, wo ebenfalls geforscht wird. Im Bereich Informatik der technischen Universitäten ist relativ viel zu holen. Nur befinden sich diese Programme fast alle noch im Entwicklungsstadium, werden selten vollständig veröffentlicht



Ein IBM steuert den HPLC und übernimmt die Auswertung der Ergebnisse

und wenn überhaupt Dokumentation vorhanden ist, dann läßt diese sehr zu wünschen übrig.
Vorteile hat diese Beschaffungsweise allerdings auch. Die Programme dürfen gar nicht kommerziell verwertet werden, sofern sie an der Hochschule entwickelt wurden. In den USA müssen mit öffentlichen Mitteln finanzierte EDV-Projekte sogar der Öffentlichkeit zugänglich gemacht werden.
Die Public-Domain-Szene ist Tummelplatz der besten Programme, mit denen Otto Normalverbraucher allerdings wenig anfangen kann. In der BRD sind wir noch nicht so weit, aber einige gute Kontakte helfen über die Hürden und bringen letztlich individuelle Software in den Computer, die unbezahlbar ist.
Aber dazu muß man fit auf dem Computer sein, denn abgesehen davon, daß die Software nicht ganz fertig und bis ins letzte Detail erprobt ist, stellt sich auch schnell heraus, daß der Programmierer kein Chemiker oder Biologe war. Wenn sich auch der Chemiker unfähig zur Programmverbesserung zeigt, kann ein ganzes Projekt ad acta gelegt werden.
Damit ist auch die Frage beantwortet, welche Aussichten Sie mit Ihrer in der Freizeit erworbenen EDV-Bildung haben. Verlangt wird sie (noch) nicht, wie es so schön heißt. Aber abgesehen davon, daß einmal erworbenes Wissen nie schadet, ist gerade der Forscher im naturwissenschaftlichen Bereich zunehmend auf Computeranwendung angewiesen. Und ohne Erfahrung muß der Selbstlehrgang eben nachgeholt werden. Da darf auch die Information nicht fehlen, mit welchen Programmiersprachen man es letztlich zu tun hat.
Bislang wird der Hauptteil der Software in Fortran eingegeben. FORmula TRANslation



Neueste Errungenschaft: Ein Wyse AT

ist nun einmal die Sprache der Wissenschaftler. Gelegentlich greift man bei kleineren Routinen auf Assembler zurück.
Die große Trendwende steht jedoch bevor. Nach den Beobachtungen von Strobl wird zunehmend in C programmiert. Ein besonderer Aspekt, da man heutzutage bereits in Homecomputern die Möglichkeiten dieser Sprache nutzen kann.

SPEKULATIONEN ZUR ZUKUNFT

Computer sind Vertreter einer zukünftigen Technik, die bereits heute schon erhältlich ist.
Was wird von seiten der Elektronik auf die Forschung zukommen?
In den Labors ist das Rennen um den Superrechner noch längst nicht gelaufen. Und es wird genau da entschieden und nicht etwa in den großen Rechenzentren. Die Forscher benötigen Rechner, die enorm leistungsfähig sind und trotzdem in einer Preiskategorie liegen, die es erlaubt, jedem einzelnen Wissenschaftler eine Konfiguration zur Verfügung zu stellen.
Um es noch einmal in Erinnerung zu bringen: Ein AT ist bereits heute mit der Darstellung von einfachen Molekülen ausgelastet. Komplexere Strukturen sind nur an einem größeren Rechner darzustellen, der wiederum von einem Wissenschaftler alleine an die Grenzen gebracht werden kann. In den Labors fehlt also nicht das gewaltige Netzwerk, sondern der Arbeitsplatz-Computer der Superlative.
Folgen hat dies natürlich auch auf die Ausbildung der Naturwissenschaftler. Falls Sie selbst ein Studium der Chemie, Medizin oder der Biologie planen und den Arbeitsbereich der Forschung im Auge haben, dann sollten Sie der EDV größte Aufmerksamkeit widmen. Bereits im privaten Sektor, der Domäne der Homecomputer, kann einiges getan werden.
Irgendwann ändert sicht die Struktur des Chemie-Studiums. Es wird wohl so aussehen, daß ein Grundwissen geschaffen wird, im Umfang vielleicht sogar etwas kleiner, als bisher. Mit höherer Semesterzahl wird es sich wie das Geäst eines Baumes, in verschiedene Spezialrichtungen verzweigen.

IN DER FORSCHUNG IST DER COMPUTER KEIN JOBKILLER

Der Druck, derartige Überlegungen anzustellen, kommt von der Computertechnik. Ein Programmierer ist kein Chemiker, und der Chemiker ist kein Informatiker. Bislang kann man sich mit Eigeninitiative, die bereits beim Studium beginnt, weiterhelfen. Aber die Anforderungen der EDV werden mit jedem neuen Rechner größer und sind nebenher nicht mehr lange zu bewältigen.
Der naturwissenschaftliche Informatiker ist noch nicht in Sicht und trotzdem kommt dem Computer eine immer größere Bedeutung zu. Von einem Jobkiller kann nicht die Rede sein. Software heißt das eigentliche Problem, mit dem man auch in der GSF zu kämpfen hat. Eine Chance für jeden Interessierten, mag er sich auch jetzt noch mit BASIC plagen.
Wer in der Computeranwendung seine Zukunft sieht, muß nicht unbedingt den Weg über das Studium einschlagen. Es besteht auch Bedarf an Laboranten jeglicher Art. Und auch hierbei ist, wie es so schön heißt, Erfahrung mit dem Computer kein Hindernis. Augenzwinkernd bedeutet dies wohl, daß Vorbildung keine finanziellen Folgen hat. Aber was macht das schon, wenn man im Berufsleben auch noch den Kontakt zu seinem Hobby pflegen kann? *GS*□

Rund um die Diskette

Laufen müssen Sie!

Wer ernsthaft mit einem Computer arbeiten will, kommt wohl kaum um die Anschaffung eines Diskettenlaufwerks herum. Für die Besitzer der CPC-Typen 664 und 6128 stellt sich diese Frage nicht, gleichwohl aber für die 464-Eigner. Doch welcher Einsteiger beherrscht sein Laufwerk wirklich?

Anfragen bei unserer Hotline am Montag zeigen uns immer wieder, daß einige von Ihnen Schwierigkeiten mit dem Diskettenlaufwerk und seinen Befehlen haben. Dabei ist der Umgang mit diesem Speichermedium gar nicht so schwer.

Zuerst einige grundsätzliche Überlegungen. Selbstverständlich können Sie auch ein Kassettenlaufwerk als Massenspeicher für Daten und Programme benutzen. Einige Nachteile hat das jedoch: Zum einen die relativ langsame Lese- und Schreibgeschwindigkeit, die sich besonders bei langen Programmen manchmal nachteilig bemerkbar macht. Zum anderen die Fehler, die durch einen versteliten Schreib/Lese-Kopf des Rekorders auftreten können.

Und zum dritten muß beim Abspeichern von Programmen, die aus mehreren Teilen bestehen, die richtige Reihenfolge eingehalten werden.



Ihre Disketten sind kein Geheimnis mehr, wenn Sie sich ein wenig mit AMSDOS und CP/M beschäftigen

Viele Fehlerquellen beim Kassettenlaufwerk

Während Sie sich den Inhalt der Diskette durch direkten Zugriff auf das Inhaltsverzeichnis ausgeben lassen können, muß beim Kassettenbetrieb erst das ganze Band durchlaufen werden. Und zu guter Letzt haben Sie beim Diskettenlaufwerk auch die Möglichkeit, direkt ein Programm aufrufen zu können, ohne erst die Stelle suchen zu müssen, an der es sich befindet.

Sie haben also eine Reihe von guten Gründen, eine Diskettenstation zum Arbeiten zu bevorzugen. Wir werden uns im weiteren Verlauf ausschließlich um die Floppy-Station kümmern, auch wenn viele Befehle, die wir besprechen werden, für den Kassettenrekorder ebenfalls zutreffen. Außerdem lassen wir die Vortex-Stationen außer acht, denn wir wollen uns in erster Linie um das Original-Laufwerk Ihres CPC kümmern.

Was passiert beim Formatieren?

Bevor Sie eine neue Diskette das erste Mal benutzen, müssen Sie sie formatieren. Was bedeutet das? Die magnetisierbare Schicht, die auf das Trägermaterial aufgetragen wurde, muß zuerst so hergerichtet werden, daß sich der Schreib/Lese-Kopf zurechtfindet. Während dieses Vorgangs wird die Diskette „eingeteilt", nämlich in insgesamt 40 Spuren (auf englisch „tracks") zu je neun Sektoren. Diese Sektoren können Sie sich als Tortenstücke vorstellen.

Der Vollständigkeit halber sei hier noch erwähnt, daß es auch noch andere Formate gibt, so zum Beispiel das IBM-Format, bei dem jedes Track lediglich acht Sektoren aufweist. Außerdem werden bei einigen käuflichen Programmen die Disketten auch nicht mit 40, sondern mit 41 oder 42 Spuren formatiert. Diese Erhöhung der Spurenanzahl wird aus Gründen des Kopierschutzes vorgenommen.

Sie haben die Wahl: CP/M oder Data-Format

Zurück zur „Standard"-Diskette. Um die Verwirrung

jetzt komplett zu machen, können Sie auswählen, ob Sie Ihre Diskette im CP/M-Format oder im Data-Only-Format formatieren.

Beim Formatieren werden aber nicht nur die Spuren und Sektoren eingeteilt, sondern auch noch die Header (Vorspänne) aufgebracht. Diese Header enthalten eine Vielzahl von Informationen, die die Diskettenstation benötigt; unter anderem auch die Sektorenkennzeichnung. Und diese Sektorenkennzeichnung ist bei den beiden CPC-üblichen Formaten unterschiedlich. Beim Data-Format, wie das Data-Only-Format allgemein abgekürzt wird, erhalten die Sektoren die Nummern &40 bis &49, während sie im CP/M- oder System-Format &C0 bis &C9 lauten.

Warum gibt es überhaupt die beiden unterschiedlichen Formate (das IBM-Format lassen wir einmal außer Betracht)?

Das CP/M-Format auf Ihrer Diskette brauchen Sie immer dann, wenn Sie CP/M-Programme laufen lassen wollen. CP/M ist ein eigenes Betriebssystem von Digital Research. Wenn Sie jedoch Programme normal unter BASIC laufen lassen, reicht Ihnen das Data-Format. Um dieses CP/M-Betriebssystem zu laden, brauchen Sie die entsprechenden Informationen für Ihren Computer. Und diese Informationen befinden sich beim CP/M-Format auf den beiden ersten Spuren der Diskette.

Zwei Spuren sind tabu bei CP/M

Im CP/M-Format haben Sie eine Speicherkapazität von 169 KByte, während beim Data-Format 178 KByte zur Verfügung stehen. Dieser Unterschied von neun KByte ergibt sich aus dem Platzbedarf für CP/M, denn dafür werden ja zwei Spuren reserviert.

Weiterhin werden vier Sektoren benötigt, um das Inhaltsverzeichnis Ihrer Diskette unterzubringen. Da ein Sektor 512 Byte hat, bleiben Ihnen also übrig:

Im Dataformat:
((39*9+5)*512):1024 =
178 KByte

Im CP/M-Format:
((37*9+5)*512):1024 =
169 KByte

Das Diskettenlaufwerk enthält einen Controller, der Ihnen den Zugriff auf die Diskette erst möglich macht. Damit das Laufwerk korrekt arbeiten kann, braucht es ein eigenes Betriebssystem. Beim CPC heißt es AMSDOS (AMStrad Disk Operating System). Dieses Betriebssystem stellt Ihnen eine Reihe von zusätzlichen Befehlen zur Verfügung.

Der Controller macht's möglich

Allen Befehlen wird der sogenannte RSX-Strich vorangestellt, erreichbar durch gleichzeitiges Drücken der Shift-Taste und der Taste mit dem „Klammeraffen".

AMSDOS-Befehle erklärt

!A
Schaltet auf das erste Laufwerk um und macht dieses zum Standardlaufwerk, das heißt, alle Diskettenbefehle, die im Programm danach folgen, werden auf das Laufwerk A bezogen.

!B
Das zweite Laufwerk wird aktiviert.

!drive,@<Laufwerk>
Schaltet ebenfalls auf das gewünschte Laufwerk um. Dazu müssen Sie allerdings das „A" als String übergeben. Also: **dr$**="A":!drive,@**dr$**. Nach diesem Befehl wird ebenfalls das Laufwerk A aktiv. Setzen Sie anstelle des „A" ein „B", so schalten Sie auf das Laufwerk B um. Das funktioniert natürlich nur, wenn Sie zwei Laufwerke haben. Da AMSDOS maximal zwei Laufwerke ansteuern kann, ist also das Umschalten auf ein „C"-Laufwerk sinnlos.

!TAPE
Mit diesem Befehl wird auf das eingebaute (CPC 464) oder ein zusätzlich angeschlossenes (664 oder 6128) Kassettenlaufwerk umgeschaltet. Alle Befehle, die danach kommen und sich auf das Speichermedium beziehen, werden auf das Kassettenlaufwerk übertragen.

!DISC
Umschalten auf das Diskettenlaufwerk.

!DISC.IN
Alle Lesebefehle, die sich auf Ihr Speichermedium beziehen, werden auf das Diskettenlaufwerk bezogen. Setzen Sie statt DISC nun TAPE, so werden die Lesebefehle auf den Kassettenrekorder angewandt.

!TAPE.OUT
Als Speichermedium wird der Kassettenrekorder benutzt. Statt TAPE können Sie natürlich auch DISC einsetzen.

Kopieren komfortabel gemacht

Praktisches Beispiel: Sie wollen mehrere BASIC-Programme von Diskette auf Kassette kopieren. Dann geben Sie ein:

!DISC.IN:!TAPE.OUT
Danach brauchen Sie nun nur noch mit load „Dateiname" und save „Dateiname" zu arbeiten. Das Umschalten von Disk und Tape brauchen Sie nun nicht mehr nach jedem Laden und Abspeichern einzugeben, dies haben Sie ja bereits mit dem !DISC.IN und dem !TAPE.OUT erledigt.

!USER,<Integer-Zahl>
Mit diesem Befehl und der nachgesetzten Zahl wird der entsprechende Userbereich auf der Diskette angesprochen. Mit !USER,9 wird zum Beispiel auf den Userbereich 9 umgeschaltet. Alle cat-, save- und load-Befehle, die dann folgen, werden auf der Diskette mit der Kennzeichnung des Userbereichs 9 abgespeichert und können nur unter dieser User-Nummer wieder aufgerufen werden. Als Usernummern steht Ihnen der Bereich von 0 bis 15 zur Verfügung.

!CPM
Aufruf des CP/M-Betriebssystems

!ERA,@<Dateiname.Ext>
Ein wichtiger Befehl, der aber einigen von Ihnen immer wieder Schwierigkeiten macht. Er ermöglicht Löschen von nicht mehr benötigten Programmen oder Dateien. Dazu einige Beispiele:

Zu löschendes Programm: "Jojo.bas". In diesem Fall müssen Sie folgendes eingeben: a**$**="Jojo.bas":!ERA,@a**$**. Dieser Befehl läuft auf allen CPC-Typen. Die 664- und 6128-Besitzer können allerdings auch eingeben: !ERA, "Jojo.bas", also direkt, ohne den zu löschenden File-Namen erst in einer Variablen abzulegen und dann mit dem Variablenpointer (@) zu löschen.

Was sind Wild Cards?

In diesem Zusammenhang ist auch das Arbeiten mit „Wild Cards" wichtig. Nehmen wir an, daß Sie unter dem File-Namen „Jojo" mehrere Programme oder Dateien abgespeichert haben, etwa:

1. Jojo.bas
2. Jojo.bin
3. Jojo.lad
4. Jojo.dat
5. Jojo.asc

Um alle diese Programme zu löschen, gibt es zwei Wege. Erste Möglichkeit: Sie löschen alle Namen einzeln mit dem ERA-Befehl. Zweite Möglichkeit: Sie arbeiten mit Wild cards. Davon gibt es zwei verschiedene, nämlich das Fragezeichen(?) und den Stern (*). Das Fragezeichen ersetzt einen Buchstaben, der Stern mehrere. Wenn Sie alle „Jojo"-Files löschen wollen, geben Sie ein: !ERA,"Jojo.*" oder !ERA,"Jojo.???". Und schon ist aufgeräumt. Wollen Sie jedoch nur Jojo.bas und Jojo.bin löschen, dann geben Sie ein: !ERA,"Jojo.b*" oder !ERA;"Jojo.b??".

Vorsicht mit dem ERA-Befehl!

Sie können auch folgendes eingeben: !ERA,"J*.*". Aber Achtung! Nun wird auf der Diskette alles gelöscht, was als Dateinamen mit „J" beginnt. Sie müssen sich also vorher vergewissern, ob Sie nicht versehentlich Files löschen, die

Sie noch benötigen. Mit !ERA,"*.*" löschen Sie übrigens Programme auf der Diskette. Deshalb sollten Sie diesen Befehl nur anwenden, wenn Sie absolut sicher sind.

Ganz verloren allerdings erst dann, wenn Sie auf diese Diskette neue Programme abspeichern, denn dann werden die durch den ERA-Befehl quasi freigegebenen Sektoren auf der Diskette für die neuen Files benutzt. Sie können, falls Sie einen Diskettenmonitor besitzen, gelöschte Programme noch retten. Aber nur, solange Sie die Diskette noch nicht wieder zum Abspeichern benutzt haben! Also, Vorsicht!

Alle Beispiele bezogen sich auf den 664 und den 6128. Besitzer von 464-Geräten müssen diese Dateinamen in einer Variablen ablegen und diese dann mit Hilfe des „Klammeraffen" löschen.

Umtaufen leicht möglich

!REN,"Neuer Dateiname","Alter Dateiname":

Dieser AMSDOS-Befehl erlaubt Ihnen auf Diskette vorhandene Dateien oder Programme umzubenennen, ohne den Inhalt selbst zu verändern. !REN,"jojo.bas","Spiel .bas" verändert also den Filenamen "Spiel.bas" in "Jojo.bas". Übrigens: Wild cards sind hierbei nicht erlaubt, Sie müssen bei der Umbenennung mehrerer Dateien jeden Namen einzeln ändern. Auch hierbei müssen 464-User die Namen erst wieder in Strings ablegen und dann diese verwenden. Also:
a$="Spiel1.bas":n$=
"Jojo.bas":1REN,@n$,@a$.

Alles klar?

!DIR, "Dateien"

Dieser Befehl führt zum Auflisten der Files auf der Diskette im Laufwerk. Allerdings werden die Namen nicht alphabetisch ausgegeben (wie beim cat-Befehl), sondern in der Reihenfolge, wie sie auf der Diskette vorhanden sind. Da bei diesem Befehl Wild cards erlaubt sind, können Sie sich, wenn Sie wollen, die Dateien ausgeben lassen, die den im String-Ausdruck enthaltenen Buchstaben führen. Beispiel:
!DIR,"*.bak"
Zeigt alle auf der Diskette enthaltenen Backup-Dateien an. Mit
!DIR,"Jojo.*"
werden alle Files angezeigt, die unter "Jojo" und verschiedenen Extensionen abgespeichert wurden.

Wollen Sie alle BAS-Dateien sehen, heißt der Befehl:
!DIR,"*.bas".

Auch hier müssen die 464-Besitzer wieder mit „Klammeraffen" und Strings arbeiten.

Jetzt geht's weiter mit BASIC!

Das waren die zusätzlichen Befehle, die Ihnen AMSDOS zur Verfügung stellt und die Sie nur aufrufen können, wenn Sie eine Diskettenstation besitzen. Alle folgenden Befehle können Sie von BASIC aus aufrufen, Sie stehen allen CPC-Besitzern offen.

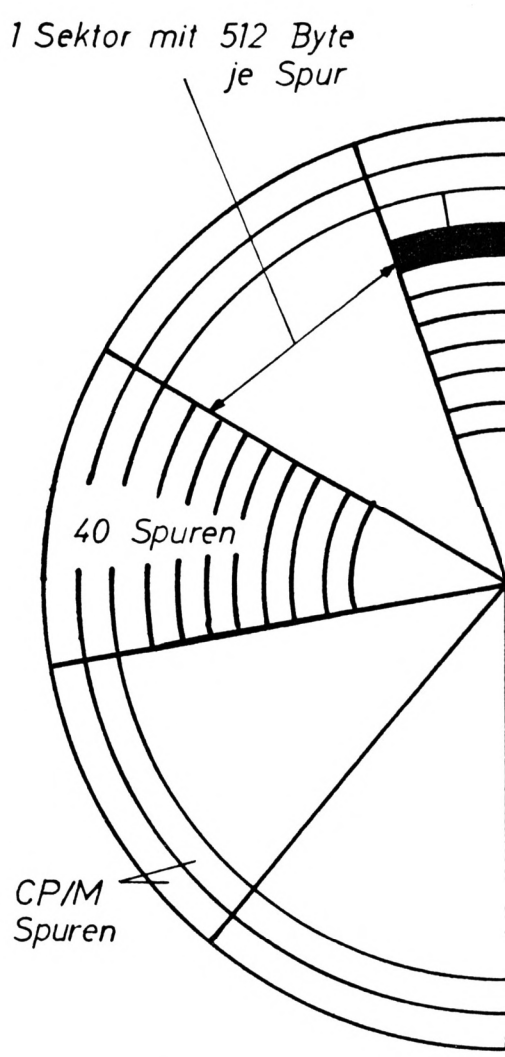
Was nützt Ihnen das beste selbstgeschriebene Programm, wenn Sie es nicht abspeichern und später wieder aufrufen können! Denn wenn Sie Ihren Computer ausschalten, ist der gesamte RAM-Speicherinhalt verloren.

Also brauchen Sie die Möglichkeit, Ihre Programme auf Dauer zu behalten. Und dazu dient der SAVE-Befehl.

Höchstens acht Zeichen!

Die Form dieses Befehls lautet:
SAVE"Dateiname.ext"

Als Dateinamen können Sie jeden beliebigen Namen eingeben, der maximal acht Buchstaben hat. Jedoch sollten Sie keine allgemeinen Namen verwenden, wie zum Beispiel „Prgramm1". Denn dann wissen Sie nach einigen Wochen garantiert nicht mehr, was sich dahinter verbirgt. Also lieber versuchen, mit den zur Verfügung stehenden acht Buchstaben Rückschlüsse auf den Inhalt des Programms zuzulassen; zum Beispiel: "Fotodat". Mit einem solchen Namen ist dann alles klar.

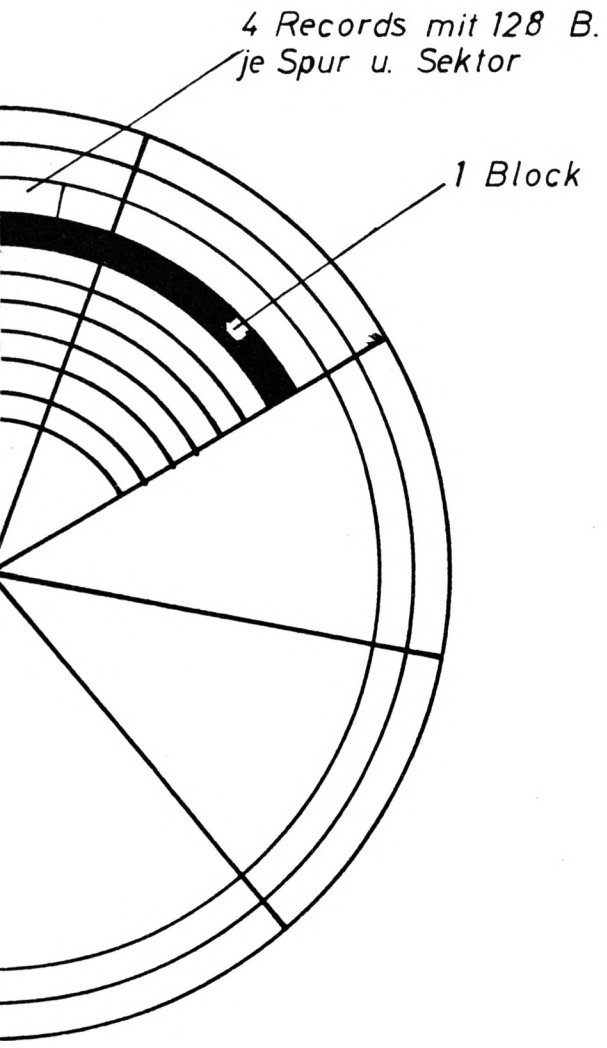


Die Erweiterung des Programm-Namens (Ext) braucht nicht mit angegeben zu werden. Fehlt sie, so wird automatisch als Extension BAS angefügt. Bei normalen BASIC-Programmen ist das okay. Anders sieht es jedoch mit Binärfiles oder Dataladern aus. Hier sollten Sie die Erweiterung einer eindeutigen Kennzeichnung angeben. Wir haben uns angewöhnt, bei sogenannten Dataladern die Extension „lad" (für Lader) anzugeben. Daß Binärfiles mit „.bin" gesichert werden sollten, ist klar. Falls Sie Bilder abspeichern wollen, ist der Zusatz „pic" zu empfehlen.

Jetzt stellen wir uns einmal vor, Sie schreiben an einem Programm, das jedoch nicht an einem Tag fertig wird. Sie speichern dieses Teilprogramm ab und laden es sich am nächsten Tag wieder ein. Wenn Sie es dann nochmals unter dem selben Namen abspeichern, so erhält die neue Version wieder den Zusatz BAS. Doch was geschieht mit der alten Variante? Die bekommt automatisch den Zusatz „bak". Daran erkennen Sie also die vorhergehende Version.

Alte Versionen werden gekennzeichnet

Wird jedoch zum dritten Mal der gleiche Programmname zum Speichern benutzt, so erhält die dritte Version den Zusatz BAS, die zweite Version „bak", und erst jetzt wird die erste Version im Inhaltsverzeichnis der Diskette (Directory) so gekennzeichnet, daß sie nicht mehr angezeigt wird. Und damit sind diese Sektoren auf



der Diskette auch wieder zur Aufnahme anderer Daten freigegeben. Diese erste Version ist dann für den normalen CPC-Benutzer verloren.

Vorsicht mit dem Buchstaben „P"

In Ihrem Handbuch steht unter dem Save-Befehl noch der Hinweis, daß Sie normale BASIC-Programme beim Abspeichern auch sichern können, indem Sie den Buchstaben „p" anhängen. Also in der Form: SAVE"Programmname",p. Doch davor können wir den CPC-Neueinsteiger nur warnen, denn Sie können diese Programme dann nicht mehr laden und listen, sondern nur noch mit „run" starten. Und alte Hasen stört dieser Listschutz durch „p" überhaupt nicht. Die können ihn nämlich durch drei einfache Poke-Befehle wieder aufheben.

In bestimmten Fällen benötigen Sie Ihr Programm jedoch nicht als normales BASIC-File, sondern als sogenannte ASCII-Datei. Was sich dahinter verbirgt, werden wir noch erläutern. An dieser Stelle nur der Hinweis, wie Programme als ASCII-Datei gesichert werden. Sie brauchen in einem solchen Fall an den Dateinamen nur noch ein „a" anzuhängen. Beispiel: SAVE"Dateiname",a.

Bildschirminhalt sichern

Wollen Sie Ihr Programm jedoch als Binärfile abspeichern, wird ein „b" angehängt. Eine solche Form der Speicherung entspricht einem Speicherauszug. Typisch für diese Art der Datensicherung ist sowohl das Abspeichern von Bildschirminhalten als auch das Sichern von Maschinenprogrammen. Damit der CPC weiß, welchen Speicherinhalt er sichern soll, müssen Sie die Grenzen mit eingeben. Beispiel:
SAVE"Bilddemo.pic,b,
&C000,&3FFF.

Mit diesem Befehl wird unter anderem der Bildschirminhalt gesichert (C000=Beginn des Bildschirmspeichers, 3FFF=Länge).

Um die Bildschirm-Meldung nicht mit abzuspeichern, müssen Sie folgendermaßen vorgehen:

```
10 mode 2
20 load"Bilddemo.pic"
30 call &BB06
40 save"Bilddemo.pic",B,
&C000,&3FFF
```

In der Zeile 10 geben Sie den Modus ein, in dem das Bild erstellt wurde. Der Call-Befehl in Zeile 30 dient nur zum Anhalten des Programmes; zum Beispiel, um einen Diskettenwechsel vorzunehmen.

Das Laden von Programmen

Um abgespeicherte Programme wieder einlesen zu können, wird der LOAD-Befehl benutzt. Mit LOAD "Jojo" bekommen Sie das Programm „Jojo" wieder in den Speicher. Eine Extension brauchen Sie nicht anzugeben. Fehlt sie, wird automatisch die BAS-Version eingelesen. Ist jedoch ein ASCII-File mit dem gleichen Namen auf der Diskette vorhanden, wird dieses geladen. Um trotzdem die BAS-Version laden zu können, müssen Sie also in diesem Fall die Namenserweiterung (BAS) mit angeben.

Wenn das Programm im Speicher ist, können Sie es bearbeiten oder mit „RUN" starten. Bedenken Sie dabei, daß bei Verwendung des LOAD-Befehls ein eventuell im Speicher vorhandenes altes Programm gelöscht wird.

Wollen Sie den Datalader „Jojo.bas" einlesen, müssen Sie auch den vollen Namen angeben. Also: load"Jojo.bas".

Wollen Sie jedoch ein Programm, das mit dem Zusatz „p" beim Speichern gesichert wurde, in den Speicher holen, so wird das natürlich nicht funktionieren. Deshalb also unser Hinweis, daß Sie beim Saven möglichst auf das „p" verzichten sollten.

Soll ein Binärfile in den Speicher gebracht werden, so müssen Sie vorweg erst den Speicherbereich schützen, damit die eingelesenen Daten nicht von BASIC aus überschrieben werden. Beispiel:
Memory &6FFF:LOAD "Jojo.bin",&7000

Starten Sie gleich mit RUN

Wollen Sie Ihr Programm gleich starten, so geben Sie RUN, gefolgt vom Dateinamen, an. Also: RUN"Jojo". Dann wird das Programm in den Speicher geholt und sofort gestartet. Dies geht auch mit Programmen, die mit SAVE"Programmname",p gesichert worden sind.

Der CPC kann auch mixen!

Haben Sie ein Programm im Speicher und möchten ein zweites hinzufügen, ohne das alte zu löschen, was ja beim LOAD-Befehl zwangsweise erfolgt, so können Sie den MERGE-Befehl anwenden. Nehmen wir an, Sie wollen „Demo1" mit „Demo2" vermischen, so können Sie folgendes machen:
load"Demo1":merge"Demo2"

Schon sind beide Programme zusammen im Speicher. Beachten Sie jedoch dabei, daß doppelte Zeilennummern von hinzugemergten überschrieben werden.

Und noch ein Hinweis: Beim CPC 464 muß das hinzuzumischende Programm als ASCII-Datei vorliegen. Dies wäre also der erste Fall, bei dem Sie auf das Abspeichern im ASCII-Format zurückgreifen müssen.

Die OPENOUT und OPENIN-Befehle werden wir in einem späteren Beitrag unter dem Stichwort Datenverwaltung behandeln. *JE*■

SZENEN EINER EHE

Sie küßten und sie schlugen sich

Wer sagt denn, daß nur die amerikanischen Karriere-Stories interressant seien? Steven Jobs und Jack Tramiel finden ihre deutsch-englischen Gegenstücke in den Gebrüdern Schneider, Fred Köster und Alan Sugar. Und bei der Feststellung, im Allgäu sei das deutsche Silicon Valley zu finden, muß man den Türkheimer Rundfunkwerken recht geben. Die Hintergründe der europäischen Computerindustrie lesen sich genauso faszinierend wie die der amerikanischen.

Auf der CeBIT im März dieses Jahres wurde alles offenbar: Die Ehe zwischen Schneider und Amstrad war geschieden, das Papier, das die beiden die letzten Monate noch verband, gilt nur noch bis zum Mai. Den Besuchern der Messe war der Vertrag ohnehin egal. Wer Augen hatte, der konnte sehen: Schneider präsentierte Neuheiten, die die bisherige Produktpalette von Amstrad empfindlich stört; Amstrad versuchte mit viel Aufwand den deutschen Markt zu erobern. Mit den Spekulationen und Gerüchten ist es nun vorbei: Es ist endgültig aus. Hinter beiden Firmen steckt aber eine eindrucksvolle Entwicklung, und die kurze Partnerschaft, die sie eingingen, sorgte noch einmal für Gesprächsstoff. Da nun ein Kapitel der Computergeschichte zugeschlagen wird, dafür aber zwei neue begonnen werden, wird es Zeit, noch einmal die merkwürdige Ehe in vier Akten zu beleuchten.

ERSTER AKT: GLEICH UND GLEICH GESELLT SICH GERN

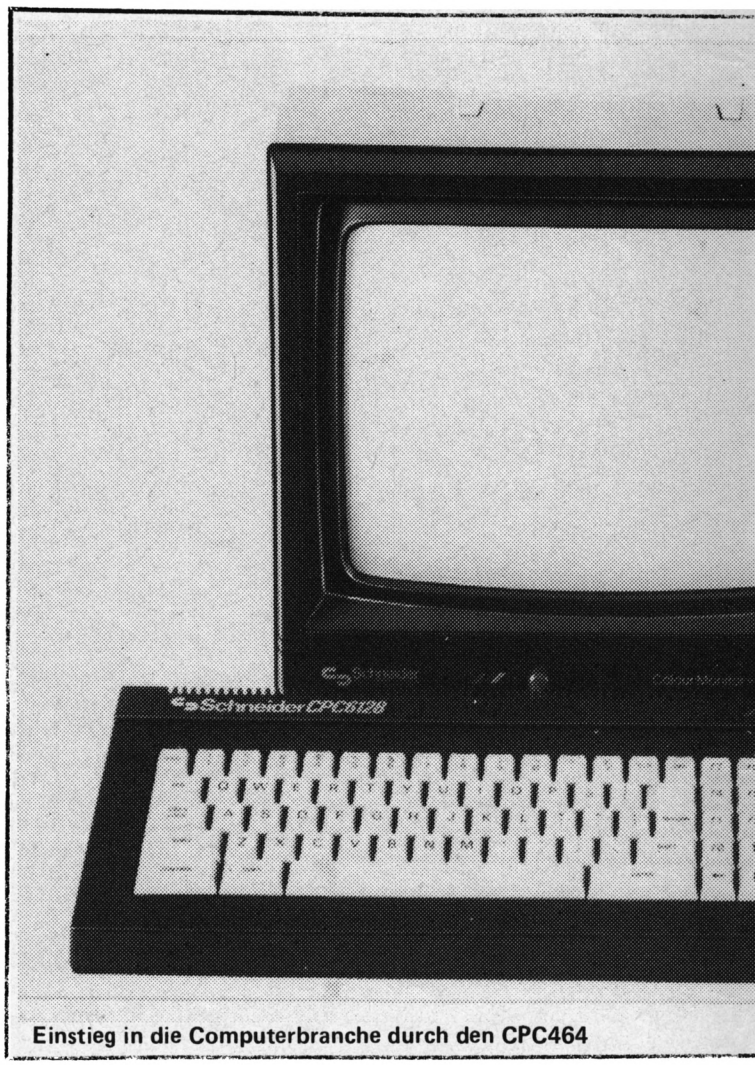
Mit nur 21 Jahren gründete Alan Sugar in Großbritannien seine Firma. Er nannte sie Amstrad, abgeleitet von Alan Sugar Trading. Genau dies, der Handel mit Fernseh- und HiFi-Geräten, war sein Bestreben. Die Bezeichnung Hersteller verdiente er sich erst in späteren Jahren. Bis dahin galt es, billig in Fernost einzukaufen, ein Amstrad-Etikett aufzukleben und das Ganze als Markengerät auf den englischen Markt zu bringen. Es klappte.
Die standesbewußten Briten griffen lieber zu einer einheimischen Marke beziehungsweise zu dem, was sie dafür hielten, bevor sie sich mit unaussprechlichen koreanischen, japanischen oder taiwanesischen Firmennamen abquälten. Hinzu kam, daß Alan Sugar ganz bewußt die Preise niedrig hielt. Die Masse der verkauften HiFi-Geräte brachte den Verdienst; Elektronik für den kleinen Geldbeutel, hieß die Devise. Es fehlte nicht viel, und man hätte den Jungmanager als Volkshelden gefeiert.
In Deutschland gab es eine ähnlich orientierte Firma; die der beiden Gebrüder Schneider, Bernhard und Albert. Auch hier wurde billig eingekauft und preiswert verkauft. Aber man war in Türkheim/Allgäu schon einen Schritt weiter: Schon längst führten die Schneider-Rundfunkwerke Einzelteile ein, die erst in Deutschland zu einem fertigen Gerät montiert wurden. Ganze Ladungen von Platinen, Bildröhren und anderen Dingen wurden zu Fernsehern, Stereoanlagen und Radios zusammengesteckt. Das hatte auch zur Folge, daß man bereits eigene Entwürfe ausbrütete und danach die Geräte in irgendeinem fernöstlichen Land in Auftrag gab.



Alan Sugar ist Amstrad

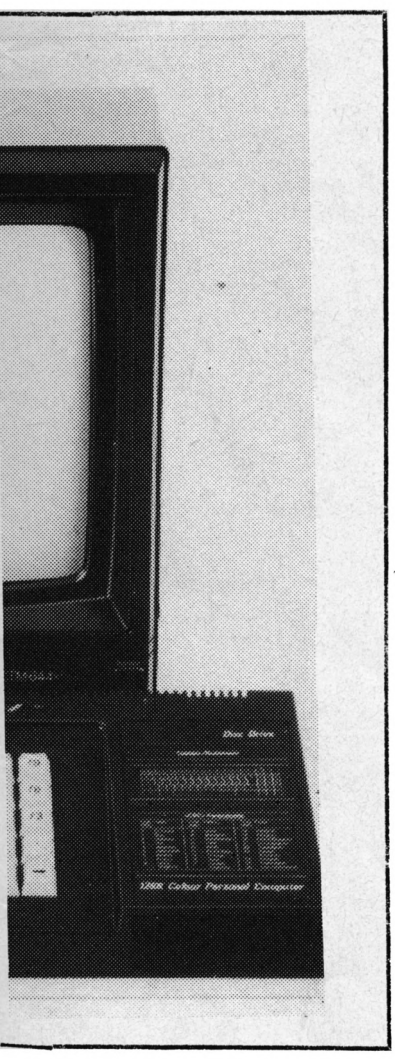
DIE WANDLUNG VOM HÄNDLER ZUM HERSTELLER

Es war eine Methode, die auch Alan Sugar bald praktizierte. Amstrad wan-



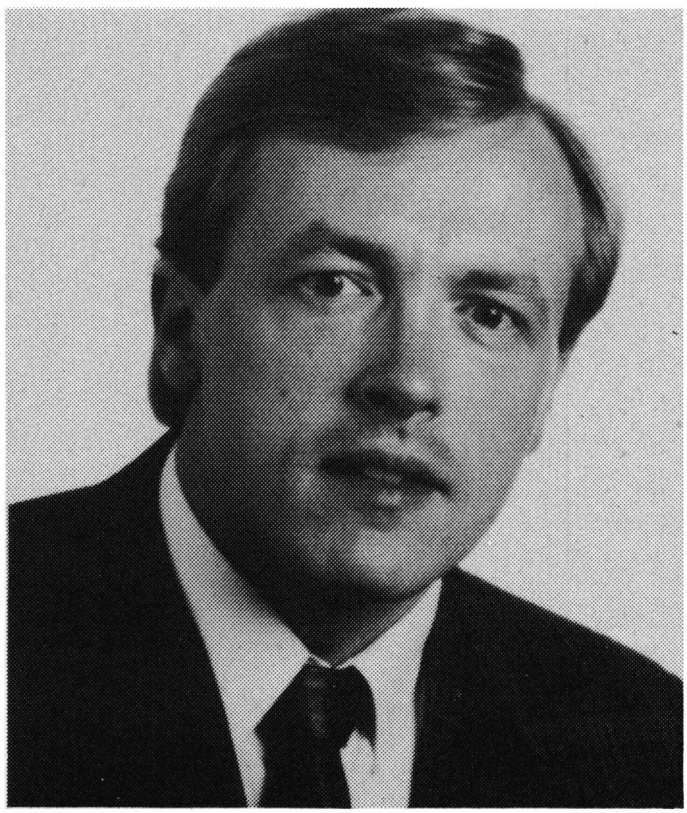
Einstieg in die Computerbranche durch den CPC464

delte sich vom Händler zum Hersteller und wurde größer und größer. Die Einäufer von Schneider mußten Alan Sugar oft in Taiwan, Hongkong oder Korea begegnet sein, wenn es darum ging, eine Charge Platinen zu ersteigern. Der englische Karrieremann gilt auch noch heute als ein Manager der guten alten Zeit: alles selbst machen und immer an der Front sein.
Kein Wunder, daß solch ein Firmenchef den richtigen Riecher hat und den Boom der Homecomputer rechtzeitig erkannte. Der wachsende Markt der Homecomputer lockte ihn. Ein Amstrad-Rechner hätte als Konkurrenz nur den Commodore C64 zu fürchten gehabt, den anderen Firmen fehlte ohnehin das richtige Marketing, um sich durchzusetzen. Lediglich in England gab es noch ein Genie namens Sinclair, dessen Spektrum-Rechner ein paar Marktanteile hatte. Ihm widmete Alan Sugar besonderes Augenmerk.



EINSTIEG IN DEN COMPUTERMARKT

Woher er Mitte des Jahres 1984 den CPC464 genommen hatte, ist bis heute ein Rätsel. Hergestellt wurde er, in bester Firmentradition, in Korea. Die Entwicklung und die Produktionsvorgaben kamen aber sehr wohl aus Großbritannien. Wer den Sinclair ZX 81 gut kannte und sich das ROM des CPC464 anschaute, der hatte schnell einen bestimmten Verdacht, wer den CPC-Rechner entwickelt haben könnte. Daß die Systemprogrammierer und Platinenlayouter einen Amstrad-Firmenausweis hatten, ist sicher; wie lange schon, das fragte man im höflichen England nicht. Dem Rechner schadeten solche Gerüchte nicht, im Gegenteil: Die Präsentation des CPC464 war eine Sensation. Für umgerechnet 1400 Mark erhielt man einen 64-KByte-Rechner mit eingebautem Datenrekorder und Farbmonitor. Und die Leistungsbeschreibung las sich wie die Wunschliste der Commodore-C64-Besitzer. Mit einem 80-Zeichen-Modus, bis zu 16 Farben, einer der besten Tastaturen und der Aussicht auf das Betriebssystem CP/M war die Komplett-Konfiguration ein Preisknüller.
Aber Computer sind keine Stereoanlagen; eine simple Feststellung, die Alan Sugar schon längst verinnerlicht hatte. Die Verkaufszahlen mußten hoch sein, damit sichSoftwarefirmen und Hersteller von Peripherie für den Rechner erwärmen konnten. Aus dem Beispiel Sinclair, dessen Firma langsam in die roten Zahlen rutschte, hatte Amstrad schon gelernt. Ein paar Marktanteile in England ließen keinen Softwareverlag in Verzückung geraten, und ein Computer ohne Spiele ließ sich nicht verkaufen. Außerdem war die Preiskalkulation des CPC464 so knapp, daß nur ein Massenverkauf Gewinn bringen konnte.
So klemmte sich Alan Sugar den Prototyp unter den Arm und ging auf Europatournee. Was er vorhatte, offenbarte eine kleine Lötbrücke auf der Platine des CPC. Bastler konnten damit bei Veränderungen eine von sieben Firmenbezeichnung auf den Monitor bringen. Das Angebot galt jedoch nicht den Lötkolbenfreaks, sondern den zukünftigen Vertriebspartnern von Amstrad. „Macht euren eigenen Computer, aber nehmt meinen dazu her", so oder ähnlich muß die Rede gewesen sein. Ein verlockendes Geschäft, und Alan Sugar konnte deshalb sogar unter mehreren Bewerbern wählen.



Wilfried Rusniok, ehemals bei Commodore, ist seit dem 1.10.1987 Entwicklungsleiter bei Schneider

SCHNEIDER STEIGT EIN

Den Zuschlag erhielt schließlich Schneider, und die Gründe lagen auf der Hand: Das Vertriebsnetz war da, der Markenname war da und auch das Know-how mit der Elektronik stimmte. Schneider baute sogar eine eigene Computerdivision auf, denn wie sagte Alan Sugar: „Computer sind keine Stereoanlagen".
Für die Leitung dieser Spezialeinheit wurde Fred Köster gewonnen, der die Feinheiten eines Vertriebes bei IBM studiert hatte und deshalb als Manager mit Sachkenntnis galt und heute noch gilt. Bei dem Deal wurde jedoch ein ent-

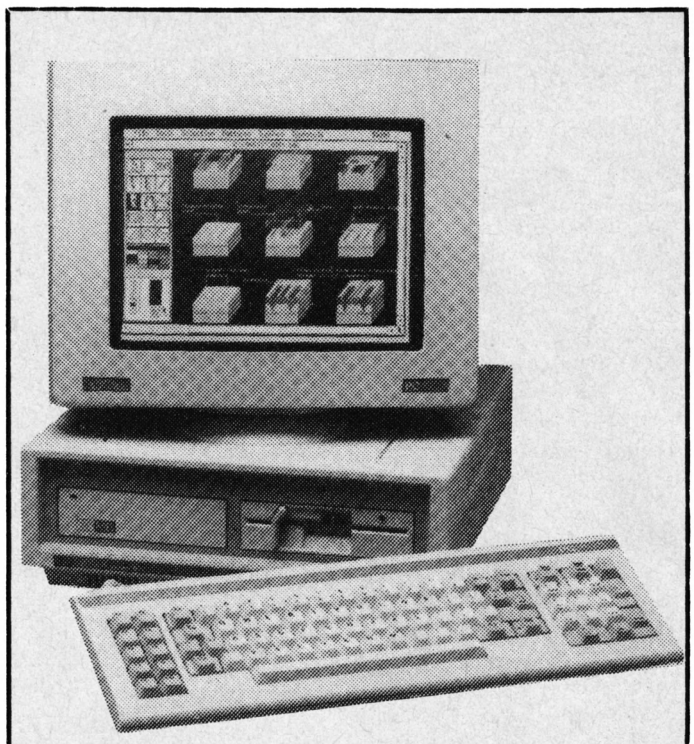
scheidender Nachteil übersehen. Er machte sich erst im dritten Akt der Ehe bemerkbar und wurde bis dahin auch gegenüber dem Endkunden kaschiert. Aber immer schön der Reihe nach.

ZWEITER AKT: AUF DER STRASSE DES ERFOLGES

1984 war die Ehe perfekt, und die Flitterwochen funktionierten ausgezeichnet. Schneider konnte gar nicht so schnell ausliefern, wie die Bestellungen eintrudelten, und in England erlebte Amstrad den gleichen Boom. Software-Firmen mußten gar nicht mehr aufgefordert werden, sich um den CPC zu kümmern. Sie wollten von alleine an dem Geschäft partizipieren. Es erschienen Bücher, Spiele und Anwendungsprogramme.
Peripherie-Hersteller gaben sich die Klinke in die Hand, um irgendein Gerät unter dem Namen Schneider oder Amstrad verkaufen zu können. So gelang es der Firma Brother als einer der ersten, einen Matrixdrucker loszuschlagen, der in seiner Klasse ähnlich preisgünstig war wie der CPC in seiner.
Aber es zeigten sich auch schon die ersten Fehler, sowohl in technischer Hinsicht als auch beim Marketing. Der Druckeranschluß hatte seine Makken. Statt, wie jeder andere Centronics-Ausgang, acht Bit zu übertragen, ließ sich der des CPC nur zu sieben Bit bewegen.
Die Palette der angebotenen Drucker wurde stark eingeschränkt, und für viele Firmen war der Rechner wieder so schnell gestorben, wie er auf den Markt kam.

DIE ERSTEN FEHLER IM MARKETING

In England versuchte man mittlerweile, das versprochene Super-Diskettenlaufwerk aufzutreiben



Der Durchbruch in den PC-Bereich: PC 1640

und geriet dabei in die Klemme. Niemand wußte zum damaligen Zeitpunkt, welches der neuen Diskettenformate sich durchsetzen würde, – 3,5 oder 3 Zoll. Alan Sugar tippte auf die kleineren Scheiben und fiel beinahe auf die Nase. Der erste Laufwerkshersteller in Fernost bekam Konstruktionsprobleme und konnte nicht liefern. Als endlich eine Ersatzfirma gefunden war, war der Ankündigungstermin längst überschritten. Die Amstrads, die sich bisher immer an bewährte, längst eingeführten Technologien orientierten, wollten damals ein einziges Mal zum Trendsetter werden und mit Massenverkäufen einen Standard schaffen. Doch dazu war es nun zu spät. Parallel zu dem 3-Zoll-Format existierte schon ein 3,5-Format, und das erwies sich als das bessere.
Von Standard konnte keine Rede mehr sein, nur der CPC benutzte die kleineren Scheiben. Zwar waren die Käufer angesichts der Leistung begeistert, auch stimmte mit 900 Mark wieder einmal der Preis, aber ein Wermutstropfen blieb doch zurück. Er offenbarte sich in teuren Disketten, da die Auflage bei der Herstellung zwangsläufig zu klein war.

AMSTRADS DEVISE: KEINE EXPERIMENTE MEHR

Amstrad und Schneider hatten ihre Lektion gelernt und riskierten vorerst keine Neuerungen mehr. Der 464 lief gut, also baute man den 664, dessen Unterschied zum Vorgänger lediglich in ein paar Verbesserungen im ROM bestand, und der statt des Datenrekorders ein eingebautes Laufwerk vorzeigte. Das einzige Verkaufsargument für den Rechner war der Preis.
Da die Kundschaft etwas mehr erwartet hatte und enttäuscht reagierte, zog man sofort nach.
Mit dem CPC6128 wurde schnell der dritte Rechner innerhalb eines Jahres vorgestellt. Hier gab es eine Erweiterung auf 128 KByte, ein eingebautes Laufwerk und natürlich wieder einen günstigen Preis, denn auch Amstrad reagierte auf den Preisverfall in der Elektronik.

LÄSTIGE KONKURRENZ WURDE AUFGEKAUFT

Amstrad, seit 1980 an der Börse notiert, machte Riesenumsätze und investierte einen großen Teil der Gewinne, um das Computergeschäft weiter auszubauen. Lästige Konkurrenz wurde einfach aufgekauft. Sinclair, mittlerweile dramatisch ins Defizit gekommen, mußte seine Firma veräußern. Der englische Markt war nun in Alan Sugars Hand, selbst Commodores Konkurrenz konnte ihn nicht mehr kratzen.
Schließlich kam 1986 der Joyce, so benannt nach der damaligen Chefsekretärin, auf den Markt. Schneider übernahm den Textcomputer und kündigte ihn als „die Wende" an. Er war es auch, wenngleich nicht in dem Sinn, wie es dem Kunden klar gemacht wurde. Es begannen nämlich die ersten Streitereien in der Ehe.

DRITTER AKT: ERFOLG MACHT ÜBERMÜTIG, DIE KRISE STELLT SICH EIN

Das Konzept des Joyce stimmte nur in Großbritannien. Dort wurde der CPC größtenteils für Spiele eingesetzt, trotz des Diskettenlaufwerkes und den damit verbundenen CP/M-Möglichkeiten. Wer ernsthafte Anwendungen, etwa eine Textverarbeitung, benutzen wollte, der dachte gar nicht daran, einen CPC zu kaufen. Für dieses Kaufverhalten kam der PCW 8256, wie der Joyce korrekt zu nennen ist, gerade richtig: Ein Rechner nur für Anwendungen, ohne deshalb gleich so teuer zu sein wie ein IBM-Kompatibler.
In der Konsequenz aus

dem Laufwerksdebakel ging Alan Sugar kein Risiko ein. Der Prozessor war ein bewährter Z80, die 256 KByte wurden durch ausgefuchtes Bankswitching erreicht. Alte Technik in neuem Gewand.

JOYCE – KEIN RECHNER FÜR DEN DEUTSCHEN MARKT

Deutsche User erwarteten von Schneider aber etwas anderes, und CP/M-Programme waren schon lange in Gebrauch. Wer schreiben wollte, der tat dies mit WordStar, und wer eine Datei benutzte, der arbeitete schon längst unter CP/M mit dBase. Der neue Rechner, den man sich wünschte, sollte wirklich neu sein, nicht wieder einen aufgedrehten Z80-Prozessor enthalten. Gerade der Aufkauf der Sinclair-Rechner veranlaßte viele dazu, auf einen verbesserten QL zu hoffen. Dieser verstaubte jedoch in den englischen Lagerhallen, und Schneider hatte keinen Einfluß auf die Vertriebspolitik von Amstrad.
Dies war der entscheidende Nachteil des Partnerschaftsvertrages. Hatte Alan Sugar 1984 denn nicht auch dafür gesorgt, daß ein großer deutscher Elektronikhersteller im Bereich der Computertech-

KEIN SCHNEIDER-SPECTRUM-RECHNER

nologie kaltgestellt wurde? Ja sicher, denn Schneider durfte nichts entwikkeln, was der Produktplanung des Hauses Amstrad widersprechen würde. In Türkheim war man auf die Innovation von Amstrad angewiesen, und die war gleich Null. Wenn es Alan Sugar in den Kram paßte, daß der QL Schimmel ansetzte, dann war daran nichts zu ändern. Sein Trick war 1984, daß man einen europäischen Konkurrenten ausspielte, indem man ihn zum Vertriebspartner machte.
Schneiders einzige Möglichkeit des Widerstands war, ein Produkt nicht in Deutschland zu vertreiben. So geschiehen mit den Sinclair-Spektrum-Rechnern. Fred Köster weigerte sich, dem CPC in Deutschland eine Konkurrenz durch die eigene Firma gegenüberzustellen. Gleichzeitig konnte man der Werbung für den Joyce eine gewisse Halbherzigkeit anmerken. Schneider wollte gar nicht behaupten, daß es sich dabei um einen Computer handelte. Zwar verkaufte sich die Textverarbeitung mit Bildschirm recht gut – es dürften mittlerweile 60000 Geräte sein, die in der Bundesrepublik benutzt werden doch fast schien es, als wolle man mit dem guten Stück nichts zu tun haben. Mit den Verbrauchern wartete man auch in Türkheim auf eine Sensation, auf etwas völlig Neues.
Es zeigte sich ganz nebenbei, daß auch Schneider keine Informationen erhielt. Mehr und mehr gerieten sie zur Nummer zwei in der Ehe, und Alan Sugar hatte die Hosen an.
Europa war ihm längst zu klein geworden. Sein Privatjet düste immer häufiger in die USA, wo er bereits den CPC6128 vorgestellt hatte. Die Ankündigung geriet allerdings zu einem Debakel. Amerika hatte kein Interesse an einem englischen Homecomputer. Wer in Übersee etwas werden wollte, der mußte die IBM-Schiene fahren, und er benötigte vor allem einen Vertriebsweg, mit dem sich die Konkurrenz aus dem Feld schlagen ließ.



Schneiders erste eigenständige Entwicklung ist der AT 2640

Alan Sugar war genau der Mann, der sich an solchen Problemen messen konnte. Während seine Niederlassungen in Europa – gerade erst in Frankreich und Spanien gegründet – mit dem CPC und dem Joyce das nötige Geld einbrachten, macht er sich an die Entwick-

VIERTER AKT: DIE EHE ENDET

lung eines MS-DOS-Rechners. Ende 1986 war das Gerät fertig, und der PC 1512, so hieß er, eroberte innerhalb von 60 Tagen einen Marktanteil von elf Prozent in England. Schneider übernahm auch diesen Rechner, und es wiederholte sich die Sensation des CPC. In Deutschland konnte gar nicht so schnell ausgeliefert werden, wie Bestellungen eingingen. Und dies trotz der augenscheinlichen Mängel, die der vermeintlich Kompatible hatte. Ein eigenes Grafiksystem, mittlerweile überholte 5,25-Zoll-Disketten und ein bis dato unbekanntes BASIC machten den Rechner zum Sonderling.

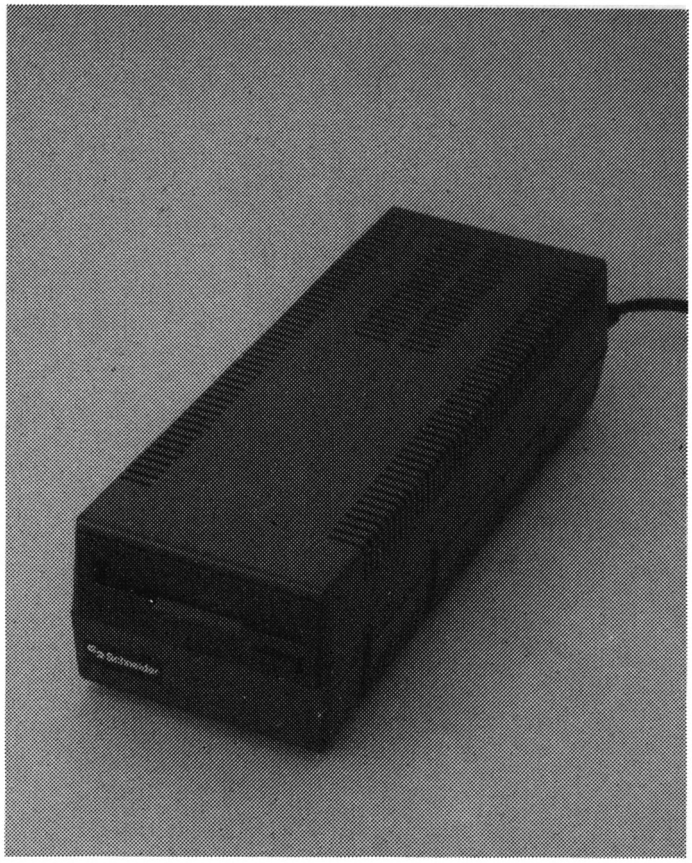
AMSTRAD LEGTE PLÖTZLICH WERT AUF DEN EIGENEN SCHRIFTZUG AM GERÄT

Vielen Käufern fiel jedoch der Schriftzug auf der Tastatur auf. Schneider by Amstrad hieß es da; die Engländer wollten jedem deutschen Kunden klarmachen, wer nun das Sagen hatte. „Sicherlich ein Schneider-Gerät, also kauft es. Aber es kommt von Amstrad", so ließ sich dieses kleine Detail wohl übersetzen.
Schneider schwammen die Felle weg. Man hatte schon längst nichts mehr zu sagen und mußte verärgert zusehen, wie sich ringsherum in Europa die Firma Amstrad breit machte. Frankreich, Spanien und Griechenland waren schon fest in der

Hand des cleveren Engländers, der so vorführte, daß er einen Partner nur kurzfristig gebrauchen konnte. Irgendwann war die Firma sein eigen und er hatte eine weitere Niederlassung.
In den USA klappte der Deal mit dem PC1512 trotzdem nicht. Es mangelte immer noch an einem kompetenten Vertriebspartner, der den Namen zu einem Markenzeichen machen konnte. Amstrad war dem amerikanischen Verbraucher unbekannt, und bei No-Name-Produkten zählt nur der Preis. Taiwan und Korea hatten diese Verkaufsstrategie schon längst für sich entdeckt und nahmen keinen Europäer in die Gemeinschaft auf.
Also öffnete Alan Sugar wieder einmal die Brieftasche, winkte mit fünf Millionen Pfund und kaufte eine Firma. Seine Wahl fiel auf die Vidco Incorporation, mit der er auf der Frühjahrs-Comdex in Atlanta den PC-1640 vorstellte. Es war ein stark verbesserter Nachfolger des 1512, und er hatte alle Aussichten auf Erfolg.

SCHNEIDER WURDE WACH

Die Präsentation brüskierte Schneider gleich doppelt. Wieder wurde man nur knapp vor der Öffentlichkeit mit dem Gerät überrascht und außerdem mit Schweigepflicht belegt. Während man in Türkheim noch darauf wartete, daß das Redeverbot aufgehoben wurde, gab Alan Sugar seine Show in den USA und war ganz stolzer Vater. Wochen später erst konnte Schneider eine Pressekonferenz veranstalten. Es muß unangenehm gewesen sein, von einer Neuvorstellung zu reden, die bereits jeder kannte. Aber dabei wurden auch jene Produktbereiche vorgestellt, in denen man noch aktiv sein durfte,



Mißglückter Versuch, einen Standard zu setzen: das Drei-Zoll-Laufwerk DDI-1

so zum Beispiel der Fernkopierer. Was Schneider damals plante, erfuhr keiner der Pressevertreter.

DER AT WURDE ZUM SYMBOL DER EMANZIPATION

Niemand redete davon, daß man sich um die Zukunft sorgte, obwohl doch an der Seite des Partners die Geschäfte glänzend liefen. Es muß aber damals schon der Zeitpunkt gewesen sein, an dem man den Gedanken an einen AT faßte. Zur Systems im Herbst 1987 war es dann so weit, der PC2640 wurde präsentiert, und in einer Pressekonferenz machte Fred Köster endlich Nägel mit Köpfen.
Er stellte Wilfried Rusniok vor, der bislang bei Commodore in Braunschweig gearbeitet hatte. Seit dem 1. Oktober 1987 war er für die Entwicklungsabteilung in Türkheim verantwortlich. Wie es hieß, war der AT nur der Beginn einer neuen Ära. Planung und Fertigung von neuen Rechnern sollten nach Deutschland kommen. Ohne daß Köster die Neuheiten konkretisierte, war doch klar, daß man sich nicht mehr durch den Vertrag knebeln lassen wollte. Diesmal war Schneider der cleverere Partner. Man machte sich den schlechten Informationsfluß von England zunutze und baute schnell ein Gerät. Mochte Alan Sugar doch einen AT in der Planung haben. Bislang hatte er nichts verlauten lassen und der Vertrag, der zu diesem Zeitpunkt noch Gültigkeit hatte, besagte nur, daß

SUGAR EROBERTE AMERIKA UND VERSUCHT ES NUN IN DEUTSCHLAND

Schneider keine Produkte herstellen durfte, die gegen die Interessen Amstrads verstoßen. Durch Ideen, von denen man nichts wußte, konnte man in Türkheim auch nicht die Abmachung verletzen. Tatsächlich war Amstrad auch noch gar nicht so weit. Man konzentrierte sich voll auf das Amerika-Geschäft und stellte dort den PPC 640 vor, einen tragbaren IBM-Computer zum Niedrigstpreis. Aber die Ehe war damit zu Ende. Alan Sugar mußte einsehen, daß er mit Schneider keinen Partner hatte, der sich irgendwann aufkaufen ließ und sein Imperium um eine deutsche Niederlassung vergrößerte. Seine Firma gehört ohnehin schon zu den Großen der Branche. Den Briefkopf schmücken Absender wie London, Dallas, Paris, Madrid, Sidney, Hongkong und Mailand.
Türkheim hätte sich da, vielleicht gar nicht so schlecht gemacht, aber auch Schneider ist eine gesunde Firma, die überhaupt nicht an Verkauf denken muß. Statt dessen setzt man auf wirkliche Entwicklung, selbst wenn man unter Zeitdruck steht. Man weiß, was man dem Kunden lange genug vorenthalten hat und holt nun das Versäumte nach.
Im Gegenzug mußte Alan Sugar reagieren. Was er 1984 nicht tun wollte, ist nun vollbracht. Neuerdings darf er seinem Briefkopf noch Neu-Isenburg hinzufügen. Denn hier baute Amstrad die deutsche Geschäftsstelle auf, die sich zum ersten Mal in Hannover vorstellte. Geschäftsführer ist Helmut Jost, ehemals im Vertrieb von Commodore beschäftigt. So suchen sich die beiden Ehepartner – der Vertrag läuft erst im Mai aus – ihre Mitarbeiter pikanterweise bei dem gleichen Konkurrenten.
Wer das Rennen um den Kunden machen wird, ist noch nicht raus. Beide Firmen verstehen es, mit guten Produkten für sich zu werben. Der Verbraucher kann sich aber auf jeden Fall freuen. Konkurrenz bringt Innovation, und an der fehlte es ja schon seit geraumer Zeit. *GS* □

KATEGORIEN FÜR LESERPROGRAMME?

Schon seit geraumer Zeit ist mir Ihre Zeitschrift eine wichtige Informationsquelle und damit ein unentbehrlicher Helfer bei meiner fast täglichen Arbeit mit dem CPC6128. Enttäuscht war ich allerdings einige Male, wenn ein abgetipptes Programm nicht hielt, was die Beschreibung versprochen hatte. Nicht immer sind die Listings so transparent, daß man von selbst auf das Endprodukt schließen kann. Die Effektivität und Qualität der CPC-Welt ließe sich erhöhen, wenn vielleicht nachfolgende Anregungen verwirklicht werden könnten:
Wenn eine Ausgabe über Drucker möglich ist, müßte immer ein Beispiel abgebildet werden. Man weiß dann, ob das Abtippen überhaupt lohnt und kann Anpassungen an die eigenen Bedürfnisse schon im Vorfeld überdenken und vornehmen.
Denkbar wäre es auch, daß die Autoren oder gar die Redaktion mögliche Abweichungen, Erweiterungen und ähnliches andeuten würden. Wünschenswert wäre es auch, wenn von Redaktionsseite die Stärken und Schwächen eines Programmes angesprochen würden.
Die Programme sollten in irgendeiner Weise klassifiziert und damit in ihrer Einsatzmöglichkeit und Leistungsfähigkeit festgelegt und definiert werden. Mögliche Kategorien: nützliches Kleinprogramm, semiprofessionell, profihaft. Die Leser könnten zu dieser Kategorie beitragen. Die Fazits der Spielbewertungen sollten manchmal negativer ausfallen. Reinen Ballerspielen stehen keine 80 Prozent Motivationscharakter zu.
Gerd Rothfuchs,
6799 Etschberg

Im wesentlichen haben Sie recht und wir haben dankbar einige Ihrer Anregungen aufgegriffen. Zwar konnte noch nicht alles verwirklicht werden, aber schauen Sie sich doch einmal dieses Heft an. Bei der Einteilung in Kategorien stehen wir allerdings vor Schwierigkeiten. Ist beispielsweise eine Hardcopy ein nützliches Kleinprogramm oder eignet sie sich auch für den professionellen Einsatz?
Spielbeschreibungen sind notwendigerweise subjektiv, wer etwas anderes sagt, dem sollten Sie nicht glauben. Da auch innerhalb der Redaktion nicht immer Einigkeit herrscht, bleiben wir bei unseren ausführlichen Beschreibungen, um keine Enttäuschung aufkommen zu lassen. Informieren Sie sich bitte nicht ausschließlich aus den Kurztabellen.

Haben Sie Probleme? Unsere Experten beantworten Ihre Fragen. CPC-Welt Postfach 1161 D8044 Lohhof

HARDCOPY-ANPASSUNG FÜR FUJITSU-DRUCKER

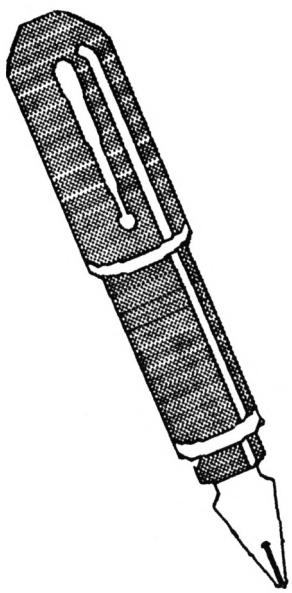
Ich besitze einen CPC464 in Verbindung mit einem Fujitsu-Drucker, welcher baugleich mit dem Panasonic KX-P1081 ist. In Ihrer Ausgabe 1/88 haben Sie ein Hardcopy-Programm abgedruckt, welches bei mir nur mit einigen Änderungen lief. Die Zeilen
35160 DATA 205,6,...,
usw. und
35170 DATA 183,254,...,
usw.
müssen in die Zeilen
35150 DATA 205,6,185,
62,27,205,,36,2,62,51,
205,,36,2,58,200,19342
35170 DATA 177,254
1,1,170,1,38,14,62,5,17,
160,0,56,17,1,10974
geändert werden. Da diese Druckertypen häufig in Verbindung mit dem CPC eingesetzt werden, wäre ein Abdruck sinnvoll.
Michael Freydorfer,
4320 Wesel
Wir und die Fujitsu-User danken Ihnen.

MISSVERSTÄNDNIS BEIM TRANSFER-PROGRAMM AUS SPECIAL 1/88

Das Programm Universaldatei in SCHNEIDER SPECIAL 1/88 wurde schon einmal von Ihnen in einer schlechteren Qualität abgedruckt. Es ist mir damals trotzdem gelungen, das Programm abzutippen, es lief jedoch nicht einwandfrei. Beim Vergleich mit dem Listing aus Heft 1/88 konnte ich feststellen, daß kein Tippfehler die Ursache war.
Der Rechner gibt eine Fehlermeldung in Zeile 3640 aus, wobei das Programm nur abgebrochen wird.
Der Directory-Eintrag auf der Diskette lautet: "Name...-DAT.BIN. Es ist kein Fehler von mir, es wird ein Komma ausgegeben, obwohl es falsch ist.
Bei meinem zweiten Problem handelt es sich um das Kopierprogramm Transfer. Meiner Meinung nach läuft es einwandfrei und ich konnte auch keinen Tippfehler feststellen. Das Kopieren von Kassette auf Diskette klappt tadellos. Sämtliche Filetypen werden übernommen und auf Diskette geschrieben. Aber die Programme sind nach dem Kopiervorgang nicht mehr lauffähig.
Da ich mir die Programmkassetten zu den Heften schicken lasse, benötige ich von den einzelnen Programmen Sicherheitskopien. Zuerst dachte ich an einen Kopierschutz. Daß das nicht der Fall ist, zeige ich Ihnen an folgendem Beispiel:
Vor einiger Zeit wurde in einer Zeitschrift das Malprogramm CAP abgedruckt. Ich habe es abgetippt und es lief tadellos. Später kaufte ich ein Diskettenlaufwerk und kopierte das Programm auf Diskette. Es ist in BASIC geschrieben und ist nach wie vor lauffähig.
Nun wollte ich Transfer probieren und kopierte CAP auf eine andere Diskette. Transfer gab nach Abschluß der Arbeit die Meldung "File kopiert" aus, aber das Programm läuft nicht. Es wird eine Fehlermeldung in Zeile 10794 ausgegeben, obwohl diese Zeile nicht existiert.
Dieter Hatwich,
5805 Breckerfeld

Die Universaldatei im SPECIAL 1/88 ist tatsächlich schon einmal erschienen, sie wurde jedoch vor dem neuerlichen Abdruck verbessert.

Bei einem Vergleich müßten Sie einige Änderungen finden. Bei Ihnen wird offensichtlich der Name der einzulesenden Datei vom Programm verfälscht, was an der Einlese- oder Abspeicherroutine liegen kann. Bitte kontrollieren Sie noch einmal beide Unterprogramme auf Tippfehler.
Das Programm Transfer fertigt Sicherheitskopien an, die nicht lauffähig sind. Der einfache Grund ist der, daß die Header in den Kassettenblöcken anders aufgebaut sind als die Header einer Diskette. Die Sicherheitskopien sind erst dann wieder lauffähig, wenn die Programme auf den alten Typ des Datenträgers zurückkopiert wurden. Also:
1. das Originalprogramm liegt auf Kassette vor.
2. Sicherheitskopie wird auf Diskette gezogen, sie ist nicht lauffähig.
3. Von der Kopie auf Diskette kann wieder eine Kopie auf eine andere Kassette gezogen werden. Diese ist lauffähig.
Bei unseren Softbox-Programmen lohnt allerdings der Einsatz von Transfer weniger. Die Softbox ist nicht geschützt, jedes Listing kann durch Laden und Abspeichern kopiert werden.



UNIVERSALDATEI AUS SPECIAL 1/88 – BILDSCHIRMMODUS BESTIMMT DIE MASKENGRÖSSE

Das Programm Universaldatei aus SPECIAL 1/88 weist einen entscheidenden Fehler auf. Eigenartigerweise speichert das Programm nur Masken mit zweistelligen Datensätzen ab, aber nicht die von mir gewünschte und benötigte Maske.
Günter Oster,
2208 Glückstadt

Vorab: Einlese- und Abspeicherroutinen funktionieren einwandfrei.
Die Maske, auf die Sie sich beziehen, wird unter dem File XXX.PAR abgelegt. Auf den Bildschirm wird jedoch in Listenform ausgegeben. Dazu stehen Ihnen, da Sie ja im Modus 2 arbeiten, 80 Zeichen zur Verfügung. Wenn Sie jedoch die Felder so dimensionieren, daß Sie insgesamt mehr als 80 Zeichen in einem Datensatz haben, so kann dies nicht gutgehen. Aus diesem Grunde haben Sie ja die Möglichkeit, sich die Zusammenstellung selbst festzulegen. Siehe dazu Seite 91 des Heftes SPECIAL 1/88.

PUZZLE UND SCHIFFE VERSENKEN: VERTAUSCHTE PROGRAMME

Ich bin CPC-Anfänger und habe mir vor kurzem Ihre Zeitschrift gekauft. Auf der Suche nach einem schnellen Erfolgserlebnis stieß ich auf Ihr Computerspiel Schiffe versenken. Ich tippte Puzzle ab und war in Schwierigkeiten. Können Sie mir mitteilen, wie ich das von Ihnen hinzugefügte Listing zur Erstellung einer leeren Highscore-Liste erhalte und wie ich vor Spielbeginn die Anleitung abrufen kann?
Eine weitere Frage: Ich habe Puzzle 2 und 3 noch nicht abgeschrieben, welche Funktion haben diese Listings?
Peter Stoß,
3544 Waldeck

Durch eine kleine Unachtsamkeit wurden leider zwei Listings vertauscht. Die Beschreibung zu Puzzle fehlte, während zum Programm Schiffe versenken das Listing fehlte. Wir bitten alle Leser, diesen Fehler zu entschuldigen. Das Programm wird selbstverständlich bald nachgereicht, hier finden Sie nähere Angaben zum Puzzle:
Das Listing 1 ist lauffähig, also abtippen, speichern und dann starten. Dann das Listing 2 abtippen und speichern. Nun das Listing 1 laden und das abgespeicherte Listing 2 hinzumergen. Anschließend als Puzzle 2 abspeichern, mit Listing 3 verfahren Sie genauso. Sie haben dann drei verschiedene Versionen zur Verfügung. Der Sinn des Spieles besteht darin, die Puzzle-Teile durch Tauschen und Drehen so hinzulegen, daß die an den Grenzen der Einzelteile abgebildeten Symbole zu den Symbolen der anderen Teile passen.

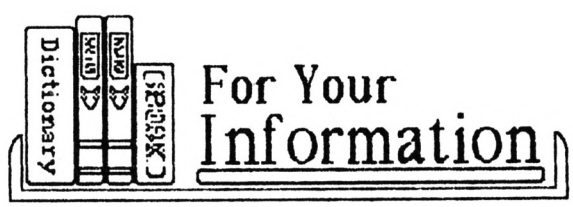
HILFE GESUCHT CP/M PLUS MIT DER DK'TRONIC-ERWEITERUNG

Mit diesem Schreiben möchte ich einen großen Hilferuf an Sie senden. Seit einiger Zeit versuche ich, genaue Herstellerangaben über die Initialisierung von CP/M Plus zu erhalten. Aber entweder kann man in England kein Wort Deutsch lesen oder die Post streikt, denn ich bin immer noch ohne Antwort. Vielleicht kann einer Ihrer Leser bei meinem Problem weiterhelfen:
Ich besitze einen Schneider CPC464 und erwarb vor drei Monaten eine dk'tronics Speichererweiterung mit 64 KByte, um das Programm AMX-Pagemaker benutzen zu können. Laut beiliegenden Handbuch soll es möglich sein, damit ein lauffähiges CP/M Plus zu erzeugen. Doch nach der angegebenen Vorgehensweise ist dies unmöglich. Vielleicht findet sich ein User der dk'tronic-Erweiterung, dem es bereits gelungen ist, das Betriebssystem zu installieren.
M. Virkus,
2392 Glücksburg

Wer helfen kann und will, richtet sein Schreiben an:
M. Virkus
Holnisstr. 17
2392 Glücksburg

NICHT OHNE TÜCKEN: SELBSTBAULAUFWERK MIT CHINON-GERÄTEN

Nach der Bauanleitung vom SPECIAL 4/87 habe ich mir ein 5.25-Zoll-Erstlaufwerk gebaut. Ich habe ein Chinon-Gerät verwendet, aber leider funktioniert es nicht.
Die Anschlüsse habe ich nach den Skizzen durchgeführt, allerdings den Platinenstecker andersherum aufgesteckt, da sonst mein CPC464 verrückt spielt. Ich habe also nur den 34-poligen Platinenstecker am Controller angebracht und die Stromversorgung angeschlossen. Eine eingelegte Diskette wurde zwar in Rotation versetzt, aber der Schreib-/Lese-Kopf nicht bewegt. Statt dessen erhalte ich folgende Fehlermeldung:



Drive A: Disc missing
Habe ich einen Fehler gemacht oder liegt ein Defekt am Laufwerk oder Controller vor?
Bertram Fritz,
6969 Hardheim

Nach Durchsicht Ihres Briefes und Ihrer Skizzen mußten wir leider feststellen, daß Sie sich nicht genau an unsere Anleitung gehalten haben. Die Anbringung des Platinenstekkers sollte so erfolgen, daß der Pin 34 auf Leitung Blau des Controllers liegt. Deshalb mußten Sie auch den Stecker am Floppybus verkehrt herum aufsetzen. Sonst resultieren daraus aber keine Probleme. Disc missing ist eine recht allgemeine Meldung, die nur bedingt Rückschlüsse auf die Fehlerquelle zuläßt. Machen Sie deshalb bitte folgenden Check:
1. Setzen Sie den zweiten Jumper (Kurzschluß) auf RM. Offensichtlich haben Sie ihn entfernt und nur den Jumper DS 0 gesetzt.
2. Bringt das Jumpersetzen nicht den gewünschten Erfolg, dann überprüfen Sie den Platinenstekker auf dem Kabel des Controllers. Ist wirklich jeder Pin mit der richtigen Ader verbunden? Falls Sie den Stecker schon geöffnet haben, sollten Sie ihn auch mit Pin 34 – wie oben angegeben – auf Blau setzen.
3. Wenn Sie dann immer noch keinen Erfolg haben, sollten Sie die Stromversorgung nach folgenden Kriterien durchprüfen:
a) Sind die Spannungen 5 V und 12 V richtig angeschlossen?
b) Hat das Netzteil genügend Leistung?
c) Gibt das Netzteil auch die gewünschten Spannungen an das Laufwerk weiter? Bitte einzeln durchmessen.
4. Als letztes könnten Sie noch das Laufwerk an einem anderen Controller und den Controller an einem anderen Laufwerk ausprobieren. Hierzu müßten Sie wahrscheinlich einen wohlmeinenden Händler aufsuchen.

FREMDE COLOR-MONITORE AM CPC: ERGÄNZUNG EINES LESERS

Wahrscheinlich bin ich nicht der einzige, der Ihnen zum obigen Thema schreibt. Ich darf Ihren ausgezeichneten Artikel vielleicht in einigen Punkten ergänzen:
Dem viel brennenderen Thema CPC am Heimfernseher mit AV-Scartbuchse hätten Sie angesichts des miserablen Bildes, das der MPS 2 über die Antennenbuchse abgibt, mehr Raum geben sollen. Wieso kann das der C64 soviel besser?



Ich benutze als Farbmonitor meinen Grundig-63-370-Fernseher mit der Scartbuchse. Folgende Anschlüsse weichen von Ihrer Darstellung ab:
Bei dem CPC-Ausgang:

Rot Pin	1
Grün	2
Blau	3
Sync	4
Ground	5
Lum	6

Bei Scart-AV-Stecker:

Rot Pin	15
Grün	11
Blau	7
Sync	20
Ground	5/9/13/17
Lum	(nicht angeschlossen)

Schaltspannung 11,4 V ––– 16
Schaltspannung Masse ––– 18
Bei der Audioverbindung bleibt es wie gehabt, Pin 2 + 6, Masse 4. Es empfiehlt sich, die RGB-Signale jeweils separat zu grounden.
Leider haben die Gerätehersteller zwar die Norm-Scartbuchse eingeführt, die Pinbelegung jedoch kann unterschiedlich sein. Es gibt sogar Geräte, bei denen Pin 19 und 20 vertauscht sind. Vom Ergebnis meiner Lötbemühungen kann ich nur ähnlich schwärmen wie Sie von den Ihren beim Fremdmonitor Highscreen.
Siegfried Bende,
2000 Norderstedt

Vielen Dank für den Hinweis. Wir möchten allerdings noch einmal auf die Industrie hinweisen, die mit einigem Erfolg aus der Norm der Scartbuchsen ein Durcheinander erzeugt. Leser sollten die Lötarbeiten nur dann vornehmen, wenn die Datenblätter Auskunft über die Pinbelegung geben.

CHECKER HÄTTE GECHECKT WERDEN SOLLEN – CPC664-BESITZER ENTDECKTE FEHLER

Ich habe erst vor kurzer Zeit mit „Schneidern und Basicen" begonnen; man wird ja dadurch nicht dümmer. Ihren Argumenten gegen den Checker stimme ich unbedingt zu – aaabeeer: Ellenlange Data sind für normal sterbliche User doch fast wie Elfmeterschießen beim Fußball. Mein Hilferuf: Der Checker geht nicht, der Fehler kann außer bei mir nur in Zeile 200, speziell:
POKE&A399,&D4,
&A390,&AA
oder einem Dreckfehler, nein Druckfehler, liegen. Ich bitte um Hilfe.
Auch alle meine Freunde haben einen Schneider CPC, ich einen CPC464, jedoch keiner einen Farbmonitor. Mir ist dies unverständlich, denn ich glaube, daß man doch einiges verpaßt. Siehe die Bilder in Ihrer Zeitschrift. Ich glaube, daß so manche Spiele durch eine Farbdarstellung doppelt so schön werden. Daß ich hinter das Geheimnis von Zorgos gekommen bin, hat mich derart beflügelt, daß ich mir einen CPC664 mit Colormonitor zugelegt habe. Aber was wird aus Schneider ohne Amstrad?
Klaus Andre,
5600 Wuppertal

Mit dem Checksummer haben Sie recht. Der Dreckfehler, pardon: der Druckfehler liegt in der von Ihnen angegebenen Zeile. Hier müssen natürlich ein Doppelpunkt und der nachfolgende POKE-Befehl angegeben werden. Aus der von Ihnen zitierten Stelle wird dann:
POKE &A399,&04:
POKE &A39D,&A...
Diese Stelle war für Sie sehr entscheidend, denn ausgerechnet hier wird überprüft, ob es sich vielleicht um einen 664 handelt. Falls nicht, dann spielt der Fehler weiter keine Rolle.

KOMPRIMIERTER AUSDRUCK BEIM DMP 2000 KARTEIPROGRAMM SPECIAL 4/87

In dem Programm Kartei des Autors Rudolf Bodinger sind für den komprimierten Ausdruck der Daten auf dem Drucker DMP 2000 Änderungen in den Zeilen 3600 und 3610 erforderlich. Zunächst muß nämlich das NLQ-Schriftbild ausgeschaltet werden. Die Zeilen müssen folgenden Inhalt haben:

```
3600 IF p$="k" THEN
PRINT#8,CHR$(27);
CHR$(120);CHR$(0).
PRINT#8,CHR$(15);
:GOTO 3620: 'komprimiert ein
3610 CLS#3.PRINT#8,
CHR$(27);:CHR$(120);
CHR$(1):PRINT#8,
CHR$(18);: 'aus
```

Siegfried Schnitger,
2890 Nordenham

CPC-SIMULATION FÜR FRÄSMASCHINEN

Wie allgemein bekannt, werden im Handwerk sowie in der Industrie immer mehr CNC-Werkzeugmaschinen eingesetzt. Im Interesse der Aus- und Weiterbildung vieler Arbeitnehmer, die CPC-Benutzer sind, wäre ein CNC-Simulationsprogramm für Dreh- und Fräsmaschinen eine echte Hilfe. Es wäre doch eine interessante Arbeit für Ihre Zeitschrift, ein solches Programm in einer Serie zur Verfügung zu stellen. Vielleicht ist Ihnen auch ein Softwarehändler bekannt, der ein solches Simulationsprogramm schon anbietet.
Otto Schomweiler,
6685 Schiffweiler

Tut uns leid, hier muß die CPC-WELT-Redaktion passen. Allein, weil uns der Zugang zu derartigen Maschinen fehlt, können wir kein Simulationsprogramm dafür entwickeln. Unsere diesbezügliche Unkenntnis wollen wir dabei einmal außer acht lassen. Da uns auch kein Softwareverlag bekannt ist, der uns oder Ihnen weiterhelfen könnte, wollen wir an die Leser appellieren. Vielleicht kann uns der eine oder andere mit dem Thema vertraut machen.

SCHLECHTE DRUCKQUALITÄT

Für mich ist SCHNEIDER AKTIV zur Zeit das beste Heft auf dem Markt. Allerdings gibt es oft erhebliche Mängel im Druck. Zum Beispiel in Heft 10/87, Listing „Hero": Zum Teil zu wenig Farbe, so daß die Buchstaben und Ziffern ausgefranst und fast nicht lesbar sind. Das Zeichen = wird oft nur noch für das Zeichen – angesehen. Der Buchstabe O und die Ziffer Null sind im Druck vollkommen identisch, was bestimmt bei vielen Listingfreunden zu erheblichen Bearbeitungsproblemen führt. Oder auf den Seiten 88 bis 96, Listing „Coin": Zuviel Farbe beim Druck, dadurch sind der Buchstabe B und die Ziffer 8, der Buchstabe D und die Ziffer 0 so „zugebatzt", daß es dringend einer Abhilfe bedarf.
Dies waren nur drei Beispiele für einen Fehler, den man vielleicht beseitigen könnte, um allen, speziell den Neulingen, das Eintippen zu erleichtern.
Reinhard Wagner,
8900 Augsburg

Wir sind an dieser Stelle bereits öfter auf das oben genannte Problem eingegangen, wollen dies jedoch noch einmal tun. Regelmäßige Leser der CPC-WELT haben vielleicht bemerkt, daß wir bei den Listings einige Experimente angestellt haben, zum Teil mit nicht zufriedenstellenden Ergebnissen. Wir glauben jedoch, mit dem jetzigen Verfahren eine Methode gefunden zu haben, mit der die Listings in der bestmöglichen Qualität abgedruckt werden können.

Jeden Montag
16-19 Uhr
Redaktions-
Hotline!
Tel 089/18 40 24
Sie fragen -
wir antworten

FUNKTIONSTASTEN-BELEGUNG UNTER CP/M PLUS

Gibt es eine Möglichkeit, unter CP/M Plus die Funktionstasten mit Strings zu belegen?
Olaf Schröder,
2251 Drelsdorf

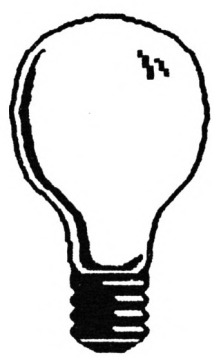
Es gibt diese Möglichkeit, nur wird sie im Handbuch zu CP/M Plus nicht beschrieben.
Zunächst muß eine Textdatei angelegt werden. Sie können hierzu ein beliebiges Textprogramm verwenden, das ASCII-Dateien erzeugt, oder den CP/M-Editor ED. Diese Textdatei hat folgenden Aufbau:

E xxx "sss"

Jede Zeile des Textes beginnt mit einem E und einer Leerstelle. Für xxx setzen Sie die Erweiterungsnummer der gewünschten Taste ein. Sie finden diese Nummern im Handbuch zum CPC 6128 in Kapitel 7, Seite 23 unten. Für f0 ist die Nummer 128, für f1 129, und so weiter. Danach folgt wieder eine Leerstelle. Zwischen den beiden Anführungszeichen können Sie den String eintragen, den Sie einer Taste zuweisen wollen. CTRL-Tasten werden dabei durch ^ dargestellt, also ^J für CTRL und J. Schließen Sie jede dieser Zeilen mit RETURN ab und speichern Sie anschließend die Datei beispielsweise unter dem Namen ERWEITER.KEY. Nachdem dies geschehen ist, können Sie mit dem CP/M-Dienstprogramm SETKEYS die Erweiterung einrichten (SETKEYS ERWEITER.KEY). Dieses Programm weist den Funktionstasten die in der Erweiterungsdatei eingetragenen Strings zu.

MODELLEISENBAHNER BENÖTIGT HILFE

Ich habe ein Problem: Seit etwa einem halben Jahr besitze ich eine Schneider-Schnittstelle RS232, die ich mit dem CPC6128 zum Steuern einer Märklin-Eisenbahn gebrauche.
Nach dem Eintippen eines Schnittstellen-Programmes erscheint jedoch immer eine Fehlermeldung auf dem Monitor. Ich war bei verschiedenen Händlern, doch keiner konnte mir helfen. Gibt es einen Kollegen, der mir schreiben aknn, wie ich die Schnittstelle zu initialisieren habe?
Robert Schulze
Schützenstraße 2
5900 Siegen □



DEM DIRECTORY AUF DER SPUR

Knacken Sie den Safe

Das Directory ist der Schlüssel zum Inhalt einer Diskette. In diesem Inhaltsverzeichnis finden Sie alle Files, die Sie gespeichert haben. Leider ermöglicht das BASIC der CPCs keinen direkten Zugriff auf diese wertvollen Informationen; der Schlüssel zur Disk ist zusätzlich in einem Safe eingesperrt. Dieser Beitrag zeigt Ihnen, wie sich der Safe knacken läßt.

Unter BASIC verwaltet Ihr Schneider-Computer die Diskette nach einem eigenen Betriebssystem, genannt AMSDOS (Amstrad Disc Operating System). Das AMSDOS-Format ist identisch mit dem CP/M-Format, so daß beide Betriebssysteme dieselben Disketten benutzen können. Die Befehle zur Anzeige eines Inhaltsverzeichnisses sind jedoch bereits unterschiedlich. Während Sie unter CP/M das Verzeichnis mit DIR erhalten, lautet der entsprechende BASIC-Befehl CAT.
Allein aus den unterschiedlichen Befehlen läßt sich jedoch nicht auf unterschiedliche Funktionsweisen schließen. Offensichtlich wird dies erst, wenn Sie das Inhaltsverzeichnis derselben Diskette einmal unter BASIC und einmal unter CP/M ausgeben lassen. CP/M zeigt Ihnen die Dateinamen unsortiert an; BASIC sortiert sie und gibt Ihnen gleichzeitig Auskunft über den noch verfügbaren Speicherplatz auf der Diskette.
Natürlich können Sie in einem BASIC-Programm das Directory jederzeit auf den Bildschirm bringen. Der CAT-Befehl funktioniert schließlich immer. Aber was machen Sie, wenn der zur Verfügung stehende Platz in einem Fenster nicht mehr ausreicht oder wenn Sie nur bestimmte Dateinamen anzeigen wollen? Hier muß BASIC passen.
Wenn Sie genau aufpassen, so werden Sie feststellen, daß der CPC zuerst auf die Diskette zugreift und erst dann das Inhaltsverzeichnis ausgibt. Würde er jeden Eintrag lesen und direkt auf den Bildschirm bringen, so müßte dies gleichzeitig geschehen. Die Folgerung liegt nahe, daß das vollständige Directory zuerst in den Speicher geladen und erst anschließend ausgegeben wird.

EINSATZ VON FIRMWARE-ROUTINEN

Diese Vermutung ist richtig, wie ein Blick ins Firmware-Handbuch beweist. AMSDOS verwendet dieselben Einsprung-Adressen wie die Kassettenverwaltung; die entsprechenden Routinen sehen freilich anders aus. Das muß Sie jedoch nicht interessieren, wenn Sie Firmware-Routinen anwenden. Sie werden nur über die Einsprungvektoren angesteuert.
Einen solchen Vektor gibt es auch für eine Routine namens CAS CATALOG. Sie ist zuständig für die Ausführung eines CAT-Befehls. Da hierzu die Adresse eines 2 KByte großen Pufferbereichs benötigt wird, die im DE-Register des Z80-Prozessors übergeben wird, kommen Sie nicht umhin, den Aufruf in Assember durchzuführen.
Das bietet Ihnen jedoch gleichzeitig die Möglichkeit, dem CPC mitzuteilen, wo er den Inhalt des Directorys ablegen soll. Sobald Sie das wissen, ist es kein Problem mehr, mit dem PEEK-Befehl auf den entsprechenden Speicherbereich zuzugreifen und das Inhaltsverzeichnis auszulesen.

```
1 'DIRSORT.BAS
2 '
3 'Demo-Hauptprogramm fuer
4 'Directory-Routinen
5 '
100 WINDOW #1,1,1,1,1:PEN #1,0:WINDOW SW
AP 0,1
110 GOSUB 10000
120 WINDOW SWAP 1,0
130 'Sortiere nach Groesse
140 sortiert=1
150 FOR i=1 TO gesamt-1
160   IF laenge(i)<= laenge(i+1) THEN 21
0
170   hilfsname$=dir$(i):hilfslaenge=lae
nge(i)
180   dir$(i)=dir$(i+1):laenge(i)=laenge
(i+1)
190   dir$(i+1)=hilfsname$:laenge(i+1)=h
ilfslaenge
200   sortiert=0
210 NEXT i
220 IF sortiert=0 THEN 140
230 'Und jetzt ausgeben
240 FOR i=1 TO gesamt
250   PRINT"Datei: ";dir$(i);"   Laenge:
 ";laenge(i);" K"
260 NEXT i
270 END
```

DIRLOAD.BAS

Das Programm DIRLOAD.BAS enthält das kleine Assembler-Programm, versteckt in den POKE-Befehlen. Der MEMORY-Befehl zu Programmbeginn setzt die obere Speichergrenze herab. Ab Adresse &A000 beginnt ein freier Speicherbereich, der gut für die Ablage von Maschinenprogrammen geeignet ist. Ab &9800, also genau zwei KByte darunter, soll das Inhaltsverzeichnis abgelegt werden. Deshalb wird die oberste zulässige Adresse noch ein Byte weiter nach unten gesetzt, was in hexadezimaler Schreibweise die Zahl &97FF ergibt. Für Ihre BASIC-Programme bleiben dann genau 38541 Byte übrig. Das reicht für die meisten Anwendungen aus.
POKE-Anweisungen haben für viele etwas Geheimnisvolles an sich. Da wird etwas in den Speicher geschrieben, was dann eine bestimmte Funktion erfüllen soll. Das dahinterstehende Assembler-Programm ist jedoch oft recht einfach, so wie in diesem Fall. In Assembler lautet der Text:

```
org &A000
LD DE,&9800
CALL &BC9B
RET
```

Selbst wenn Sie Assembler nicht beherrschen, dürfte es Ihnen nicht schwerfallen, dieses Programm zu verstehen. Die erste Zeile bewirkt nichts anderes, als daß das Programm ab der angegebenen Adresse im Speicher

abgelegt wird. Der Org-Befehl gehört nicht zum Sprachumfang der Maschinensprache, sondern stellt einen Pseudo-Befehl für das Übersetzungsprogramm dar. Er wirkt also nicht anders als die erste Zahl eines POKEs.
Die Abkürzung LD bedeutet "load", also „lade".
Hier wird der Wert &9800 in das DE-Register des Prozessors geladen. Der Z80 besitzt mehrere dieser Register, die entsprechende BASIC-Anweisung.
Die Zahlen hinter den POKE-Befehlen stellen die Umsetzung dieser Assembler-Befehle dar. Da der Prozessor nur mit Zahlen rechnen kann, müssen Sie ihm die Anweisungen in dieser Form darbieten.
Wenn Sie den BASIC-Lader starten, wird Ihre neue CAT-Routine in den Speicher geladen. Durch den MEMORY-Befehl den Inhalt dieser Byte. Jedes druckbare Zeichen wurde dargestellt, jedes nicht druckbare durch sein hexadezimales Äquivalent ersetzt. Als Platzgründen können wir natürlich nur den Anfang des Directorys zeigen.
Die Systematik, die hinter den Einträgen steckt, ist nicht schwer zu erkennen. Jeder Dateiname beginnt mit dem Byte-Wert 255 beziehungsweise &FF. Danach folgt der Dateiden Dateinamen automatisch mit Leerzeichen auf, falls er kürzer als acht Zeichen ist.
Damit stehen Dateinamen und Extension immer an den selben Stellen und können voneinander unterschieden werden. Der trennende Punkt ist dazu nicht mehr nötig. Bei 64 möglichen Einträgen spart man insgesamt 64 Byte Platz ein. Das ist nicht viel, aber besser als nichts. Die Bedeutung des Byte

```
&FF  B  S  P  1                          B  A  S &01 &00
&FF  B  S  P  2                          B  A  S &01 &00
&FF  B  S  P  3                          B  A  S &01 &00
&FF  B  S  P  4                          B  A  S &01 &00
&FF  B  S  P  5                          B  A  S &01 &00
&FF  B  S  P  6                          B  A  S &01 &00
&FF  D  I  R  L  O  A  D                 B  A  S &01 &00
&FF  D  I  R  P  R  I  N  T              B  A  S &02 &00
&FF  D  I  R  R  O  U  T                 A  S  M &01 &00
&FF  D  I  R  R  O  U  T                 B  A  S &01 &00
&FF  D  I  R  P  O  U  T                 M  R  G &01 &00
&FF  D  I  R  S  O  R  T                 B  A  S &01 &00
&00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00
&00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00
&00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00
&00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00
&00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00
&00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00 &00
```

```
100 'DIRLOAD.BAS
110 '
120 'BASIC-Lader fuer
130 'Directory-Routine
140 '
150 MEMORY &97FF
160 POKE &A000,&11
170 POKE &A001,0
180 POKE &A002,&98
190 POKE &A003,&CD
200 POKE &A004,&9B
210 POKE &A005,&BC
220 POKE &A006,&C9
```

Oben: Das notwendige BASIC-Programm
Links: Ein Auszug aus dem Speicher

man – vereinfacht gesagt – mit Variablen in BASIC gleichsetzen kann. Der Wert &9800 ist nichts anderes als die Startadresse des Zwei-KByte-Puffers, den Sie für das Directory benötigen.
CALL ruft, wie in BASIC, eine bestimmte Routine auf, indem es die Programmausführung an der angegebenen Speicherstelle fortsetzt. In diesem Fall ist das die Adresse &BC9B. Dort befindet sich die oben erwähnte Routine CAS CATALOG. Sie legt das Inhaltsverzeichnis einer Diskette in dem durch DE bestimmten Speicherbereich ab und bringt es anschließend auf den Bildschirm.
Der Befehl RET bedeutet RETURN und erfüllt diselbe Funktion wie die

werden die entsprechenden Speicherbereiche geschützt. Auch ein NEW kann ihnen nichts anhaben. Erst durch Rücksetzen des Computers erhalten Sie wieder Zugriff auf diese Speicherbereiche.

BILDSCHIRM UND SPEICHER: GLEICHER INHALT, ANDERES AUSSEHEN

So einfach, wie es vielleicht scheinen mag, ist der Zugriff auf das Directory jedoch nicht. Wenn Sie mit CALL &A000 die neue Routine aufrufen, erscheint das Verzeichnis in gewohnter Form auf dem Bildschirm. Es steht anschließend aber auch in dem Speicherbereich ab &9800.
In *Abbildung 1* sehen Sie

name in Klartext. Auffällig ist dabei nur, daß der gewohnte Punkt vor den letzten drei Zeichen fehlt. Anschließend steht immer ein Zahlenwert, der allerdings wechseln kann. Das letzte Byte eines Eintrags ist immer Null.
Warum fehlt hier der Punkt im Dateinamen? AMSDOS und CP/M erkennen Bezeichnungen mit maximal acht Zeichen an. Die letzten drei, durch Punkt abgetrennten Zeichen dienen der Kennzeichnung der Datei. BASIC-Files erhalten zum Beispiel automatisch ein "BAS" angehängt. Dieses Anhängsel bezeichnet man als Erweiterung oder Extension. Wie in der Abbildung ebenfalls erkennbar wird, füllt AMSDOS

hinter dem Dateinamen ist nicht schwer zu erraten. Mit CAT erfahren Sie auch die Größe der jeweiligen Datei, und irgendwo muß sich diese Information ja befinden. Wo stünde sie sonst, wenn nicht hier?
Abbildung 2 faßt den Aufbau eines Directory-Eintrages zusammen. Jeder Eintrag besteht aus 14 Byte.
Diese Länge scheint nicht so recht zur maximalen Eintragszahl zu passen. Diese beträgt, wie oben erwähnt, 64. Insgesamt würden also nur 896 Byte benötigt. Weshalb aber werden gleich zwei KByte reserviert, obwohl hier Speicherplatz verschwendet wird?
Eine Erklärung ist, daß BASIC standardmäßig

den selben Speicherbereich verwendet, der auch den Ein-/Ausgabepuffer für Textdateien enthält. Dieser Puffer umfaßt ebenfalls zwei KByte. Andererseits muß auch berücksichtigt werden, daß das Directory tatsächlich diesen Umfang hat. Pro Eintrag sind nämlich 32 Byte vorgesehen und werden auch von CP/M verwendet. AMSDOS unterschlägt allerdings einige Informationen, die für den CAT-Befehl nicht benötigt werden.

DIRROUT.BAS

Nachdem der Aufbau des Inhaltsverzeichnisses bekannt ist, ist es nicht mehr schwierig, die entsprechenden Speicherbereiche auszulesen und in BASIC-Variablen zu übertragen. Das Unterprogramm DIRROUT.BAS zeigt eine Möglichkeit dazu auf.
Zeile 10 ist für ein späteres Hauptprogramm vorgesehen. Hier wird Platz für 64 Strings und 64 Zahlen geschaffen – die Namen und Längen der einzelnen Dateien. Die Variable "base" gibt die Startadresse des Directory-Puffers an.
Ab Zeile 10000 beginnt das eigentliche Unterprogramm. Die Variable "curr" (für "current address", also „derzeitige Adresse") ist nichts anderes als ein Zähler, der mit dem Wert von "base" initialisiert wird. In der Variablen „Gesamt" ist nach Abschluß der Routine die Anzahl der vorhandenen Einträge enthalten. Sie muß vorab mit Null belegt werden, da ihr Wert sich nur durch ein Hochzählen ändert, das mit jedem gefundenen Eintrag erfolgt. Und um überhaupt ein Directory lesen zu können, müssen Sie die Assembler-Routine bei &A000 aufrufen, die Sie mit DIRLOAD installiert haben.
Jetzt befindet sich das Directory der gerade eingelegten Diskette im Speicher, und die Übertragung kann beginnen. Zeile 10010 überprüft, ob bereits in der ersten Speicherstelle eine Null steht. Ist dies der Fall, kann das Unterprogramm verlassen werden; die Diskette ist nämlich leer. Wurde ein anderer Wert als Null gefunden, kann der Eintragszähler bereits um Eins erhöht werden. Dieser erste Dateiname wird in den folgenden Programmzeilen in eine Stringvariable des Arrays "dir$" übertragen, wobei auch der Punkt vor der Extension hinzugefügt wird. Zeile 10100 liest schließlich die Länge der Datei und speichert den erhaltenen Wert in der entsprechenden Variable des Feldes „laenge". Zeile 10110 hat die Aufgabe, zu überprüfen, ob der letzte Eintrag erreicht ist. Nur wenn an 14. Stelle des Eintrages der Wert &FF steht, befinden sich noch weitere Files auf der Diskette. In diesem Fall wird der Zähler "curr" um 14 erhöht und zeigt damit auf den nächsten Eintrag. Die Prozedur beginnt dann von vorne. Wurde ein anderer Wert gefunden, hat das Unterprogramm seine Aufgabe erfüllt.
Nach dem Aufruf dieser Subroutine befinden sich die Namen und Längen aller Dateien auf der Diskette in den Arrays "dir$" und „laenge"; die Variable „Gesamt" enthält die Anzahl der Dateien auf der Diskette. Mit diesen Variablen können Sie anfangen, was Sie wollen. Ein Beispiel ist das Programm DIRSORT.BAS. Es sortiert die Dateien nicht alphabetisch, sondern nach ihrer Größe.

```
1 'DIRROUT.BAS
2 '
3 'Unterprogramm zum Laden
4 'eines Directorys
5 '
10 DIM dir$(64),laenge(64):base=&9800
10000 curr=base:gesamt=0:CALL &A000
10010 IF PEEK(base)=0 THEN 10120
10020 gesamt=gesamt+1
10030 FOR ii=1 TO 8
10040    dir$(gesamt)=dir$(gesamt)+CHR$(P
EEK(curr+ii))
10050 NEXT ii
10060 dir$(gesamt)=dir$(gesamt)+"."
10070 FOR ii=9 TO 11
10080    dir$(gesamt)=dir$(gesamt)+CHR$(P
EEK(curr+ii))
10090 NEXT ii
10100 laenge(gesamt)=PEEK(curr+12)
10110 IF PEEK(curr+14)=&FF THEN curr=cur
r+14:GOTO 10020
10120 RETURN
```

Byte	Inhalt
0	&FF, wenn Eintrag vorhanden Null, wenn kein Eintrag vorhanden
1 2 3 4 5 6 7 8	Name der ersten Datei; wenn nötig, mit Leerzeichen aufgefüllt
9 10 11	Extension der ersten Datei; wenn nötig, mit Leerzeichen aufgefüllt
12	Länge der Datei in KByte
13	immer Null
14	&FF, wenn weiterer Eintrag vorhanden Null, wenn kein weiterer Eintrag

DIRSORT.BAS

Um DIRSORT anwenden zu können, muß das Maschinenprogramm installiert sein. Außerdem müssen Sie das Unterprogramm und die DIM-Anweisung für die Arrays

hinzuMERGEn. Da der MERGE-Befehl unter AMSDOS nicht so funktioniert, wie er eigentlich sollte, speichern Sie die Subroutine DIRROUT.BAS am besten noch einmal als ASCII-Datei. Dazu ergänzen Sie den SAVE-Befehl einfach um die beiden Zeichen ",A". Wenn Sie das Hauptprogramm auf diese Weise vervollständigt haben, lassen Sie es laufen. Sie sehen, daß das Directory hier nicht auf dem Bildschirm angezeigt wird. Wenn vorher jedoch in der oberen linken Ecke ein Zeichen gestanden hat, so ist dieses nun verschwunden.
Hier handelt es sich um einen einfachen Trick, die oft lästige Anzeige des Inhaltsverzeichnisses zu unterdrücken. Vereinbaren Sie einfach ein Fenster an irgendeiner Bildschirmposition, an der es niemanden stört. Danach setzen Sie die Schreibfarbe für dieses Fenster gleich der Hintergrundfarbe.
Mit einem WINDOW-SWAP-Befehl vertauschen Sie Stream 0, auf den normalerweise die Directory-Ausgabe erfolgt, mit der Stream-Nummer des neu definierten Fensters. Erst dann rufen Sie das Unterprogramm mit GOSUB 10000 auf. Sie werden sehen, daß Sie nichts sehen. Vergessen Sie aber nicht, nach dem Einlesen des Directorys wieder auf das aktuelle Fenster zurückschalten.
Weitere Anwendungen für dieses Unterprogramm liegen nahe. Sie können es zum Beispiel gewinnbringend in einer Dateiverwaltung verwenden, die nur Dateien anzeigt, die mit ihr angelegt wurden. Dazu müssen Sie diese Dateien nur mit einer besonderen Extension versehen. Leichter wird es Ihnen vielleicht fallen, das hier vorgestellte Sortierprogramm um einige Wahlmöglichkeiten zu bereichern. Man könnte die Files wieder alphabetisch sortieren oder anhand ihrer Extension. Und schließlich wäre auch die Realisation einer Benutzeroberfläche à la GEM möglich. *AE*□

Haben
Sie Fragen.
Jeden Montag
16-19 Uhr
Redaktions-
Hotline!
Tel 089/18 40 24
Sie fragen -
wir antworten

GRAFIK-KURS

Einfach und effektvoll

Die Grafikfähigkeiten des Schneider CPC sind hervorragend. Sie müssen nur verstehen, sie richtig einzusetzen. So können Sie bereits mit den grundlegenden Zeichenbefehlen des CPC eindrucksvolle Effekte hervorrufen.



Abbildung 1

Bei den Grafikbefehlen haben die Autoren des Locomotive BASIC besondere Sorgfalt walten lassen. Das zeigt sich nicht nur im Umfang des vorhandenen Befehlssatzes, sondern auch in der Tatsache, daß die großen Brüder des CPC 464 einige Ergänzungen aufzuweisen haben. Es gibt jedoch fast nichts, was Sie nicht auf dem 464 ohne Probleme realisieren könnten.
Die einzige Ausnahme stellt der Fill-Befehl zum Ausfüllen von Flächen dar. Da alle Programmbeispiele auf sämtlichen Rechnern der CPC-Familie laufen sollen, wollen wir darauf verzichten, auf diesen Befehl näher einzugehen.

DIE EINFACHSTEN BEFEHLE

Solch komplizierte Grafik-Anweisungen sind für viele Zwecke nicht einmal notwendig. Das erkennen Sie an *Listing 1*, einem kleinen Spiel für die CPCs. Das Programm verwendet lediglich den Befehl PLOT und die Funktion TEST.
Ihre Aufgabe bei diesem Spiel ist es, die eigene orange Linie mit Hilfe der Cursortasten über den Bildschirm zu steuern. Dabei dürfen Sie einige Dinge nicht berühren. Wenn Sie in die eigene Linie laufen, ist das Spiel ebenso zu Ende, als würden Sie eine der grünen Linien berühren, die der Computer auf den Bildschirm malt. Auch eine Kollision mit dem Rand bricht das Programm ab. Die Berechnung der erreichten Punktzahl und einige andere Verwaltungsaufgaben sollen uns nicht interessieren. Betrachten wir aber einmal, wie der Rechner das Ziehen der Linien durchführt: Ab Zeile 210 wird der neue Zug des Spielers berechnet. Die Variablen xs und ys enthalten die aktuelle Position. Diese Werte werden als x und y an ein Unterprogramm ab Zeile 340 übergeben. Dort wird die neue Position entsprechend der ge-

```
100 'Linienspiel                          (122F)
110 'mit Zufallswerten                    (18DA)
120 '                                     (07FB)
130 'Bildschirm vorbereiten               (1DB8)
140 INK 0,0:PAPER 0                       (0CA5)
150 INK 1,15:INK 2,21:INK 3,11            (16C3)
160 BORDER 26:MODE 1:PEN 3                (0E4F)
170 'Startwerte                           (11AB)
180 xs=320:ys=200                         (178A)
190 xc=10:yc=10                           (161D)
200 score=0                               (0FC4)
210 'Spielerzug                           (11DE)
220 x=xs:y=ys:GOSUB 340                   (1F5F)
230 xs=x:ys=y                             (1A35)
240 IF t$=""THEN 280'Wenn keine Ta
ste gedrueckt, Computerzug ausf.          (40BB)
250 score=score+1:LOCATE 1,1:PRINT
 score                                    (278D)
260 IF TEST(xs,ys)=2 OR TEST(xs,ys
)=1 THEN PEN 3:LOCATE 1,1:PRINT" G
 A M E   O V E R    Score: ";score
:LOCATE 1,5:END                           (62A1)
270 PLOT xs,ys,1                          (138F)
280 'Computerzug                          (12DC)
290 x=xc:y=yc:GOSUB 420                   (1F36)
300 xc=x:yc=y                             (1A79)
310 IF xc=xs AND yc=ys THEN PEN 3:
LOCATE 1,1:PRINT" G A M E   O V E
R    Score: ";score:LOCATE 1,5:END
                                          (56E3)
320 PLOT xc,yc,2                          (13B6)
330 GOTO 210                              (091A)
340 'Tastaturabfrage und                  (1AC8)
350 'Bestimmung Spielerpos.               (1DD9)
360 t$=INKEY$:IF t$=""THEN RETURN         (1737)
370 IF t$=CHR$(240)THEN y=y+2:IF y
>398 THEN y=398'Pfeil nach oben           (41C4)
380 IF t$=CHR$(241)THEN y=y-2:IF y
<0 THEN y=0'Pfeil nach unten              (3E92)
390 IF t$=CHR$(242)THEN x=x-2:IF x
<0 THEN x=0'Pfeil nach links              (3EF6)
400 IF t$=CHR$(243)THEN x=x+2:IF x
>638 THEN x=638'Pfeil nach rechts         (433D)
410 RETURN                                (065B)
420 'Zufallswert-Bestimmung               (1D97)
430 'und Best. Computerpos.               (1DF3)
440 RANDOMIZE TIME                        (08AD)
450 xce=500+INT(100*RND)'Ende der
Linie                                     (295D)
460 x=x+2:IF x<xce THEN RETURN            (1F8C)
470 PRINT CHR$(7);:x=INT(100*RND):
y=INT(400*RND)'Neuer Start                (38A4)
480 RETURN                                (06E7)
```

drückten Pfeiltaste ermittelt. In der Variablen t$ erhalten Sie außerdem den Wert dieser Taste.
Zeile 230 hat die Aufgabe, den Variablen des Hauptprogramms die neuen Werte aus dem Unterprogramm mitzuteilen. Diese Vorgehensweise bezeichnet man als Parameterübergabe. Sie ist besonders beim Testen von Programmen von Vorteil, da Sie die im Unterprogramm ermittelten Werte von denen des Hauptprogramms unterscheiden können. Wenn ein Fehler auftritt, kann er somit genauer lokalisiert werden.
Zeile 260 überprüft, welche Farbe sich an der neuen Position befindet. Dafür ist die BASIC-Funktion TEST(x,y) zuständig. Ihr Ergebnis ist der Pen, der an der Stelle (x;y) zum Zeichen verwendet wurde. Beachten Sie, daß Sie nur die Nummer eines Farbstiftes erhalten, nicht aber die Farbe, die ihm zugeteilt wurde. So können Sie nur sagen, daß etwa Stift Nummer drei verwendet wurde, nicht aber, daß der Rechner einen roten Punkt gefunden hat.

TESTEN MIT TEST

So gerne Sie manchmal die tatsächliche Farbe erfahren möchten: Die Funktionsweise von TEST ist gut durchdacht. Sie können nämlich die dargestellten Farben jederzeit durch eine INK-Anweisung ändern. Da Sie nur den PEN-Wert erhalten, ist die tatsächliche Farbe nicht von Interesse. Das erspart es Ihnen, im ganzen Programm nach Werten zu suchen, die Sie neu angeben müssen. Meistens würde dabei doch eine Zeile vergessen, und die Suche nähme einige Zeit in Anspruch.
Der Computer zeichnet seine Linien mit Pen 2, Sie malen mit Pen 1. Wenn Sie eine der beiden Linien berühren, ist das Spiel zu Ende. Dies wird in Zeile 260 kontrolliert. Anschließend setzt Zeile 270 einen Punkt in Farbe 1 an die neue Stelle. Dies darf erst jetzt geschehen, da sonst die TEST-Funktion immer nur den Wert 1 liefern würde – nämlich die Farbe des letzten Punktes Ihrer Linie. Ist der neue Punkt gesetzt, kann der Computer seinen Zug ausführen.
Sie werden jetzt einwenden, daß die dritte Bedingung noch nicht überprüft wurde: Der Spieler darf auch den Bildschirmrand nicht berühren; so haben wir dies in unseren Regeln vereinbart.
Diese Kontrolle findet statt, indem Haupt- und Unterprogramm zusammenarbeiten. In t$ erhalten Sie den Wert der zur Steuerung verwendeten Taste. Haben Sie keine Taste gedrückt, ist dieser String leer. Wenn Sie ferner versuchen, über den Rand hinauszufahren, wird der neue Wert im Unterprogramm sofort wieder zurückgesetzt. Sie finden diese Operationen in den Zeilen 370 bis 400.

GRAFIK-SPIELEREIEN

Wenn Sie also den Rand berührt haben, stehen Sie wieder am Endpunkt Ihrer Linie. Dort aber ist bereits ein Punkt gesetzt. Zeile 260 stoppt daher das Spiel. Dies ist nur dann der Fall, wenn Sie zuvor eine Taste betätigt haben. Die Programmzeile 240 läßt den Rechner sofort beim Computerzug weiterarbeiten, wenn Sie stehengeblieben sind.
Der Zug des Rechners läuft etwas anders ab als der des Spielers. Es werden nur waagrechte Linien von links nach rechts eingezeichnet. Der Endpunkt einer solchen Linie wird bei jedem Aufruf des Unterprogramms neu bestimmt. Durch den Zufallsfaktor in Zeile 450 befindet er sich irgendwo zwischen 500 und 600. Wurde dieser Endpunkt überschritten, so bestimmt Zeile 470 einen neuen Start.
Das Zeichnen des Punktes geschieht wieder im Hauptprogramm. Zeile 320 setzt einen Punkt in der PEN-Farbe 2. Zuvor überprüft Zeile 310, ob die neuen Koordinaten des Rechners mit dem

Endpunkt der Spielerlinie übereinstimmen. In diesem Fall ist das Spiel ebenfalls beendet.
Der Rechner darf die Linie des Spielers an einer beliebigen Stelle kreuzen. Mit Hilfe der TEST-Funktion können Sie das Programm so ändern, daß die grüne Linie sofort stoppt, wenn sie auf die orange trifft. Eine entsprechende Abfrage könnte im Unterprogramm für den Computerzug oder vor dem Setzen des neuen Punktes stattfinden.

GRAFIKMODI – NÄHER BETRACHTET

Wenn Sie sich die Routinen für die Berechnung der neuen Positionen genauer ansehen, wird Ihnen auffallen, daß die x- und y-Werte jeweils um einen Betrag von zwei geändert werden. Dennoch sehen Sie auf dem Bildschirm durchgezogene Linien. Obwohl die Annahme naheliegt, der Punkt (0;0) und der Punkt (1;1) seien verschieden, scheint der CPC es hier nicht allzu genau zu nehmen.
Der Grund dafür liegt in der Beschränkung des Bildschirmspeichers auf 16 KByte. In Modus 2 stehen Ihnen nur zwei Farben zur Verfügung. Da ein Bit die Werte Null und Eins annehmen kann, genügt dies zur Unterscheidung. In einem Byte kann sich der Rechner bereits acht Bildpunkte merken. Die verfügbaren 16 KByte stellen 16 mal 1024 Byte dar, was Platz für 131.056 Bildpunkte bedeuten würde. Auch dieser Wert ist noch nicht ausreichend, wie eine kurze Rechnung ergibt. Multiplizieren Sie 640 mit 400, und Sie erhalten den Wert 256.000. Da die 16 KByte nicht ausschließlich für Bildpunkte zur Verfügung stehen, hat man bei Schneider einen Kompromiß geschlossen. Es werden nur 128.000 Punkte gespeichert. Damit ergibt sich eine tatsächliche Auflösung von 640 mal 200 Punkten in Mode 2. In y-Richtung kann nur jeder zweite Punkt optisch unterschieden werden, während in x-Richtung tatsächlich einzelne Punkte dargestellt werden.
Was aber bezweckten die Autoren des BASIC mit diesem Verfahren? Sollte der Anwender durch diese Werte an eine höhere Auflösung glauben, als sie der CPC tatsächlich aufweist? Das ist mit Sicherheit nicht der Fall, da der CPC auch mit 640 mal 200 Punkten nicht schlecht abschneidet. Immerhin besitzt er damit das gleiche grafische Auflösungsvermögen wie ein IBM-Rechner mit der Standard-Grafikkarte CGA.
Der Grund ist vielmehr in einer Vereinfachung von Rechenoperationen zu sehen. Das Verhältnis Länge zu Breite des Schneider-Monitors entspricht etwa dem Wert 640 zu 400. Wenn Sie einen Kreis zeichnen wollen, können Sie in x- und y-Richtung denselben Wert für den Radius verwenden, um den Kreis wirklich rund darzustellen. Könnten Sie nur 200 y-Punkte ansteuern, so müßten Sie dort den Radius halbieren. Um einen Kreis auf den Bildschirm zu bringen, würden Sie in Wirklichkeit eine Ellipse berechnen.
Damit ist die maximale Auflösung in nur 200 mal 640 Punkte erklärt. Weshalb aber nimmt die Anzahl der verfügbaren Punkte in den niedrigeren Modi ab?

BINÄRDARSTELLUNG IST GANZ EINFACH

Diese Grafik-Modi stellen Ihnen eine höhere Anzahl von Farben zur Verfügung. Während Sie in Mode 2 nur zwei Farben darstellen können, sind es in Mode 1 bereits vier. Zwei Farben können durch Null und Eins unterschieden werden. Vier Farbwerte hingegen benötigen bereits zwei Bit: Mit zweimal zwei möglichen Zuständen können Sie vier unterschiedliche Aussagen machen. Die Werte 00, 01, 10 und 11 stehen im Dualsystem für die Zahlen Null bis Drei. Das ist das Geheimnis des ersten Parameters beim INK-Befehl. Dem Rechner wird dort mitgeteilt, welcher Bit-Kombination er welche Farbe zuteilen soll. Mit dem Befehl PEN und dem dritten Parameter bei PLOT oder DRAW geben Sie nur noch diese Bits an, nicht aber die tatsächliche Farbe.
Es dürfte jetzt nicht mehr schwerfallen, die Anzahl der darstellbaren Bildpunkte in Modus 0 zu errechnen. Da Sie hier 16 verschiedene Farben darstellen können, werden pro Bildpunkt vier Bit benötigt. Damit reduzieren sich die Bildpunkte auf der x-Achse nochmals um die Hälfte auf einen Wert von 160. Insgesamt sind hier noch 32.000 Punkte darstellbar.
Bei der Arbeit mit dem CPC haben Sie den Vorteil, daß der Bildschirm so angesprochen werden kann, als würden Sie in

```
100 'Zeichnen eines                          <159B>
110 'n-Ecks                                  <0DD5>
120 '                                        <07FB>
130 'Farben einstellen                       <18A0>
140 INK 0,0:PAPER 0:BORDER 0                 <0FD1>
150 INK 1,6                                  <0922>
160 MODE 1                                   <07E8>
170 'Startwerte                              <11AB>
180 INPUT"Wieviele Ecken";ecken              <1F18>
190 xz=320:yz=200                            <17AA>
200 r=150                                    <0CE8>
210 winkel=0                                 <10D8>
220 schritt=(2*PI)/ecken                     <1F4D>
230 'Zeichnen                                <0F3E>
240 MOVE xz+r,yz                             <16B0>
250 WHILE winkel<2*PI                        <1498>
260 winkel=winkel+schritt                    <2312>
270 DRAW xz+r*COS(winkel),yz+r*SIN
(winkel)                                     <373E>
280 WEND                                     <0688>
```

```
100 'Moiree-Muster                           <147D>
110 '                                        <07E7>
120 'Farben einstellen                       <188C>
130 INK 0,0:PAPER 0:INK 1,12:INK 2
,7:INK 3,18                                  <1D7B>
140 BORDER 0:MODE 1                          <0A31>
150 'Muster erzeugen                         <16D2>
160 zaehler=0                                <117D>
170 FOR i=0 TO 640 STEP 2                    <1228>
180 GOSUB 240                                <09BC>
190 MOVE 320,0:DRAW i,400,zaehler            <2061>
200 GOSUB 240                                <09E4>
210 MOVE 320,400:DRAW i,0,zaehler            <20E5>
220 NEXT i                                   <0A29>
230 GOTO 230'Ready unterdruecken             <1E70>
240 zaehler=zaehler+1                        <1C28>
250 IF zaehler>3 THEN zaehler=1              <1F8B>
260 RETURN                                   <0630>
```

einem kartesischen Koordinatensystem arbeiten. Das bedeutet, daß sich der Punkt (0;0) unten links auf dem Bildschirm befindet. Beim Umsetzen einer mathematischen Formel haben Sie daher keine Schwierigkeiten. Sie können sie fast ohne Änderungen übernehmen; lediglich den Definitions- und Wertebereich müssen Sie an die Koordinaten anpassen, die Sie auf dem CPC ansteuern können. Das geschieht in der Regel durch Verschiebung oder Streckung. Um die benötigten Faktoren zu ermitteln, können Sie *Abbildung 1* zu Rate ziehen. Dort ist der Aufbau des Bildschirms in allen drei Modi dargestellt.

KEINE ANGST VOR MATHE

Da bereits die Rede von mathematischen Anwendungen war, soll als nächstes eine etwas schwierigere Aufgabe in Angriff genommen werden. Der CPC soll ein n-Eck auf dem Bildschirm darstellen. Nachdem Sie die gewünschte Anzahl von Ecken eingegeben haben, erscheint die entsprechende Figur.
Das klingt zwar recht kompliziert, ist es aber nicht. Einige mathematische Grundlagen sind allerdings nötig. Wichtig ist zunächst die allgemeine Formel zur Darstellung einer Ellipse mit den Radien r1 und r2. Sie lautet:

```
x = cos(w)*r1
y = sin(w)*r2
```

Um eine vollständige Ellipse darzustellen, muß der Winkel w alle Werte von Null bis zwei mal Pi durchlaufen. Da die Menge dieser Werte unendlich ist, müssen Sie eine akzeptable Schrittweite wählen, so daß die errechneten Punkte zu der Grafikauflösung Ihres Rechners passen. Die allgemeine Formel läßt sich für den Sonderfall einer Kreisdarstellung leicht abwandeln, indem Sie für r1 und r2 die gleichen Werte vergeben. Eine Ellipse mit zwei gleich langen Halbachsen ist nichts anderes als ein Kreis.
So mancher wird durch diese mathematischen Grundlagen von der Grafikprogrammierung abgeschreckt. Dazu besteht jedoch kein Grund. Ein Programmierer muß nicht zugleich Mathematiker sein. Die benötigten Formeln finden Sie in jeder Formelsammlung; Sie müssen nicht wissen, wie sie hergeleitet werden. Allein die Anwendung ist wichtig.
Was hat nun die Kreisformel mit der Darstellung eines n-Ecks zu tun? Wie oben beschrieben, müssen Sie die Schrittweite für den Abstand der einzelnen Punkte nur klein genug wählen, um einen Kreis auf den Bildschirm zu bringen.
Doch nehmen Sie einmal den umgekehrten Fall einer großen Schrittweite an, so daß für einen Kreis nur drei Punkte ermittelt werden. Diese drei Punkte befinden sich im gleichen Abstand zueinander und liegen auf einem Kreisbogen. Das bewirkt unsere Formel. Wenn Sie diese drei Punkte verbinden, so erhalten Sie ein perfektes gleichseitiges Dreieck. Ebenso können Sie mit vier, fünf oder mehr Punkten verfahren. *Listing 2* demonstriert, wie kurz ein solches Programm sein kann. Nach der obligatorischen Festlegung der Farben werden Sie nach der Anzahl der Ekken gefragt. Anschließend vergibt das Programm einige Startwerte. Die Variablen xz und yz geben das Zentrum des Kreises an, das in der Mitte des Bildschirms liegen soll. Der Wert r bestimmt den Radius, und winkel gibt einen Startwert für den aktuellen Winkel an – wie der Name verrät.
Eine wichtige Berechnung wird in Zeile 220 durchgeführt. Die Variable schritt enthält die für Ihr n-Eck benötigte Schrittweite. Die einzelnen Stützpunkte, wie der Fachausdruck lautet, müssen im richtigen Abstand auf dem Umfang des Kreises verteilt werden. Der Wert der Variablen schritt ergibt sich daher, indem man den Wert für den Vollkreis, zwei mal Pi, durch die Anzahl der Ecken dividiert. Von Null ausgehend, muß der Rechner die auf diese Art ermittelten Punkte miteinander verbinden. Damit dies möglich wird, müssen Sie erst einmal den Grafikcursor auf den Startpunkt des Kreises setzen. Das geschieht in Zeile 240. Der Startpunkt befindet sich bei einem Winkel von Null. Wenn Sie die Kreisformel anwenden, ergibt sich:

```
x = xz+cos(0)*r
y = yz+sin(0)*r
```

Da der Cosinus von Null den Wert Eins und der Sinus von Null den Wert Null hat, können Sie diese Formeln vereinfachen zu

```
x = xz+1*r = xz+r
y = yz+0*r = yz
```

Damit haben Sie die Parameter für die MOVE-Anweisung in Zeile 240 ermittelt. Wenn Sie sich diese Rechenarbeit ersparen wollen, so können Sie auch die Formel übernehmen, ohne sie zu vereinfachen. Der Funktion

INK farbregister,farbe

weist dem Farbregister die Farbe mit der angegebenen Kennziffer zu

PEN farbregister

setzt die Farbe für auszugebenden Text auf das angegebene Farbregister

MODE bildschirmmodus

stellt den angegebenen Bildschirmmodus ein (siehe auch Abbildung 1)

PLOT x,y[,farbregister]

setzt einen Punkt an der Stelle (x;y) in der aktuellen oder angegebenen Farbe. Der Grafikcursor befindet sich anschließend an der Stelle (x;y).

MOVE x,y

plaziert den Grafikcursor an der Stelle (x;y)

DRAW x,y[,farbregister]

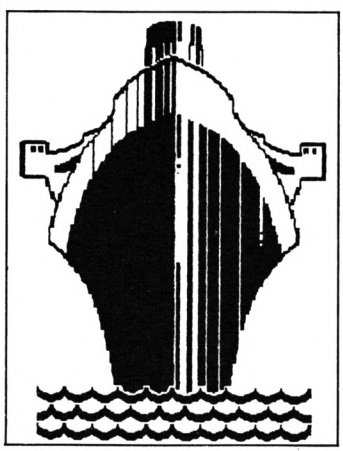
zieht eine Linie von der aktuellen Position des Grafikcursors an die Position (x;y) in der aktuellen oder angegebenen Farbe. Der Grafikcursor befindet sich anschließend an der Stelle (x;y).

Abb 2: Die wichtigsten Grafikbefehle

des Programms tut dies keinen Abbruch.

DER UNSICHTBARE CURSOR

Der Befehl MOVE verändert die Position des Grafikcursors, ohne auf dem Bildschirm eine Änderung zu bewirken. Dieser Grafikcursor ist kein sichtbarer Punkt wie der Cursor, der etwa bei einem INPUT-Befehl erscheint. Konkret handelt es sich dabei nur um einige Stellen im Speicher des Computers, in denen die Position vermerkt ist, an der der letzte Zeichenbefehl endete.
Der Begriff Cursor ist jedem Anwender von seinen ersten Übungen am Computer her bekannt, so daß zur Veranschaulichung der Ausdruck Grafikcursor eingeführt wurde. Gewöhnungsbedürftig ist dabei nur, daß Sie diesen Grafikcursor nicht sehen können. Seine Position hingegen erfahren Sie jederzeit durch die BASIC-Funktionen XPOS und YPOS.
Der Befehl DRAW, den Sie in Zeile 270 finden, zieht eine Linie von der aktuellen Position des Grafikcursors zu dem dahinter angegebenen Punkt. Sie könnten hier, wie bei PLOT, als dritten Parameter die gewünschte Farbe angeben. Der Grafikcursor steht nach Ausführung von DRAW an den angegebenen Koordinaten.
Ändern Sie Listing 2 so ab, daß der Radius des Kreises über den Bildschirmbereich hinausgeht. Sie werden feststellen, daß das Ihren CPC überhaupt nicht stört. Auch das ist ein Vorteil gegenüber vielen anderen Rechnern. Durch Änderung weniger Faktoren können Sie eine Grafik ausschnittweise vergrößern. Die Teile, die nicht mehr auf den Bildschirm passen, werden einfach nicht dargestellt.
Dabei erhalten Sie keine Fehlermeldung. Diese Tatsache kann aber auch Anlaß zu einer großen Suchaktion werden. Das ist dann der Fall, wenn Sie auf dem Monitor nichts sehen, obwohl etwas dargestellt werden müßte. An welcher Stelle Ihr CPC versucht, etwas zu malen, können Sie wieder mit XPOS und YPOS erfahren. Meist gelingt es damit, den Fehler im Programm einzugrenzen und schnell zu beseitigen.



EFFEKTE DURCH FARBÄNDERUNG

Den Beweis, daß bereits mit einfachen Befehlen wie MOVE und DRAW eindrucksvolle Grafiken erzeugt werden können, liefert *Listing 3*. Denken Sie daran, daß Sie die INK-Belegungen nachträglich ändern können. Die bereits in dieser Farbe dargestellten Bildteile erhalten dadurch ebenfalls eine andere Farbe. Denken Sie an den Zusammenhang zwischen der Farbbelegung und der sie repräsentierenden Bit-Kombination! Das eröffnet Ihnen die Möglichkeit, Bewegungen auf dem Bildschirm durch Farbänderungen zu simulieren. Mit diesem Programm können Sie testen, wie eindrucksvoll solche kleinen Tricks wirken können.
In *Abbildung 2* finden Sie eine Zusammenfassung der Grafikbefehle, die in dieser Folge vorgestellt wurden. Auch die bekannten Anweisungen INK, PEN und MODE werden dort noch einmal erläutert. *Alwin Ertl*□

Werfen Sie Ihren CPC nicht gleich zum Fenster hinaus!

Die Fehlermeldungen des CPC haben schon viele zur Verzweiflung gebracht. Werfen Sie Ihren Computer deshalb nicht gleich zum Fenster hinaus. Einige hilfreiche Befehle, dazu die Auflistung der BASIC-Fehlermeldungen, werden Ihnen den nötigen Durchblick verschaffen.

Scheitert der CPC beim Versuch, einen Befehl auszuführen, gibt er Fehlermeldungen aus. Dabei sind die Möglichkeiten, Fehler in Programme durch falsche Programmierung einzubauen, vielfältig. Die wohl häufigste Meldung ist: "Syntax Error".
Aber es gibt noch andere Fehlerarten. Nämlich die, die nicht durch eine falsche Programmierung auftreten, sondern Systemmeldungen, die auf eine fehlerhafte Handhabung hindeuten oder einen sonstigen Fehler melden. Beispiele sind die "Read Error b"-Meldung bei Kassettenstationen und der "Read fail"-Hinweis bei den Diskettenlaufwerken.

FEHLER ÜBER FEHLER – ABER WELCHE?

Bei BASIC haben Sie die Möglichkeit, den Fehlermeldungen auf die Spur zu kommen. Es gibt Befehle, die Ihnen hilfreich unter die Arme greifen. Wie so oft, ist auch hier der BASIC-Befehlssatz Ihres CPC anderen Rechnern überlegen. Denn diese Befehle kennt nicht jeder Computer der Acht-Bit-Klasse.
Zunächst zum ERR-Befehl: Wenn ein BASIC-Fehler auftritt, können Sie sich mit Hilfe einer Fehlerabfangroutine die Nummer des Fehlers ausgeben lassen. Fügen Sie möglichst am Anfang des Programms, folgende Zeile ein:

ON ERROR GOTO <Zeilennummer>

Damit wird der CPC, wenn er bei der Abarbeitung eines von Ihnen geschriebenen Programmes auf einen Fehler trifft, zu dieser Zeilennummer verzweigen. Dort haben Sie die Möglichkeit, sich die Fehlernummer ausgeben zu lassen. Die BASIC-Fehlermeldungen haben laufende Nummern, so daß Sie im Handbuch Ihren Fehler nachlesen können.

KOMFORTABLE AUSGABE DER FEHLERQUELLE

Der zweite Befehl, den der CPC zur Verfügung stellt, ist der ERL-Befehl. Er enthält nach dem Auftreten eines Fehlers dessen Zeilennummer.

Lesen Sie bitte weiter a.S. 31

1. ALLGEMEINE BASIC-FEHLERMELDUNGEN

Nummer	Fehlertext	Ursache
1	Unexpected NEXT	Es wurde im Programm ein NEXT gefunden, ohne das ein FOR/TO-Befehl angetroffen wurde.
2	Syntax Error	Ihr CPC versteht einen Befehl nicht. Meist falsche Schreibweise oder Tippfehler
3	Unexpected RETURN	Es wurde ein Return ohne das zugehörige GOSUB angetroffen.
4	DATA exhausted	Wenn die in den DATA-Zeilen abgelegten Daten schon durch READ alle gelesen wurden und trotzdem noch mehr gelesen werden sollen, wird diese Meldung ausgegeben
5	Improper Argument	Der Wert eines Arguments in einer Funktion oder ein Parameter ist falsch.
6	Owerflow	Eine Zahl ist zu groß
7	Memory full	Der Speicher ist voll, entweder weil das Programm oder die Variablen zu umfangreich sind. Oder das Programm ist zu verschachtelt.
8	Line does not exist	Eine Zeile, die angesprochen wurde, ist nicht vorhanden
9	Subscript out of range	In einem Feld (Matrix) ist eine Indexzahl zu groß oder zu klein.
10	Array already dimensioned	Das Feld wurde durch DIM bereits dimensioniert
11	Division by zero	Division durch Null. Bitte den Wert des Quotienten kontrollieren
12	Invalid direct command	Das eingegebene Kommando darf im Direktmodus nicht eingegeben werden.
13	Type mismatch	Typenunstimmigkeit. Zum Beispiel wurde durch READ versucht, Strings zu lesen, es wurde aber eine Zahl angetroffen. Oder umgekehrt.
14	String space full	Der Speicher ist voll, es können keine weiteren Strings mehr verwaltet werden.

15	String too long	Ein String ist länger als 255 Zeichen
16	String expression too complex	Ein String ist zu komplex
17	Cannot CONTinue	Der CONT-Befehl kann nicht ausgeführt werden. Das ist zum Beispiel dann der Fall, wenn nach einem Programmabbruch eine Zeile geändert wurde.
18	Unknown user command	Eine FN-Funktion wurde aufgerufen, ohne daß sie definiert wurde
19	RESUME missing	In einer Fehlerbehandlungsroutine wurde das Programmende erreicht, ohne daß ein RESUME folgt.
2Ø	Unexpected RESUME	Im Programmlauf wurde ein RESUME angetroffen. Dies ist jedoch nur in einer ON ERROR GOTO-Routine erlaubt.
21	Direct command	Beim Laden einer Datei als Programm wurde eine Zeile ohne Zeilennummern gefunden. Meist ASCII-Texte.
22	Operand missing	Die Anweisung ist unvollständig
23	Line to long	Eine Basiczeile ist länger als 255 Zeichen
24	EOF met	Trotz Fileende wurden noch Leseversuche unternommen.
25	File type error	Die Datei, die gelesen werden soll, liegt im falschen Format vor (zum Beispiel können mit OPENIN keine Basicprogramme eingelesen werden)
26	Next missing	In einer FOR-NEXT-Schleife fehlt das NEXT
27	File already open	Eine Datei, die gelesen werden soll, wurde schon mit OPENIN geöffnet.
28	Unknown command	Ein unbekanntes Kommando wurde aufgerufen.
29	WEND missing	In einer WHILE-WEND-Schleife fehlt das WEND.
3Ø	Unexpected WEND	Der WEND-Befehl wurde außerhalb einer WHILE-WEND-Schleife angetroffen.

```
================================================================

Spezielle AMSOS-Fehlermeldungen bei Diskettenlaufwerken
(nur bei den CPC-Typen 664 und 6128)

DERR-Wert       Fehlerursache

142             Stream ist nicht im richtigen Zustand

143             Ende der Datei(Hardware-Ende) erreicht

144             Fehlerhaftes Kommando, zum Beispiel falscher Dateiname

145             Datei dieses Namens besteht bereits

146             Datei kann nicht gefunden werden

147             Directory ist voll

148             Diskette ist voll

149             Diskettenwechsel, während Dateien mit OPENIN oder
                OPENOUT geöffnet waren.

150             Datei ist nur lesbar und kann nicht bearbeitet werden.

154             Dateiende (soft end) erreicht

193             Adresse fehlt

194             Diskette ist schreibgeschützt

196             Sektor unbekannt

200             Diskette fehlt, Laufwerk nicht bereit

208             Überlauffehler

224             Datenfehler
```

Die Fehlermeldungen des CPC sind recht ausführlich, aber gerade deshalb oft schwer zu verstehen

Damit haben Sie bereits alle Befehle, um sich neben der Fehlernummer auch die fehlerhafte Zeile ausgeben zu lassen. Die Zeile, die Sie an der Stelle einfügen müssen, zu der Ihr Computer springen soll, muß heißen:
<Zeilennummer>
PRINT"Fehler Nummer"
;ERR;"in Zeile";ERL
Mit dem RESUME-Befehl in der nachfolgenden Zeile teilen Sie Ihrem Rechner mit, was er anschließend tun soll.
Wenn er in die fehlerhafte Zeile zurückkehren soll, genügt ein RESUME allein. Soll er jedoch die darauf folgende Zeile abarbeiten, so heißt der Befehl RESUME NEXT.
Aber Achtung: Dieses NEXT benötigt kein FOR, hat also nichts mit einer FOR-NEXT-Schleife zu tun.
Als dritte RESUME-Möglichkeit können Sie auch eine Zeilennummer angeben. Dann wird dort das Programm weiter abgearbeitet. Ihre Fehlerausgabe-Routine muß so aussehen:

1. Am Anfang des Programms:
   ON ERROR GOTO
   <Zeilennummer>

2. Fehlerausgabe:
   2.1
   PRINT "Fehler
   Nummer";ERR;"in
   Zeile";ERL
   2.2
   RESUME oder
   RESUME NEXT
   oder RESUME
   <Zeilennummer>

Die Befehle ERR, ERL und RESUME gelten für alle CPC-Besitzer.
Es gibt noch einen speziellen Befehl für die Ausgabe von Disketten-Fehlermeldungen. Er heißt DERR und funktioniert nur in Verbindung mit einem AMSDOS-Laufwerk, also dem üblichen beim CPC.

Tritt ein solcher Diskettenfehler auf, so wird als Fehlernummer bei Verwendung des ERR-Befehls stets die Zahl 32 ausgegeben. Wollen Sie wissen, welcher spezielle Diskettenfehler aufgetreten ist, müssen Sie den DERR-Befehl anwenden. Damit erhalten Sie die genauere Fehlerspezifikation (siehe hierzu unter AMSDOS-Fehler). *JE* □

Gregors Rache

Ordnung muß sein, dachten sich schon die alten Römer, und gliederten das Jahr. Aus der Zeit der Cäsaren stammen immer noch einige Monatsnamen. Nicht nur die lateinisch Vorgebildeten, sondern auch Computerkenner wissen jedoch, daß mit den Bezeichnungen nicht alles korrekt ist.

Oktober ist zum Beispiel die Bezeichnung des achten Monats. Octo bedeutet im Lateinischen acht, Programmierer kennen ja die Oktalzahlen. Der Dezember sollte einmal der zehnte Monat des Jahres sein. Im alten Rom war dies ganz richtig, denn das Jahr wurde ab März gezählt.

Es lag nicht einfach an der Zählweise, daß es noch allerhand Schwierigkeiten mit dem Kalender gab. Die Erde kreiste einfach nicht so um die Sonne, wie es die Mathematiker vorschrieben, sie war zu langsam. Für ein ganzes Jahr benötigte Mutter Erde nicht 365 Tage, wie amtlich veranschlagt, sondern sie ließ sich Zeit und brauchte etwa einen viertel Tag mehr.

Die Lösung war ein zusätzlicher Tag, den man alle vier Jahre dem Februar zukommen ließ. Leider klappte es immer noch nicht so richtig, denn es waren ja nicht exakt sechs Stunden, also 360 Minuten, sondern 348 Minuten und 46 Sekunden.

Gar so schlimm ist dies nicht, aber im Laufe der Jahrhunderte kamen doch ein paar Tage zusammen. Im 16. Jahrhundert war es dann einigen Leuten zuviel. Die Kirche, aus der sich damals die Wissenschaftler rekrutierten, dachte als erste darüber nach.

Schließlich erließ Papst Gregor XIII. (1502–85) ein Dekret, daß der Kalender neu zu ordnen sei. Das bedeutete für seine Mannen erst einmal eine Menge Rechenarbeit, aber 1582 war es dann soweit: Der Gregorianische Kalender löste den Julianischen ab. Nach dieser Neuordnung richten wir uns auch heute noch, wenngleich sie noch einmal überarbeitet wurde.

BEDINGUNGEN ZUM RECHNEN

Um überhaupt mit der Arbeit des Programmierens anfangen zu können, rufen wir uns kurz die Regeln dieses Kalenders in Erinnerung: Jedes Standardjahr hat 365 Tage. Alle vier Jahre gibt es ein Schaltjahr mit 366 Tagen, der zusätzliche Tag ist der 29. Februar. Der Einfachheit halber sind es die Jahre, die sich durch vier teilen lassen; 1988 ist also wieder einmal soweit.

So viel wußte man aber schon im alten Rom, und trotzdem wollte es nicht klappen. Papst Gregor dachte jedoch weiter. Auch jedes volle Jahrhundert erhält deshalb diesen Bonustag, sonst geht die Rechnung nicht auf. Dies war übrigens der Punkt des päpstlichen Dekrets, der erst nachträglich korrigiert wurde. Ursprünglich hantierte man noch etwas ungeschickt mit den Jahrhunderten und wollte nur jenen einen Schalttag gönnen, deren erste beiden Ziffern sich nicht durch vier teilen ließen.

Mit diesen Angaben sind alle Voraussetzungen dafür geschaffen, um die Summe der Tage zwischen zwei Daten berechnen zu können. Jetzt hat der Computerfreak das Wort. Für alle, die nur am Ergebnis interessiert sind, ist das *Listing 2* gedacht. Um zu erfahren, wie wir auf die kurze Formel kommen, müssen wir uns durch *Listing 1* quälen. Wir müssen zuerst wissen, was wir errechnen wollen, mit einem Kalender kann man ja allerhand anfangen. Entweder wir berechnen alle Tage zwischen zwei verschiedenen Daten, wie es zum Beispiel für ein Biorhythmus-Programm notwendig ist. Oder wir ermitteln den aktuellen Wochentag, um eine Notizdatei zu erstellen. Listing 1 erledigt die erste Aufgabe, die Kurzform Nummer 2 ist für letztgenannte Routine notwendig.

Die Dimensionierung der Datenfelder ist logisch: Das Jahr hat zwölf Monate und damit Monatsnamen. Diese stehen in der Variablen monat$,

```
10 REM ***       Listing 1       ***
20 REM *** Kalenderberechnungen ***
30 DIM monat$(12),monat(12)
40 FOR i=1 TO 12:READ monat$(i):READ mon
at(i):NEXT i
50 PRINT"Heutiges Datum"
60 INPUT"",dat1$:ht=VAL(MID$(dat1$,1,2))
:hm=VAL(MID$(dat1$,4,2)):hj=VAL(MID$(dat
1$,7,4))
70 PRINT"Geburtsdatum    ";
80 INPUT"",dat2$:gt=VAL(MID$(dat2$,1,2))
:gm=VAL(MID$(dat2$,4,2)):gj=VAL(MID$(dat
2$,7,4))
90 REM ***    Berechnung        ***
100 REM *** Tage im Geburtsjahr ***
110 gestage=monat(gm)-gt
120 IF gm<=2 AND gj/4=INT(gj/4) THEN ges
tage=gestage+1
130 IF gm=12 THEN 150
140 FOR i=gm+1 TO 12:gestage=gestage+mon
at(i):NEXT i
150 REM ***   Berechnung    ***
160 REM *** Tage der Jahre ***
170 FOR i=gj+1 TO hj-1:gestage=gestage+3
65
180 IF i/4=INT(i/4) THEN gestage=gestage
+1
190 NEXT i
200 REM *** Tage / aktuelles Jahres ***
210 IF hm=1 THEN 240
220 IF hm>2 AND hj/4=INT(hj/4) THEN gest
age=gestage+1
230 FOR i=1 TO hm-1:gestage=gestage+mona
t(i):NEXT i
240 gestage=gestage+ht
250 PRINT"Gesamttage ",gestage
260 DATA Januar,31,Februar,28,Maerz,31,A
pril,30,Mai,31,Juni,30,Juli,31,August,31
,September,30,Oktober,31,November,30,Dez
ember,31
```

die dazugehörigen Tage in der Zahlvariablen monat. Die Data werden am Ende des Programms untergebracht. Die dazugehörige Einleseroutine reduziert sich auf die Zeile 40.
Um schnell zur Sache zu kommen, wird kein Modus definiert, und auch die üblichen Spielereien mit Farben und Windows entfallen. In Zeile 50 wird das heutige Datum verlangt, zunächst noch als Textstring, und deswegen heißt es aufpassen.

MID$- UND VAL-ANWEISUNG WANDELN DEN EINGABESTRING IN ZAHLEN

Die nachfolgende MID$ Konstruktion erwartet nämlich die richtige Zahl am richtigen Platz des Textes. Der aktuelle Tag ist zweistellig und steht an erster Stelle. Ebenfalls zweistellig ist der Monat, er beginnt an vierter Stelle, ab der sechsten Stelle steht die vierstellige Jahreszahl.
Bei der Eingabe des Datums ist also ein Trennungszeichen zwischen Tag und Monat und zwischen Monat und Jahr vorgesehen. Ob Sie hierfür den Punkt oder den Doppelpunkt verwenden, spielt keine Rolle. Richtig wäre die Eingabe von 04.04.1988 für den vierten April. Geben Sie 4.4.88 ein, dann kommt es zu einem Syntax-Error: Das Programm würde das Jahr 8 herausfischen, der Monat hätte nur eine Bruchzahl, nämlich 0.8.

DIE VARIABLEN

Nach der MID$-Routine sind folgende Variablen mit den entsprechenden Werten belegt:

ht: Heutiger Tag
hm: Heutiger Monat
hj: Heutiges Jahr

Und analog dazu:

gt: Geburtstag
gm: Geburtsmonat
gj: Geburtsjahr

```
10 REM ***          Listing 2          ***
20 REM *** Berechnung Wochentages ***
30 'Dimensionierung & Variablenbelegung
40 '
50 DIM mt$(12):DIM tm(12)
60 t$(1)="Sonntag":t$(2)="Montag":t$(3)=
"Dienstag":t$(4)="Mittwoch":t$(5)="Donne
rstag":t$(6)="Freitag":t$(7)="Samstag"
70 mt$(1)="Januar":tm(1)=31:mt$(2)="Febr
uar":tm(2)=28:mt$(3)="Maerz":tm(3)=31:mt
$(4)="April":tm(4)=30:mt$(5)="Mai":tm(5)
=31:mt$(6)="Juni":tm(6)=30:mt$(7)="Juli"
:tm(7)=31:mt$(8)="August":tm(8)=31
80 mt$(9)="September":tm(9)=30:mt$(10)="
Oktober":tm(10)=31:mt$(11)="November":tm
(11)=30:mt$(12)="Dezember":tm(12)=31
90 CLS:PRINT,"Bitte geben Sie das heutig
e Datum ein":PRINT:PRINT,"TTMMJJJJ":d$="
":INPUT "",akd$
100 '
110 ' Eingabestring umwandeln
120 '
130 CLS:t=VAL(LEFT$(akd$,2)):m=VAL(MID$(
akd$,3,2)):j=VAL(RIGHT$(akd$,4))
140 '
150 ' Brechnung
160 '
170 r1=j-(INT(0.6+(1/m))):r2=m+12*(INT(0
.6+(1/m))):r3=r1/100
180 w1=INT(13*(r2+1)/5)+INT((5*r1)/4)-IN
T(r3)+INT(r3/4)+t-1
190 w=w1-(7*(INT(w1/7)))+1
200 PRINT "Heute ist ";t$(w);" ";t;m;j
```

Die einzige Variable, die dann noch benötigt wird, repräsentiert auch schon das Ergebnis: gestage ist von gesamten Tagen abgeleitet.

DIE UNTER-ROUTINEN: TAGE IM GEBURTS-JAHR

Es gibt nur drei Rechengänge: Zunächst sind alle Tage zu zählen, die im Geburtsjahr bis zum 31. Dezember noch anfallen. Die Tage der ganzen Monate werden in der Schleife in Zeile 140 addiert. Hinzuzuzählen sind nur noch die im Geburtsmonat vergangenen Tage. Doch halt, erinnern wir uns an die Bedingungen, die Papst Gregor festlegte. Falls der Geburtstag vor oder im Februar lag und die Jahreszahl glatt durch vier zu teilen ist, muß dem Ergebnis noch ein Tag hinzugefügt werden.

TAGE DER VERGANGENEN GANZEN JAHRE

Das nächste Programm zählt für jedes Jahr 365 Tage hinzu. Jedoch nicht für das aktuelle Jahr, dies kommt in einem anderen Unterprogramm an die Reihe. Und das Geburtsjahr wurde ja bereits berechnet. Es kann zwar sein, daß zwischen diesen beiden Daten kein ganzes Jahr vergangen ist, aber die Schleife ist so konstruiert, daß in diesem Fall nichts gezählt wird. Eine IF-THEN-Abfrage sorgt dafür, daß bei einem Schaltjahr ein zusätzlicher Tag berechnet wird. Hierzu stünde zwar der MOD-Befehl zur Verfügung, der immer den Restwert ausgibt, aber der erklärt sich auch nicht von alleine. Also bedienen wir uns der INT-Anweisung. Ist der durch vier geteilte Betrag gleich dem gerundeten Ergebnis, dann handelt es sich um ein Schaltjahr. Die Jahrhunderte als Schaltjahr können wir vergessen, sie sind ja immer durch vier teilbar.

TAGE DES AKTUELLEN JAHRES

Das Verfahren, die Tage des aktuellen Jahres zu zählen, haben Sie schon im Teilprogramm für das Geburtsjahr kennengelernt. Es muß lediglich nicht mehr bis zum 31. Dezember gerechnet werden, sondern vom ersten Januar bis zum Geburtstag.
Auch beim Schaltjahr müssen Sie wieder aufpassen. Diesmal heißt es in Zeile 220: Falls der aktuelle Monat über zwei liegt, dann ist der Summe aller Tage ein Tag hinzuzufügen.
Was wir jetzt haben, ist die Lebensspanne in Tagen. Allerdings gibt es noch einen wesentlichen Nachteil, den Sie aber bei Bedarf selbst korrigieren können: Durch die Schleifenkonstruktion ist es zunächst nicht möglich, das Alter zu berechnen, wenn Geburtsjahr und aktuelles Jahr gleich sind. Hier müßte bei der Berechnung im Geburtsjahr eine Abfrage eingebaut werden.

DIE BERECHNUNG DES WOCHENTAGES

Das zweite Problem, nämlich daß der Wochentag nicht genannt wird, wollen wir mit dem zweiten Listing aus der Welt schaffen. Welche Berechnungen hier durchgeführt werden müssen, haben Sie bereits erfahren. Wir müssen alle Tage bis zu einem festen Bezugspunkt, von dem bekannt ist, wel-

cher Wochentag es war, ermitteln.
In Listing 1 haben Sie gesehen, warum man welche Berechnungen durchführt. Listing 2 zeigt Ihnen, auf welch knappe Form man die mathematischen Regeln bringen kann. Für den Einbau in ein Programm dürfte es sicherlich besser geeignet sein.
Auch die Einleseroutine wurde verändert, damit Sie sehen, daß es auch anders geht. Die Werte wurden fest eingegeben und müssen nicht mehr durch Data eingelesen werden.
Als Variablen wurden belegt:

mt$:	Monatsnamen
tm:	Tage im Monat
t$:	Wochentag

Die Variable w liefert schließlich eine Zahl, die dem jeweiligen t$ entspricht. r1, r2 und w1 dienen nur zur Aufnahme eines Zwischenergebnisses.

EINGABEMETHODEN BEI ANDEREN PROGRAMMEN

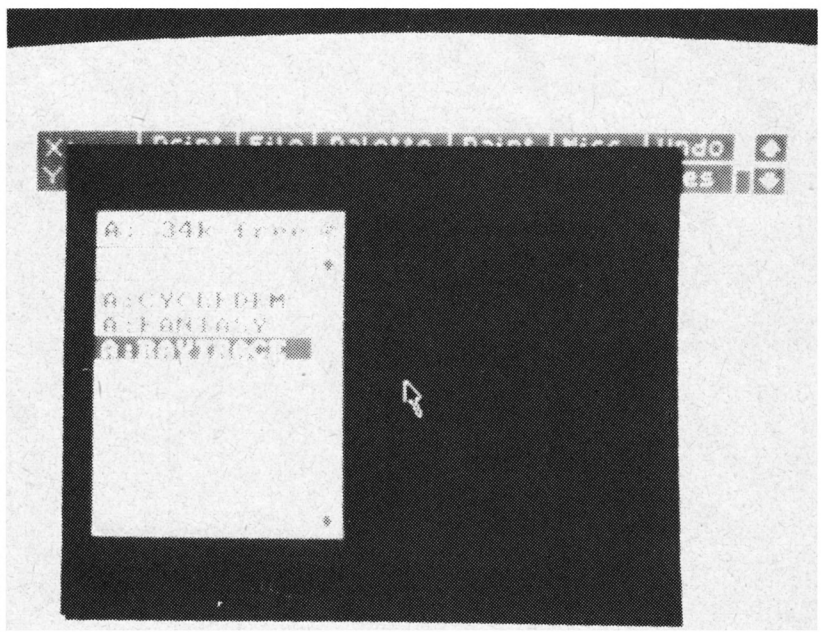
Da es immer wieder Schwierigkeiten bei der Eingabe des Datums gibt, hier eine Form, bei der auf ein Trennungszeichen, etwa einen Doppelpunkt, verzichtet wurde. Generell benötigt eine Berechnungsroutine immer die volle Jahreszahl, also bitte niemals 88 für 1988 eingeben. Sollten Trennungszeichen eingegeben werden, dann handelt es sich ohnehin um einen String, bei dem die MID$-Anweisung und der VAL-Befehl sich die richtigen Zahlen heraussuchen.
Auf das Trennungszeichen dürfte es also nicht ankommen.
In Zukunft kann Ihr Programm also wissen, welcher Wochentag gerade ist.
Probieren Sie einmal aus, ob der Computer auch montags Lust zum Arbeiten hat. Wenn ja, dann viel Spaß dabei. *GS* □

WINDOWS OHNE WINDOW

Fenster zum Hof

Mancher BASIC-Programmierer hat auf seinem CPC schon einen Befehl vermißt, um ein Zeichen vom Bildschirm zu lesen. Ihr Rechner stellt jedoch eine Routine bereit, mit der Sie dieses Problem lösen können.

Im Speicher Ihres CPC befindet sich eine stattliche Anzahl vorprogrammierter Routinen, die sogenannte Firmware. Diese Assembler-Programme zieht BASIC heran, wenn es eine bestimmte Aufgabe bewältigen soll. So gibt es etwa Routinen zum Ziehen von Linien oder zur Ausgabe eines Textzeichens. Auch ein Programm, das ein Zeichen auf dem Bildschirm identifiziert, ist vorhanden.
Warum die Autoren des Locomotive BASIC diese Routine nicht genutzt haben, um einen entsprechenden BASIC-Befehl zu schaffen, wird wohl unerklärt bleiben. Diese fehlende Anweisung können Sie aber auf einfache Weise ersetzen.
Zum Beispiel haben Sie die Möglichkeit, einen RSX-Befehl zu schaffen, der die gewünschte Aufgabe erfüllt. Die Anwendung eines solchen Befehls ist sehr einfach, der Aufwand für seine Installierung sehr groß. Wenn Sie aber bereit sind, einige Unbequemlichkeiten bei der Anwendung in Kauf zu nehmen, hält sich der Aufwand in Grenzen. Sie benötigen nur drei Zeilen Assembler, um ein Zeichen vom Bildschirm lesen zu können. Außerdem müssen Sie kein Assembler-Experte sein, um das Programm zu verstehen.



Windows sind nicht nur den professionellen Programmierern vorbehalten.

ASSEMBLER IST UNVERMEIDBAR

Um Routinen in Maschinensprache in den Speicher des CPC zu bekommen, wird in der Regel ein sogenannter DATA-Lader verwendet. Assembler ist eine symbolische Programmiersprache, deren Anweisungen von einem entsprechenden Programm in Zahlen umgesetzt werden. Diese Werte kann der Prozessor direkt verarbeiten. In Zeile 10080 des Programms SCREEN.MRG finden Sie die DATAs, die von unserer Assembler-Routine erzeugt werden. In vollständiger Notation liest sich das so:

```
ORG &A000
DEFB 0
CALL &BB60
LD (&A000),A
RET
```

Die beiden ersten Befehle sind Pseudo-Anweisungen, die dem Übersetzungsprogramm mitteilen, was es tun soll. Die ORG-Anweisung sagt ihm, daß der erzeugte Code ab der Speicherstelle &A000 abgelegt werden soll. DEFB steht für „define byte“ und belegt die erste Speicherstelle mit Null.
Nun beginnt das eigentliche Programm: Mit CALL wird eine Firmware-Routine an der Adresse &BB60 aufgerufen. Das funktioniert ähnlich dem GOSUB-Befehl in BASIC. In der Variablen A steht nach dem Verlassen des Unterprogramms der Character-Code des Zeichens, das an der aktuellen Cursor-

Position gefunden wurde. Dieser Code wird durch die nachfolgende LD-Anweisung (für "load") an der Adresse &A000 abgelegt. Mit RETurn wird das Unterprogramm verlassen, und BASIC macht weiter, als wäre nichts geschehen. Dieses Unterprogramm können Sie, nachdem es sich im Speicher Ihres Rechners befindet, mit CALL &A001 starten. Es legt den Code des an der Cursor-Position befindlichen Zeichens in &A000 ab. Mit einem entsprechenden PEEK-Befehl können Sie diesen Wert erfahren. Beachten Sie bei der Anwendung, daß der Cursor über dem Zeichen stehen muß. Eventuell muß zuvor ein LOCATE-Befehl ausgeführt werden.
Die Firmware-Routine bei &BB60 heißt TXT RD CHAR. Das ist die Abkürzung von "text read character". Das Wörtchen text zeigt an, daß das Unterprogramm aus einer Sammlung von Routinen stammt, die zusammen den Text-Manager bilden. Der Rest der Bezeichnung gibt die Funktion der Routine ab.
Das Identifizieren eines Zeichens auf dem Bildschirm des CPC ist keine leichte Aufgabe. Die meisten Rechner treffen eine strenge Unterscheidung zwischen Text- und Grafikbildschirm. Wäre dies beim CPC der Fall, bestünde nicht die Möglichkeit, Texte und grafische Darstellung auf eine derart komfortable Weise zu mischen, wie Sie es gewohnt sind. Die Arbeit mit dem Textbildschirm wäre jedoch einfacher, da jedes Zeichen nur mit seinem Code im Speicher abgelegt wäre. Mit einem PEEK an der richtigen Adresse wäre das Problem gelöst.
Der Schneider CPC jedoch stellt alle Zeichen anhand ihrer Punktmatrix dar. Das bedeutet, daß die Routine TXT RD CHAR alle Punkte einer Zeichenposition abtasten und das Ergebnis mit den bekannten Zeichenmatrizen vergleichen muß. Dabei müssen auch Möglichkeiten wie andersfarbige oder gar inverse Darstellung eines Zeichens berücksichtigt werden.
Da Sie die Möglichkeit haben, Text und Grafik beliebig zu mischen, kann es schließlich sein, daß eine Linie mitten durch einen Buchstaben verläuft. In diesem Fall muß aber auch die Firmware passen. Die dadurch entstandene Zeichenmatrix deckt sich mit keinem der bekannten Muster, und Sie erhalten als Ergebnis den Wert Null.
Wenn Sie in Ihrem Programm nur hin und wieder Zeichen vom Bildschirm lesen wollen, genügt es, die Routinensammlung SCREEN.MRG bis zur Zeile 10100 abzutippen und als ASCII-File zu speichern. Sie können das Unterprogramm mit dem MERGE-Befehl an jedes beliebige BASIC-Programm anhängen. Mit LOCATE x,y:CALL &A001 bestimmen Sie das Zeichen an der Stelle (x,y). Den Character-Code erfahren Sie mit PEEK(&A000).

```
10000 ''''''''''''''''''''''''''  (21C3)
10010 'Routinen f. Bildschirmbe-'  (210E)
10020 '        arbeitung         '  (21FC)
10030 '                          '  (214A)
10040 '       SCREEN.MRG         '  (2170)
10050 ''''''''''''''''''''''''''  (2128)
10060 '                          (07CA)
10070 'Routine installieren      (1B66)
10080 DATA 00,cd,60,bb,32,00,a0,c9
                                 (1EA2)
10090 RESTORE 10080:MEMORY &9FFF:F
OR i=1 TO 8:READ a$:POKE &9FFF+i,V
AL("&"+a$):NEXT i                (3B0C)
10100 RETURN                     (0635)
10110 'Puffer reservieren        (1977)
10120 MEMORY &9FFF-bufsize       (14B6)
10130 start=&9FFF-bufsize+1:akt=st
art                              (2E29)
10140 DIM screen(maxscr)         (1ACF)
10150 scrcnt=0                   (105B)
10160 RETURN                     (06AD)
10170 'Bildschirm speichern      (1BC4)
10180 scrcnt=scrcnt+1:IF scrcnt>ma
xscr THEN scrcnt=scrcnt-1:PRINT:PR
INT"PUFFER VOLL !!!":RETURN      (5C9C)
10190 screen(scrcnt)=akt+1       (225D)
10200 FOR i=xs TO xe:FOR j=ys TO y
e                                (28DE)
10210 LOCATE i,j:CALL &A001:akt=ak
t+1:IF akt>&9FFF THEN PRINT:PRINT"
PUFFER VOLL!!!":RETURN           (464E)
10220 POKE akt,PEEK(&A000):NEXT j,
i                                (1FC2)
10230 RETURN                     (063A)
10240 'Bildschirm holen          (17FD)
10250 IF scrcnt=0 THEN PRINT:PRINT
"PUFFER LEER !!!":RETURN         (2833)
10260 zaehl=screen(scrcnt)       (22DD)
10270 FOR i=xs TO xe:FOR j=ys TO y
e                                (286A)
10280 LOCATE i,j:PRINT CHR$(PEEK(z
aehl));:zaehl=zaehl+1:NEXT j,i   (41F1)
10290 scrcnt=scrcnt-1:IF scrcnt<0
THEN scrcnt=0                    (33FD)
10300 IF scrcnt=0 THEN akt=start-1
 ELSE akt=screen(scrcnt+1)-1     (4496)
10310 RETURN                     (06DB)
```

ZWISCHENSPEICHER FÜR DEN BILDSCHIRM

Sie können diese Möglichkeit zum Beispiel einsetzen, um Teile des Bildschirms zwischenzuspeichern und deren alten Inhalt bei Bedarf wiederherzustellen. Dazu dient der Rest von SCREEN.MRG. Natürlich haben Sie die Möglichkeit, Arrays als Zwischenspeicher zu verwenden. Da Sie aber nicht in jedem Fall die Größe des zu speichernden Teils als bekannt voraussetzen dürfen, müßten Sie die Felder so groß wie möglich definieren. Dadurch würde unnötig Speicherplatz vergeudet. Es ist daher günstiger, die Inhalte einzelner Bildschirmfenster direkt im Speicher abzulegen.
Um das Programm zu vereinfachen, wollen wir davon ausgehen, daß im Speicher befindliche Bildschirmbereiche in umgekehrter Reihenfolge ihrer Ablage wieder dargestellt werden sollen. Das Überspringen von Zwischenstufen wird nicht zugelassen. Auch bei dieser Ablageform müssen Sie zuvor einen Pufferbereich bestimmen, dem Sie eine bestimmte Größe geben. Gegenüber Arrays hat das immer noch den Vorteil, daß Sie von einer durchschnittlichen Fenstergröße ausgehen können und nicht mit mehreren Variablen arbeiten müssen.
Beansprucht etwa ein Fenster ein Viertel des Bildschirms und ein wei-

teres den kompletten Bildschirm, so können Sie einen Puffer definieren, der 1,25 Prozent der gesamten Bildfläche aufnehmen kann. Würden Sie Arrays verwenden, so würde entweder das Unterprogramm umfangreicher, da es mit zwei verschiedenen Variablen arbeiten muß, oder das Array größer, da Sie es in zwei Dimensionen mit der maximalen Ausdehnung vereinbaren müßten.
In der Variablen bufsize geben Sie an, wie groß der Pufferbereich werden soll. Mit GOSUB 10110 rufen Sie das entsprechende Unterprogramm auf. Es setzt die oberste Speichergrenze so weit herab, daß sich BASIC-Programm und Bildschirmpuffer nicht gegenseitig überschreiben können.
Je mehr Platz Sie also für den Puffer reservieren, desto kürzer muß Ihr Programm werden.
Unsere Subroutine leistet aber noch mehr: Zunächst belegt sie eine Variable namens start, die den Anfang des Puffers angibt. Der Wert akt zeigt später auf das oberste belegte Byte des Bildschirmspeichers und ist zu Anfang identisch mit start.
Die Variable maxscr müssen Sie im Hauptprogramm vorbelegen. Damit bestimmen Sie die Anzahl der Bildschirmfenster, die im Höchstfall abgelegt werden können. Das Array screen soll später die Startadressen der einzelnen Fenster enthalten; der Zähler scrcnt gibt an, wie viele Fenster bereits gespeichert sind.

```
100 'Testprogramm                          (13B6)
110 '                                      (07E7)
120 'Vor Programmstart:                    (199A)
130 '   MERGE"SCREEN.MRG"                  (1A1B)
140 '                                      (0722)
150 INK 0,0:PAPER 0:INK 1,21:PEN 1
:BORDER 0:MODE 2                           (1BC6)
160 GOSUB 10070'Installierung              (18D9)
170 bufsize=2*(25*80):maxscr=6:GOS
UB 10110'Puffer vorbereiten                (3E3D)
180 PRINT"Bitte geben Sie 20 Zeile
n beliebigen Text ein!"                    (362F)
190 FOR i=1 TO 20:LINE INPUT"",a$:
NEXT i                                     (1FC2)
200 xs=20:ys=10:xe=40:ye=20:GOSUB
10170'Bildteil speichern                   (4151)
210 LOCATE 20,10:PRINT"**********
*********":LOCATE 20,20:PRINT"****
****************"                         (42E5)
220 FOR i=11 TO 19:LOCATE 20,i:PRI
NT"*                    *":NEXT i          (37D1)
230 xs=20:ys=10:xe=40:ye=20:GOSUB
10170'Bildteil speichern                   (418D)
240 LOCATE 22,14:PRINT"Dies koennt
e":LOCATE 23,16:PRINT"ein Menuefen
-":LOCATE 25,18:PRINT"ster sein !"
                                           (4995)
250 FOR i=1 TO 1000:NEXT i                 (16DD)
260 xs=20:ys=10:xe=40:ye=20:GOSUB
10240'Letztes Bild holen                   (4137)
270 xs=20:ys=10:xe=40:ye=20:GOSUB
10240'Letztes Bild holen                   (414B)
280 FOR i=1 TO 1000:NEXT i                 (1619)
290 END                                    (06A7)
```

FENSTER IM HINTERGRUND

Nachdem all diese Werte definiert sind, ist der Puffer aufnahmebereit. Sie können jetzt ein Fenster mit GOSUB 10170 im Speicher ablegen. Allerdings müssen Sie zuvor die Grenzen des gewünschten Bildschirmbereichs bestimmen. Dazu dienen die Variablen xs, xe, ys und ye. Die Speicherroutine zählt zunächst scrcnt hoch und überprüft, ob die maximal zulässige Anzahl bereits erreicht ist. Ist das der Fall, so erhalten Sie die Meldung PUFFER VOLL. Das Unterprogramm wird verlassen, und Ihr Programm läuft weiter, als wäre nichts Besonderes geschehen. Bei der Wiederherstellung der Bildschirminhalte treten dann allerdings seltsame Effekte auf.
Sofern noch Platz im Puffer ist, wird die neue Startadresse an der nächsten Position im Array Screen abgelegt. Anschließend tastet das Programm den von Ihnen gewählten Bildbereich ab und speichert die gefundenen Zeichen. Beachten Sie dabei, daß auf Farben und invers dargestellte Buchstaben keine Rücksicht genommen wird. Bei der Wiederherstellung des Bildschirms erscheint alles in der gerade aktuellen Farbe.
Während des Speichervorgangs wird laufend überprüft, ob das Ende des Puffers erreicht ist. Es könnte ja sein, daß Sie den Bereich zu klein definiert haben. Ist das der Fall, so wird die Subroutine vorzeitig abgebrochen. Ein Teil des Bildschirms ist jedoch gespeichert und kann, wenn Sie Glück haben, wiederhergestellt werden.
Diese Wiederherstellung übernimmt das Unterprogramm ab Zeile 10240. Bevor es jedoch irgendeine Aktion in Angriff nimmt, wird geprüft, ob sich überhaupt etwas im Puffer befindet. Ist er leer, so erhalten Sie eine entsprechende Meldung, und das Hauptprogramm läuft weiter. Wurde ein Bildschirm gefunden, so wird er Zeichen für Zeichen auf den Monitor gebracht. Die Fensterbegrenzungen müssen Sie allerdings vorher bestimmen, da sie nicht abgelegt werden. Diese Methode bringt den Vorteil mit sich, daß Sie Fenster an einer beliebigen Position wiedererscheinen lassen können.
Wurde beim Abspeichern eines Textfensters das Ende des Puffers erreicht, so stört das diese Routine nicht im geringsten. Sie liest einfach über das Ende hinaus, wenn die angegebenen Fenstergrenzen dies verlangen. Dadurch kann es geschehen, daß einige Steuerzeichen, die sich zufällig in diesem Speicherbereich befinden, ausgegeben werden und Ihre Bildschirm-Darstellung ungewollt verändern. Achten Sie deshalb darauf, daß der zur Verfügung stehende Speicherplatz ausreicht.
Der Rest der Routine bedarf fast keiner Erläuterung mehr. Das Programm geht genau umgekehrt vor wie die Speicherroutine: scrcnt wird um Eins reduziert, und das Unterprogramm kann verlassen werden.
Das Testprogramm demonstriert die Anwendung dieser neuen Möglichkeiten. Sie werden feststellen, daß das Speichern und Wiederherstellen eines Fensters etwas Zeit in Anspruch nimmt. Mit einer umfangreichen Assembler-Routine könnte man dies zwar vermeiden; da BASIC aber wesentlich leichter zu verstehen ist, sind die hier vorgestellten Unterprogramme einfacher anzuwenden. Wenn Sie Programmierkenntnisse in Assembler haben, können Sie die verwendeten Verfahren sicher leicht in dieser Sprache impementieren. *AE* □

HISOFT-PASCAL WIRD ERWEITERT

Vektor-grafik

Nach der Blockgrafik in der letzten Folge (CPC-WELT 3/88) soll dieses Mal eine andere Form der grafischen Darstellung an die Reihe kommen: Vektorgrafiken oder Polygonzüge.
Wenn Sie, wie bei der Blockgrafik, Bilder aus einzelnen Punkten zusammensetzen, so ergibt sich ein gravierender Nachteil: Bei einer Vergrößerung werden aus den Punkten Kästchen beziehungsweise Klötze, je nach Vergrößerungsfaktor.
Beschreiben Sie eine Grafik jedoch nur durch ihre Verbindungslinien, so ist die Größe, in der sie definiert wurde, gleichgültig. Die Linien werden lediglich länger, nicht aber gröber.

ROUTINEN FÜR EIGENE PROGRAMME

Diese Methode soll in den heute vorgestellten Routinen angewandt werden. Von einem fiktiven Nullpunkt ausgehend, wird eine aufeinanderfolgende Reihe von Strecken festgelegt. Ein Dreieck ließe sich etwa so beschreiben: „Gehe gleichzeitig zehn Einheiten nach rechts und nach oben, dann gleichzeitig zehn Einheiten nach rechts und nach unten, zuletzt zwanzig Einheiten nach links."
Als Abfolge von drei Strecken liest sich das so:

```
 10/  10
 10/–10
–20/   0
```

Den Nullpunkt können Sie an jede beliebige Stelle auf dem Bildschirm legen – es entsteht immer das gleiche Dreieck.

Von der Theorie zur Praxis: Überlegen wir uns zuerst, wie ein solcher Linienzug im Rechner dargestellt werden könnte.
Pascal bietet als geeignete Datenstruktur hierfür einen Record an. Der Datentyp polygon aus unserem Listing enthält die Eckenzahl sowie die x- und y-Längen (nicht -Koordinaten) der einzelnen Strecken.
Als erste Funktion treffen Sie im Programmtext auf die Vorzeichenfunktion sgn. Da HiSoft-Pascal sich beharrlich weigert, hier Integer-Werte zu akzeptieren, wurde sie kurzerhand neu definiert. Die Prozedur move kennen Sie bereits aus den vielen Grafikroutinen der früheren Folgen. Drawr hingegen ist neu, entspricht aber dem gleichnamigen BASIC-Befehl. Sie dient dazu, von einem Punkt aus eine Linie zu zeichnen, deren Endpunkt nicht direkt, sondern in relativen Koordinaten, mit dem Ausgangspunkt als Ursprung, angegeben ist.
Mit der Prozedur clear löschen Sie einen Polygonzug, indem Sie die Anzahl der Ecken auf Null setzen. Werte, die eventuell noch im Koordinaten-Array enthalten sind, interessieren dabei nicht.
Mit der Prozedur Ecke können Sie einen bestehenden Polygonzug erweitern, indem Sie eine Linie hinzufügen. Die Koordinaten werden im Array abgelegt und der Eckenzähler um eins erhöht. Damit können Sie Vektorgrafiken wie unser Dreieck bereits definieren:

```
clear(dreieck):
ecke (dreieck,10,10);
ecke (dreieck,10,10);
ecke (dreieck,–20,0);
```

Die Variable dreieck muß vorher als Typ polygon deklariert werden.
Mit der Prozedur drawpoly haben Sie das Werkzeug zur Hand, um die definierte Grafik an einem beliebigen Startpunkt (xs/ys) auf den Bildschirm zu bannen.
Die Prozedur translate ist etwas komplexer. Ihre Parameterliste ließe sich in ausführlicher Schreibweise etwa so darstellen:

```
translate (Start-Polygon,
           Ziel-Polygon,
           Linienverlängerung in
           x-Richtung,
           Linienverlängerung in
           y-Richtung,
           Streckungsfaktor in
           x-Richtung,
           Streckungsfaktor in
           y-Richtung);
```



```
100 {*************************************************************************}
110 {*   HISOFT POLYGONS              Version 1.0              06/01/1987    *}
120 {*                              by Alwin Ertl                             *}
130 {*************************************************************************}
140
150 PROGRAM polygons;
160
170  CONST
180   maxecken = 50;
190
200  TYPE
210   polygon = RECORD
220               ecken : integer;
230               position : ARRAY ['x'..'y',1..maxecken] OF integer
240             END; {OF RECORD}
250
260  VAR
270   poly, grosspoly : polygon;
280   i : integer; {alle Variablen nur fuer Demo!}
290
300  FUNCTION sgn (x : integer) : integer;
310   BEGIN
320    IF x = 0
330     THEN
340      sgn := 0
350     ELSE
360      IF x > 0
370       THEN
380        sgn := 1
390       ELSE
400        sgn := -1
410  END;
```

```
420
430  PROCEDURE move (x, y : integer);
440   BEGIN
450    rde := x;
460    rhl := y;
470    user (#bbc0) {gra move absolute}
480   END;
490
500  PROCEDURE drawr (x, y : integer);
510   BEGIN
520    rde := x;
530    rhl := y;
540    user (#bbf9) {gra line relative}
550   END;
560
570  PROCEDURE clear (VAR p : polygon);
580   BEGIN
590    p.ecken := 0
600   END;
610
620  PROCEDURE ecke (VAR p : polygon; x, y :
     integer);
630   BEGIN
640    IF p.ecken < maxecken
650     THEN
660      BEGIN
670       p.ecken := succ (p.ecken);
680       p.position ['x',p.ecken] := x;
690       p.position ['y',p.ecken] := y
700      END
710   END;
720
730  PROCEDURE drawpoly (p : polygon; xs, ys : integer);
740   BEGIN
750    move (xs,ys);
760    FOR i := 1 TO p.ecken DO
770     drawr (p.position['x',i],p.position['y',i])
780   END;
790
800  PROCEDURE translate (p1 : polygon; VAR p2 : polygon;
810                       xadd, yadd, xfak, yfak : integer);
820   VAR
830    i : integer;
840   BEGIN
850    p2.ecken := p1.ecken;
860    FOR i := 1 TO p1.ecken DO
870     BEGIN
880      p2.position ['x',i] := p1.position ['x',i] * xfak
890                             + xadd*sgn(p1.position['x',i]);
900      p2.position ['y',i] := p1.position ['y',i] * yfak
910                             +yadd*sgn(p1.position['y',i])

920     END
930   END;
940
950 BEGIN
960  clear (poly);
970  ecke (poly,20,0);
980  ecke (poly,0,20);
990  ecke (poly,-20,0);
1000 ecke (poly,0,-20);
1010 ecke (poly,20,20);
1020 ecke (poly,-10,10);
1030 ecke (poly,-10,-10);
1040 ecke (poly,20,-20);
1050 page;
1060 drawpoly (poly,100,100);
1070 translate (poly,grosspoly,0,0,5,7);
1080 drawpoly (grosspoly,400,100)
1090 END.
```

Vektorgrafik und Polygonzüge mit erweitertem HiSoft-Pascal: Vergrößern ohne Verzerrung; auch für bewegte Grafiken geeignet.

Hier wird die Variable p1 in p2 übertragen, wobei sich, je nach Wunsch, das Aussehen des Polygons verändert. Wenn Sie Vergrößerungsfaktoren gesetzt haben, werden alle Linien gestreckt. Nach dieser Umwandlung sind sie xfak beziehungsweise yfak mal so lang, wie sie ursprünglich waren.
Außerdem können Sie durch xadd und yadd eine Verlängerungskonstante setzen, zum Beispiel alle Linien um fünf Einheiten länger machen. Die Orientierung des Vektors wird durch das Vorzeichen sgn berücksichtigt.
Einen geschlossenen Polygonzug, bei dem die letzte Linie wieder auf den Ausgangspunkt stößt, erhalten Sie nur dann, wenn die Summe der x-Werte und die Summe der y-Werte jeweils gleich Null sind.

Die Verwendung von Verlängerungskonstanten erzeugt in den meisten Fällen keine geschlossenen Polygonzüge mehr, wie für xadd=yadd=5 leicht an dem Dreiecksbeispiel zu erkennen ist:

10+5/	10+5	15/	15
10+5/	−10−5	15/	−15
−20−5/0		−25/	0
Summe:		5/	0

Streckungsfaktoren hingegen verändern die Summe nicht. Nehmen wir xfak als 5 und yfak als 2 an:

10*5/	10*2	50/	20
10*5/	−10*2	50/	−20
−20*5/	0*2	−100/	0
Summe:		0/	0

Hierdurch wird die Figur also ohne Verzerrung vergrößert. Das Hauptprogramm demonstriert dies an dem bekannten Häuschen, das sich in einer durchgehenden Linie zeichnen läßt. Schaffen Sie es noch auf Anhieb? Wenn Sie eine Figur haben, die sich nicht in einer geschlossenen Linie darstellen läßt, müssen Sie mehrere Polygonzüge an den passenden Stellen auf den Bildschirm bringen. Da diese Figuren sehr schnell gezeichnet werden, eignen sie sich für bewegte Grafiken außerordentlich gut. Die Methode, nach der Sie Bilder bewegen können, haben Sie in der letzten Folge kennengelernt.

UND JETZT: TURBO

Mit dieser Ausgabe endet die vorerst letzte Folge zum Thema HiSoft-Pascal. Doch auch in Zukunft werden die zahlreichen Pascal-Programmierer nicht zu kurz kommen. Die SCHNEIDER CPC-WELT wird sich verstärkt jenem Pascal-Dialekten zuwenden, der inzwischen zum Marktstandard geworden ist: Turbo Pascal. Auch hier freuen wir uns über Ihre Mitarbeit in Form von Programmen und Berichten. *AE* □

SPRECHEN SIE C?

C wie Cäsar nur schneller!

Bei einer Funktion handelt es sich nicht nur um eine Programmroutine im Quelltext, sondern auch ein einzelner Befehl wird als Funktion bezeichnet. Es handelt sich bei letzterem ja ebenfalls um ein Miniprogramm.

Die Main-Funktion stellt das Hauptprogramm dar und ist unbedingt notwendig. Zur Programmausführung – nicht nur beim Compilieren – wird diese Routine zuerst gesucht und abgearbeitet. Deshalb hat der Programmierer auch nicht freie Wahl bei der Namensgebung, die Bezeichnung Main steht fest.

ARNOR-C: ARBEITSDISKETTE ANFERTIGEN

Die Handhabung des Arnor-C-Compilers macht, wenn wir den Leserbriefen Glauben schenken, einige Schwierigkeiten. So leistungsfähig diese Software ist, so kompliziert scheint auch der Umgang damit zu sein. Machen Sie sich deshalb zuerst eine Arbeitsdiskette, wie es im Handbuch auf Seite 3–2 der Einleitung beschrieben ist. Den wohlklingenden Titel Start Of The Day dürfen Sie dabei wörtlich nehmen. Spätere Anpassungen sind nicht ausgeschlossen. Auf eine systemformatierte Diskette gehört noch das File C10CPM 3.EMS. Sie finden es auf der Seite eins der Systemdiskette. Ansonsten kopieren Sie nur noch die aufgeführten Dateien. Bei der Arbeitsdiskette zum Compiler können Sie auf CP/M verzichten. Zu diesem Zeitpunkt sollte ja schon längst das Betriebssystem geladen sein. Kopieren Sie deshalb sämtliche Dateien von Seite zwei der Originaldiskette auf die Arbeitsdiskette. Aber Vorsicht: Bei unserer Lieferung war die Seite zwei auf Seite A der Diskette. Falls dies so üblich ist, müssen Sie sich nach dem Etikett richten.
Gestartet wird Arnor C mit der Start Of The Day-Diskette und der Eingabe von ICPM. Sie kommen damit automatisch in den Editor APED und können von dort aus auch den Compiler oder Linker aufrufen. Ideal ist es, wenn Sie über eine zweite Diskettenstation verfügen, denn sonst müssen Sie zum Diskjockey werden.
Arnor ist kompatibel zu den üblichen Compilern, vor allem aber zum Standard, den die Herren Kernighan und Ritchie geschaffen haben.
Wenn Editor, Compiler und Linker ihre Arbeit getan haben, haben Sie drei verschiedene Files auf der Diskette, die alle eine andere Endung tragen. Beispiel:

TEST.C
(abgespeicherter Quellcode)
TEST.L
(vom Compiler erzeugt)
TEST.O
(vom Linker erzeugt und lauffähig).

Die Endung C bezeichnet einen Quellcode, den Originaltext. Sie mußten beim Abspeichern darauf achten, daß diese Extension angegeben wird. Mit L wird das Link-File dieser Datei bezeichnet. Der Compiler fügt die

Abbildung 1: Standard der Steuerzeichen nach K. & R.

```
\n = newline
     (neue Zeile)
\r = carriage Return
     (Anfang der aktuellen Zeile)
\t = tab
     (Tabulatorstelle)
\b = backspace
     (ein Zeichen zurück)
\f = form feed
     (neue Seite)
\\ = backslash
\' = single quote
     (einfaches Anführungszeichen)
\ddd = octal constant
       (ddd = Oktalzahl)
\0 = null Character
```

Abbildung 2: Zusätzliche Steuerzeichen in anderen Interpretern

```
\" = double quote
     (Anführungszeichen)
\v = vertikal tab
     (Vertikaler Tabulator)
\xhh = hexadeziml Constant
       (xF1 = Hexadezimalzahl)
\c = conditional newline
```

Abbildung 3: Demolisting für Steuerzeichen

```
main ()
{
printf ("Text 1 in erster Zeile");
  /* kein Steuerzeichen - Kein Zwischenraum */
printf ("Text 2 ohne Zwischenraum\n");
  /* neue Zeile durch /n */
printf ("Text 3\t");
  /* Weiter auf Tab.-Stelle */
printf ("Text 4");
}
```

Kennung selbst hinzu. Mit dem Link-Programm allein ist nichts anzufangen, es ist ein Mittelstück zwischen dem Source-Code und dem Objektcode. Letzterer wird mit O auf der Diskette abgelegt und ist wirklich lauffähig. Dabei benötigen Sie wiederum einen C-Interpreter, hin und wieder auch als Runtime-Modul bezeichnet.

PRINTF UND SEINE FOLGEN

Kommen wir wieder zur Programmierung. Mit printf wurde bereits der erste Befehl erklärt. Er bewirkt eine Bildschirmausgabe, wobei der Text, solange es sich nicht um eine Variable handelt, in Klammern und Anführungszeichen stehen muß, also:

printf("Dies ist der Bildschirmtext");

Beachten Sie auch das Semikolon, welches zum Abschluß eines Befehles zwingend notwendig ist. Große Bedeutung kommt dabei der formatierten Ausgabe zu. Das heißt, in C muß schon bei der Eingabe des Quelltextes an die Position der Zeichen gedacht werden. Dabei ist nicht einmal ein Zeilenvorschub, wie in BASIC üblich, selbstverständlich. Der Programmierer muß entsprechende Steuerzeichen anfügen.

Abbildung 4: Demolisting zur Eingabe

```
main ()
{
int weiter;
  /* Menueaufbau */
printf("1. Menuepunkt\t");
printf("2. Menuepunkt\n");
printf("3. Menuepunkt\t");
printf("4. Menuepunkt\n\n");
  /* Eingabe der Zahl */
scanf("%d", &weiter);
  /* Verzweigungsmöglichkeit */
printf("\nGewaehlt: %d", weiter);
}
```

Nach dem C-Standard von Kernighan und Ritchie sollte ein Compiler sämtliche in *Abbildung 1* aufgeführten Möglichkeiten berücksichtigen. Andererseits ist aber nichts vollkommen, daher wurde auch diese Tabelle überarbeitet.
Viele Compiler bieten über den Standard hinausreichende Steuerzeichen, die nicht unerwähnt bleiben sollten. *Abbildung 2* zählt auf, was Sie von einem modernen C-Interpreter erwarten dürfen. Über die Anwendung dieser Steuerzeichen gibt das Listing in *Abbildung 3* Auskunft. Zu beachten ist lediglich, daß sie innerhalb der Anführungszeichen des Textes liegen müssen. BASIC-Programmierer kennen dies bereits von den dort üblichen Controll-Codes.
An dieser Stelle noch etwas zur optischen Aufbereitung eines Quelltextes. Da es dem Compiler relativ egal ist, welche Tabulatoren Sie verwenden, sollten Sie das Listing so übersichtlich wie möglich gestalten. Das Einrücken von Kommentarzeilen ist nur Mittel zum Zweck.
Es ist auch guter Stil, wenn die geschweiften Klammern in einer eigenen Zeile untergebracht und so die Programmroutinen klar erkennbar werden.
Wie jede Compilersprache nimmt es C mit den Variablentypen sehr genau. Das bringt den Nachteil mit sich, daß Sie fast in jeder Unterroutine aufs neue

VOR- UND NACHTEILE DER VARIABLEN-BEHANDLUNG UNTER C

deklarieren müssen. Es hat aber auch Vorteile: Es fällt zum einen leicht, globale Typen festzulegen. Zum anderen können Sie sich darauf verlassen, daß Werte nur innerhalb des Unterprogrammes Gültigkeit haben. Es hat eben jedes Ding seine zwei Seiten.
Versuchen wir jetzt, mit dem zweiten Listing aus *Abbildung 4* in die Praxis einzusteigen.

ERSTE BEDINGUNG: VARIABLENDEKLARATION IN JEDEM UNTERPROGRAMM

Egal, welche Text- oder Zahlentypen innerhalb einer C-Funktion verlangt werden, sie müssen zuerst definiert werden. In der dritten Zeile des Listings bereiten wir den Compiler darauf vor, daß er es in dieser Routine mit Integerzahlen zu tun hat. Dabei handelt es sich um Ganzzahlen aus dem Bereich von –32768 bis +32767, sofern von einem Acht-Bit-Rechner die Rede ist. BASIC-Programmierer kennen den Begriff Integerzahl sicher schon.
Von der fünften bis zur neunten Zeile wird ein Menü aufgebaut, welches hier nur als Beispiel gelten kann. Eine neue Funktion steht in Zeile elf, die Scanf-Funktion. Sie entspricht etwa der LINE INPUT-Anweisung bei BASIC. Der Ausdruck innerhalb der Klammern besagt nichts anderes, als daß eine Integerzahl erwartet und diese der Variablen weiter zugewiesen wird.
Da keine Unterprogramme programmiert wurden, haben wir nur noch eine Zeile angefügt, die bestätigen kann, welche Zahl gewählt wurde. Sollte es sich um größere Programme handeln, müßte hier eine Verzweigung stattfinden.
Soviel für heute. Mit den Steuerzeichen sollten Sie etwas experimentieren. Sie sind das einzige Mittel, dem Bildschirmausdruck etwas Format zu verleihen. Die Variablen werden über den vorgestellten Typ der Integerzahlen das nächste Mal erweitert. Hierzu gehören auch die mathematischen Zeichen, die sogenannten Operatoren. *GS* □

AM FALSCHEN PLATZ GESPART

Hilfe, mein Laufwerk ist kaputt!

Als ich mir meinen CPC vor zwei Jahren gekauft hatte, wurde mir der hohe Preis der Drei-Zoll-Disketten nicht so recht bewußt. Aber im Laufe der Zeit wuchs meine Programmsammlung immer mehr an und damit auch der Diskettenbestand. Und das Loch in der Brieftasche wurde immer größer.

Irgendwann war mir der Preis für die Drei-Zoll-Disketten zu hoch. Zufällig fiel mir in jenen Tagen eine Anzeige in die Hände, in der ein 5,25-Zoll-Zweitlaufwerk angeboten wurde. Es sollte 350 Mark kosten, dafür aber mit den Disketten im 5,25-Zoll-Format, zu einem Stückpreis von etwas über einer Mark, laufen.
Bei dem damaligen Preisunterschied von etwa acht Mark für jede Diskette konnte ich mir ausrechnen, wieviel Geld ich, bei meinem Diskettenbestand von etwa 300 Stück, sparen würde. Enorm, was im Geldbeutel noch übrigbliebe.

TEURER ALS ERWARTET

Das versprach jedenfalls die Anzeige. Verschämt verschwiegen wurde jedoch, daß das neue Laufwerk keinen eigenen Controller besitzt und vom eingebauten CPC-Controller mitverwaltet wurde.
Ich hatte also, wie bei den kleinen Disketten auch, 40 Spuren mit je neun Sektoren und einer Kapazität von 178 KByte im DATA-Format zur Verfügung. Das störte mich noch nicht sonderlich.
Der Ärger begann, als das Laufwerk per Nachnahme eintraf. Von wegen 350 Mark, es kam noch etliches hinzu: Porto, Verpak kung, Mehrwertsteuer. Jetzt war der Preis bereits auf über 400 Mark gestiegen.
Auch das hätte ich noch geschluckt. Aber es nahm kein Ende: Als ich das Laufwerk ansprechen wollte, indem ich es als Zweitlaufwerk mit !B aufrief, meldete der CPC „Disk missing in B", obwohl eine Diskette im Laufwerk war. Ich schaltete mit !A auf das eingebaute Laufwerk meines 664 zurück und wollte dort ein Programm aufrufen. Auch diesmal vermißte AMSDOS eine Diskette.
Also, das Zweitlaufwerk abgeklemmt und im eingebauten Laufwerk eine Diskette eingelegt: Alles normal. Nach dem Anstecken des zweiten wieder der gleiche Effekt. Egal, welches Laufwerk ich ansprechen wollte, ich bekam immer die stereotype Antwort: Disk missing. Außerdem lief das Laufwerk ständig und war nicht mehr abzustellen.

AHNUNGSLOSE FACHLEUTE

War es kaputt? Ich wollte das Geschäft anrufen, bei dem ich die Bestellung aufgegeben hatte. Doch ich landete beim Anrufbeantworter, der mir mitteilte, daß das Büro nur abends von 18 bis 20 Uhr besetzt sei – ein Feierabendversand-Geschäft offenbar.
Nach etlichen Versuchen hatte ich endlich den richtigen Ansprechpartner am Hörer: „Ich kann mir gar nicht erklären, was fehlen könnte. Die Laufwerke arbeiten alle einwandfrei."
Bei mir lief es jedenfalls nicht. Doch dann fiel dem Fachmann ein: „Ach ja, es kann sein, daß die Brücke xy verkehrt eingelegt ist." Ich solle das Gehäuse aufschrauben und an den Brücken ein bißchen herumprobieren.
Gesagt, getan. Kaum hatte ich den Deckel abgeschraubt, lief das Laufwerk einwandfrei. Den Deckel schnell wieder daraufgemacht, und prompt kam die Fehlermeldung. Was nun? Das Laufwerk ohne Deckel betreiben? Das kann ja wohl nicht wahr sein.

REPARATURVERSUCH ERFOLGLOS

Ich hatte eine Idee: Vielleicht berührte ein Bauteil das Gehäuse und verursachte einen Kurzschluß? Ich schnitt mir ein passendes Stück Isoliermaterial zurecht, legte es oben auf die Platine und schraubte vorsichtig den Deckel wieder zu.
Hurra, es lief. Fehler gefunden, alles okay.
Ich lehnte mich entspannt zurück. Nun noch eine Tasse Kaffee, und dann kann's losgehen. Als ich aus der Küche zurückkam, meldete der CPC: Disk missing.
Vor lauter Wut schaltete ich alles aus. Am nächsten Tag wieder ein Versuch: alles in Ordnung.
Ich schrieb mein eigenes Dateiprogramm so um, daß die Daten immer auf das Zweitlaufwerk abgespeichert wurden. Es funktionierte hervorragend.
Es läutete. Mein Nachbar kam, um mein neues Laufwerk zu bestaunen. Stolz wollte ich ihm alles vorführen. Nur leider: Disk missing.
Wutschnaubend rief ich meinen Fachmann an, doch „...das Büro ist nur in der Zeit...". Endlich, zwei Wochen später, erreichte ich ihn. Ich sollte das Laufwerk wieder einsenden.
Nach drei kurzen Wochen kam es zurück. Es war dasselbe Laufwerk, ich erkannte es an einem Kratzer im Gehäuse. Jetzt lief es problemlos. Allerdings fehlte die Information, was defekt war. Der Sache wollte ich auf den Grund gehen. Also wieder ans Telefon. Es sei nur die 5-Volt- (oder war es die 12-Volt-?)Versorgung nicht korrekt geklemmt gewesen, hieß es.
Was denn mit meinen Auslagen für die vielen Telefongespräche, für das Porto sei, wollte ich wissen. Höhnisch klang es aus dem Hörer: Durch das Öffnen des Laufwerks hätte ich jeden Garantieanspruch verloren. Daß es überhaupt instandgesetzt worden sei, hätte ich nur der grenzenlosen Kulanz der Firma zu danken.
Wieder etwas dazugelernt. Doch was sollte es, das Laufwerk lief ja. Bis zum vorletzten Wochenende jedenfalls. Denn da stiegen plötzlich kleine, hübsch anzusehende

RAUCHSIGNALE

Qualmwolken auf. Sicherheitshalber habe ich schnell alles ausgemacht und das Gerät geöffnet. Ich entdeckte, daß die eine Diode der Siebkette in der Stromversorgung ganz anders aussah als die anderen drei. Sie war eigenartig verformt und auch verfärbt. Also wieder den Händler gefragt. Die Antwort: „Was für ein Laufwerk haben Sie? Den Typ XYZ? Der wird schon lange nicht mehr produziert, dafür gibt es keine Ersatzteile. Außerdem ist das Nachfolgemodell viel besser. Wenn ich Ihnen einmal kurz vorrechnen darf, was Sie damit sparen....."
Ganz schnell legte ich den Hörer auf. An diesem Tag habe ich erwogen, die Computerei sein zu lassen und lieber wieder mit Murmeln zu spielen. *JE* □

Von PIP bis TYPE

CP/M ist – so einfach es klingt – nichts anderes als eine Software, die den Computer steuert. Aber der Dolmetscher zwischen Mensch und Maschine verlangt auch dem Anwender einiges ab, und nicht immer ist das Handbuch von Schneider eine Hilfe. Weiter geht es deshalb mit einer ausführlichen Beschreibung des Betriebssystems, gespickt mit Tips und guten Ratschlägen.

Neben den Grundbegriffen wurde bereits der DIR-Befehl mitsamt seiner Optionen, den Jokern und Wild cards erwähnt. Es handelt sich dabei um die Fragezeichen, die als Ersatz für ein Zeichen, und um die Sterne, die für ganze Buchstabengruppen stehen. Vielleicht haben Sie ein wenig herumprobiert, und unter Umständen kam es zu einem Lesefehler, obwohl die Diskette in Ordnung war, wie ein zweiter oder dritter Versuch zeigte. Bevor Zweifel an der Funktionstüchtigkeit des CPC oder des CP/M-aufkommen, lassen Sie sich sagen, daß der Lesefehler eine ganz natürliche Ursache hat.

Ohne Logbuch hat der Controller Orientierungsschwierigkeiten

Der Controller kann schlicht und einfach das Logbuch nicht lesen. Der Begriff aus der christlichen Seefahrt hat auch in die Computertechnik Einzug gehalten; um seine Bedeutung zu erklären, muß allerdings etwas weiter ausgeholt werden.

Noch bevor Sie auf der Diskette lesen (durch DIR) oder etwas schreiben können, muß der Controller wissen, ob und welche Diskette eingelegt ist, bzw. in welchem Format diese vorliegt. Andernfalls passiert überhaupt nichts, es kommt zu einem Lesefehler. CP/M verlangt ein Logbuch, in dem als wichtigster Eintrag das Diskettenformat festgelegt ist.

Den Lesevorgang des Kursbuches nennt man Einloggen, und nur in den wenigsten Fällen merken Sie überhaupt etwas davon. Jedesmal, wenn Sie auf die Benutzerebene von CP/M kommen, etwa nachdem ein Programm abgearbeitet wurde, holt sich der Controller erst einmal die notwendigen Informationen. Waren Sie aber zu schnell und haben die Diskette ausgetauscht, dann stimmt plötzlich etwas nicht mehr, das System meldet Bdos Error on A &oder B). Wenn es sich bei der neuen Diskette ebenfalls um eine CP/M Diskette handelt, dann können Sie einfach durch R (für Retry) einloggen, und der Fall ist erledigt.

Wie aber, wenn ein Utility, etwa der nachladbare Befehl Filecopy, den Wechsel der Diskette verlangt? Machen Sie sich darum keine Sorgen. Diese Programme sind so gestaltet, daß hier zwei Logbücher berücksichtigt werden. Erst, wenn das Programm abgearbeitet ist, werden Sie wieder mit der Problematik konfrontiert, indem Sie aufgefordert werden, eine System-Diskette einzulegen, weil der Controller nicht mehr weiter weiß.

Dateiausdruck mit TYPE ist Charaktersache

Falls es zuviel Theorie war – wie gesagt, Sie bekommen ja außer einer gelegentlichen Fehlermeldung nichts mit – soll kurz auf den Befehl TYPE hingewiesen werden; endlich wieder etwas Praxis! Mit diesem nachladbaren Befehl können Sie jede Datei auf den Bildschirm bringen, ohne zuvor ein Anwenderprogramm geladen zu haben. Selbst diese Programme werden Ihnen aufgelistet. Die Anweisung ist im übrigen identisch mit dem MS-DOS-Befehl.

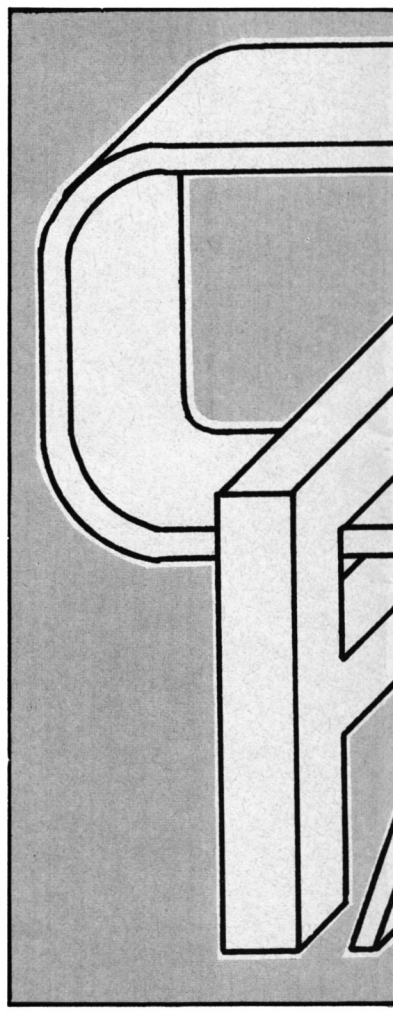
Für den IBM steht zwar noch PRINT zur Verfügung, damit die Datei auf den Drukker kommt, aber auch beim CPC herrscht unter CP/M daran kein Mangel. Betätigen Sie einfach den Schalter für die Druckerausgabe (Ctrl und P), bevor Sie TYPE benutzen.

In den meisten Fällen werden Sie von dem, was Sie auf dem Bildschirm oder auf Papier sehen, allerdings enttäuscht sein. Kaum ein Anwenderprogramm speichert seine Dateien als reinen ASCII-Code ab. Meist sind Steuerzeichen vorhanden, bei einer Textverarbeitung etwa Absatzkennzeichen oder Schriftänderungen. Diese Steuerzeichen ergeben keinen Sinn, ja schlimmer noch: Wenn sie unter der Codenummer 32 (hexadezimal: 2C) liegen, haben sie auch Wirkung auf die Ausgabe. Auch bei COM-Dateien wird es Schwierigkeiten geben.

Trotzdem stimmt unsere Behauptung, der Ausdruck (fast) jeder Datei sei möglich. Welche Wirkung die Steuerzeichen haben, ist ja nicht das Problem des TYPE-Befehls. Im schlimmsten Fall stürzt der Rechner ab. Aber es gibt auch mal recht nette Ergebnisse, zum Beispiel beim CP/M 2.2 von Vortex. Wer einmal CP/M im Modus 1 benutzen will, der sollte folgenden Befehl (das Betriebssystem muß bereits aktiviert sein) benutzen:

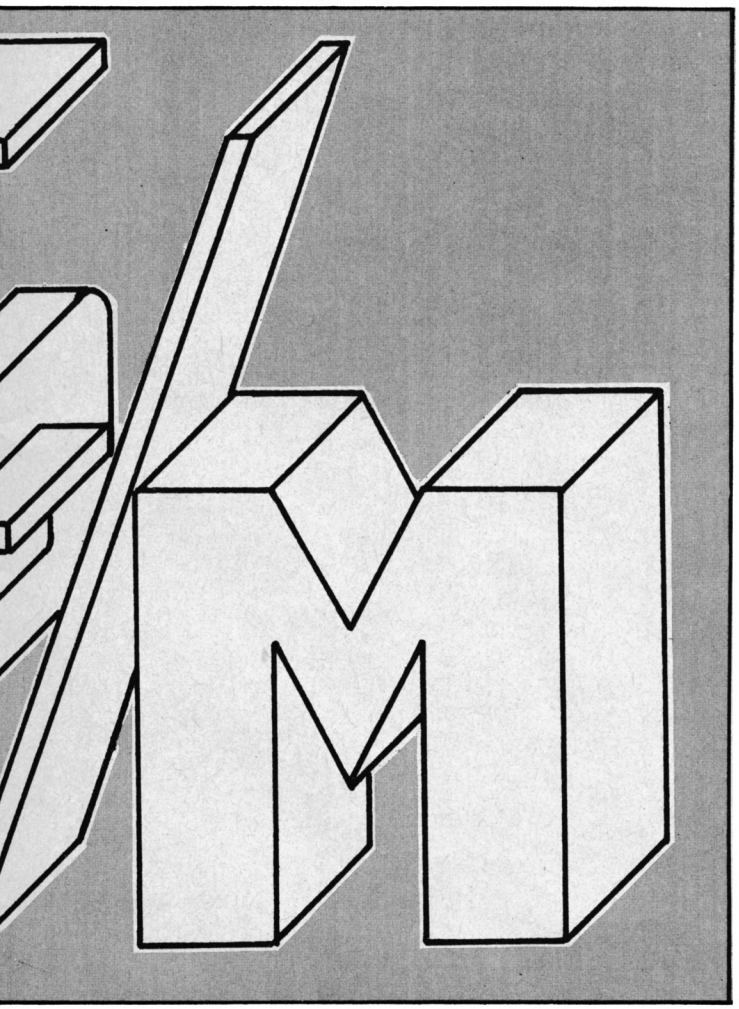
TYPE FORMAT.COM

Wenn dann ein paar unverständliche Zeichen erschienen sind, drücken Sie einfach ENTER, und Sie arbeiten mit der 40-Zeilen-Darstellung. Die kleine Spielerei beweist, daß die Steuerzeichen durchaus wirksam sind, auch wenn sie nicht dargestellt werden können. Bei diesen Versuchen sollte der Drucker aber nicht angeschaltet sein. Da hier eventuell ein oder mehrere Steuerzeichen einen Seitenvorschub bewirken, wäre es schade um das Papier.



Aber der wirkliche Nutzeffekt von TYPE liegt natürlich nicht darin, ein möglichst großes Chaos zu erzeugen. Solange Sie auf ASCII-Dateien zugreifen, haben Sie einen einwandfreien Ausdruck auf dem Bildschirm oder auf dem Drucker.

Zur Erinnerung noch einmal: ASCII-Dateien sind Datensätze, bei denen der Character-Code eines jeden Zeichens nicht größer als 128 und nicht kleiner als 32 ist.



Probieren Sie den Befehl doch einmal mit einem BASIC-Programm aus, welches zuvor mit SAVE"Name",A abgespeichert wurde. Oder verwenden Sie eine Textdatei (etwa von Tasword), bei der keine Steuerzeichen benutzt wurden.

Da wir gerade bei den Kontroll-Befehlen sind, soll die spärliche Sammlung, die unter CP/M zur Verfügung steht, noch einmal aufgelistet werden. Die Steuerzeichen werden in Verbindung mit der Ctrl-Taste aktiviert.

Minimalvorrat an Control-Codes

Ctrl + P haben Sie ja schon kennengelernt. Das Steuerzeichen dient als Ein- und Ausschalter für die Druckerausgabe (P für Print).

Ctrl + S erlaubt es Ihnen, den momentan ablaufenden Ausgabevorgang anzuhalten. Längere Dateien, die mit TYPE auf dem Monitor erscheinen, können durch Ctrl und S angehalten werden. Eine beliebige Taste setzt die Ausgabe fort.

Ctrl + C bricht eine Befehlsausführung ab. Voraussetzung ist allerdings, daß ein Programm diese Tastenkombination überhaupt zuläßt bzw. abfragt, wie etwa TYPE. Nach Ctrl + C befinden Sie sich wieder auf der Eingabe-Ebene des Betriebssystems.

Wenn wir an dieser Stelle sagen, dies sei es schon gewesen, dann wecken wir den Widerspruch einiger „alter Hasen". Tatsächlich gibt es schon noch ein paar Ctrl-Befehle, jedoch werden sie erst in bestimmten Programmen wichtig.

Besonders gefürchtet, wenn auch zu Unrecht, ist der Editor ED. Seine umfangreichen Leistungen verdienen jedoch in der CPC-WELT eine gesonderte CP/M-Folge, zumal er im Handbuch zur Diskettenstation von Schneider schnöde übergangen wird.

Aus alt mach neu – RENAME

Es ist eine Selbstverständlichkeit, daß nur mit einer Kopie der Systemdiskette gearbeitet wird. Wer sich daran hält, der darf sich dann auch einmal den Spaß machen und die fantasielosen Programmnamen austauschen. Die REN-Anweisung leistet dabei Hilfe. Dabei ist die Syntax jedoch etwas verwirrend, wie ein Beispiel zeigen soll:

REN neu.BAS=alt.BAS

Verwirrend deshalb, weil ja zuerst der neue Name genannt wird, der Befehl sich aber aus RENAME, also Umbenennen, ableitet. Aber damit läßt sich leben. Weiterhin ist auch das Gleichheitszeichen zu berücksichtigen.

Wer auch mit MS-DOS arbeitet (hier steht ein Leerzeichen zwischen altem und neuem Dateinamen), wird gelegentlich durcheinander kommen. Probieren Sie es ruhig einmal aus, achten Sie aber darauf, daß auch die neuen Dateikennungen noch Sinn ergeben. Sonst kann es Ihnen passieren, daß das Format-Programm abläuft, weil Sie es unter einem unsinnigen Namen gestartet haben.

Eine der mächtigsten CP/M-Kommandos soll am Schluß der heutigen Folge behandelt werden. PIP ist die Abkürzung von Peripheral Interchange Program (Datenaustausch zwischen Peripheriegeräten, nicht, wie vielfach angenommen, von Pipeline. Trotzdem läßt es sich so besser merken. Wenn es um Einzelheiten geht, dann vergessen Sie einmal schnell das Handbuch, denn dort stehen nur ein paar Stichwörter, die bestenfalls neugierig machen, aber keine Hilfe bieten.

Pipeline zur Peripherie – Peripherial Interchange Programm

Wir wollen nicht den gleichen Fehler in der CPC-WELT machen, und PIP wird uns deshalb im wesentlichen in der nächsten Folge beschäftigen. Vorbereitend nur ein paar Stichwörter:

Peripherie ist für das Betriebssystem ja auch die Tastatur oder etwa eine serielle Schnittstelle (ja, ich weiß: keine beim CPC vorhanden – kann aber nachgekauft werden). Auch der Drucker oder der Bildschirm gehört dazu. Zwischen all diesen Ein/Ausgabeeinheiten können Dateien transferiert werden. Teilweise ist auch hier die Einschränkung zu machen, daß Steuerzeichen das Ergebnis verfälschen oder zunichte machen. Wir haben das Problem ja schon bei der Ausgabe auf den Bildschirm (mittels TYPE) kennengelernt, und nichts anderes passiert ja mit PIP, wenn als Ziel der Monitor angegeben wurde.

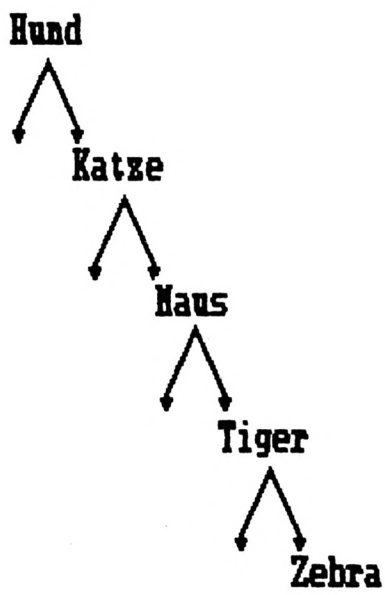
Beim Kopieren der Programme (bei PIP gilt auch der Druckvorgang als Kopie eben auf den Drucker) dürfen sogar mehrere Files aneinander gehängt werden. Im Fachjargon spricht man von Verknüpfungen. Und wem dies noch nicht komfortabel genug ist, der darf Joker und Wild cards einsetzen. Der Fantasie sind keine Grenzen gesetzt, und es können endlich einmal alle Textdateien in nur einem File kopiert werden (als Ablage). Mit PIP hat jeder das richtige Werkzeug zur Hand. Wie damit umzugehen ist, wird in der nächsten Folge besprochen. ■ *GS*■

BINÄRBÄUME IN BASIC

Auch in Basic können Sie Bäume pflanzen

Ein binärer Baum sei eine Datenstruktur, die nur in Programmiersprachen wie etwa Pascal realisierbar sei. So lautet eine oft vorgebrachte Behauptung. Daran ist nicht viel Wahres. Auch in BASIC können Sie Bäume pflanzen.

Abbildung 1



Reihenfolge der Eintragungen:
Hund - Katze - Maus - Tiger - Zebra

Die Struktur eines Binärbaumes ist abhängig von der Reihenfolge der Eingaben. Dies verdeutlichen die beiden Abbildungen auf dieser Seite.

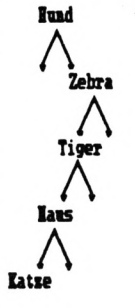
Ein Binärbaum ist eine Struktur aus theoretisch beliebig vielen Datensätzen, die jeweils ein Datenfeld und zwei Zeigerfelder enthalten. Die Zeiger weisen dabei nach links und rechts. Diese beiden Richtungen wurden von den Informatikern vereinbart; es würde nichts dagegen sprechen, wenn Sie sie nach oben und unten weisen ließen.
Doch wie so vieles in der Computerei dient auch die Baumdarstellung nur zur Veranschaulichung.
In der Realität geben die Zeiger zwei Speicherstellen an; und die finden Sie eben nicht kreuz und quer in Ihrem Computer verteilt, sondern schön hintereinander aufgereiht.
Dazu kommt noch etwas Erstaunliches: Informatiker-Bäume haben die Wurzel immer oben. Das scheint ein ganz widernatürlicher Zustand zu sein; doch wenn man die theoretisch unendliche oder zumindest im voraus unbekannte Ausdehnung eines solchen Gebildes bedenkt, liegt der Sinn dieser Regelung auf der Hand.

Wenn Sie mit der Wurzel oben auf einem Stück Papier beginnen, haben Sie nach unten hin viel Platz. Umgekehrt müßte der Platzbedarf kalkuliert werden. In viel stärkerem Maße gilt dies bei Verwendung eines Textverarbeitungsprogramms.
Die Wurzel unterscheidet sich in ihrem Aussehen nicht von den anderen Datensätzen im Baum. Der Inhalt des Datenfeldes (auch Schlüsselfeld, Schlüsseleintrag oder key entry genannt) ist meist ein Dummy-Datum, also ein Datum ohne Bedeutung. Die beiden Zeiger (pointer) weisen in einem leeren Baum zunächst auf nichts.
Wollen Sie neue Einträge in den Baum aufnehmen, so geschieht dies nach einem bestimmten Schema. Zum Beispiel können Sie festlegen, daß der Datensatz mit dem jeweils kleineren Schlüsselfeld den linken und der mit dem größeren Schlüsselfeld den rechten Ast hinabwandern soll. In unserem Demonstrationsprogramm wurde das so realisiert. Gegen die umgekehrte Regelung ist jedoch nichts einzuwenden. Dieses Hinabwandern entlang eines Astes läßt sich am besten in einer Grafik darstellen. Die *Abbildungen 1 und 2* zeigen, daß die Reihenfolge, in der Eintragungen in einen Binärbaum erfolgen, von erheblicher Bedeutung für dessen Aussehen ist.

ES MUSS NICHT IMMER PASCAL SEIN

Ein solcher Baum kann auch in BASIC realisiert werden. Natürlich müssen Sie sich in dieser Sprache auf eine maximale Größe festlegen. In anderen Programmiersprachen, wie etwa Pascal, ist dies augenscheinlich nicht nötig.

Abbildung 2



Reihenfolge der Eintragungen:
Hund - Zebra - Tiger - Maus - Katze

Jedoch wird auch dort eine bestimmte maximale Ausdehnung vorgegeben: durch das RAM Ihres Rechners.

Das Argument, Pascal ermögliche eine effizientere Speicherplatznutzung, läßt sich ebenfalls ad absurdum führen. Es ist zwar unbestritten, daß ein Binärbaum in Pascal nur den seiner Größe entsprechenden Speicherplatz in Anspruch nimmt. Um eine sinnvolle Anwendung zu ermöglichen, müssen Sie aber in jedem Fall soviel RAM reservieren, daß der für das zu lösende Problem größte denkbare Baum darin Platz findet. Auch die Behauptung, durch die in Pascal mögliche Rekursion sei der Aufbau einer solchen Datenstruktur einfacher zu bewerkstelligen, kann nicht ohne Vorbehalt akzeptiert werden.

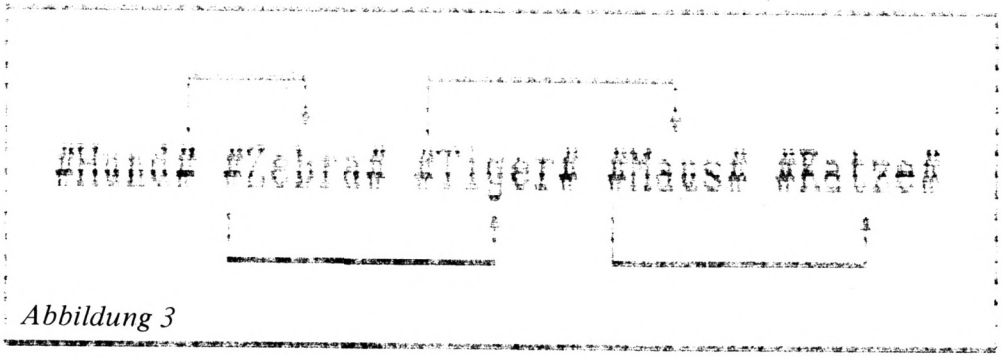


Abbildung 3

Für den Programmierer bedeutet diese Möglichkeit sicherlich weniger Aufwand. Die Menge an Speicherplatz, die dadurch unnötigerweise verbraucht wird, sollte jedoch auch berücksichtigt werden. Der CPC hat nur 64 KByte, mit denen Sie gut haushalten müssen.

DIE BAUMSTRUKTUR

Deshalb soll versucht werden, einen Binärbaum in einer nicht-rekursiven Sprache, nämlich BASIC, aufzubauen. Wir bestimmen zunächst eine maximale Größe. Sie soll bei 100 liegen und findet ihren Niederschlag in den DIM-Anweisungen der Zeile 130. Hier ist begriff$ das oben erwähnte Schlüsselfeld, links und rechts sind die beiden Zeiger. In Zeile 140 bestimmen wir einen Eintrag für die Wurzel des Baumes. Das Schlüsselfeld ist mit "zzz" vielleicht nicht gerade glücklich gewählt, da dies der größte mögliche Eintrag ist, sofern Sie sinnvolle Begriffe eingeben.
Der anzulegende Binärbaum kann daher nur nach links wachsen. Sie können diese Vorbelegung natürlich ändern; doch unser Baum soll Ihnen schließlich Suchoperationen erleichtern und keinen Schönheitspreis bei der Gartenbauausstellung gewinnen.
Die Variable zähl gibt an, wieviele Eintragungen sich bereits im Baum befinden. Obwohl BASIC dies automatisch macht, belegen wir sie sicherheitshalber mit Null vor.

AUFNAHME NEUER KNOTEN

Das nachfolgende Menü ist einfach zu verstehen, so daß wir uns gleich Zeile 280 zuwenden. Hier finden Sie die Routine, die einen Begriff in den Binärbaum einträgt.
Nach dem Einlesen des Schlüsselfeldes in die Variable neu$ wird zunächst der Zähler um Eins erhöht. Danach wird stelle auf Null gesetzt. Dieser Wert zeigt an, wo ein neuer Zeiger eingerichtet werden soll. Das Schlüsselfeld selbst wird auf jeden Fall an der nächsten freien Position im Array eingetragen.
Jetzt muß nur noch der richtige Wert für stelle ermittelt werden. Zeile 340 überprüft zunächst, ob der neue Wert kleiner oder größer ist als das gerade betrachtete Schlüsselfeld. Entsprechend wird stelle der Wert des linken oder rechten Zeigers zugewiesen.
Ist dieser Zeigerwert nicht gleich Null, so ist der Pointer bereits belegt, da er auf einen anderen Datensatz weist. Daher muß im Baum eine weitere Ebene nach unten gegangen werden. Eine entsprechende Überprüfung findet in Zeile 350 statt.
Wurde eine freie Stelle (ein freier Knoten, wie die Informatiker sagen) im Baum gefunden, so ermittelt Zeile 360 noch einmal, ob der linke oder der rechte Zeiger des übergeordneten Knotens dorthin weisen soll. Mit der Änderung dieses Pointers ist der neue Datensatz in den Baum übernommen.
Eine Darstellung des Binärbaumes aus *Abbildung 2* finden Sie in einer modifizierten Form in *Abbildung 3* wieder. Dort ist das Array so abgebildet, wie es der Darstellung im Speicher am nächsten kommt. Aus Gründen der Übersichtlichkeit wurden die Pfeile, die die Pointer darstellen sollen, abwechselnd über und unter den eigentlichen Daten eingetragen.

Speicherdarstellung des Binärbaumes

BLITZSCHNELLE SUCHOPERATIONEN

Binärbäume dienen in erster Linie dazu, eine schnelle Suche nach einem bestimmten Datensatz zu ermöglichen. Diese Aufgabe übernimmt das Unterprogramm ab Zeile 380.
Nach dem Einlesen des Suchbegriffes in such$ wird stelle wieder auf Null gesetzt. Startpunkt für die Suche ist immer die Wurzel des Baumes. Das ist leicht einzusehen, da andernfalls nicht die gesamte Datenmenge erfaßt werden könnte.
In Zeile 420 wird eine

```
100 'Binaerbaum                                    (11EA)
110 '- Versuchsprogramm -                          (1B38)
120 '                                              (07FB)
130 DIM begriff$(100),links(100),r
echts(100)                                         (2F72)
140 begriff$(0)="zzz":links(0)=0:r
echts(0)=0                                         (3536)
150 zaehl=0                                        (0F1B)
160 INK 0,0:PAPER 0:INK 1,15:PEN 1
:BORDER 0:MODE 1                                   (1BA9)
170 PRINT"Binaerbaum":PRINT:PRINT:
PRINT                                              (1890)
180 PRINT"1 = Begriff eintragen":P
RINT                                               (1F47)
190 PRINT"2 = Begriff suchen":PRIN
T                                                  (1C8B)
200 PRINT"3 = Baum speichern":PRIN
T                                                  (1C5F)
```

```
210 PRINT"4 = Baum laden":PRINT      <18DE>
220 PRINT"5 = Ende":PRINT:PRINT      <142C>
230 INPUT"Ihre Wahl: ",wahl          <1B0D>
240 IF wahl<1 OR wahl>5 THEN 160     <1D3E>
250 CLS:ON wahl GOSUB 280,380,460,
510,560                              <238F>
260 PRINT:PRINT"Unterprogramm beende
t.":PRINT"Zurueck zum Menue mit
beliebiger Taste.":CALL &BB06        <5026>
270 GOTO 160                         <0910>
280 'Eintragen                       <1026>
290 INPUT"Neuer Begriff: ",neu$      <1E0F>
300 zaehl=zaehl+1                    <18BB>
310 PRINT"Dies ist Begriff Nummer
";zaehl                              <294F>
320 stelle=0                         <10E5>
330 begriff$(zaehl)=neu$:links(zae
hl)=0:rechts(zaehl)=0                <4BF1>
340 IF neu$<begriff$(stelle)THEN e
intrag=links(stelle)ELSE eintrag=r
echts(stelle)                        <62DA>
350 IF eintrag<>0 THEN stelle=eint
rag:GOTO 340                         <2C5F>
360 IF neu$<begriff$(stelle)THEN l
inks(stelle)=zaehl ELSE rechts(ste
lle)=zaehl                           <5E79>
370 RETURN                           <060B>
380 'Suchen                          <0D55>
390 INPUT"Gesuchter Begriff: ",suc
h$                                   <2361>
400 PRINT"Ich suche den Begriff ";
such$                                <263A>
410 stelle=0                         <109A>
420 IF such$<begriff$(stelle)THEN
stelle=links(stelle)ELSE stelle=re
chts(stelle)                         <618F>
430 IF begriff$(stelle)=such$THEN
PRINT"Gefunden an Position ";stell
e:GOTO 450                           <4B25>
440 IF stelle=0 THEN PRINT"Begriff
 nicht gefunden!"ELSE 420            <31EE>
450 RETURN                           <06AB>
460 'Speichern                       <105E>
470 OPENOUT"BBAUM.DAT"               <1185>
480 FOR i=0 TO 100:PRINT#9,begriff
$(i):PRINT#9,links(i):PRINT#9,rech
ts(i):NEXT i:PRINT#9,zaehl           <5EB2>
490 CLOSEOUT                         <06FC>
500 RETURN                           <0610>
510 'Laden                           <0CA0>
520 OPENIN"BBAUM.DAT"                <114B>
530 FOR i=0 TO 100:INPUT#9,begriff
$(i):INPUT#9,links(i):INPUT#9,rech
ts(i):NEXT i:INPUT#9,zaehl           <5EF3>
540 CLOSEIN                          <065C>
550 RETURN                           <0675>
560 'Programmende                    <1398>
570 END                              <06D8>
```

ähnliche Berechnung wie beim Eintragen eines Elementes in den Baum durchgeführt. Abhängig vom Wert des zu suchenden Strings wird nach links oder rechts verzweigt. Befindet sich dort der gesuchte Begriff im Schlüsselfeld, so kann Zeile 430 bereits die Erfolgsmeldung ausgeben.
War der gewünschte String dort nicht auffindbar, so bestehen zwei Möglichkeiten. Die unangenehmere davon ist, daß der Zeiger den Wert Null hatte und somit zurück in die Wurzel verwies. Ist das der Fall, so befindet sich der gesuchte String nicht im Baum.

SCHNELL GEFUNDEN

Die zweite Möglichkeit besteht darin, daß zwar ein Zeiger auf einen weiteren Knoten vorhanden war, daß jedoch dessen Schlüsselfeld den gesuchten Eintrag nicht enthält. In diesem Fall muß die Suche mit dem neuen Knoten fortgesetzt werden. Diese Abfrage finden Sie in Zeile 440.
Die Suche im Binärbaum ist einer der schnellsten Suchalgorithmen. In einem ideal ausbalancierten Baum, der rechts und links dieselbe Anzahl von Knoten enthält, würden Sie nur zehn Zugriffe benötigen, um den gewünschten Datensatz unter 1024 Eintragungen herauszufinden. Da unsere Baumroutinen nicht so effizient ausgelegt sind, daß sie den Baum automatisch ausgleichen würden, erhöht sich die Zahl der Zugriffe gegenüber dem Idealfall.
Dennoch sparen Sie im Vergleich zur sequentiellen Suche, bei der Sie einen Datensatz nach dem anderen überprüfen müssen, einiges an Zeit. Außerdem liegt der Binärbaum immer in sortierter Form vor, so daß Sie ihn für eine effiziente Suche nicht aufbereiten müssen. Da bei den üblichen Sortiermethoden mit der Vertauschung von Strings gearbeitet wird, entsteht einiger Abfall, den der CPC in einer garbage collection wieder aus seinem Speicher entfernen muß. Diese zeitraubende Müllbeseitigung können Sie sich hier ersparen.

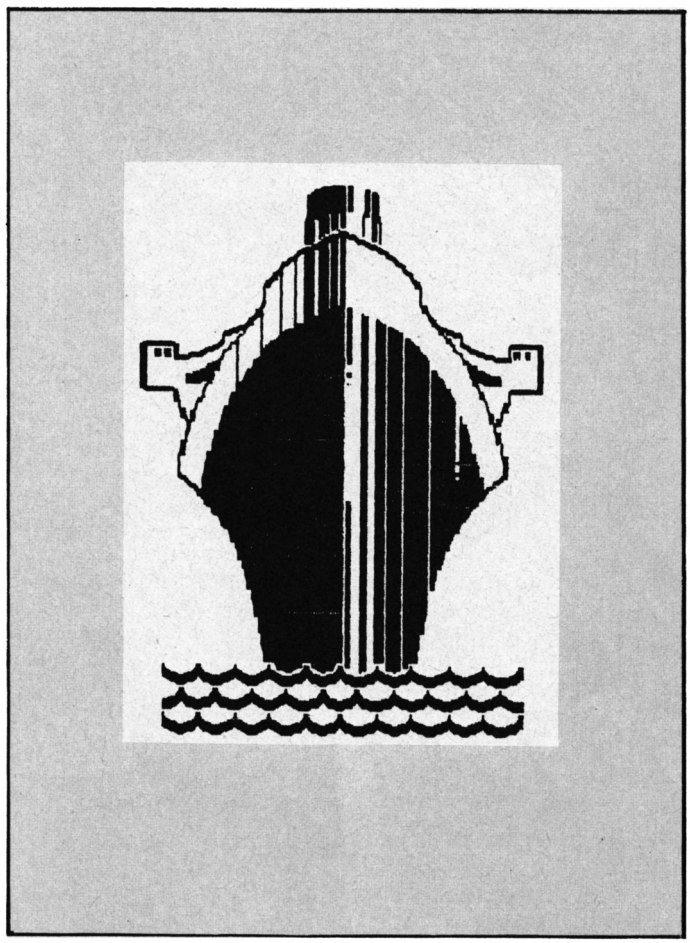
OFFEN FÜR ERWEITERUNGEN

Ein Nachteil dieser Datenstruktur sei nicht verschwiegen. Versuchen Sie, auf einem Blatt Papier einen Binärbaum zu skizzieren, der zwei gleiche Schlüsselfelder enthält. Irgendwann kommen Sie an einen Punkt, an dem es nicht mehr möglich ist, einen weiteren Datensatz einzufügen. Auch bei der Suche blockiert der identische Eintrag einen Teil des Baumes. Das Programm verabschiedet sich mit einem eleganten Absturz.
Um solche Bruchlandungen zu vermeiden, müssen Sie darauf achten, daß niemals zwei inhaltlich gleiche Schlüsselfelder auftauchen können.
Mit der Auskunft, daß der gesuchte Begriff gefunden ist, ist Ihnen in der Regel nicht gedient. Normalerweise gehören zu einem Begriff noch weitere Informationen, die jedoch nicht Bestandteil des Schlüsselfeldes sind.
Ein Beispiel dafür wäre eine Adreßverwaltung. Schlüsselfeld ist allein der Name; der Rest des Datensatzes enthält Zusatzinformationen.
Sie können das hier vorgestellte Versuchsprogramm ohne Probleme für diese Anwendung erweitern, indem Sie zusätzliche Arrays hinzufügen. Eine Lade- und Speicherroutine ist ebenfalls schon integriert. Denken Sie daran, daß Sie die Zusatzinformationen zwar in den Baum eintragen müssen, sie jedoch bei der Suche nach einem Datensatz keine Rolle mehr spielen. *AE* □

Print macht Power

Mit dem Befehl PRINT können Sie in BASIC einiges mehr bewerkstelligen, als einfach nur Zeichen auf dem Bildschirm auszugeben. Bei richtiger Anwendung einiger Steuercodes werden Ihre Programme kürzer, schneller und leistungsfähiger.

Die Anweisung PRINT kann nicht nur sichtbare Zeichen auf den Bildschirm bringen. Der Schneider CPC kennt insgesamt 32 in der Regel nicht druckbare Zeichen, welche jeweils bestimmte Funktionen ausführen. Ein Großteil davon dient der Simulation von BASIC-Befehlen.
Es gibt aber auch ein paar Steuerzeichen, die mit Funktionen belegt sind, die Ihnen in BASIC sonst nicht zur Verfügung stehen. Auf diese Pseudo-Befehle wollen wir einmal unser Augenmerk richten.



BUCHSTABEN

Der CPC verwendet Druckzeichen mit Codes von 0 bis 255, die nach ASCII (American Standard Code for Information Interchange) codiert sind. In diesem Code entspricht jedem Zeichen eine bestimmte Zahl. Das große „A" liegt zum Beispiel bei 65, das Leerzeichen auf 32, das Zeichen „1" bei 49.
Da der CPC, wie fast alle heutigen Computer, ein Zeichen jeweils in einem Byte verschlüsselt, gibt es 256 mögliche Belegungen. Der ASCII jedoch stammt noch aus den Tagen des Fernschreibers und ist nur mit sieben Bit, also 128 Zeichen, belegt. Daher hat Schneider ab Codenummer 128 einen eigenen Zeichensatz definiert.
Im ASCII sind die Zeichen 0 bis 31, die kein druckbares Äquivalent besitzen und folglich nur durch ihre Codenummern dargestellt werden können, zur Ausführung bestimmter Steuerfunktionen vorgesehen. Das war und ist bei der Datenfernübertragung nötig, um etwa Textstart- und -endemarkierungen setzen zu können oder beim Empfänger ein akustisches Signal auszulösen.
Bei der Ausgabe auf dem Bildschirm Ihres CPC sind all diese Funktionen jedoch überflüssig. Wir könnten jetzt diese Zeichen einfach für nicht existent erklären und ihr Auftreten in einem Text ignorieren – mit einigen Ausnahmen, auf die wir später noch zu sprechen kommen werden.
Amstrad ging bei der Entwicklung des CPC-BASIC jedoch einen anderen Weg. Fast jedes Steuerzeichen erfüllt hier eine Funktion, deren Ergebnis auf dem Bildschirm sichtbar ist, oder simuliert einen BASIC-Befehl. Sie müssen nur das entsprechende Zeichen via PRINT ausgeben lassen. Alles andere erledigt der Computer für Sie.
Es mag vielleicht etwas ungewohnt sein, mit PRINT einen Befehl ausführen zu lassen. Wenn Sie aber einmal damit gearbeitet haben, ist es auch nicht schwieriger als „normale" BASIC-Kommandos.
Eine Rücksetz-Routine für die Bildschirmfarben könnte zum Beispiel aus folgenden Anweisungen bestehen:

```
INK 0,0
INK 1,21
PAPER 0
PEN 1
BORDER 0
MODE 2
```

Wenn Sie im BASIC-Handbuch nachschlagen, finden Sie für jeden der oben genannten Befehle einen Steuercode:
4 für MODE, 28 für INK, 14 für PAPER, 15 für PEN und 29 für BORDER.
Folgende Definition am Anfang eines Programms erspart Ihnen einen Unterprogrammaufruf:

```
reset$=CHR$(28)+
CHR$(0)+CHR$(0)+
CHR$(0)+CHR$(28)+
CHR$(1)+CHR$(21)+
CHR$(21)+CHR$(14)+
CHR$(0)+CHR$(15)+
CHR$(1)+CHR$(4)+
CHR$(2)
```

Mit dem einfachen Befehl

```
PRINT reset$
```

haben Sie die gewünschte Farbkombination eingestellt.
Das mag Ihnen vielleicht etwas umständlich erscheinen, und vielleicht wäre ein Unterprogramm auch wirklich besser.
Die wahre Stärke der Steuerzeichen zeigt sich erst, wenn BASIC plötzlich für die gewünschte Anwendung nicht mehr ausreicht.

STEUERZEICHEN SICHTBAR GEMACHT

Das erste Steuerzeichen, das Sie vielleicht einmal benötigen werden, trägt die Nummer 1. Wenn Sie nach diesem Code ein weiteres Steuerzeichen ausgeben, so verwandelt es sich – welch Wunder! – in ein sichtbares Zeichen. Auch die Codes 0 bis 31 wurden nämlich mit Zeichenmatrizen, einer „Gestalt" also, versehen. *Beispiel 1* gibt Ihnen diesen Zeichensatz auf dem Bildschirm aus. Denken Sie bei Verwendung dieser Figuren bitte daran, daß ein Standarddrucker sie genauso-

wenig darstellen kann wie die Zeichen über 127. Auch Drucker verwenden diese Codes als Steuersymbole; sie wirken jedoch meist völlig anders als auf dem Bildschirm. Deshalb sollten Sie diese Zeichen auch immer über die CHR$-Funktion ansprechen, nie direkt mit der CTRL-Taste – obwohl auch das möglich ist.

CURSORSTEUERUNG...

Doch sehen wir uns zunächst ein weiteres Steuerzeichen an; es hat die Nummer 5. *Beispiel 2* führt Ihnen seine Wirkung vor. Hier wird das Programm tatsächlich kompakter als in reinem BASIC. Die Zeile 70 müßte lauten:

```
70 TAG:PRINT MID$
   (a$,i,1);:TAGOFF
```

Natürlich gibt es auch eine Reihe Steuercodes, die noch den Zweck erfüllen, für den sie ursprünglich vorgesehen waren – nämlich die Bewegung des Cursors oder, auf Druckern, von Wagen und Papier. Dazu gehört das Zeichen Nummer 13, genannt „carriage return" (Wagenrücklauf). Es setzt den Cursor auf den linken Rand der gerade bearbeiteten Zeile beziehungsweise läßt den Druckkopf dorthin zurücklaufen.
Auch das Zeichen „line feed" (Zeilenvorschub) erfüllt noch seine ursprüngliche Funktion, nämlich eine Verschiebung des Cursors um eine Zeile nach unten beziehungsweise einen Papiervorschub um eine Zeile.
Diese beiden Zeichen sendet BASIC automatisch nach jedem PRINT, das nicht von einem Komma oder Semikolon abgeschlossen wird. Deshalb finden Sie sich nach einem solchen Befehl am linken Rand einer neuen Zeile wieder.
Bei den übrigen Cursorsteuerzeichen wird die Anwendung schwieriger: CHR$(11) setzt den Cursor in die vorhergehende Zeile, CHR$(8) bewegt ihn um ein Zeichen nach links, und CHR$(9) läßt ihn ein Zeichen nach rechts wandern. Das entspricht aber nicht der Ausgabe eines Leerzeichens, da mit CHR$(9) der Hintergrund erhalten bleibt. Wenn Sie diese Zeichen benutzen, müssen Sie bei einer eventuellen Druckerausgabe darauf Rücksicht nehmen, ob Ihr Printer versteht, was Sie von ihm wollen.
Die Bewegung des Cursors nach oben entspricht einem Zurückschieben des Papiers. Das kann längst nicht jeder Drucker; und viele, die's können, verwenden wieder einen anderen Code.
In *Beispiel 3* sehen Sie die Auswirkung dieser Steuerbefehle auf den Bildschirm. Der Cursor bewegt sich dauernd, nur leider können Sie ihn nicht sehen. Durch einmaligen Druck auf ESC hält das Programm an, ein Druck auf eine andere Taste setzt es fort. Wenn Sie es auf diese Weise mehrmals anhalten, sehen Sie den Cursor über verschiedenen Ziffern irgendwo in der Mitte des Bildschirms.

```
10 'Beispiel 1
20 FOR i=0 TO 31
30   PRINT i,CHR$(1);CHR$(i)
40   FOR j=1 TO 500:NEXT j
50 NEXT i
```

```
10 'Beispiel 2
20 CLS
30 MOVE 100,150
40 a$="Dies ist ein Test."
50 FOR i=1 TO LEN(a$)
60   PRINT MID$(a$,i,1);
70   PRINT CHR$(5);MID$(a$,i,1);
80 NEXT i
```

```
10 'Beispiel 3
20 MODE 1
30 FOR i=1 TO 25
40   PRINT"12345678901234567890123456789
01234567890";
50 NEXT i
60 LOCATE 20,12
70 WHILE INKEY$=""
80   PRINT CHR$(8);
90   PRINT CHR$(9);
100   PRINT CHR$(10);
110   PRINT CHR$(11);
120 WEND
```

```
10 'Beispiel 4
20 MODE 2
30 PRINT"Dies ist ein Test..."
40 PRINT CHR$(21);
50 PRINT"Dies ist ein Test..."
60 PRINT CHR$(6);
70 PRINT"Dies ist ein Test..."
```

```
10 'Beispiel 5
20 MODE 1
30 FOR i=1 TO 25
40   PRINT"12345678901234567890123456789
01234567890";
50 NEXT i
60 LOCATE 20,12
70 PRINT CHR$(19);
80 FOR j=1 TO 500:NEXT j
90 PRINT CHR$(20);
100 PRINT"Hier bin ich!"
110 WHILE INKEY$="":WEND
```

```
10 'Beispiel 6
20 CLS
30 a$="Dies ist ein Test..."
40 FOR i=1 TO LEN(a$)
50   PRINT CHR$(24);MID$(a$,i,1);
60 NEXT i
```

...UND GRAFIK-TRICKS

Diese Cursorsteuerung kann in Verbindung mit anderen Kontroll-Codes dazu dienen, verschiedene Effekte herbeizuführen. Ein nützliches Beispiel ist die Überlagerung verschiedener Zeichen. Normalerweise löscht der CPC ein Zeichen, wenn ein neues an dieselbe Position geschrieben wird. Das ist jedoch nur ein Betriebsmodus für den Textbildschirm, der von Schneider „Opaque-Modus" genannt wurde. Es existiert auch noch ein sogenannter Transparent-Modus, bei dem die automatische Hintergrundlöschung abgeschaltet ist. Zwischen diesen beiden

Modi können Sie nur über einen Steuercode hin- und herschalten. Er trägt die Nummer 22. Geben Sie danach das Zeichen mit dem Code 1 aus, so ist der Transparent-Modus eingeschaltet. Sie sollten allerdings das Ausschalten (Codenummer 0) nicht vergessen, sonst erleben Sie meistens eine unangenehme Überraschung.
Die Anweisung

```
PRINT "e"; CHR$(22);
           CHR$(1);
           CHR$(8);
           " ";
           CHR$(22);
           CHR$(0)
```

erzeugt ein "e" mit Akzent, das auf der Tastatur üblicherweise nicht vorhanden ist. So können Sie aufwendigen Zeichendefinitionen aus dem Weg gehen oder mehrfarbige Zeichen darstellen. Bestimmen Sie zwischendurch einen neuen PEN – vielleicht auch über Steuercodes.
Mit dem Steuerzeichen Nummer 23 werden verschiedene Grafikmodi eingestellt. Davon existieren vier Stück, die bestimmen, in welcher Art neu zu zeichnende Punkte mit dem Hintergrund verrechnet werden. Die Parameter dieses Codes gehen entsprechend von 0 bis 3. Dabei steht 0 für FORCE, 1 für XOR, 2 für AND und 3 für OR. Wer sich mit logischen Verknüpfungen bereits intensiver beschäftigt hat, benötigt nur noch für den FORCE-Modus eine Erklärung. Er ist standardmäßig eingestellt und dürfte damit jedem bekannt sein: Die angegebene Farbe wird an jedem Punkt ohne Berücksichtigung des dargestellten Hintergrunds auf den Bildschirm gebracht.
Die anderen drei Modi verknüpfen die aktuelle Grafikfarbe an jedem Punkt mit der bereits auf dem Bildschirm vorhandenen, und zwar bitweise in Abhängigkeit von den jeweiligen logischen Operatoren (XOR, OR, AND). Das kann sehr hübsche Effekte ergeben. Sie müssen nur eifrig experimentieren.
Eine mögliche Anwendung für den XOR-Modus sei noch verraten: Sobald Sie zweimal dieselben Punkte auf dem Bildschirm in derselben Farbe zeichnen lassen (und XOR aktiviert ist), ist der alte Hintergrund wieder zu sehen. Damit können Sie einen Zeiger über den Grafikschirm bewegen, ohne das Bild dauernd neu aufbauen zu müssen. Eine Maus-Simulation über Joystick liegt damit im Bereich des in BASIC Machbaren.
Wenden wir uns wieder dem Textbildschirm zu. *Beispiel 4* zeigt Ihnen zwei Steuercodes, mit denen Sie die Textausgabe ab- und anschalten können. Steuercodes werden trotzdem akzeptiert, wie Sie an der Leerzeile erkennen können. Sie wird natürlich durch die automatisch gesendeten Zeichen "carriage return" und "line feed" verursacht.
Das Abschalten der Textausgabe kann zum Beispiel dann erwünscht sein, wenn der Benutzer ein Kennwort eingeben soll, das nicht jeder zufällig Anwesende wissen muß. Der normale INPUT-Befehl liefert bekanntlich ein Echo der Eingabe auf dem Bildschirm.
Es gibt mehrere Steuercodes, mit denen Sie einen Teilbereich des gerade aktiven Textfensters oder des ganzen Textbildschirms löschen können. Alle diese Codes sind in Ihrem BASIC-Handbuch beschrieben. Sie haben zwei schwerwiegende Vorteile gegenüber dem Überschreiben mit Leerzeichen aufzuweisen. Zum einen ist ihre Ausführung wesentlich schneller; zum anderen wird bei ihrer Anwendung die aktuelle Cursorposition nicht verändert, so daß Sie Ihren Text ungehindert fortsetzen können. Wie das aussieht, demonstriert *Beispiel 5*.
Die am häufigsten verwendeten Codes sind die Zeichen 7 und 24.
Die Nummer 7 trägt im ASCII die Kurzbezeichnung BEL für "bell".
Eine Klingel ertönt dabei allerdings nicht mehr; der CPC beschränkt sich auf ein kurzes Piepen. Für Sound-Anwendungen mag es Sie vielleicht interessieren, daß CHR$(7) alle Tonwarteschlangen leert.
Das Invertierungszeichen 24 wird in *Beispiel 6* vorgeführt. Texthervorhebung durch Invertierung ist wohl die dezenteste Art, den Programmanwender auf etwas aufmerksam zu machen. Deshalb wird dieses Zeichen oft und gern verwendet.

DRUCKER „FRESSEN" KEINE STEUERZEICHEN!

Manche BASIC-Programmierer wollen sich Tipparbeit ersparen, indem sie CTRL-X, das Äquivalent zu CHR$(24), direkt über die Tastatur eingeben. Spätestens bei der Ausgabe des Listings auf dem Drucker schlägt jedoch CTRL-X unbarmherzig zu. Oft fehlen ganze Teile des Ausdrucks. Ein Blick in die ASCII-Tabelle bringt uns ein Stück weiter. Dort finden wir als Abkürzung für diesen heimtückischen Code "CAN". Das steht für "cancel", zu Deutsch: stornieren. Die meisten Drucker haben einen Zwischenspeicher, einen sogenannten Puffer, der einen Teil der gesendeten Daten aufnimmt, bevor der Text gedruckt wird. Das Zeichen "cancel" wird von den meisten Printern als Aufforderung verstanden, diesen Puffer zu löschen. Je nach Größe des Puffers und Menge der bereits gesendeten Daten kann das bis zu 16 KByte und mehr vernichten.
Genauso kann es Ihnen mit CTRL-G (CHR$(7) oder BEL) ergehen: Viele Drucker haben einen eingebauten Summer, der dann piept – im Listing keine Spur von dem gewünschten Zeichen.
Steuerzeichen können also die Effektivität Ihres Programms erhöhen, wenn Sie sie gezielt anwenden. Die Ansteuerung sollte nur über CHR$ und, wenn möglich, nur durch am Programmanfang definierte Strings erfolgen, deren Bezeichner halbwegs aussagekräftig sind.
"PRINT transparent$;" sagt jedem etwas; mit "PRINT CHR$(22); CHR$(1);" können nur eingefleischte Schneider-Freaks etwas anfangen.

AE □

COMPUTER UND KIRCHE

Den Computer als Arbeitshelfer hat ein Dekan einer evangelisch-lutherischen Gemeinde nahe Nürnberg entdeckt. Ein dort ansässiges Unternehmen entwickelte speziell für Pfarreien das Programm Diakonos. Damit können im wesentlichen die Schreib- und Verwaltungsarbeiten erledigt werden, die ein Pfarrer heute eben auch hat.
Der Einsatz des Computers führte bei „Testpfarrer" Christoph Schirmer nicht etwa zu mehr Bürokratie, sondern zu mehr Qualität seiner seelsorgerischen Arbeit.
Aber auch das Bibelstudium kann effektiver gestaltet werden: Altes und Neues Testament sind auf Diskette erhältlich, und mit der kompletten Lutherischen Bibelübersetzung auf CD-Platte ist eine EDV-Pfarrei sogar auf dem Höhepunkt der Technik. Probleme, etwa mit dem Datenschutz bei Geburtslisten, sieht man nicht. □

PROGRAMMIERSPRACHEN ZUR AUSWAHL

SPRACHEN-GEWIRR

Der vielschichtige, schnell wachsende Markt der Programmiersprachen und ihrer Dialekte ist unüberschaubar geworden. Unsere Serie erläutert die Hauptunterschiede.

Das komplexe Thema der Programmiersprachen von Ada bis V.I.P. ist nicht nur für den Profi-Programmierer interessant, sondern auch für die Einsteiger geeignet, die sich einen Überblick der babylonischen Sprachverwirrung beschaffen wollen. Wir werden dieses Thema in mehreren Teilen aufbereiten.

Es gibt zur Zeit weltweit mehr als 160 höhere Programmiersprachen. Nur etwa 15 Prozent unterscheiden sich erheblich voneinander.

Von einigen existieren mehrere Versionen, sogenannte Dialekte. Hinzu kommen Besonderheiten der verschiedenen Computeranlagen und deren Compiler, auf denen mit den höheren Programmiersprachen gearbeitet wird. Dies fördert die Entwicklung weiterer Dialekte. Addiert man alle Variationen und Dialekte, dürfte die Anzahl der bekannt gewordenen Programmiersprachen zwischen 2000 und 2500 liegen.

Ein weiterer Faktor, der die Übersichtlichkeit auf diesem Markt einschränkt, ist die Weiterentwicklung der Sprachen. Dies bringt neben Vorteilen sicherlich auch Probleme mit sich: Bestehende Programme müssen teilweise umgestellt werden.

Hier soll nur auf die Hauptunterschiede der verschiedenen Sprachen eingegangen werden. Verbesserungen in den einzelnen Weiterentwicklungen und Dialekten ändern den Charakter der betreffenden Sprache nicht und den Anwendungsbereich nur unwesentlich. Nur dort, wo eine Sprache ihr Erscheinungsbild grundlegend geändert hat, wird im Text darauf eingegangen.

Die gebräuchlichsten Programmiersprachen werden etwas ausführlicher behandelt. Bei den einzelnen Beschreibungen werden Probleme angesprochen, die auch für andere Sprachen gelten. So greift die Besprechung über zu anderen Sprachen und allgemeinen Problemen und zeichnet Besonderheiten aus den Anwendungsgebieten auf. Zu einigen Programmiersprachen werden Beispiele gegeben; je nach Besonderheit und Bedeutung der Sprache ausführlicher dokumentiert. Sie sollen nur dazu dienen, einen optischen Eindruck zu vermitteln.

WICHTIGE ENTSCHEIDUNGSHILFE

Strukturierung und Modularität einer Sprache lassen sich so schon auf den ersten Blick erkennen. Auch verdeutlicht das Beispiel die Lesbarkeit einer Programmiersprache oder des Programms; der Charakter einer Sprache ist zu erkennen.

Die künstliche Intelligenz und Expertensysteme werden erläutert sowie Begriffe aus der Welt der Programmiersprachen. Der Leser erhält einen Eindruck, wie vielschichtig Programmiersprachen sein können. Vielleicht kann daraufhin eine Kauf- oder Lern-Entscheidung einfacher gestaltet werden. Zumindest wird der Leser sich ein eigenes Urteil bilden können, wenn die eine oder andere Programmiersprache angepriesen oder verurteilt wird. In den meisten Fällen werden hier nur Vor- und Nachteile einer Programmiersprache genannt und das vorrangige Einsatzgebiet. Es gibt keine „schlechten" Programmiersprachen, sie müssen immer in ihrem Umfeld betrachtet werden, und sei es nur das zeitliche. Auf veraltete Sprachen wird gesondert hingewiesen.

Die vorliegende Auswahl von Programmiersprachen ist nicht vollständig. Sie kann dies zu keinem Zeitpunkt sein, da in immer kürzeren Abständen neue Dialekte und „Verbesserungen" auf den Markt geworfen werden. Es erschien auch nicht sinnvoll, hier nur auf genormte Sprachen einzugehen, da sie zum einen rar sind, zum anderen keinen Überblick über die momentane Marktsituation geben können, denn Normierungen sind langwierige Unterfangen.

Andererseits kann unmöglich jeder firmeneigene Sprach-Dialekt ausführlich besprochen werden, daher sind sie zum Teil nur der Übersicht wegen namentlich aufgelistet oder wegen ihrer Sonderstellung kurz angesprochen.

Auch sprachliche Programmiermethoden sind in diese Liste aufgenommen. Die Grenze zwischen Methode und Sprache ist oft fließend. Daher erscheint manchmal der Begriff Programmiersystem anstelle der Programmiersprache. Dies sollte nicht verwirren, denn auch eine Programmiersprache ist immer in einem System zu sehen, und sei es nur der spracheigene Compiler oder Interpreter.



Moderne Computersysteme verlangen moderne Programmiersprachen

1. Ada

Ada ist wie HOL (siehe dort) eine Systemimplementierungssprache, die in den USA vom US-Verteidigungsministerium für den militärischen Einsatz entwickelt wurde. Seit dem 1. Januar 1985 ist Ada für alle US-Waffen-

systeme obligatorische Programmiersprache und löste damit etwa 800 (achthundert!) Programmiersprachen aus diesem Anwendungsbereich ab. Da Ada aus dem militärischen Bereich kommt, hatte sie ursprünglich einen Decknamen: green.
Ende der 70er Jahre hatte das DoD (US Departement of Defense) die Entwicklung einer Programmiersprache ausgeschrieben. Es wurden insgesamt 17 Vorschläge eingereicht, von denen vier in die engere Wahl gezogen wurden. Diesen ordnete man Tarnnamen zu:

green – Honeywell Bull
red – Intermetrics
blue – Softech



yellow – SRI International.

Das DoD entschied sich für „green" und so bekam Honeywell Bull den Zuschlag.
Ada ist eine Real-Time-Sprache (= Echtzeitsprache) ähnlich Pearl, jedoch unterscheiden sich beide unter anderem in dem aufzubringenden Programmieraufwand stark.
Ada wurde ursprünglich entwickelt für den Einsatz in embedded systems, das sind integrierte Computersysteme, die innerhalb größerer Systeme „eingebettet" sind und dort bestimmte Aufgaben übernehmen.
Beispiele für embedded systems sind unter anderem: Flugüberwachungssysteme, Feuerleitsysteme usw., aber auch mikroprozessorgesteuerte Waschmaschinen. Die Entwicklung von Programmen für solche Systeme ist in der Regel wesentlich kostenintensiver als technisch-wissenschaftliche oder kommerzielle Programme, da die letztgenannten Bereiche mit standardisierten Sprachen abgedeckt sind (Fortran, Cobol . . .).
Ada hat sich zwischenzeitlich auch auf dem nichtmilitärischen Bereich etabliert.

Programmiersprachen

1 Ada
2 ADAPT
3 ADF
4 Algol
5 APL
6 APS
7 APT
8 Assembler
9 ATLAS
10 Autocode
11 Autopit
12 Autopol
13 Autoproc
14 Autopromt
15 Autospot
16 BASIC
17 BCPL
18 BLISS-32
19 C
20 CDL/2
21 Cobol
22 Comal 80
23 Coursewriter
24 CSL
25 DIPOL
26 DL
27 Elan
28 EOL
29 Exapt
30 FOCUS
31 Forth
32 Fortran
33 GPL
34 HOL
35 HOPE
36 Lisp
37 Logo
38 LPG
39 MACRO 80
40 MANTIS
41 Modula 2
42 mu-SIMP
43 Natural
44 NEAT
45 NELIAC
46 NPL
47 NUCOL
48 OPS-5
49 OCCAM
50 Pascal
51 Pearl
52 PILOT
53 PLANKALKÜL
54 PL/1
55 PROLOG
56 PROSA
57 QBE
58 Query Language(QL)
59 PRG
60 RPNL
61 SIMULA
62 SL 3
63 Smalltalk
64 SQL
65 SYMAP
66 V.I.P
67 Verschiedene

ADA: WECHSEL INS ZIVILLEBEN

Obwohl Ada eine höhere Programmiersprache, also maschinenunabhängig ist, enthält ihr Sprachumfang maschinennahe Elemente, um dem Benutzer die Möglichkeit zu geben, für die – im militärischen Bereich besonders aufwendigen – Spezialanwendungen Ein- und Ausgabefunktionen anwendungsspezifisch zu erstellen.
Ada gibt es seit 1981 und hat in außermilitärischen Anwendungsgebieten Fuß gefaßt; sowohl im technisch-wissenschaftlichen Bereich, als auch in der kommerziell-administrativen Datenverarbeitung.
Ada ist inzwischen eine leistungsfähige, universelle Programmiersprache mit integrierten Spracheinrichtungen, die eine Vielzahl von modernen Programmiertechniken unterstützt. Ada wurde unter dem Aspekt der Wartungsfreundlichkeit sowie der gesteigerten Zuverlässigkeit und Übertragbarkeit entwickelt. Die Vorteile kommen besonders zur Geltung bei rechenintensiven Programmen in Echtzeitumgebung.
Ada – vorwiegend auf größeren Rechenanlagen einsetzbar – verlangt eine strenge Zuweisung von Datentypen in der ersten Phase der Programmentwicklung. Da die Überprüfung der Datentypenzuweisungen schon während der Compilierung stattfindet, können folgenschwere Fehler vermieden werden.
Die Datenabstraktion ermöglicht es dem Benutzer, sich auf die wesentlichen Punkte seines Programmes zu konzentrieren, ohne sich um die darunterliegenden Einzelheiten kümmern zu müssen. Die Details der Implementierung bleiben für ihn verborgen, während ihm gleichzeitig alle Mechanismen für ihre Verwendung zur Verfügung stehen.
Auffallend ist hier die Vielzahl der zur Verfügung stehenden Datentypen im Gegensatz zu anderen Programmiersprachen.
Die *Datentypen* gliedern sich in Ada wie folgt auf:

1. Ada Datentypen
1.1. Verweistypen
1.2. strukturierte Datentypen
- Ableitungstypen
- Unterbereichstypen
- abstrakte Typen

1.2.1. Felder (Strings)
1.2.2. Verbunde
1.2.2.1. mit Varianten
1.2.2.2. ohne Varianten
1.3. skalare Datentypen
- Ableitungstypen
- Unterbereichstypen
- abstrakte Typen

1.3.1. diskrete Datentypen (indiex-/iterationsfähig)
1.3.1.1. ganzzahlige Datentypen
1.3.1.2. Aufzählungstypen (z.B. Boolean, Character)
1.3.2. reelle Datentypen
1.3.2.1. Festpunkttypen (mit absoluter Genauigkeit)
1.3.2.2. Gleitpunkttypen (mit relativer Genauigkeit).

Ada bietet hervorragende Sprachelemente zur Steuerung paralleler Programmabläufe, ohne daß auf die Systemdienste des Betriebssystems zurückgegriffen werden muß. Der Programmierer hat die Möglichkeit, große Programme zu unterteilen und getrennt zu compilieren. Dabei liefert die Bibliotheksverwaltung Informationen zu den übrigen Compiler-Einheiten. Andere Programmiersprachen stellen nur wenig Information bei getrennt compilierten, aber voneinander abhängigen Programm-Modulen bereit.
Eine besondere Stärke von Ada sind die generischen Definitionen, spezielle Formen von Programmeinheiten, die Algorithmen definieren, analog zur Definition von Unterprogrammen, wobei die Datentypen variabel bleiben und erst bei der Compilierung festgelegt werden.

EINFACHE FEHLERBEHANDLUNG

Die Fehlerbehandlung bei Ada ist vorbildlich und trägt entscheidend zur Wartungsfreundlichkeit bei.
Die Namensgebung von Ada hat nichts mit einer Abkürzung zu tun, wie dies bei vielen anderen Programmiersprachen der Fall ist, sondern hat folgenden Ursprung: Der britische Mathematiker Charles Babbage (1792 bis 1871) entwickelte ein Gedankenmodell einer Analytical Engine, eines Computers. Dieses Modell verfügte über eine heute noch aktuelle Architektur: Eingabe, Ausgabe, Verarbeitung, Speicher. Seine Maschine konnte niemals gebaut werden, hätte doch alles mechanisch bewerkstelligt werden müssen, mit unendlich vielen Zahnrädern und ähnlichem. Babbage wurde nicht für ernst genommen. Moralisch und finanziell wurde er unterstützt von einer Edeldame namens Ada Augusta Countess of Lovelace. Ihr wurde diese Programmiersprache gewidmet.
Ada unterstützt sowohl die Top-Down- als auch die Bottom-Up-Entwicklung; beide Methoden sind gleichermaßen gebräuchlich, wenn sie auch mit sehr unterschiedlichen Vor- und Nachteilen behaftet sind.
Beide Verfahren beschreiben die Vorgehensweise bei der Programmerstellung im allgemeinen – also nicht nur bei Ada, sondern bei Programmiersprachen überhaupt.

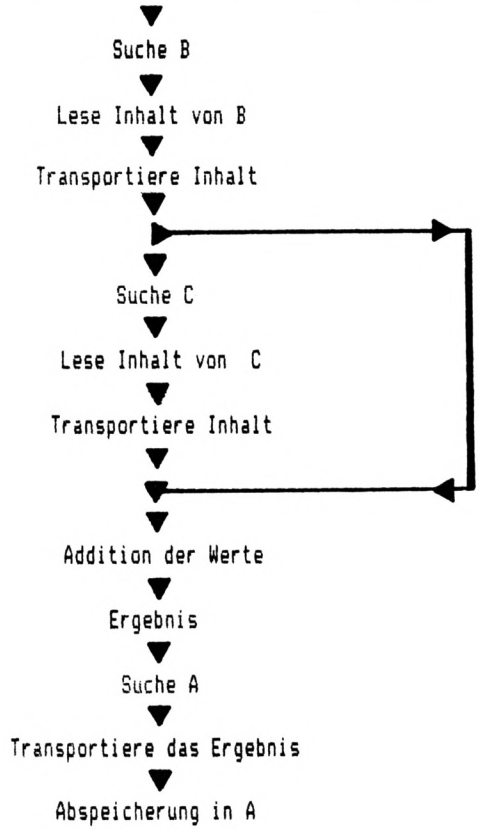
1. Top-Down-Methode
Das zu erstellende Programm wird durch schrittweise Verfeinerung entwickelt. In der Praxis sieht das so aus, daß die Problemlösung – der Weg – grob vorgezeichnet wird, so daß ein Programmgerüst entsteht. Verschiedene diffizile und oft benötigte Randprobleme, wie Datei-Ein- und -Ausgabe, werden dann in Form von Unterprogrammen realisiert. Ada unterstützt die Top-Down-Methode unter anderem durch die Möglichkeit der Einflechtung von body stubs anstelle der Unterprogramme. Body stubs sind Platzhalter für den Korpus (Rumpf) eines Unterprogrammes. Die eigentlichen Unterprogramme können später erstellt und sogar separat compiliert werden.

2. Bottom-Up-Methode
Diese Vorgehensweise bei der Programmierung steht in krassem Gegensatz zur Top-Down-Entwicklung. Hier wird das Problem zunächst untergliedert, und Einzellösungen in Form von Unterprogrammen werden ausgearbeitet. Daraufhin kann der Rumpf des Programms mit den entsprechenden Unterprogrammaufrufen erstellt werden.

Der große Vorteil von Ada ist die strenge Standardisierung. Darüber wacht das Ada Joint Programm Office, dem sogar Lehrbücher über Ada vorgelegt werden müssen.
Ada ist als Universalsprache zu bezeichnen, die dem professionellen Softwareentwickler eine umfassende Sprachunterstützung bietet.
Der Programmieranfänger wird sich mit Ada jedoch nicht leicht tun, das ist eben der Preis der Komplexität einer höheren Programmiersprache.
Ada-Experten gehen sogar so weit, zu behaupten, daß man Fähigkeiten dieser mächtigen Sprache nur mit einem abgeschlossenen Informatikstudium ausnutzen könne. Mit anderen Worten: Ada ist nicht geeignet für Anfänger.

Tabelle 1:

Beispiel Assemblerprogrammierung Ablaufplan:



2. ADAPT
NC(Numerical Controlled)-Steuersprache (siehe APT, EXAPT).

3. ADF
Dedizierte Abfrage-Sprache im Bereich informativer Systeme (Datenbanken etc.), siehe hierzu auch Query Language oder SQL.

4. Algol
Algorithmic **L**anguage.
Algorithmische Sprache.
Eine mathematisch-wissenschaftlich orientierte Sprache, die in der Hauptsache an Hochschulen angewendet wird.
Algol ist seit 1959 auf dem Markt und hat starke Ähnlichkeiten mit Fortran, ist jedoch systematischer in ihrem Aufbau des Sprachschatzes und umfangreicher. Die Ausgabe von 1959 wurde bereits 1960 revidiert (Algol 60). 1968 erschien dann Algol 68, in ihrem Wortschatz wesentlich umfassender und ausgereifter. Wie bei vielen anderen Programmiersprachen sind auch hier die

Tabelle 2:

Programmbeispiel in Assembler für CPU 6502 von MOS-Technology, zu finden in VC20/C64/Apple etc.
Source-Listing Ausschnitt zu Assembler:

```
1A24          91 26     STAIZ TEMPL     Abspeichern und nächste
1A26  GENADR  20 BD 19  JSR NXTADR      Adresse holen, wenn 0,
1A29          F0 D8     BEQ AGAIN       dann nicht fertig, sonst
1A2B          20 A6 19  JSR RSTORE      Rücksprung zu RSTORE
1A2E          69        RTS
1A2F  NBRS    98        TYA             X und Y in Stack speichern
1A30          48        PHA
1A31          8A        TXA
1A32          48        PHA
1A33          A0 00     LDYIM $00       Y und N gleich Null setzen
1A35          84 32     STYZ
1A37          A2 08     LDYIM $08       Acht Nachbarn testen
1A39  OFFS    B5 29     LDAZX OFFSET
1A3B          10 15     BPL ADD         Springe nach ADD, wenn
1A3D          49 FF     EORIM $FF       OFFSET positiv
1A3F          85 37     STAZ TMP        sonst Subtraktion vorbereiten,
1A41          38        SEC             Carry-Bit setzen
1A42          A5 39     LDAZ RCSH
1A44          E5 37     SBCZ TMP        Subtrahieren, damit die
1A46          85 22     STAZ CHL        korrekte Adresse des
1A48          A5 3A     LDAZ RCSH       Nachbarn entsteht,
1A4A          85 23     STAZ CHH        Sprung zur
1A4C          B0 11     BCS EXAM        Untersuchungsroutine
1A4E          C6 23     DECZ CHH        Page-Grenze überschreiten
1A50          D0 0D     BNE EXAM        weiter zu EXAM
1A52  ADD     18        CLC             Addieren vorbereiten, wenn
1A53          65 39     ADCZ RCSL       OFFSET positiv, Ergebnis
1A55          85 22     STAZ CHL        in CHL ablegen.
1A57          A5 3A     LDAZ RCSH
1A59          85 23     STAZ CHH
1A5B          90 02     BCC EXAM        Sprung zur EXAM-Routine
1A5D          E6 23     INCZ N          Page-Grenze überschreiten
1A5E  EXAM    B1 22     LDAIY CHL       Nachbardatenbyte holen
1A61          C9 51     CMPIM DOT       und prüfen, ob besetzt
1A63          D0 02     BNE NEXT
1A65          E6 32     INCZ N          Anzahl der Nachbarn
1A67  NEXT    CA        DEX             erhöhen
1A68          D0 0F     BNE OFFS        Sprung nach OFFS
1A6A          68        PLA             X und Y Werte aus Stack
1A6B          AA        TAX             zurückholen
1A6C          68        PLA
1A6D          A8        TAY
1A6E          60        RTS
```

verschiedenen Ausgaben kaum oder nur bedingt kompatibel.
Diese Sprache weist starke Nachteile in der Ein- und Ausgabe von Daten auf. Ihre Vorteile liegen in der streng mathematischen Ausdrucksform.
Algol ist für Deutschland in der DIN 66026 „Informationsverarbeitung; Programmiersprache Algol" festgelegt.
Nach Algol 60 und Algol 68 folgte noch ein Versuch, diese „veraltete" Programmiersprache zu retten: Algol W, aber ohne nennenswerten Erfolg. Pascal war der nächste Entwicklungsschritt und löste damit die Algol-Familie ab.

5. APL

A Programming Language. APL ist ähnlich wie BASIC eine problemorientierte Sprache. Sie ist 1960 in den USA von dem Mathematiker K. Iverson entwickelt worden und wurde in erster Linie bei dem Time Sharing und bei größeren Rechenanlagen eingesetzt. 1962 wurde APL publiziert, aber erst 1966 gab es die erste Implementation für das IBM-System /360.
Mittlerweile gibt es auch schon mehrere Interpreter für diverse Microcomputer.
Time Sharing (Zeit-Aufteilung) einer Rechenanlage liegt dann vor, wenn mehrere Benutzer zur selben Zeit an einem Rechner angeschlossen sind. Der Rechner kann aber immer nur ein Programm bearbeiten, daher muß die Benutzerzeit aufgeteilt werden.
Time Sharing ist das Zuweisen von Arbeitszeit (meist im Sekunden-Bereich) der Zentraleinheit des Rechners an die verschiedenen Benutzer; es ist jedoch auch mit anderen Programmiersprachen möglich. APL spielt heute im Zeitalter der Microcomputer keine große Rolle mehr in bezug auf das Time Sharing.
Das Hauptanwendungsgebiet für APL liegt in der Wirtschaftsmathematik und Statistik sowie in der Mathematik bei der Vektor- und Matrizenrechnung. Vorteile gegenüber Fortran und Algol: bessere Datenhandhabung und bessere Dialogmöglichkeiten. Im Gegensatz zu diesen ist APL jedoch interpretativ: Jeder Befehl wird sofort interpretiert und in eine Maschinensprache umgeformt (kein Compiler notwendig). Außerdem ist APL fünf- bis achtmal schneller als Cobol und Fortran.

Weitere Vorteile von APL:
- APL bietet sowohl gute Berechnungsmöglichkeiten als auch eine gute Textverarbeitung;
- Durch wenige Reglements und wenige Symbole ist APL sehr kompakt und leicht erlernbar für den Anfänger;
- Die zu bearbeitenden Daten müssen nicht wie in anderen Sprachen vom Benutzer genau definiert werden. Es muß keine Rücksicht auf vorgegebene Formate und Größenordnungen der Daten genommen werden;
- Problemlose Matritzen- und Vektorverarbeitung sowie einfache Programmierung von Tabellenausgaben etc.

APL unterscheidet sich in einigen Punkten sehr stark von anderen Programmiersprachen, und obwohl schon etwas betagt, gibt es keine vergleichbare Sprache.
Sie ist nicht nur eine höhere Programmiersprache zur Beschreibung von Algorithmen (problemorientiert), sondern auch eine moderne Notation für die Darstellung mathematischer Sachverhalte. Komplizierte mathematische Zusammenhänge können in APL vereinfacht dargestellt werden. In APL ist der Umgang mit komplexen Objekten wie Vektoren, Matritzen und Folgen genauso einfach wie mit einfachen Zahlen. Komplizierte mathematische Zusammenhänge können so einfach und verständlich dargestellt werden.

In APL gibt es drei Gruppen von *Operatoren:*

1. Skalare Operatoren. Diese sind für skalare Argumente definiert, die als Ergebnisse wiederum Skalare liefern. Die skalaren Operatoren sind jedoch auch auf beliebige Strukturgrößen anwendbar, allerdings muß bei dyadischen Funktionen entweder die Struktur von Links- und Rechtsargument identisch oder eines der beiden Argumente ein Skalar sein.
2. Gemischte Operatoren. Hier treten stets Vektoren als Argumente oder als Ergebnis auf. Sie lassen sich daher nicht ohne weiteres auf Argumente höheren Ranges erweitern.
3. Spezielle Operatoren. Alle übrigen Operatoren, die nicht zu 1. oder 2. gehören, werden als spezielle Operatoren bezeichnet.

Das Hauptmerkmal von APL liegt darin, daß ein

in der Notation von Iverson dargestelltes Problem bereits das Programm zu dessen Lösung darstellt. Zur Programmgestaltung stehen ergänzend zur Verfügung: die Möglichkeit zur Programmstrukturierung (Verzweigung, Schleifen, Unterprogrammbildung) und die Namensgebung für das Programm. Nicht zuletzt zu nennen ist die Möglichkeit, Programme zu editieren, korrigieren, abzuspeichern etc.; aber das zeichnet ja alle Programmiersprachen aus.
Oft ist es nicht erforderlich, ein Problem als APL-Programm zu bearbeiten. Vielmehr wird das Problem mit mehreren Anweisungen Schritt für Schritt der Lösung zugeführt. Sobald ein Problem mehrfach auftritt – nur mit wechselnden Parametern – wird eine Funktion auch programmiert.
APL ist ein interpretatives System. Jede eingegebene Anweisung wird sofort in eine systeminterne tabellarische Form umgewandelt und ausgeführt. Einzige Ausnahme sind Programmzeilen mit definierten Funktionen. Diese werden zum späteren Gebrauch abgespeichert.
Wie bei allen interpretativen Programmiersprachen erfolgt hier eine direkte Fehlererkennung und -meldung, was die Programmierung erleichtert.

DESKTOP FÜR PROGRAMMIERER

Die Arbeit in APL ist vergleichbar mit der Arbeit am Schreibtisch (nicht zu verwechseln mit Desk Top Publishing): Man macht sich Notizen, führt Berechnungen durch, hält Zwischenergebnisse fest, faßt Ergebnisfolgen in Tabellen zusammen und räumt von Zeit zu Zeit den Schreibtisch wieder auf, indem man nicht mehr benötigte Notizen in den Papierkorb wirft.

Tabelle 3:

```
Programmbeispiel zu Autopit:
Listing
```

```
PARTNO/4848 FLANSCH
MACHIN/1
REMARK/BESCHREIBUNG DES ROHLINGS
CONTUR/BLANK
BEGIN/-3,40,YLARGE,PLAN,-3
RGT/DIA,150
RGT/PLAN,57
RGT/DIA,80
TERMCO
REMARK/BESCHREIBUNG DES ROHLINGS
CONTUR/PART
SURFIN/FIN
M10,N10,BEGINN/13,77.5,XSMALL,DIA,85,BEVEL,1,FIT,3,OSETNO,2
M0,RGT/PLAN,0
L2=LINE/0,68.9,ATANGL,60,
M1,RGT/L2
M2,RDT/DIA,146,RGH,FIT,4,OSTENO,1
L4=LINE/0,68.5,80,ATANGL,-75
M3,RGT/L4,FEED,0.8
M4,N5,RGT/PLAN,26,ROUND,5
M6,N7,LFT/DIA,120,BEVEL,3,FIT,3
M8,N9,RGT/PLAN,56,BEVEL,1
L9=LINE/53.43.1,15,42.5
RGT/L9
TERMCO
REMARK/TECHNOLOGISCHE DEFINITIONEN
PART/CTR,250,TORLIM,100
CLAMP/NORMAL,DRL,3,ZPS,15
CHUCK/1,DCL,40,MHJ,60,DJ,300
PPRINT/WERKSTUECK AM AUSSENDURCHMESSER EINSPANNEN
REMARK/BEARBEITUNGSDEFINATIONEN
A1=FACE/RGH,TOOL,112,9,SPEED,145
A2=TURN/RGH,TOOL,113,9,SPEED,145
A4=CONT/RGH,TOOL,114,9,SPEED,145
A5=TURN/FIN,TOOL,212,11,SPEED,120
A6=CONT/FIN,TOOL,214,11,SPEED,80
REMARK/BEARBEITUNG DES WERKSTUECKES
CUT/A1,M8,TO,N9
CUT/A4,N9,TO,N10
CUT/A2,M8,RE,M3
CUT/A4,M8,RE,M3
CUT/A6,N9,RE,M3,FEED,.01
CUT/A6,N9,TO,M10,FEED,.05
CLAMP/INVERS;DRL;56,ZPS,20
PPRINT/WERKSTUECK UMSPANNEN
B1=THREAD/TOOL,217,1,SPEED,120
CUT/A2,M2,TO,M3
CUT/A1,M1,RE,M0
CUT/A6,M0,TO,M3,FEED,.05
CUT/A5,M10,RE,M10,FEED,.02
CUT/B1,M2,TO,M3,DEPTH,2.0,DIRECT,25,PITCH,2,1,PASSES,6,1
FINI/
```

Dieser Arbeitsumgebung entspricht in APL der aktive Workspace (Arbeitsbereich).
Jedem APL-Benutzer steht ein solcher Workspace mit jeweils 32 K-Byte Speicherplatz zur Verfügung. Hierin finden alle Eintragungen, Programmabläufe, Löschungen usw. statt. Werden Programme gestartet oder soll auf Variablen zugegriffen werden, so müssen diese im aktiven Workspace vorhanden sein. Verschiedene Workspaces werden in Bibliotheken gespeichert. Jedem Benutzer ist eine Bibliothek

DATENSCHUTZ BLEIBT GEWAHRT

zugeordnet, mit der Möglichkeit einer Schutzkennung. Die hier abgelegten Workspaces werden als passiv bezeichnet. Aktiv ist immer nur eines – das aktuelle Workspace. Dieses befindet sich im Arbeitsspeicher des Rechners (bei mehreren Anwendern auch mehrere Workspaces), während sich die passiven Workspaces immer auf externen Datenträgern befinden (Festplatte o.ä.). Workspaces können vor dem Zugriff anderer Benutzer geschützt werden. Auf ungeschützte Workspaces können alle Benutzer zugreifen, allerdings nur lesend, nie schreibend.
Der Inhalt des aktiven Workspace kann mit Hilfe von Systemanweisungen geändert werden. Außerdem existieren Anweisungen für die Manipulation der Workspace-Bibliotheken und für die Datenübertragung zwischen verschiedenen Benutzern.
Alle Systemanweisungen beginnen mit einer Klammer ")", nach links offen. Zwischen den Systemanweisungen und den APL-Ausdrücken muß streng unterschieden werden; so dürfen diese beiden Befehlstypen nicht gemischt werden.
Jedem Workspace muß

Tabelle 4:

```
-------------------------------------------------------------------------------
Programmbeispiel in BASIC
in Standard-BASIC
Listing-Ausschnitt
-------------------------------------------------------------------------------
5 REM ********************************************* DIMENSIONIERUNG 100 PROBEN *
10 DIM P(100)
20 GOSUB 500
25 REM ************************************************** BILDSCHIRM LOESCHEN *
30 PRINTCHR$(147)
40 PRINT"WIEVIELE PROBEN SOLLEN UNTERSUCHT WERDEN"
45 PRINT
50 INPUT N
60 FOR I = 1 TO N
70 IF I = 9 THEN 90
80 PRINT I;PRINT TAB(2) "PROBE":GOTO 100
90 PRINT I;PRINT TAB(3) "PROBE"
100 INPUT P(I)
110 NEXT I
120 PRINT "ANZAHL DER PROBEN   =" N
130 REM ************************************************** ORDNEN DER PROBEN *
140 LET L = N - 1
150 FOR J = 1 TO L
160 IF P(J) = P(K) THEN GOTO 190
170 LET C = P(J)
180 LET P(J) = P(K)
190 NEXT K
200 NEXT J
210 PRINT "KLEINSTER WERT      =" P(N)
220 PRINT "GROESSTER WERT      =" P(I)
230 REM ********************************************************* MITTELWERT *
240 LET M = 0
250 FOR I = 1 TO N
260 LET M = M + P(I)
270 NEXT I
280 LET M = M/N
290 PRINT "MITTELWERT          =" M
300 LET X = 0
310 FOR I = 1 TO N
320 LET X = X + (P(I) - M).2
330 NEXT I
340 REM ************************************* VARIANZ UND STANDARTABWEICHUNG *
350 LET S2 = 1 / (N - 1) * X
360 LET S = SQR (S2)
370 PRINT "VARIANZ             =" S2
380 PRINT "STANDARTABWEICHUNG  =" S
390 REM ************************************************** VARIANZKOEFFIZIENT *
400 LET V = (S / M) * 100
410 PRINT "VARIANZKOEFFIZENT   =" V "%"
420 GET R$ : IF R$ = "" THEN GOTO 420
430 REM ***************************** AUSDRUCK DER PROBEN IN GEORDNETER REIHE *
440 FOR I = 1 TO N
450 PRINT P(I)
460 NEXT I
500 PRINT "TEST"
510 RETURN
-------------------------------------------------------------------------------
```

ein Name gegeben werden, unter dem es aufgerufen und bearbeitet oder aber abgespeichert und in der Bibliothek wieder aufgefunden werden kann. Der Name des Workspace kann jeder beliebige in APL zugelassene sein, allerdings sind nur die ersten elf Zeichen relevant für die Identifikation. Der Workspace in APL besteht in der Regel aus Programmen und Daten (in Form von Variablen). Jede Funktion und jede Variable erhält einen eindeutigen Namen. Die Namensgebung unterliegt strengen Definitionsregeln. Zusätzlich ist im Workspace ein Bereich für Symboltabellen reserviert, in dem alle Namen der vorhandenen Objekte eingetragen werden. In der Regel sind dies nicht mehr als 256 Eintragungen. Diese Voreinstellung kann jedoch geändert werden.

TASTATURBELEGUNG NOTWENDIG

Namen können zu Gruppen zusammengefaßt werden, welche ihrerseits wieder mit Namen gekennzeichnet werden. Auch hier sind genaue Regeln zu beachten. Namen in diesen Listen müssen nicht zwangsläufig auch belegt sein. Diese Namensliste ist manipulierbar mit verschiedenen Befehlen. So kann man sich zum Beispiel eine Liste alphabetisch ordnen lassen, getrennt nach Variablen, Funktionen oder Gruppen.

APL benötigt eine besondere Tastatur oder zumindest eine Neubelegung der Tasten, denn APL arbeitet mit einer ganzen Reihe von Sonderzeichen. Diese sind in erster Linie Definitionen von APL-Funktionen (wie zum Beispiel das Wurzelzeichen in der Mathematik).

APL kennt keine speziellen Ein- und Ausgabeanweisungen. Daten werden zusammen mit den Operatoren eingegeben. Alphanumerische Daten

werden in Hochkommas gesetzt. Eingaben werden stets um sechs Zeichen nach rechts eingerückt eingegeben. Ausgaben erfolgen in der ersten Spalte beginnend.
Verschiedene Rechnerimplementationen unterstützen die Hard- und Softwareeigenschaften spezieller Computer.
APL-Plus für den Apple-Macintosh beispielsweise verwaltet Pull-Down-Menüs, Ikondarstellung, grafische Eigenschaften sowie Mauseingabe.

6. APS
Assembly **P**rogramming **S**ystem.
Maschinenorientierte Programmiersprache.
APS ist eine spezielle Ausführung von Assembler (siehe dort).

7. APT
Automatically **P**rogrammed **T**ools.
Anwendung nur im Maschinenbau zum numerischen Steuern von Werkzeugmaschinen. Es gibt viele Versionen von APT, die alle in erster Linie von den zu steuernden Maschinen abhängen.
Beispiel: die Teilsprache EXAPT. Sie wird für Maschinen eingesetzt, die bohren, fräsen und drehen.
APT und EXAPT sind die wichtigsten Sprachen zur Steuerung von NC-Maschinen.
APT ist besonders für Mehrachsprobleme geeignet.
Die gewünschte Bahnkurve wird in Polygonpunkte zerlegt, die geradlinig verbunden werden. Die räumliche Kurve wird somit durch lineare Interpolation erzeugt.
APT wurde 1963 in den USA erstellt und zwei Jahre später in Deutschland zu EXAPT weiterentwickelt.
Seit 1967 besteht in Deutschland ein „APT und EXAPT-Verein" in Aachen.

8. Assembler
Maschinenorientierte Programmiersprache.

Tabelle 5:

```
-------------------------------------------------------------
Programmbeispiel zu C
Strukturiertes Listing
-------------------------------------------------------------
/*      Primzahlenprogramm "Sieb des Erathostenes"
        (nach BYTE 1/1983, mit Korrekturen)
*/

$include (stdio.h)

$define TRUE            1
$define FALSE           0
$define DURCHL          250
$define MENGE           8190

char flags(MENGE +1) = (0);

main () (

    register int i, k;
    static int lauf, zaehler;

    printf("%d Durchlaeufe: ", DURCHL);
    for(lauf = 1;lauf(= DURCHL; lauf++) (
            zaehler = 0;
            for(i = 0; i = (= MENGE; i++)
                flags(i) = TRUE;
            for(i = 2; i (= MENGE; i++) (
                if(flags(i)) (
                    for(k = i + i; k (= MENGE; k += i)
                        flags(k) = FALSE;
                    zaehler++;

                    )
                )
            )
    printf("%d Primzahlen/n" , zaehler);
    exit (0);
    )
-------------------------------------------------------------
```

Assembler-Befehle sind dem eigentlichen Maschinencode sehr ähnlich: Jeder Befehl, jedes Sprachelement entspricht genau einem Maschinenbefehl.
Die Struktur der einzelnen Befehle ist primitiv einfach: Befehlscode, Adresse 1, Adresse 2, Adresse 3. Es gibt Ein-, Zwei- und Drei-Adreß-Assembler, je nachdem, wie viele Adressen hinter dem eigentlichen Befehl stehen können.
Liegt ein sogenanntes Ein-Adreß-System vor, wird ein Akkumulator (zentrales Register) benötigt. Jeder Befehl steuert den Akkumulator an und verknüpft dessen Inhalt mit der im Befehl angegebenen Adresse.
So müssen selbst einfache Anweisungen bei der Programmierung schrittweise zergliedert werden (siehe Tabelle 1).
Zwei- und Drei-Adreß-Assembler sind entsprechend kompakter und schneller in der Ausführung, setzen aber einen anderen Maschinentyp voraus.
Für den Befehlscode sind Kürzel eingeführt worden, die sogenannten mnemotechnischen Abkürzungen (Mnemotechnik = Ersetzen von Zeichenketten durch Worte und Kürzel, die Bezug zur Bedeutung der Zeichenkette haben).
Programme müssen mit einem Assembler-Programm (ähnlich einem Compiler oder Interpreter bei anderen Sprachen) übersetzt werden, was allerdings durch die Kürze und Einfachheit der Befehle sehr rasch geschieht.
Das Assembler-Programm liefert zu dem vom Programmierer geschriebenen Quellprogramm ein Objektprogramm. Für den umgekehrten Fall kann man einen Disassembler verwenden, der den Maschinencode (Objektprogramm) wieder in lesbare mnemotechnische Befehlsfolgen umwandelt (Quelle).

ASSEMBLER BRINGT GESCHWINDIGKEIT

Hierin sind auch die Vorteile von Assembler zu sehen: die Ausführungsgeschwindigkeit der Programme und der geringe Speicherplatzbedarf. Beides Vorteile, die im Vergleich zu anderen Sprachen sehr ins Gewicht fallen können, vor allen Dingen bei kleineren Computeranlagen.
Neben diesen Vorteilen hat Assembler allerdings sehr schwerwiegende Nachteile aufzuweisen: Assembler-Programme sind nicht zu lesen, da sie nicht mit Befehlen geschrieben werden, welche sofort ihre Wirkung auf den Programmablauf erkennen lassen, sondern in Form eines Code.
An dieser Stelle sollte vielleicht einmal gesagt werden, was ein Befehl in einem Programm ist: Mit der Bezeichnung Befehl ist fast immer ein Makro-Befehl gemeint. Er enthält meist mehrere Schritte, die für den Pro-

grammierer allerdings nicht interessant sind.

Beispiel: Eine Addition in irgendeiner höheren Programmiersprache

A = B + C

In den meisten Programmiersprachen kann eine Addition in einer solchen Form angegeben werden.
Der Wert der Variablen B soll mit dem Wert der Variablen C addiert und das Ergebnis in der Variablen A gespeichert werden.
Für den Computer ist dies allerdings nicht ein Befehl, sondern gleich ein kleines Bündel von Befehlen:

1. Zunächst muß geprüft werden, wo sich im Speicher die Variablen B und C befinden. Die Stellen sind durch Adressen markiert.
2. Sind die Adressen gefunden, werden die Inhalte (Werte) von B und C gelesen.
3. Der Wert von B muß nun in das Addierwerk gebracht werden.
4. Der Wert von C muß ebenfalls in das Addierwerk übermittelt werden.
5. Dort wird der Wert von B und der Wert von C verrechnet und es entsteht ein neuer Zahlenwert, die Summe aus B und C.
6. Nun wird der Speicherplatz, der für die Variable A reserviert wurde, gesucht.
7. Der Inhalt des Addierwerkes muß nun in diesem Speicherplatz abgespeichert werden.

Will man das Ergebnis der Addition erfahren, so muß der Inhalt der Variablen A abgerufen werden.
In Assembler muß nun jeder Schritt für sich durchgeführt werden, wozu in anderen Sprachen eine kurze Zeile genügt. Außerdem sind andere Sprachen meist genormt, das bedeutet, die Anpassung einer bestimmten Sprache an einen bestimmten Computertyp geschieht mit Hilfe von Compilern oder Interpretern, die zur Ausstattung einer Anlage gehören. Da Assembler aber praktisch direkt mit den Maschinencodes arbeitet, gibt es für jeden Anlagentyp ein anderes Assembler als Sprache und ein anderes Assembler als Programm zur Umwandlung des Quelltextes in den Maschinencode.
Assembler gibt es seit 1953.

9. ATLAS

Anwenderspezifische Realzeitsprache.
ATLAS ist maschinenunabhängig und kann daher als „höhere" Programmiersprache bezeichnet werden, jedoch eignet sie sich vorwiegend zum Testen von Geräten, wozu sie auch eigentlich entwickelt wurde.

10. Autocode

Maschinenorientierte, sehr vereinfache Programmiersprache.
Autocode ist der Vorläufer der Assembler-Sprachen und wurde 1953 entwickelt.
Autocode war durch seine Einfachheit leicht erlernbar. Die Nachteile lagen in der sehr starken Orientierung an der entsprechenden Rechenanlage (im Gegensatz zur Problemorientierung), daher sind die Programme nicht von einer auf eine andere Rechenanlage gleichen Typs übertragbar (wie bei Assembler).
Aus Autocode entstand später Autocoder. Beide Sprachen sind heute kaum noch gebräuchlich.
Assembler hat sich wegen seiner besseren Strukturen durchgesetzt.
Autocode war die Programmiersprache der zweiten Anlagengeneration (etwa 1955 bis 1965).

11. Autopit

Automatisches **P**rogrammieren **I**nclusive **T**echnologie.
Eigenentwicklung der Firma Pittler. Autopit hat deutsche Befehlsfolgen und einige Vorteile in der Materialbearbeitung zu bieten.
Das System von Autopit berücksichtigt Abmessungen und Materialbesonderheiten und trifft daraufhin automatisch Entscheidungen.
Autopit ist maschinenorientiert und kann auch bei weniger großen Anlagen sinnvoll eingesetzt werden, ist aber speziell für einen Maschinentypus der Firma Pittler entwickelt worden.

12. Autopol

Automated **P**rogramming **O**f **L**athes.
Für Drehbanksteuerung und andere Maschinen mit zweidimensionalen Bewegungsabläufen (siehe APT).
Trotz eines sehr kleinen Sprachumfangs können alle erforderlichen Vorgänge gesteuert werden, wodurch Autopol sehr benutzerfreundlich ist.
Das fertig erstellte Programm muß mit Hilfe eines Anpassungsprogrammes übersetzt werden (ähnlich dem Compiler- oder Assemblerprinzip).

13. AUTOPROG

Entwickelt in der CSSR für numerische Steuerung.
Anwendung nur regional.

14. Autopromt

Automated **P**rogramming **O**f **M**achine **T**ools.
Für die Positionierung von Werkzeugen im Maschinenbau (NC-Steuerung).
Diese Sprache zeichnet sich aus durch ihren konkreten Bezug zu der Bezeichnungsweise in ihrem Anwendungsbereich, dem Maschinenbau. Es ist daher relativ einfach, anhand von technischen Zeichnungen das Steuerprogramm für die Werkzeugmaschinen zur Fertigung eines Werkstückes herzustellen (siehe APT).
Da die übliche Terminologie aus dem Maschinenbau in symbolischer Form in die Syntax von Autopromt übernommen wurde, ist diese Programmiersprache besonders benutzerfreundlich und hat auch selbstdokumentierenden Charakter. Autopromt ist daher für die Werkzeugmacher leicht erlernbar.

15. Autospot

Automated **S**ystem **F**or **P**osition **O**f **T**ools.
Für Bohr- und Drehmaschinensteuerung (siehe APT).
Auch Autospot-Programme müssen nach der Erstellung zunächst mittels eines Anpassungsprogramms übersetzt werden. Der vom Computer daraufhin erstellte Lochstreifen wird zur Maschinensteuerung benutzt (NC-Steuerung).

16. BASIC

Beginners **A**ll purpose **S**ymbolic **I**nformation **C**ode.
BASIC wurde ursprünglich entwickelt, um Anfängern den Einstieg in die Programmiertechnik zu vereinfachen, hat sich aber durch die vielseitigen Möglichkeiten der Ein- und Ausgabe von Daten zur Dialogsprache schlechthin entwickelt. BASIC zeichnet sich aus durch gute String-Verarbeitung (Zeichenketten, Texte) und einfach aufgebaute Befehle.

BASIC – PROGRAMMIERSPRACHE FÜR JEDERMANN

BASIC entstand etwa 1964 an der Dartmouth University in den USA unter John Kemeny und Thomas Kurtz.
Es dürfte die wohl bekannteste Programmiersprache sein, nicht zuletzt durch die Technologie der Mikrocomputersysteme, die heute etwa zu 90 Prozent mit BASIC und mit Dialekten von BASIC arbeiten können.
Die ersten Microcomputer (heute würde man sie als Homecomputer klassi-

fizieren) konnten zum Teil sogar nur in BASIC programmiert werden, weil der BASIC-Interpreter fest „verdrahtet" war (ROM).
BASIC ist eine dialogorientierte Sprache, was nicht heißt, daß mit BASIC keine mathematischen Probleme gelöst werden können, da große Ähnlichkeiten zu Fortran und Algol bestehen.
Da BASIC im Gegensatz zu den genannten anderen Programmiersprachen mit einem Interpreter arbeitet, ist das Programmieren, Fehlersuchen und Austesten von Programmen sehr komfortabel und schnell. Ein Editor wird beim Programmieren nicht benötigt, da die Programmzeilen direkt vom Bildschirm in den Interpreter übernommen werden. Noch vor der Ausführung des Programms (oder dessen Interpretation) wird jede Zeile bei der Eingabe auf Syntaxfehler untersucht.

Beispiel:
Eingabe der Zeile
15 PINT A
Meldung des Interpreters
? SYNTAX ERROR
Korrektur der Zeile
15 PRINT A

Der Anwendungsbereich von BASIC übergreift den vieler anderer Programmiersprachen.

SCHWIERIGE STRUKTURIERUNG

Der größte Nachteil von BASIC für den Programmierer ist die schlechte Lesbarkeit von umfangreicheren fertigen Programmen, da es keine strengen Strukturierungsmodi gibt, an die sich der Programmierer halten muß, wie in anderen Programmiersprachen. So ist die Gefahr der Verschachtelung von Programmen weitaus höher als etwa bei Pascal, Fortran oder Cobol.
Daher wird BASIC gerne nachgesagt, es forme beim Anwender einen schlechten Programmierstil und sei daher für Anfänger nur bedingt geeignet.
In BASIC müssen keine Variablennamen deklariert werden, jedoch sind die Namen nur zwei Zeichen lang, was die Lesbarkeit der Programme erschwert.
Die BASIC-Syntax selbst ist leicht verständlich und die meisten Befehle sind selbstdokumentierend.
Unterprogramme werden in BASIC nicht mit einem Label (Namen) versehen, sondern können direkt angesteuert werden. Nachteil: Fehlerquelle und Unübersichtlichkeit. Außerdem werden Parameter immer automatisch den Unterprogrammen zur Verfügung gestellt, was die Flexibilität einschränkt.
BASIC-Dialekte sind zum Beispiel:
BASIC 80, Integer BASIC, Microsoft BASIC, BASIC 86, Advanced BASIC, XY-BASIC, EXTRABASIC, C-BASIC-2, Multiuser BASIC, Level III BASIC, Better BASIC, usw., die Reihe könnte beliebig fortgesetzt werden. Nur wenige BASIC-Versionen bringen wirklich interessante Neuerungen, wie GfA-BASIC: Hierbei handelt es sich um einen BASIC-Dialekt mit Pascal- und C-Elementen. GfA-BASIC arbeitet ohne Zeilennummern und kann compiliert und im Direktmodus angewendet werden, ähnlich dem True BASIC für den Amiga.

Tabelle 6:

```
------------------------------------------------------------
Programmbeispiel zu Cobol
------------------------------------------------------------
PROCEDURE DIVISION
   OPEN INPUT FILES
   OUTPUT PRINTER
   READ FILE8
FOLLOWING
   IF NUMBER IN B IS EQUAL LAST
   GO TO END ELSE NEXT SENTENCE.
   MOVE NUMBER IN B TO NUMBER IN
   ACAT, INTER
CONTINUE
   ADD TOTAL TO S IN B
   GIVING TO TOTAL
   READ FILE8
   IF NUMMBER IN P IS EQUAL INTER
   GO TO CONTINUE, ELSE NEXT SENTENCE
   MOVE TOTAL TO GRAM IN ACAT
   WRITE ACAT
   GO TO FOLLOWING
END
   CLOSE FILE8,PRINTER
   STOP RUN
------------------------------------------------------------
```

Die Standard-BASIC-Syntax:

Anweisungen
LET
GOTO
STOP
END
REM
IF..THEN
FOR..NEXT
GOSUB
RETURN
DIM
DEF
RESTORE

Standardfunktionen
SQR Wurzelfunktion
ABS Absolutfunktion
SGN Vorzeichenfunktion
LOG Logarithmusfunktion
EXP Exponentialfunktion
SIN Sinusfunktion
COS Cosinusfunktion
TAN Tangensfunktion
ATN Arcustangensfunktion
INT Ganzzahlfunktion
RND Zufallszahlenfunktion

Systembefehle
LIST
NEW
RUN
SAVE
LOAD

E/A Anweisungen
READ
OPEN
CLOSE
DATA
INPUT
GET
PRINT
PRINT USING

Operatoren
\+ Addition
– Subtraktion
* Multiplikation
/ Division
↑ Potenzierung

AND Konjunktion
OR Disjunktion
NOT Negation

Obwohl BASIC starke Konkurrenz bekommen hat, ist es immer noch *die* Programmiersprache für den Anfänger. Wer autodidaktisch arbeitet, also sich selbst Programmierkenntnisse aneignen will, wird in der Regel auf BASIC zurückgreifen. Fast alle Homecomputer arbeiten standardmäßig mit BASIC-Dialekten.
BASIC-Interpreter benötigen wenig Speicherplatz und sind bei den meisten Home-Computern in ROMs (Read Only Memory) untergebracht, sonst auf Diskette verfügbar.
Zur Codeoptimierung gibt es auch BASIC-Compiler. Durch sie wird ein geringer Geschwindigkeitsgewinn erzielt. Im professionellen oder semi-professionellen Programmbereich ist BASIC selten zu finden.

Seit Mitte 1987 gibt es eine weitere Version mit dem Namen True BASIC (das wahre BASIC). Seiner Entwicklung geht eine Geschichte voraus: Viele BASIC-Dialekte, die aufgrund der eingeschränkten Möglichkeiten der verfügbaren Hardware entwickelt wurden, ließen eben auch nur eingeschränkte Möglichkeiten der Programmiersprache und der damit erstellten Software zu. Zu dieser engen Verknüpfung von BASIC mit der Hardware eines bestimmten Computers kam eine große Einschränkung der Portabilität, obwohl gerade diese ein Hauptkriterium für die Entwicklung von BASIC war.

TRUE BASIC: BESSER ALS DER STANDARD

Die hier genannten Nachteile, insbesondere die Einschränkungen aufgrund der Hardwarevoraussetzungen, wurden BASIC im allgemeinen nachgesagt, obwohl sie im Grunde nicht der Sprache an sich anzulasten waren.
Die Entwickler hatten weder eine unübersichtliche, noch eine schwerfällige Sprache konzipiert. Daher wurde True BASIC geschaffen. Hiermit haben die ursprünglichen Entwickler von BASIC, Kurtz und Kemeny, versucht, die damaligen Vorstellungen von BASIC wieder zu realisieren, da heutige Microcomputer wesentlich bessere Voraussetzungen für die Realisation dieses Projektes besitzen. True BASIC entspricht vollkommen dem National Standard For BASIC und garantiert damit ein hohes Maß an Portabilität. Es ist ein Versuch der Rehabilitation dieser Sprache.

17. BCPL

BASIC Combined Programming Language. Assemblerähnliche Programmiersprache. Vorläufer von C (siehe dort).

1969 auf den Markt gebracht, verbindet BCPL die Struktur von Algol mit der Leistungsfähigkeit von Assembler.
BCPL wurde von Martin Richards am Massachusetts Institute of Technology entwickelt. Bis Ende der 60er Jahre wurde BCPL überwiegend und mit Erfolg bei der Systemprogrammierung eingesetzt.
Am Bell Laboratorium in New Jersey entwickelte Ken Thompson hieraus die typenlose Sprache B. BCPL war ebenfalls typenlos, wenn der Typ des Maschinenwortes unberücksichtigt bleibt.
Aus B, einer aus BCPL abgeleiteten problemorientierten Programmiersprache, wurde Mitte der 70er Jahre C entwickelt.

18. BLISS-32

Systemimplementierungssprache für VAX-Systeme. BLISS-32 unterstützt die Entwicklung modularer Software nach dem Konzept der strukturierten Programmierung.
Durch die Möglichkeit des direkten Zugriffes auf alle Hardware-Komponenten der VAX-Systeme wird damit die Programmierung von Echtzeit- und Hardware-abhängigen Anwendungen erleichtert.
BLISS-32 wird hauptsächlich zur Entwicklung von Betriebssystemen, Compilern, Laufzeitsystembausteinen, Datenbanksystemen, Kommunikationssoftware und Dienstprogrammen eingesetzt.
Die Version VAX-BLISS-32 erlaubt insbesondere die Entwicklung von systemnaher Software für die VAX. Die Basisfunktion der VAX-Architektur können direkt angesprochen werden.

19. C

Weiterentwicklung von BCPL.
C ist eine Systemimplementierungssprache. Neben der Programmierung von Betriebssystemen wird C aber auch in technisch-wissenschaftlichen Bereichen eingesetzt, da ihr Niveau dem von Algol, PL/1 oder Pascal angeglichen ist. C ist stark blockstrukturiert.
C gibt es seit 1972 und wurde von Dennis Ritchie aus der Systemimplementierungssprache B durch Erweiterungen entwickelt, zum Beispiel durch Einfügen von Datentypen.
Die Sprache C zeichnet sich vor allem durch ihre hohe Verarbeitungsgeschwindigkeit und ihre Maschinennähe trotz hohem Programmniveau aus.
Der Grad der Komplexibilität des Programmcodes ist frei wählbar, wobei sich Lesbarkeit und Schreibaufwand natürlich konträr verhalten.
Weitere Vorteile liegen in der weniger strengen Abfassung als etwa PL/1, was aber zugleich einen großen Nachteil mit sich bringt: Die Fehlersuche wird erheblich erschwert und kompliziert hierdurch und durch einige andere Hindernisse das Erlernen der Sprache und deren Anwendung beträchtlich.

C: DIE NÄCHSTE GENERATION

C ist sicherlich eine wichtige Entwicklung auch für die zukünftigen Rechnergenerationen, mit Sicherheit aber keine Sprache für Anfänger.
C ist Hardware-unabhängig und daher universell einsetzbar.
Die Leistungsfähigkeit von C konnte zum ersten Mal 1972 unter Beweis gestellt werden. In den Bell Laboratories wurde zu dieser Zeit das Betriebssystem Unix vorgestellt. Dieses mittlerweile weit verbreitete Betriebssystem ist vollständig in C programmiert.
C ist heute die Standardsprache für professionelle Software-Entwicklungen in allen Bereichen. Kompaktheit, Effizienz und Schnelligkeit, sowie ein hohes Maß an Portabilität zwischen verschiedenen Hardware-Konfigurationen und Betriebssystemen zeichnen C aus.
Die Maschinennähe von C-Programmen – bei gleichzeitiger Hardware-Unabhängigkeit – läßt die Erstellung universell lauffähiger Programme zu. Zeitaufwendige Implementierungen für andere Rechnertypen können praktisch entfallen.
Die Hochsprache C ist sicherlich nichts für Anfänger. Um hier sinnvoll und routiniert mit den Sprachmitteln umgehen zu können, bedarf es einigen Aufwandes. Über die Grundsätze der Software-Entwicklung sollte man Bescheid wissen, bevor man sich mit C befaßt.
Von Vorteil ist sicherlich die Kenntnis einer anderen höheren Programmiersprache.

SCHWIERIGE SYNTAX

Die Schwierigkeiten der Anwender resultieren unter anderem aus der schlechten Lesbarkeit von C-Programmen und der unter Umständen schwierigen Fehlersuche.
Die C-Syntax ist für Fehler geradezu prädestiniert, da manche Symbole je nach Zugehörigkeit zu anderen Operatoren auch andere Bedeutungen erhalten. Das Minuszeichen etwa bedeutet sowohl den zweistelligen Subtraktions-Operator, als auch das einstellige Minus.
Viele Operationen bestehen zudem aus mehr als einem Zeichen. Eine Zeichenfolge kann, je nach Kontext, also auf die eine oder andere Art interpretiert werden. Die Zeile: A + + + B hat in C die Bedeutung (A + +) + B, könnte aber im Programm auch ganz anders gelesen werden, etwa als A + (+ + B), was in C durchaus zugelassen ist; jedoch liefern beide Zeilen verschiedene Ergebnisse, was vom Programmierer nicht gewollt sein kann.
Für Microcomputer gibt es bereits eine Reihe von

Implementierungen für C, so zum Beispiel:

Turbo-C von Borland und Heimsoeth,
Quick-C von Microsoft,
Zorland-C von Zorland,
Aztec-C von Manx.

Kaum hat sich C auf dem Markt etabliert, gibt es schon einen Nachfolger: C plus plus oder, um bei der Syntax von C zu bleiben C++.
Wie das alte C wurde auch C++ in den Bell Laboratories entwickelt, allerdings von Bjarne Stroustrup. C++ erschien zum ersten Mal 1983 unter dem Namen: C with Classes. C++ ist aufwärtskompatibel, das heißt, „alte" C-Programme laufen auch unter C++.
Dieses Konzept wurde zum ersten Mal konsequent bei Betriebssystemen durchgeführt. Bekanntestes Beispiel ist hier MS-DOS.

20. CDL/2

Systemimplementierungs-Sprache SIL (siehe SL/3). SIL-Sprachen sind zwar Code-effizient, aber recht unbequem in der Programmierung.

21. Cobol

Common **B**usiness **O**riented **L**anguage.
Sprache für den kaufmännischen Bereich.
Die Sprache Cobol hat ihre Stärken dort, wo Fortran Schwächen aufweist: in der Handhabung großer Datenmengen und dem Aufbau von Dateien.
Cobol gibt es seit Ende der 50er Jahre und wird ständig überarbeitet: vom **A**merican **N**ational **S**tandards **I**nstitute zu ANSI-Cobol oder von Codasyl, der **C**onference of **D**ata **S**ystems **L**anguage.
Cobol wurde wie Ada im Auftrag des US-Verteidigungsministeriums entwickelt.
Codasyl ist ein Gremium zur Weiterentwicklung von Programmiersprachen, im besonderen von Cobol.
In Deutschland ist Cobol in der DIN 66028 festgelegt: „Informationsverarbeitung; Programmiersprache Cobol".
Cobol kommt besonders dort zum Einsatz, wo große Mengen von kommerziellen Daten verarbeitet werden sollen. Es gehört zu den sogenannten selbstdokumentierenden Programmiersprachen. Das bedeutet, die Programme sind leicht lesbar, was die teilweise umständliche Programmierung ausgleicht. Durch die guten selbstdokumentierenden Eigenschaften ist Cobol leicht erlernbar und auch von Nichtprogrammierern gut anzuwenden.
Cobol ist in starkem Maße rechnerunabhängig, also eine echte problemorientierte Sprache. Der Einsatz fand bisher meist auf größeren Anlagen statt. Heute gibt es jedoch leistungsfähige Compiler für viele Kleinrechner und Microcomputer. Ab 1985 beginnt Cobol, auch die MS-DOS-Welt auf dem IBM-PC zu erobern. Es gibt bereits sehr gute Compiler für IBM-kompatible Personal Computer. Diese Programme sind sogar portabel auf Großrechneranlagen.

Cobol-Programme bestehen immer aus vier *Programmteilen* (Divisions). Dadurch wird eine starke Strukturierung erreicht:

1. Erkennungsteil –
   Identification Division.
   Hier stehen Angabe zur Kennzeichnung des übersetzten Programmes in Form einer Liste.
2. Maschinenteil –
   Environment Division.
   Es muß angegeben werden, auf welcher Anlage dieses Programm laufen soll und welche peripheren Geräte vom Programm angesprochen werden. Hierunter fallen auch Dateien.
3. Datenteil –
   Data Division.
   Die Form der Daten, die vom Programm verarbeitet werden, muß eindeutig festgelegt werden, außerdem die Speicherbereiche und ob es sich um Variablen oder Konstanten handelt.
4. Prozedurteil –
   Procedure Division.
   Hier werden alle Abläufe und Operationen beschrieben, die ausgeführt werden sollen.

Im letzten Teil, dem Prozedurteil, steht das eigentliche Cobol-Programm. Die übrigen Divisions definieren lediglich den Programmnamen, die Peripherie (im Programm angesprochene Geräte wie Drucker, Floppy) und die Dateien und Speicherbereiche der Daten.
Das Standard-Cobol enthält folgende Befehle:

Operatoren:
ADD
SUBTRACT
MULTIPLY
DIVIDE
COMPUTE

Steueranweisungen:
IF
ALTER
PERFORM
GO
STOP

E/A-Anweisungen:
READ
WRITE
ACCEPT
DISPLAY
OPEN
CLOSE

Übersetzer-Anweisungen:
USE
ENTER
EXIT
NOTE

Daten-Übertragung:
MOVE
TRANSFORM
EXAMINE

In Cobol können sogenannte Wörter definiert werden. Ein Wort besteht aus maximal 30 Zeichen: Zahlen 0 bis 9, Zeichen A bis Z.

Beispiele:
Brutto-Lohn
Netto-Lohn
Personenkennziffer.

Leerzeichen dürfen nicht in Wörter geschrieben werden, denn sie dienen der Trennung.
Außer den frei wählbaren Wörtern gibt es die Cobol-Wörter. Sie sind bereits definiert und haben feste Bedeutungen. Es gibt drei Arten von Cobol-Wörtern:

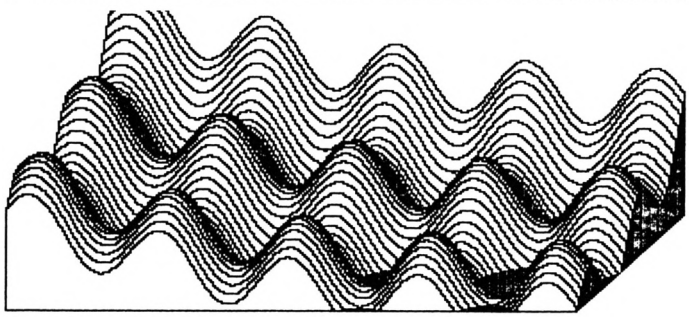
1. Bindewörter
   Sie werden dazu verwendet, die Anwesenheit eines Kennzeichens anzuzeigen (In und Off), bzw. Boolsche Operatoren zu bilden (And, Or, Not . . .)
2. Wahlwörter
   Sie dienen der besseren Dokumentation der Programme und steigern so die Lesbarkeit. Das Wahlwort ändert nichts an der Interpretation einer Anweisung durch den Übersetzer, muß aber der vorgegebenen Syntax entsprechen.
3. Schlüsselwörter
   Sie sind notwendig zur Vollständigkeit einer Erklärung und müssen so verwendet werden, wie in den Formaten angegeben.

Wie in anderen Programmiersprachen gibt es zwischenzeitlich auch in Cobol einige Versionen und Dialekte.
Wie kaum eine andere Programmiersprache wurde Cobol erweitert und verbessert. Nur so ist zu erklären, daß sich Cobol über Jahrzehnte auf dem Markt behaupten konnte.
Hier die chronologische Entwicklung von Cobol:
Cobol 59
Cobol 60
Cobol 61
Cobol 63 (extended)
Cobol Edition 65
Cobol 68
Cobol 69
Cobol 70
Cobol 73
ANSI 74 Cobol
Cobol 80
ANSI 85 Cobol.
Die vorläufig letzte Version von Cobol, der ANSI-85-Standard, hat der betagten Programmiersprache einen neuen Charakter gegeben.
Die Programme sind übersichtlicher und die Cobol-Syntax wurde entrümpelt. Dennoch finden

Tabelle 7:

Programmbeispiel zu Comal 80
Listing

```
 10 DIM  ANTWORT$ OF 10
 20 PRINT "WEITER MIT START, CONT ODER STOP"
 30 INPUT ANTWORT$
 40 CASE ANTWORT$ OF
 50 // ES WURDE 'START' EINGEGEBEN
 60 WHEN "START"
      GOSUB PROC STARTEN
 70 // ES WURDE 'STOP' EINGEGEBEN
 80 WHEN "STOP"
      PRINT "DAS PROGRAMM WURDE BEENDET"; STOP
 90 // ES WURDE 'CONT' EINGEGEBEN
100 WHEN "CONT"
      PRINT "ES WIRD FORTGEFAHREN"
110 GO TO 200
120 // ES WURDE EINE FALSCHE EINGABE GEMACHT
130 OTHERWISE
140 PRINT "SIE HABEN EINE FALSCHE ANWEISUNG GEGEBEN"
150 GO TO 20
160 ENDCASE
200 // HIER WIRD FORTGEFAHREN
```



sich im ANSI 85 Cobol mehr als 40 neue Funktionen.
Es ist auch wichtig, daß sich eine Sprache der technischen Entwicklung anpaßt, denn vor 20 Jahren gab es noch keine Bildschirmterminals, so Bildschirmterminals. So fehlen in alten Programbebefehle für den Bildschirm. Überflüssig auf der anderen Seite sind Lochkartenbefehle – sie können heute aus einer Programmiersprache eliminiert werden. Diese Reihe könnte beliebig fortgesetzt werden und gilt natürlich nicht nur für Cobol.

22. Comal 80

Common **A**lgorithmic **L**anguage.
Comal wurde 1980 in Dänemark von Borge Christensen (State Teachers College) und Benedict Loftsted (Department für Computerwissenschaft der Universität Aarhus) entwickelt.
Die Grundstruktur von Comal 80 ist der des BASIC gleich, jedoch um einige Elemente aus anderen Programmiersprachen, zum Beispiel Pascal, erweitert worden.
Die Ähnlichkeiten mit BASIC sind allerdings wesentlich. Comal 80 ist somit als eine Erweiterung von BASIC anzusehen, wird dieses aber in Zukunft nicht verdrängen können.
Im Gegensatz zu BASIC und vielen anderen Programmiersprachen sind in Comal längere Variablennamen zulässig (maximal 16 Zeichen).
In Comal 80 ist es möglich, Unterprogramme zu benennen und Parameter zu übergeben, was in BASIC nicht machbar ist. Comal 80 ist mehr als BASIC selbst dokumentierend, behält aber alle Vorteile der Dialogsprachen.
Der Befehlsvorrat entspricht dem von BASIC in leicht modifizierter Form, zuzüglich der folgenden Anweisungen:

CASE .. ENDCASE
WHILE .. ENDWHILE
REPEAT .. UNTIL
IF .. THEN ..ELSE

23. Coursewriter

Spezielles System für computerunterstütztes Lernen (CAI=Computer-Aided Instruction).
Im Bereich der Ausbildung tritt der Computer mit Coursewriter-Programmen an die Stelle der Lehrperson.
CAI ist maschinenaufwendig, ebenso die Coursewriter-Programme.
Vorteil der Methode ist, daß der Schüler das Lerntempo individuell bestimmen kann, und auch die Erfolge werden individuell erlebt.

DIE UNBEKANNTEN

Das Erlernen des Systems ist auf minimale Notwendigkeiten beschränkt, so bleibt dem Schüler mehr Raum für den eigentlichen Lerninhalt.
Der Nachteil hieraus wird durch den hohen Aufwand verursacht, die hohen Kosten, wodurch sich Coursewriter nicht durchsetzen konnte.

24. CSL

Control and **S**imulation **L**anguage.
Spezielle Programmiersprache zu Simulationszwecken und Modelluntersuchungen (siehe auch SIMULA).

25. DIPOL

Anwenderspezifische, problemorientierte Echtzeitsprache für die Automatisierung von Stückprozessen.

26. DL

Data **L**anguage.
Abkürzung für verschiedene Datenbanksprachen (zum Beispiel Query Language).

27. Elan

Extentable **Lan**guage.
Elan ist wie C (siehe dort) eine Systemimplementierungssprache. Das Betriebssystem Eumel ist zum Beispiel in Elan geschrieben.
Elan ist kaum verbreitet.

28. EOL

Expression **O**riented **L**anguage.
EOL wurde um 1960 von dem polnischen Mathematiker Jan Lukasiewicz entwickelt.
Aufsehen erregte sie durch die ungewöhnliche Notation zur Formulierung arithmetischer Ausdrücke.
Diese Notation wurde zunächst nach ihrem Erfinder Lukasiewicz benannt.
Aufgrund der schwierigen und ungewöhnlichen Schreibweise dieses Namens wurde diese Notation in den USA umbenannt in PN (Polish Notation).
Bei der PN werden immer zuerst die Operatoren, dann die zugehörigen Operanden geschrieben. Daher wird PN auch als Präfix-Schreibweise bezeichnet.

Beispiel:
5+4, – in PN: + 5 4

PN wird bei einigen Programmiersprachen auch umgedreht angewendet, also zuerst die Operanden und dann der oder die Operatoren. Diese Schreibweise bezeichnet man als RPN, also Reverse Polish Notation, bzw. Postfix-Schreibweise.

Beispiel:
5+4,– in RPN: 5 4 +

Bekannt wurde RPN durch die Taschenrechner von Hewlett Packard.

Dipl.-Ing. FH
Oliver Rosenbaum□

Fortsetzung im nächsten Heft

Kein Spiel!

Erfinder von BASIC-Erweiterungen und ähnlichem stehen immer wieder vor dem Problem, ihre Programme möglichst gut in das BASIC des CPC zu integrieren. Nicht selten verunstalten sie dabei Ihre Programme auf unschöne Weise, da sie versuchen, alle benötigten Unterprogramme selbst zu erstellen. Dabei ist das nicht nötig! Das gesamte Betriebssystem des CPC ist recht offen konzipiert; jeder kann die Routinen selbst nutzen. In dieser Folge der CPC-Trickkiste gibt es hierzu ein paar Beispiele.

Kein Programmierer hat es besonders gern, wenn man in seinen Programmen herumpfuscht. Ein „Break" ist oft ein Ärgernis, zerstört es doch beispielsweise den Bildschirmaufbau.

ABSTURZ VERHINDERN

Zudem gibt es beim CPC-464 keine Möglichkeit, die ESCape-Taste generell zu sperren (beim CPC664/6128 kann der Befehl ON BREAK CONT verwendet werden). Doch es gibt zwei Möglichkeiten, eine Programmbrechung zu vermeiden:

Man definiert die ESCape-Taste um. Dies schützt zwar umfassend vor jeglichem Break in einem BASIC-Programm, verhindert aber keineswegs das Zurücksetzen des Rechners durch den „Affengriff" <SHIFT>+<CTRL>+<ESCape>. Auch die Kassetten-I/O-Routinen können weiterhin mit der ESCape-Taste unterbrochen werden; trotzdem ist diese Möglichkeit empfehlenswert, insbesondere, wenn man eine andere Funktion auf die ESCape-Taste legen möchte:

```
KEY DEF 66,0,&FF,&FF,&FF
```

Nach diesem Befehl ignoriert BASIC die ESCape-Taste. Setzt man anstatt &FF andere Werte ein, etwa 65, wird bei Betätigung der ESCape-Taste das entsprechende Zeichen ausgegeben.

Wer übrigens zu Test-Zwecken die ESCape-Taste zwar umdefinieren, trotzdem sein Programm aber noch unterbrechen können möchte, kann die Break-Funktion auf jede andere Taste legen:

```
KEY DEF 18,0,13,13,&FC
```

Mit diesem Befehl wird die große <ENTER>-Taste zusammen mit <CTRL> zur Break-Taste. Der Code & FC steht also für Break.

Wem diese Möglichkeiten zu „unprofessionell" sind, kann den Vektor in KM TEST BREAK umbiegen. Hierbei sind vielerlei Variationen denkbar. Die einfachste Möglichkeit ist, an die Adresse BDEE ein C9 zu POKEn, welches in Z80-Assembler der Op-Code für den Befehl RET, beende die Routine, ist. In diesem Fall testet der Rechner gar nicht mehr, ob die ESCape-Taste gedrückt wurde oder nicht. Das Unterbrechen von Kassetten-I/O- und von Tastatureingaben über INPUT und LINE INPUT sind jedoch weiterhin möglich.

Allerdings gibt es beim UmPOKEn noch die Möglichkeit, den Rechner beim Druck einer Tastenkombination mit ESCape, also beispielsweise <SHIFT><ESCape>, zu einer bestimmten eigenen Routine zu schicken, etwa zu einer Hardcopy, die den aktuellen Spielstand dann auf den Drucker bringt.

Möchte man sein Programm allerdings nur vor Unterbrechungen schützen, so kombiniert man am besten beide Methoden: Man definiert die ESCape-Taste um und POKEd C9 nach BDEE.

Die meisten RSX-Programmierer lösen das Problem der Parameterkontrolle und des Fehlerabfangs auf höchst unbefriedigende Weise: Die einen schreiben irgendeine Fehlermeldung auf den Bildschirm, was nicht besonders schön ist, denn zum einen läßt sich diese Fehlermeldung im Programm nicht abfangen, da ON ERROR GOTO ja nur auf die BASIC-Fehler reagiert, zum anderen ist die Meldung „Fehler im Befehl!" oder ähnliches ja nicht besonders informativ.

Die anderen schreiben bei einem Fehler einen Code in eine Speicherstelle, die der Benutzer dann gefälligst auszulesen und zu analysieren hat.

Um ehrlich zu sein: Mir gefällt keine dieser Möglichkeiten besonders gut. Mein Lösungsvorschlag: Nutzen wir doch die BASIC-Error-Routine! In BASIC gibt es eine Unterroutine, die zu einem bestimmten Fehlercode eine Meldung heraussucht, diese, falls kein ON ERROR aktiv ist, auf dem Bildschirm ausgibt und zurück ins BASIC springt. Die Betonung liegt auf „falls kein ON ERROR aktiv ist", das heißt, die Fehler können von BASIC durch-

Abbildung 1:
Fehlercodes und Meldungen

00	Unknown Error
01	Unexpected NEXT
02	Syntax Error
03	Unexpected RETURN
04	DATA Exhausted
05	Improper Argument
06	Overflow
07	Memory Full
08	Line does not exist
09	Subscript out of range
10	Array already dimensioned
11	Division by zero
12	Invalid direct command
13	Type mismatch
14	String space full
15	String too long
16	String expression too complex
17	Cannot CONTinue
18	Unknown user function
19	RESUME missing
20	Unexpected RESUME
21	Direct command found
22	Operand missing
23	Line too long
24	EOF met
25	File type error
26	NEXT missing
27	File already open
28	Unknown command
29	WEND missing
30	Unexpected WEND
31ff.	Unknown error

aus abgefangen und weiterverarbeitet werden, wie alle anderen Fehler im BASIC auch.

Und wer jetzt immer noch nicht ganz überzeugt ist, der soll es bleiben lassen, auch wenn die BASIC-Fehlermeldungen alle anderen Fehler, nämlich mit einem einfachen CALL ins BASIC-ROM ausgeben.
Adressen:

CA94 beim CPC 464
CB58 beim CPC 664
CB55 beim CPC6128,

aber die Typenunterscheidung dürfte nicht schwerfallen. In CPC-WELT 12/86 wurde gezeigt, wie einfach man den CPC-Typ ausfindig macht.

Das ROM muß zuvor eingeschaltet worden sein (geht mit CALL B900), und die Fehlernummer wird beim CPC464 im Register E, bei den andern CPCs im Register A übergeben. Eine Übersicht aller Fehler und die entsprecenden Fehlernummern finden Sie in Abbildung 1.

Alle Klarheiten beseitigt? Dann braucht man es nicht wie die Programmierer des AMSTRAD-Disketten-ROM zu machen, sondern gibt seine Fehler professionell aus!

Nun, wenn die Error-Routine eines Programms nicht ganz hundertprozentig sitzt, läßt sich das unter Umständen noch verkraften, aber eines bringt mich häufig zur Verzweiflung: Da muß nun in einem Assemblerprogramm eine Eingabe gemacht werden, und wie sieht die dann aus? Da hüpft irgendwo in der Gegend ein Cursor 'rum, die Anzeige flackert, Korrekturen sind sowieso nur mit der DEL-Taste möglich, an COPY CURSOR oder ähnliche Raffinessen ist erst gar nicht zu denken.

Die EDIT-Routine nutzen

Bei solchen Programmen frage ich mich immer: „Konnte oder wollte der das nicht besser machen?". Nun, falls er es nicht konnte, dem läßt sich abhelfen: Das CPC-Betriebssystem bietet jedem eine sehr gute Eingaberoutine, die normale BASIC-Routine nämlich, mit COPY CURSOR und all den anderen Sondertasten (vgl. Abbildung 2). Also in Zukunft die BASIC EDIT Routine nehmen, okay?

Es ist auch ganz einfach: Man sagt der Routine, wo der zu editierende Text steht oder, bei Neueingaben, wo der LOCATE-Befehl wirkt (in der CPC-Sprungleiste). Nach Angabe der richtigen Parameter wird die EDIT-Routine angesprungen.

In Assembler geht das dann wie in Abbildung 3: Man kopiert den zu editierenden Text in einen Zwischenspeicher, schreibt ein Null-Byte an das Ende des Textes und lädt HL mit der Bufferanfangsadresse. Dann ruft man EDIT auf, und zwar mit einem CALL...

BD3A beim CPC464
BD5B beim CPC664
BD5E beim CPC6128

(Typunterscheidung nicht vergessen), und schon läßt sich der Text editieren.

Als Rückgabe bekommt man HL wieder an den Anfang des Buffers, und falls die Eingabe mit der <ESC>-Taste abgebrochen wurde, ist das ZERO-Flag gesetzt.

So, jetzt können Sie die neuen Tricks gleich mal ausprobieren. Nächsten Monat behandelt die CPC Trickkiste das Thema „CPC – Deine Disketten". ■

Abbildung 3:

EDIT-Nutzung unter Assembler

```
BD5E              8 edit:   equ #bd5e       ;Adresse CPC6128
                  9 ;edit:  equ #5d5b       ;Adresse CPC 664
                 10 ;edit : equ #bd3a       ;Adresse CPC 464

8000             13         org #8000
8000             14         ent $

8000 212680      17 start:  ld  hl,text_1   ;Eingabeprompt
8003 CD1D80      18         call print      ;ausgeben
8006 216880      19         ld  hl,inut     ;Zeiger auf Input-Buffer
8009 CD5EBD      20         CALL EDIT       ;EDIT-Routine aufrufen
800C 215F30      21         ld  hl,text_2   ;Bestätigungsmeldung
800F CD1D80      22         call print      ;ausgeben
8012 216881      23         ld  hl,text_3   ;Zweiter Teil und Meldung
8015 CD1D80      24         call print      ;ausgeben
8018 CD06BB      25         call #bb06      ;auf eine Taste warten
801B 18E3        26         jr  start       ;noch einmal
                 27
801D 7E          28 print:  ld  a,(hl)      ;Textzeichen holen
801E B7          29         or  a           ;Endmarkierung ?
801F C8          30         ret z           ;Ja, zurück
8020 CD5ABB      31         call #bb5a      ;Zeichen ausgeben
8023 23          32         inc hl          ;Textzeiger erhöhen
1824 18F7        33         jr  print       ;und weiter
                 34
8026 0402        35 text_1: defb 4,2        ;Mode 2
8028 44656D6F    36         defm "Demonstartion der EDIT-Nutzung"
8047 0D0D0D      37         defb 10,10,13   ;LF,LF,CR
804A 67696520    38         defm "Wie heißen Sie..."
805E 00          39         defb 0          ;Endmarkierung
805F 0D0D0D      40 text_2: defb 10,10,13   ;LF,LFCR
8062 48616c6c    41         defm "Hallo "   ;Antwort
                 42
8068             43 input:  defs 256        ;Input Buffer
                 44
8168 2C207769    45 text_3  defm ",wie geht's denn ?"
817B 0D0D0D0D    46         defb 10,10,13   ;LF,LF,CR
81F7 54617374    47         defm "Taste drücken ..."
8190 00          48         defb 0
```

Beispiele für Ein- und Ausgaberoutinen

Abbildung 2: Sondertasten in der System Eingaberoutine

Pfeil links	**Cursor ein Zeichen weiter**
Pfeil rechts	**Cursor ein Zeichne zurück**
Pfeil hoch	**Cursor eine Zeile hoch**
Pfeil ab	**Cursor eine Zeile tiefer**
DEL	**Zeichen vor Cursor löschen**
CLR	**Zeichen auf Cursor löschen**
CTRL+Pfeil links	**Cursor an das Ende der Bildschirmzeile**
CTRL+Pfeil rechts	**Cursor an den Anfang der Bildschirmzeile**
CTRL+Pfeil hoch	**Cursor an den Anfang der Eingabe**
CTRL+Pfeil ab	**Cursor an das Ende der Eingabe**
CAPS LOCK	**Schalter (ein/aus) für Groß- und Kleinschreibung**
CTRL+CAPS LOCK	**SHIFT LOCK ein-/ausschalten**
CTRL+TAB	**Einfügemodus ein-/ausschalten**
ESC (Escape)	**Eingabe abbrechen**
SHIFT+Pfeiltasten	**Copy-Cursor bewegen**
COPY	**Zeichen auf Copy-Cursor kopieren**
ENTER	**Eingabe beenden**

TRICKREICHE STRINGS

Basic an die Kette gelegt

Die Arbeit mit Zeichenketten ist ein Kapitel, das vielen BASIC-Einsteigern Schwierigkeiten bereitet. Wir zeigen einige Tricks, wie Sie diesen Datentyp effektiv einsetzen können.

Wie man mit der Funktion MID$ Teile aus einem String erhält, ist Ihnen sicher bereits bekannt. So würde etwa das folgende Programm die Ausgabe "CPC" auf dem Bildschirm liefern:

```
10 a$="Schneider
   CPC"
20 b$=MID$
   (a$,11,3)
30 PRINT b$
```

Nur wenige wissen, daß sie MID$ auch als praktischen Ersatz für RIGHT$ benutzen können. Wenn Sie in unserem Beispiel einfach den dritten Parameter, der die Länge der zu kopierenden Zeichenkette angibt, weglassen, erhalten Sie das gleiche Ergebnis. Dadurch ist es möglich, den rechten Teil eines Strings mit unbekannter Länge zu ermitteln. Wenn Sie eine eingegebene Zeichenkette ohne das erste Zeichen ausgeben wollen, könnten Sie dies so formulieren:

```
10 INPUT a$
20 l=LEN(a$)
30 PRINT RIGHT$
   (a$,l-1)
```

Sie können sich jedoch auch eine Programmzeile und eine Variable sparen:

```
10 INPUT a$
20 PRINT MID$
   (a$,2)
```

Wenn Sie dann noch das so unterschlagene erste Zeichen wieder hinten anfügen, können Sie eine Laufschrift auf den Bildschirm bringen. Solche Effekte sind hin und wieder ganz nützlich, um ein selbstgeschriebenes Programm in ansprechender Aufmachung darbieten zu können.

MID$ ALS PSEUDO-VARIABLE

Mit dem Weglassen des dritten Parameters sind die Möglichkeiten, die in der MID$-Funktion stek-



Künstler im Umgang mit den Strings – Der CPC

ken, noch nicht erschöpft. Sie können diese Funktion auch anstelle einer Variablen verwenden. Man bezeichnet sie in diesem Zusammenhang oft auch als Pseudo-Variable.
Definieren Sie sich einen String a$, der vierzig Sternchen enthält, und versuchen Sie folgende Zuweisung:

```
MID$(a$,1,2)="abcd
efgh"
```

Wenn Sie sich anschließend a$ ausgeben lassen, werden Sie feststellen, daß nur die ersten zwei Zeichen durch "ab" ersetzt worden sind. Ist es schon erstaunlich, daß man eine BASIC-Funktion überhaupt als Variable einsetzen kann, so ist es noch erstaunlicher, daß solche Zuweisungen auch sinnvolle Ergebnisse zustande bringen. Der Längenparameter kann hier verwendet werden, um anzugeben, wieviele Zeichen in den String übernommen werden sollen.
Sofern Sie noch nicht weiter experimentiert haben, können Sie jetzt folgende Anweisung ausprobieren:

```
MID$(a$,2,2)
=MID$(a$,1,2)
```

Sie werden feststellen, daß das "a" am Anfang der Zeichenkette sich verdoppelt hat. Das ist kein Wunder, denn mit der genannten Anweisung haben Sie nichts anderes getan, als die ersten beiden Zeichen des alten Strings an Position zwei und drei des neuen Strings zu kopieren. Dadurch bleibt das erste Zeichen erhalten.
Solche MID$-Zuweisungen sind vor allem dann nützlich, wenn Sie in Ihrem Programm zahlreiche String-Variablen verwenden. Um das näher erläutern zu können, müssen wir uns etwas intensiver mit dem Innenleben des CPC auseinandersetzen.

DIE MÜLLABFUHR DES CPC

Für jeden String, der neu definiert wird, legt der CPC zunächst drei Byte im Speicher ab. Das ist der sogenannte String-Descriptor. An welcher Speicheradresse er sich befindet, erfahren Sie mit der Adreß-Funktion, die über das Zeichen "@" erreichbar ist. Setzen Sie Ihren Computer zurück und geben Sie ein:

```
a$="Dies ist ein Test."
```

Die Abfrage "PRINT @a$" liefert Ihnen eine Zahl, die beim CPC464 den Wert 374 haben muß. Geben Sie jetzt ein:

```
PRINT PEEK(374)
```

Es erscheint die Zahl 18 auf Ihrem Bildschirm. Geben Sie jetzt ein:

```
PRINT PEEK(375)
+256*PEEK(376)
```

Wenn Sie einen anderen Wert als 374 erhalten haben, müssen Sie natürlich die darauffolgenden Zahlen verwenden. Der Wert, der jetzt auf Ihrem Monitor angezeigt wird, gibt Ihnen die Speicherstelle an, ab der der eigentliche Inhalt der Zeichenkette abgelegt ist. Wenn wir wieder vom 464 ausgehen, ist dies die Adresse 43886. Setzen wir unser kleines Experiment einmal fort. Geben Sie ein:

```
a$="Und noch'n Test!"
```

Sehen wir uns einmal an, wo sich der String-Descriptor jetzt befindet:

```
PRINT @a$
```

Es hat sich nichts geändert. Der CPC hat erkannt, daß Sie dieselbe Variable (a$ nämlich) neu belegt haben. Aber lassen Sie sich doch einmal die Adresse ausgeben. Auf dem 464 sehen Sie jetzt den Wert 43870. Sie hat sich also nach unten verschoben, und zwar genau um die Länge der neuen Zeichenkette. Dieser Sache wollen wir mit folgendem Testprogramm auf den Grund gehen:

```
10 a$=CHR$(INT
   (256*RND))
20 PRINT FRE(0)
30 GOTO 10
```

Die FRE-Funktion gibt Ihnen an, wieviele Byte an freiem Speicherplatz noch zur Verfügung ste-

hen. Wenn Sie das Programm starten, werden Sie feststellen, daß diese Zahl mit jedem Durchlauf um Eins abnimmt. Und darin liegt des Rätsels Lösung. Für jede Zuweisung, auch an denselben String, verwendet Ihr Schneider-Computer einen neuen Speicherbereich. Nur der „Ablageplatz" für den zugehörigen Descriptor ändert sich nicht.
Diese Methode hat gewisse Vorteile. Da sich die Längen von Strings jederzeit ändern können und die Zeichenketten im Speicher hintereinander liegen, sparen Sie sich das Hin- und Herschieben von Byte. Dadurch beschleunigt sich der Programmablauf erheblich. Dieser Vorteil ist jedoch schnell zunichte gemacht, wenn Sie zu viele Strings einsetzen. Irgendwann ist der Speicherplatz verbraucht, und es geschieht das Unvermeidliche: Die Müllabfuhr rückt an.
Ihr CPC sieht sich die Liste der Descriptoren an und vernichtet all jene Strings, auf die kein Zeiger mehr weist. Offensichtlich werden diese Zeichenketten ja nicht mehr benötigt. Anschließend werden die übriggebliebenen Zeichenketten so dicht wie möglich aneinandergepackt. Und weil sich dadurch ihre Adressen ändern, muß der CPC auch noch die Descriptoren bearbeiten.
Diese gefürchtete "garbage collection" – so der Fachausdruck – läßt sich nur dann vermeiden, wenn Sie möglichst wenig Strings verwenden. In manchen Programmen ist der Einsatz dieses Datentyps allerdings unumgänglich. Hier kann ein überlegter Einsatz von String-Funktionen nützlich sein. Dies soll Ihnen an einem eindrucksvollen Beispiel näher erläutert werden.
Bei den folgenden Angaben zum freien Speicherplatz handelt es sich um Werte, die für den CPC 464 ohne Diskettenlaufwerk gelten. Der Platzverbrauch ist jedoch für jedes CPC-Modell derselbe; nur Anfangswerte können anders liegen.
Wir wollen ein Array aus 100 Strings mit Zeichenketten belegen, die jeweils vierzig Sternchen enthalten. Das könnte zum Beispiel in einem Programm der Fall sein, das den Inhalt einer Textdatei bis zur letzten Zeile auffüllt, um sie zu vervollständigen. Das können Sie mit folgendem Programm machen:

```
10 DIM a$(100)
20 FOR i=1 TO 100
30 FOR j=1 TO 40:
   a$=a$+"*":NEXT j
40 NEXT i
```

Nachdem das Array dimensioniert ist, haben Sie 43103 Byte frei. Wenn Sie es belegt haben, bleiben davon noch 2051 übrig.
Um einen String mit Sternchen zu belegen, haben Sie nämlich vierzig Zuweisungen benötigt. Mit jeder wurde der String um ein Zeichen länger, und der alte Inhalt wurde verworfen.
Die ganze Prozedur wurde einhundertmal durchlaufen. Normalerweise hätten Sie jetzt 78000 Byte verbraucht.
Da der CPC jedoch nicht soviel Platz hat, macht er zwischendurch schon einmal eine kleine garbage collection. Deshalb dauert diese Zuweisung auch so lange.
Ersetzen Sie jetzt die Zeile 30 durch:

```
30 a$(i)=STRING$
   (40,"*")
```

Die Funktion STRING$ erzeugt einen String, der aus der Wiederholung eines einzigen Zeichens besteht. In diesem Fall wollen wir vierzig Sternchen erzeugen.
Sie werden sehen, daß der CPC nicht nur wesentlich schneller arbeitet, sondern Ihnen auch erheblich mehr Speicherplatz übrig läßt. Statt 2051 Byte haben Sie jetzt noch 39120 freie Speicherplätze.
Nehmen wir an, Sie wollen die eingetragenen Sternchen in Leerzeichen umwandeln. So etwas könnte durchaus einmal notwendig werden, zum Beispiel, nachdem Sie in einem Text etwas markiert haben und diese Markierung wieder aufheben wollen. Sie könnten das Listing jetzt einfach ergänzen:

```
50 FOR i=1 TO 100
60 a$(i)=STRING$
   (40," ")
70 NEXT i
```

Nachdem das Programm gelaufen ist, werden Sie feststellen, daß der verfügbare Speicherplatz von ehemals 39054 auf 35054 Byte geschrumpft ist. Das sind genau jene vierzig mal einhundert Zeichen, die Sie neu eingetragen haben. In einem Programm, das Strings intensiver nutzt, wären Sie bereits mit längeren Wartezeiten, oft bis zu zwanzig Minuten, konfrontiert worden. Was also ist zu tun?
Die Lösung dieses Problems steckt in MID$ verborgen. Ersetzen Sie Zeile 60 durch:

```
60 MID$(a$(i),1)
   =STRING$(40," ")
```

Sie werden staunen: Es sind immer noch 39049 Byte frei. Die von Ihnen verwendete Pseudo-Variable legt nämlich keinen neuen String an, sondern beschreibt direkt die von der Zeichenkette belegten Speicherstellen. Deshalb müssen die Strings, die Sie einander zuweisen, die gleiche Länge haben. Ist dies nicht der Fall, so bleiben entweder Teile der alten Zeichenkette erhalten, oder der neue Inhalt wird nicht vollständig übertragen.
Die Pseudo-Variable MID$ hat leider einen kleinen Schönheitsfehler: Sie läßt sich nicht in Zusammenhang mit INPUT verwenden.
Zum Schluß noch ein Tip für Besitzer eines CPC464. Falls Sie den DEC$-Befehl vermissen, der eine hexadezimale oder binäre Zahl in ihr dezimales Äquivalent umwandelt: Es gibt ihn auch auf dem 464.

ES GIBT IHN DOCH: DEC$

Den Autoren des BASIC 1.0 ist dabei jedoch ein kleiner Schönheitsfehler unterlaufen. Und da kein Programmierer gerne zugibt, daß er auch Fehler macht, wurde diese Funktion einfach nicht ins Handbuch aufgenommen.
Die Syntax des DEC$-Befehles auf dem 464 lautet:

```
DEC$((Zahlenwert,
Format)
```

Beachten Sie die Doppelklammer am Anfang. Sie ist nämlich der Schönheitsfehler.
Bei dem anzugebenden Zahlenwert handelt es sich um eine im Hex- oder Binär-Format angegebene Zahl oder um eine beliebige numerische Variable. Der Parameter "Format" ist ein String, wie Sie ihn auch zur Formatangabe bei PRINT USING verwenden. So liefert etwa der Befehl

```
PRINT DEC$((&AB,
"###")
```

die Ausgabe 171. Um zu kontrollieren, ob das richtig ist, können Sie den HEX$-Befehl verwenden:

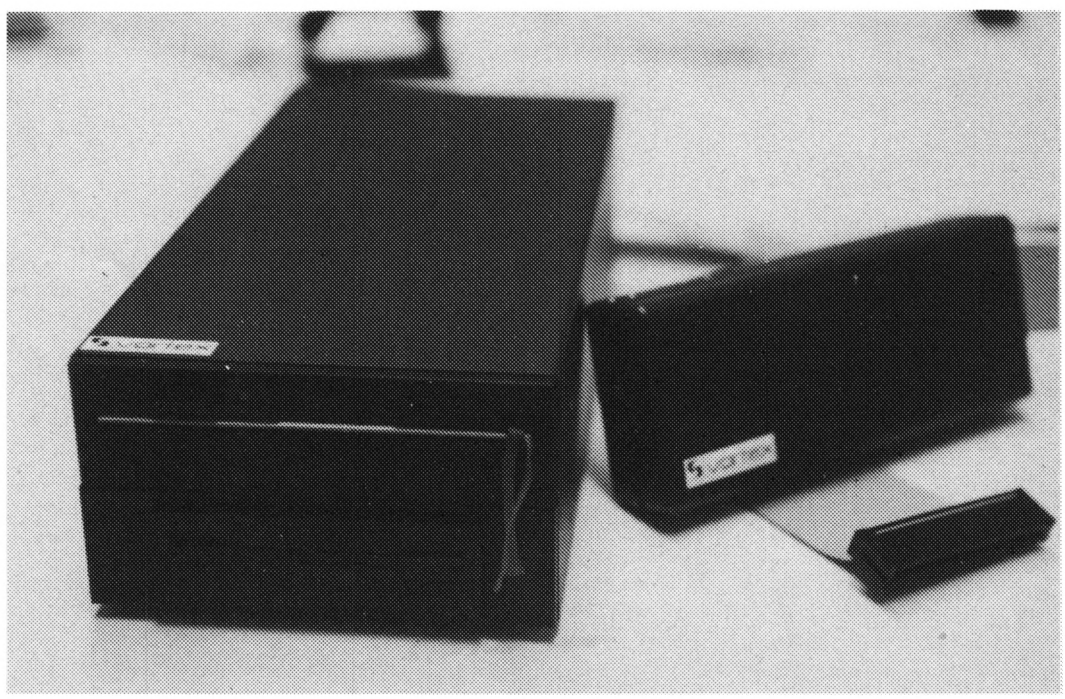
```
PRINT HEX$(171,4)
```

In diesem Fall erhalten Sie die Hex-Zahl mit zwei führenden Nullen. Zwei Ziffern würden in diesem Zahlensystem ausreichen, um den Dezimalwert 171 darzustellen. Diese beiden Ziffern lauten natürlich "AB" – womit klar ist, daß der DEC$-Befehl das richtige Ergebnis liefert. *AE* □

Vortex-Rom geknackt

Viele Vortex-Anwender interessieren sich auch für das „Innenleben" Ihres Diskettenlaufwerks. Problematisch ist dabei nur, daß Vortex seine ROMs seit einiger Zeit mit trickreichen Verfahren gegen Auslesen schützt. Unser Leser Ludger Papenkort hat eine Methode gefunden, wie man trotzdem an den begehrten Inhalt dieser ROMs kommt. Hier ist sein Bericht über das Leben und Leiden mit der Vortex-Floppy.

Software-Schutz ist ein legitimes Anliegen der Programmersteller, da oft viele arbeitsreiche Monate, wenn nicht sogar, wie bei professionellen Programmen (WordStar, dBase), viele (Mann-)Jahre zwischen der Idee und dem lauffähigen Programm liegen. Private Software-Tüftler begnügen sich häufig mit einem einfachen Listschutz ihres BASIC-Programmes. Andere wiederum verwenden Kopierschutzverfahren verschiedenster Raffinesse.
Wie auch der Software-Schutz geartet ist, es gibt immer ein Mittel (Programm), ihn zu umgehen oder unwirksam zu machen. Den Hackern unter den Lesern sage ich nichts Neues.
Es überraschte mich aber, festzustellen, daß auch Programme, die in ROMs abgelegt sind, geschützt sein können. Ich kam darauf, als ich mit dem Disassembler der Vortex-Floppy F1-X die ROMs meines CPC durchforsten wollte, um zu sehen, was sich da so alles abspielt.
Es ist immer interessant, zu erfahren, wie die Profis programmieren, und mitunter auch sehr lehrreich für eigene Routinen in Maschinensprache.
Da die Erweiterungs-ROMs alle dem gleichen Standard entsprechen müssen, um vom Betriebssystem beim Einschalten des Rechners aktiviert werden zu können, ist das Auffinden bestimmter Routinen recht einfach. Nach den ersten vier Byte eines jeden ROM kommt die Tabelle der RSX-Befehle. Sie besteht aus lauter JP nnnn, wobei nnnn das Sprungziel ist *(Abbildung 1)*.



Kein Geheimnis mehr: Die Vortex-Floppy-Station

Der erste Eintrag in der Sprungtabelle ist kein Jump, sondern die Adresse der Befehlsnamen-Tabelle. Hier finden Sie einen Befehl, zählen seine Position ab und sehen dann in der Sprungtabelle an der gleichen Position die Adresse der gesuchten Routine.
Das BASIC-ROM, das ROM der Schneider Drei-Zoll-Floppy und auch das ROM der Vortex-Speichererweiterung verhielten sich vorschriftsmäßig: Beim Disassemblieren traten keinerlei Probleme auf, sinnvollen Code zu erzeugen. Natürlich mit Ausnahme der Datenbereiche, wie etwa die Tabellen für die Disketten-Formate im Floppy-ROM. Bei Durchsicht des Vortex-Floppy-ROMs gab's dann aber die große Überraschung: Noch nicht einmal die Sprungtabelle konnte richtig disassembliert werden *(Abbildung 2)*.
Meine erste Vermutung, der Vortex-Disassembler würde die ROM-Nummer erkennen und dann den Code (ebenfalls von der Firma Vortex) verschlüsselt ausgeben, wurde durch Betrachtung mit dem DDTZ entkräftet. Da standen immer noch für eine Sprungtabelle sehr untypische Dinge wie "CALL PE,nnnn", verschiedene "SUB"s und "AND"s oder auch nur ein banales "HALT". Manchmal war sogar das resignierende "???" des DDTZ zu sehen, der mit dem Code auch nichts mehr anzufangen wußte.
Beispiele für einen entschlüsselten und einen nicht-entschlüsselten ROM-Header finden Sie in den *Abbildungen 1 und 2*. Sie werden selbst erkennen, daß sich mit der Originaltabelle nicht viel anfangen läßt.
Darauf setzte bei mir die Denkphase ein und ich schrieb mir die Byte so untereinander, wie sie eigentlich kommen müßten. Parallel dazu die entsprechenden Adressen und die Byte als Binärzahl. In *Abbildung 3* sehen Sie einen Ausschnitt aus der Sprungtabelle, die das System der Vortex-Verschlüsselung zeigt: Zunächst erfordert ein Sprungbefehl als erstes Byte immer ein "C3", was den Op-Code für den Jump darstellt. In der Tabelle fällt auf, daß das Byte, welches eigentlich immer C3 enthalten müßte, zwischen E3, CB und EB wechselt (Marke 1).

Adresse	Inhalt	Bemerkungen
C000	01	ROM-Typ
C001	01	ROM-Markierungs-Nummer
C002	21	ROM-Versions-Nummer
C003	00	ROM-Modifiktions-Ebene
C004	5F E9	Adresse Befehlsnamen-Tabelle
C006	C3 EC C1	Sprungtabellen-Eintrag 0 (Initialisierungs-Routine)
C009	C3 DC C2	Sprungtabellen-Eintrag 1
C00C	C3 A1 B6	usw.
C00F	C3 A4 B6	
usw.		

Abb. 1: Beispiel ROM-Header (Vortex-Floppy entschlüsselt)

Adresse	Inhalt	Bemerkungen
C006	E3	EX (SP),HL
C007	EC C1 C3	CALL PE,C3C1
C00A	DC C2 E3	CALL C,E3C2
C00D	A1	AND C
C00E	D6 E3	SUB E3
usw.		

Abb. 2: ROM-Header (Vortex-Floppy nicht entschlüsselt)

Adresse	Inhalt	1. Byte binär: 7654 3210	2. Byte binär: 7654 3210
C006	E3 EC C1	1110 0011	0000 0110
C009	C3 DC C2	1100 0011	0000 1001
C00C	E3 A1 D6	1110 0011	0000 1100
C00F	E3 A4 D6	1110 0011	0000 1111
C012	CB B2 D6	1100 1011	0001 0010
C015	EB 76 D6	1110 1011	0001 0101
C018	CB 79 D6	1100 1011	0001 1000
C01B	CB 8F D6	1100 1011	0001 1011
usw.			

Marke 1 — Marke 2 — Marke 3

Abb. 3: Beginn der Sprungtabelle bei C006 laut Konvention der Firma Amstrad

In der Binärdarstellung ist zu erkennen, daß hierfür nur das dritte beziehungsweise fünfte Bit (von null bis sieben gezählt) verantwortlich sein kann. Noch etwas fällt auf: Wenn das dritte Bit gesetzt ist, also anstatt einer 3 ein B im unteren Nibble des Op-Codes steht, dann ist das vierte Bit in der Adresse auch gesetzt (Marke 2). Das selbe gilt auch für das fünfte Bit des Op-Codes und das zweite Bit der Adresse (Marke 3). Nachdem ich diesen Zusammenhang herausgefunden hatte, baute ich ein Filterprogramm. Es invertierte mir die Bit drei und fünf eines jeden Byte des Codes im ROM entsprechend den Bit vier und zwei des unteren Byte der jeweiligen Adresse. Also:

Adresse unteres Byte, Bit 2 = 1: Bit 5 an der Adresse invertieren;

Adresse unteres Byte, Bit 2 = 0: Bit 5 bleibt wie es ist;

Adresse unteres Byte, Bit 4 = 1: Bit 3 an der Adresse invertieren;

Adresse unteres Byte, Bit 4 = 0: Bit 3 bleibt wie es ist.

Die Sache war also klar. Vortex hat einfach die Adreßleitungen mit den Datenleitungen am ROM entsprechend über Gatter und Inverter verknüpft. Daß das Problem gelöst war, zeigte sich beim DDTZ-Durchlauf des gefilterten Codes: Es kamen funktionierende Sprungtabellen heraus, und auch sonst wurde sinnvoller Code disassembliert.
Wenn Sie jetzt das Filterprogramm suchen sollten, um schnell mal den ROM-Code durchzunudeln, muß ich Sie leider enttäuschen. Bei genauerem Hinsehen stellte sich nämlich die nächste Hürde ein: Die Sprünge der Befehlstabelle gingen zum Teil auf Adressen, die mitten in einem Befehl lagen, zum Beispiel auf das dritte Byte eines Vier-Byte-Befehls. Das ist der Funktion eines Programms nicht gerade zuträglich. Des Rätsels Lösung: Die oben geschilderte Invertierung betrifft nur die Byte, die den Op-Code eines Befehls enthalten.
Alle anderen Byte, die nur Daten wie Sprungadressen oder irgendwelche Konstanten enthalten, machen diese Invertierung nicht mit. Das ist schon an der Tabelle der Befehlsnamen zu sehen, die im Klartext im ROM stehen und für jedermann mittels Hex-Dump zu lesen sind.
Die zusätzlichen Gatter der Firma Vortex befinden sich wahrscheinlich in den kleinen schwarzen Särgen, die man nach guter alter Unsitte der Typaufschrift durch Abschleifen beraubt, und somit in die Anonymität getrieben hat – wenigstens in meinem Floppy-Controller. Aber wie können Sie unterscheiden, ob der Prozessor gerade ein Befehlsbyte oder ein Datenbyte adressiert? Aber genau das meldet der Prozessor seiner Umwelt mittels seiner M1-Steuerleitung.

DER ABLAUF IM PROZESSOR

Diese Steuerleitung gibt Auskunft über den Stand der prozessor-internen Befehlsverarbeitung. Sie hat drei wichtige Phasen: die Holphase (Fetch-Zyklus), die Dekodierphase und die Ausführungsphase.
Die Holphase ist die interessanteste, denn in ihr wird der Op-Code eines Befehls geholt, was mit der M1-Steuerleitung angezeigt wird. Dies ist wichtig für externe Einheiten wie DMA-Bausteine, die einen Interrupt beim Prozessor angefordert haben und ihrerseits einen Befehl auf den Datenbus legen wollen, meistens ein RST (Restart) oder CALL. Der Prozessor meldet den externen Einheiten mit der M1-Steuerleitung und dem IORQ, daß er einen Befehl empfangen will.
In der Dekodierungspha-

Kein Software-Schutz ist gegen Mißbrauch gefeit. Das verständliche Anliegen der Hersteller reizt private Software-Tüftler, den Schutz mit Hilfe eines Programms zu umgehen oder unwirksam zu machen.

se wird der Befehl intern weiterverarbeitet, also dekodiert, und in der Ausführungsphase ausgeführt.
Die Ausführung kann nicht nur darin bestehen, zum Beispiel etwas zu addieren, sondern auch weiter Datenbyte zu holen. Aber für uns ist wichtig: Dieses Holen gehört nicht zur Hol-, sondern zur Ausführungsphase. Mit der M1-Steuerleitung kann also zwischen der Adressierung von Op-Code-Byte und der von Datenbyte unterschieden werden.
Daraus folgt, daß die Vortex-Entwickler nicht nur die Adreßleitungen, sondern auch die M1-Leitung mißbraucht haben, um ihren Code zu schützen. Aber das ist für Sie jetzt kein Problem mehr. Sie brauchen nur noch einen Disassembler zu programmieren, der immer weiß, was für ein Byte er vor sich hat und es entsprechend der oben aufgestellten Regel entschlüsselt oder auch nicht.

Ich begnüge mich mit zwei Listings: einem entschlüsselten für die Op-Codes und einem Original-Hexdump für die Adressen und Daten, die sich nebeneinander auf meinem Schreibtisch meterweise herumräkeln und mich hämisch angrinsen, wenn ich wieder mal vergessen habe, woher der Sprung eigentlich kam, den ich gerade verfolge. *Ludger Papenkort*□

Hinweis der Redaktion: Für Informationen wie in obigem Beitrag übernehmen wir keine Verantwortung. Die Vorschriften des Urheberrechtsgesetzes sind zu beachten.

Wie mit MS-DOS

Immer wieder wird behauptet, CP/M sei dem Betriebssystem MS-DOS unterlegen. Als Beispiel werden häufig die Batch-Dateien, I/O-Redirection und ähnliches angeführt. Was diese Schlagwörter bedeuten und wie man solche Techniken auch unter CP/M Plus einsetzen kann, soll Ihnen hier vorgeführt werden.

Batch-Files und „Batch Processing" sind zwei Ausdrücke aus der IBM-Welt, die im Deutschen mit „Stapeldatei" und „Stapelverarbeitung" übersetzt werden. Die Bezeichnungen sind nicht gerade sinnfällig und bedürfen einer weiteren Erklärung.
Eine Stapeldatei ist nichts anderes als eine Textdatei, die mehrere Kommandozeilen enthält. Sie können alle Befehle verwenden, die auch im Direktmodus des jeweiligen Betriebssystems, MS-DOS oder CP/M, möglich sind. Der Rechner arbeitet diese Datei Zeile für Zeile ab und verhält sich so, als würden Sie die entsprechenden Befehle direkt über die Tastatur eingeben.
Unter MS-DOS wird eine Batch-Datei genauso aufgerufen wie ein COM-File, also einfach mit dem Namen. CP/M macht es Ihnen nicht ganz so einfach. Es stellt für die Bearbeitung von Stapeldateien ein eigenes Dienstprogramm namens SUBMIT zur Verfügung.

Nehmen wir an, Sie wollen mit WordStar arbeiten und gleich anschließend alle Backup-Files löschen. Natürlich soll das Ganze automatisch ablaufen.
In diesem Fall benutzen Sie einen Texteditor, der seine Texte als ASCII-Dateien ablegt, und legen eine Batch- beziehungsweise Submit-Datei mit folgendem Inhalt an:

```
WS
ERA *.BAK
```

Texteditoren können WordStar im Programmmodus, der CP/M-Editor ED oder Turbo-Pascal sein – um nur die gängigsten zu nennen.
Sie haben diese kleine Datei unter dem Namen WS.SUB auf Ihrer WordStar-Diskette gespeichert, das SUBMIT-Dienstprogramm befindet sich ebenfalls auf dieser Diskette. Jetzt können Sie mit dem Befehl

```
SUBMIT WS
```

WordStar starten und darin arbeiten – dafür sorgt die erste Zeile. Nachdem Sie das Programm verlassen haben, wird die zweite Zeile der Batch-Datei ausgeführt. Sie bewirkt, daß alle Dateien mit dem Suffix BAK gelöscht werden.
Vergessen Sie nicht, den ERA-Befehl in der Datei mit RETURN abzuschließen, da in einem Submit-File jedes Zeichen so interpretiert wird, als käme es von der Tastatur. Bei diesem File ist übrigens nur das Suffix SVB erlaubt.
Wer MS-DOS kennt, wird vielleicht einwenden, eine AUTOEXEC-Datei sei in CP/M nicht anzulegen. Überzeugte CP/M-User werden diesem Argument sofort begegnen. Eine Autoexec-Datei (von „Automatic Execution") ist nichts anderes als eine Batch-Datei, die beim Start des Betriebssystems automatisch ausgeführt wird. Auch so etwas gibt es unter CP/M. Die Datei hat lediglich eine andere Bezeichnung, nämlich PROFILE.SUB. Findet der Rechner ein File dieses Namens, so wird es ausgeführt, sobald Sie CP/M starten.
Legen Sie sich eine Datei mit folgendem Inhalt unter dem Namen PROFILE.SUB an:

```
DIR [DRIVE=ALL,
   DATE,FULL]
DATE SET
DATE CONTINUOUS
```

Damit diese Datei lauffähig ist, benötigen Sie die Dienstprogramme DIR.COM und DATE.COM auf der selben Diskette.
Wenn Sie die so vorbereitete CP/M-Diskette das erste Mal laden wollen, werden Sie vielleicht gleich beim ersten Befehl eine Fehlermeldung erhalten. Sie werden feststellen, daß das Ihren CPC nicht im geringsten stört und er mit der Abarbeitung der Stapeldatei fortfährt. Wir wollen dieser Fehlermeldung trotzdem auf den Grund gehen.
In der Regel können Sie unter CP/M zu Ihren Dateien keine Zeit- und Datumsangaben speichern. Unter CP/M Plus ist dies jedoch möglich – wenn Sie wollen. Und es ist hier sogar komfortabler als MS-DOS.
Während MS-DOS nur dann ein neues Datum speichert, wenn Sie eine Datei ändern, können Sie unter CP/M auch erfahren, wann eine Datei zuletzt gelesen wurde. Damit wissen Sie, ob und wann ein Unbefugter an Ihren Daten war. Daß Sie diese Daten auch schützen können, ist jetzt fast schon selbstverständlich.
Doch zunächst zum DATE-Befehl: Da er transient ist, sollte die entsprechende Datei auf Ihrer Arbeitsdiskette vorhanden sein. Sie können dann mit

```
DATE SET
```

Datum und Uhrzeit ein-

stellen. Leider besitzt der CPC keine eingebaute Echtzeituhr, so daß Sie diese Prozedur vor der Arbeit mit CP/M Plus jedesmal durchlaufen müssen.
Wem die Bildschirmabfrage zu lange dauert, der kann auch direkt mit der Anweisung DATE die Zeit einstellen:

DATE mm/tt/jj hh:mm:ss

lautet die Reihenfolge der benötigten Parameter. Beachten Sie in jedem Fall die amerikanische Schreibweise des Datums. Der letzte Befehl unserer Datei PROFILE.SUB steht ebenfalls im Zusammenhang mit der Uhrzeit. Die Anweisung

DATE CONTINUOUS

oder kurz

DATE C

läßt auf dem Bildschirm eine „Digitaluhr" erscheinen. Sie läuft allerdings nur bis zum nächsten Druck auf RETURN. Damit kann der CPC intern die aktuelle Zeit speichern. Jetzt bleibt nur noch das Problem, sie auf die Diskette zu bringen, und zwar ins Inhaltsverzeichnis und zur richtigen Datei.
Die Zeitspeicherung müssen Sie mit einem Verlust an Speicherplatz bezahlen. Die Anzahl der verfügbaren Directory-Einträge nimmt um ein Viertel ab. Doch Sie können den verlorenen Speicherplatz jederzeit wiedergewinnen. Das Dienstprogramm

INITDIR

erledigt beides für Sie. Wenn Sie es auf eine Diskette anwenden, die für Zeitspeicherung noch nicht eingerichtet ist, so wird deren Inhaltsverzeichnis umformatiert. Wenden Sie es auf eine bereits vorbereitete Disk an, so können Sie den zuvor vergebenen Platz im Directory zurückgewinnen oder die vorhandenen Zeiteinträge löschen – etwa wenn diese falsch sind, weil die Uhr nicht richtig eingestellt war.
Mit INITDIR ist die Diskette noch nicht ausreichend vorbereitet. Ein weiteres Dienstprogramm muß her, nämlich SET. Damit bestimmen Sie, zu welchem Anlaß eine Zeitmarke (englisch: Time Stamp) eingetragen wird. Die Syntax ist

SET Laufwerk: [Option]

Die Laufwerksangabe kann entfallen, wenn Sie sich auf den aktuellen Drive beziehen wollen. An Optionen bestehen für das Setzen von Zeitmarken folgende Möglichkeiten:

CREATE=ON
ACCESS=ON
UPDATE=ON

Dabei sind mehrere Dinge zu beachten: Zunächst müssen Sie wissen, daß CREATE für „Erstellen", ACCESS für „Zugriff" und UPDATE für „Ändern" steht. Außerdem können Sie keine Zeitmarken für Erstellung und Zugriff gleichzeitig setzen. Die Optionen CREATE und ACCESS schließen sich also gegenseitig aus. Nur die zuletzt eingestellte ist wirksam. Die günstigste Lösung ist, Zugriffs- und Änderungszeiten zu speichern. Dies bedeutet nichts anderes als einen Vermerk über den jeweils letzten Lese- oder Schreibzugriff auf eine Datei. Da auch bei einer Neuerstellung ein Schreibvorgang vorliegt, ist die Option CREATE damit weitgehend abgedeckt. Sie können zwar nicht feststellen, ob die Datei mit dem letzten Schreibzugriff neu erstellt oder nur geändert wurde. In der Regel kommt eine Änderung aber einer Neuerstellung fast gleich, ein veralteter Inhalt einer Textdatei interessiert Sie wohl kaum.
Damit Sie sich die Zeiteintragung ansehen können, gibt es in CP/M Plus einen zweiten, diesmal transienten, DIR-Befehl. Er bietet Ihnen die Möglichkeit, Optionen anzugeben. Hier die wichtigsten:

DIR [DATE]

zeigt Ihnen das erweiterte Directory-Listing, sofern Zeitmarken zugelassen sind. Das also ist der Trick bei der Sache.

DIR [DRIVE=x]

bringt das Inhaltsverzeichnis des für x angegebenen Laufwerks auf den Bildschirm. Statt eines Laufwerksbezeichners (hier ohne Doppelpunkt) ist auch die Angabe ALL möglich. CP/M zeigt Ihnen dann ein Inhaltsverzeichnis der Disketten in allen angeschlossenen Laufwerken.
Leider hat diese Sache einen kleinen Haken: DIR erkennt nur Laufwerke, auf die seit dem Start von CP/M bereits einmal zugegriffen wurde.

DIR [FULL]

benötigen Sie, um ein erweitertes Directory für sämtliche Files zu erhalten. Ein Beispiel für ein komplettes Directory finden Sie in der nebenstehenden *Abbildung*.
Die weiteren Optionen im Dienstprogramm DIR können frei miteinander kombiniert werden – ein Beispiel dafür ist die erste Zeile unserer Datei PROFILE.SUB.
Der oben bereits erwähnte SET-Befehl erlaubt noch weitere Anwendungen wie etwa das Benennen einer Diskette:

SET A: [NAME=CPC1]

gibt der Diskette in Laufwerk A den Namen „CPC1". Damit ist es möglich, Disketten nicht nur per Aufkleber zu archivieren.
Eine viel wichtigere Anwendung aber ist das Schützen einer Diskette oder einzelner Files durch ein Password. Dafür müssen Sie CP/M nur mitteilen, daß Sie Dateien schützen wollen:

SET A: [PROTECT=ON]

Nehmen wir an, Sie wollen die Datei ADRESS.LST mit dem Schutzwort HACKER versehen. Die Anweisung würde dann lauten:

SET A:ADRESS.LST [PASSWORD=HACKER]

Der Schutz wirkt beim Lesen, Schreiben und Löschen einer Datei. Es ist jedoch möglich, eine

```
dir Ædrive=a,date,fullA

Scanning Directory...

Sorting  Directory...

Directory For Drive A:  User  0

     Name      Bytes   Recs   Attrib  Prot   Update      Access
------------ ------ ------ -------- ------ ------ --------------

CI0CPM3  EMS     25k    200 Dir RW  None
DATE     COM      3k     23 Dir RW  None
DEVICE   COM      8k     58 Dir RW  None
DIR      COM     15k    114 Dir RW  None          01/18/88 10:40
GET      COM      7k     51 Dir RW  None
INITDIR  COM     32k    250 Dir RW  None
PROFILE  SUB      1k      1 Dir RW  None
PUT      COM      7k     55 Dir RW  None
SET      COM     11k     81 Dir RW  None          01/18/88 10:37
SUBMIT   COM      6k     42 Dir RW  None

Total Bytes     =115k Total Records =     875  Files Found =  10
Total 1k Blocks =115 Used/Max Dir Entries For Drive A:   17/ 64
```

Datei nur gegen einzelne Operationen zu schützen. So können Sie etwa das Eintragen neuer Adressen in Ihre Liste erlauben, jedoch nicht das Lesen aus derselben.
Auch das bewirkt der SET-Befehl:

```
SET A:ADRESS.LST
  [PROTECT=xxx]
```

Dabei steht für „xxx" READ, WRITE, DELETE oder NONE. Letztere Option hebt den Schutz wieder auf, die ersten drei dürften sich von selbst erklären. Eine solche Einstellung ist auch nachträglich möglich.
Um einzelne Dateien sinnvoll schützen zu können, muß auch das Directory-Label selbst mit einem Schutzwort versehen sein. Ist das nicht der Fall, so kann jedermann durch die Anweisung

```
SET A: [PROTECT
  =OFF]
```

die Sicherung desaktivieren – ohne ein Schutzwort zu kennen. Daher ist es unerläßlich, auch das Directory entsprechend abzusichern. Lassen Sie dafür einfach den Dateibezeichner bei der Password-Vergabe weg.

SCHUTZ FÜR DATEIEN

Mit SET werden auch Dateiattribute vergeben, von denen hier nur die zwei wichtigsten angeführt sind. Die Anweisung

```
SET A:WICHTIG.TXT
  [RO]
```

setzt bei dem File WICHTIG.TXT ein Read-Only-Attribut. Das bedeutet, daß jeder Versuch, die Datei neu zu beschreiben, zu einer Fehlermeldung führt. Mit der Option RW (Read–Write) heben Sie diesen Schutz wieder auf.
Auch hier ist es möglich, den Dateinamen wegzulassen. Dies bewirkt einen Schreibschutz für eine ganze Diskette. Wenn Sie selbst Anwendungsprogramme unter CP/M schreiben, kann das von Nutzen sein. Versuchen Sie es mit folgender PROFILE-Datei:

```
SET A: [RO]
SET B: [RW]
ANWENDUNG
```

Natürlich steht statt „Anwendung" der Name Ihres COM-Files. Es wird automatisch mit dem Aufruf von CP/M gestartet. Der Benutzer ist anschließend gezwungen, seine Arbeitsdateien auf dem zweiten Laufwerk anzulegen, da das erste nicht beschrieben werden kann. Somit ist die wertvolle Programmdiskette vor versehentlichem Löschen geschützt.
Der Schreibschutz für eine Datei bleibt auf der Diskette gespeichert, nicht aber der einer Disk. Sobald Sie CP/M verlassen und neu starten, haben Sie wieder eine „normale" Diskette in Laufwerk A. Jetzt wird auch klar, warum das Setzen des Schreibschutzes in der Submit-Datei erfolgen muß: Sie kommen nicht umhin, ihn vor jedem Programmstart erneut zu aktivieren.
Unter diesen Aspekten kann CP/M also durchaus mit MS-DOS konkurrieren: Stapelverarbeitung, Schreib-/Leseschutz und sogar Passwords sind machbar.
Was aber ist mit der vielgenannten Ein-/Ausgabe-Umleitung, auf Neudeutsch: I/O-Redirection? Wir würden sie nicht erwähnen, wenn der CPC sie nicht beherrschen würde. Die transienten Befehle GET und PUT sind dafür zuständig.
Haben Sie einen langsamen Drucker? Wenn es einmal eilt, können Sie die Druckerausgaben auch in eine Datei auf Diskette umleiten. Der Befehl lautet

```
PUT PRINTER
OUTPUT TO
Dateiname
```

Da in einem solchen Fall der gedruckte Text gleichzeitig auf dem Bildschirm erscheint, schaltet man dieses „Echo" am besten mit der Option

```
[NOECHO]
```

ab. Mit ECHO wird es wieder aktiviert.
Eine weitere wichtige Option lautet FILTER. Wenn Sie diesen Kenner angeben, werden alle ESC- und CTRL-Zeichen in druckbare Zeichen umgesetzt. Das ist dann nützlich, wenn Sie die Druckerausgabe auf den Bildschirm umleiten, etwa bei einem Programmtest, bei dem Sie Papier sparen wollen.
Statt PRINTER kann auch CONSOLE als Ausgabe verwendet werden. In diesem Fall gilt die Umleitung für alle Ausgaben, die normalerweise auf dem Bildschirm erfolgen. So vertauschen Sie Bildschirm und Drucker:

DRUCKER UND BILDSCHIRM

```
PUT PRINTER
OUTPUT TO
CONSOLE

PUT CONSOLE
OUTPUT TO
PRINTER
```

Denken Sie bei der Anwendung dieser Befehle daran, daß die Einstellung nur für das nachfolgend aufgerufene Programm gilt. Die Option SYSTEM veranlaßt den Rechner, die Ausgaben solange umzuleiten, bis ein weiterer PUT-Befehl wieder Ordnung schafft.
Wenn Sie schon die Ausgabe nicht auf die Standardgeräte lenken müssen, so müssen Sie natürlich auch die Eingabe nicht unbedingt von der Tastatur holen. Dafür existiert das Dienstprogramm GET.

```
GET CONSOLE
INPUT FROM FILE
Dateiname
```

holt die Eingabe von einer beliebigen Textdatei. Dies wollen wir mit einem praktischen Tip für Anwender von Turbo Pascal demonstrieren. Dieser Compiler stellt zuerst die Frage, ob der Klartext für die Fehlermeldungen geladen werden soll. Sie wird in der Regel mit Y und RETURN beantwortet. Danach erscheint ein Menü, in dem Sie als erstes ein W (ohne RETURN) eingeben werden, um den Dateinamen für die Arbeitsdatei zu bestimmen.
Es gibt aufwendige Patches für Turbo Pascal, damit der Compiler all dies automatisch erledigt. Das ist unter CP/M Plus jedoch überhaupt nicht nötig. Legen Sie sich einfach eine Datei T.SUB an, die folgende Zeilen enthält, die jeweils mit RETURN abgeschlossen werden.

```
GET CONSOLE
INPUT FROM FILE
T.TXT
TURBO
```

Anschließend schreiben Sie in die Datei T.TXT alle Eingaben, die Sie normalerweise machen würden, also:

```
Y
W
```

Dem W darf kein RETURN folgen!
Um jetzt Turbo Pascal zu starten, müssen Sie nur SUBMIT T eingeben – oder statt T.SUB eine Datei PROFILE.SUB anlegen, die Turbo sofort beim Start von CP/M lädt.

DATEIENDE WIRD ÜBERGANGEN

Dabei erkennen Sie einen weiteren wichtigen Effekt von GET: Turbo Pascal erwartet noch weitere Eingaben von Ihnen, die jedoch von Fall zu Fall verschieden sein werden. CP/M dagegen hat vorgesorgt und nimmt automatisch wieder Ein-

Bitte lesen Sie weiter a.S. 89

Jetzt schlägt's Dreizehn!

CPC-WELT EXTRA kommt direkt zu Ihnen ins Haus!
7 x pro Jahr!
Für nur 60 DM!
Sie sparen über 40 DM!





Finden Sie Ihre CPC-WELT nicht immer am Kiosk? Vielleicht, weil schon ausverkauft? Möchten Sie die CPC-WELT schon vor der Kioskbelieferung in Händen haben? Dann gibt es jetzt die Möglichkeit! Wir beliefern Sie im Abonnement mit sechs plus einer Ausgabe für ganze 60 DM (Inland) oder 80 DM (Ausland). Sie erhalten dann das jeweils druckfrische Heft, in der Regel sogar früher, als es am Kiosk hängt (so die Bundespost will). Sechsmal. Und außerdem gehört zum Abo noch unser jährlicher CPC-SPECIAL Sammelband im Wert von DM 14,80. Einzige Bedingung: Das Abo muß zum Zeitpunkt des Erscheinens dieses Bandes noch bestehen und bezahlt sein. Ist das ein Angebot?

WICHTIGE RECHTLICHE GARANTIE!

Sie können diesen Abo-Auftrag binnen einer Woche nach Zugang der Abo-Bestätigung durch den Verlag an Sie widerrufen. Postkarte genügt! Zur Wahrung der Frist genügt die rechtzeitige Absendung. Bitte, bestätigen Sie durch Ihre zweite Unterschrift, daß Sie von diesem Widerspruchsrecht Kenntnis genommen haben.

Sie können dieses Abonnement jeweils mit einer Frist von einem Monat zum Ende des Bezugszeitraumes (sechs Hefte) kündigen. Unterlassen Sie diese Kündigung, wird die Belieferung mit weiteren sechs Heften zu den gleichen Bedingungen fortgesetzt! Die Lieferung beginnt nach Eingang der Abo-Gebühr.

ABO-SERVICE-KARTE

Ich nehme zur Kenntnis, daß die Belieferung erst beginnt, wenn die Abo-Gebühr dem Verlag zugegangen ist!

Ja, ich möchte von Ihrem Angebot Gebrauch machen.
Bitte senden Sie mir bis auf Widerruf ab sofort jeweils die nächsten sechs Ausgaben an untenstehende Anschrift. Wenn ich nicht vier Wochen vor Ablauf kündige, läuft diese Abmachung automatisch weiter.

Name _____
Vorname _____
Straße/Hausnr. _____
PLZ/Ort _____
Ich bezahle:
☐ per beiliegendem Verrechnungsscheck
☐ gegen Rechnung
☐ bargeldlos per Bankeinzug von meinem Konto
bei (Bank und Ort) _____
Kontonummer _____
Bankleitzahl _____
(steht auf jedem Kontoauszug)

Unterschrift _____
Von meinem Widerspruchsrecht habe ich Kenntnis genommen.

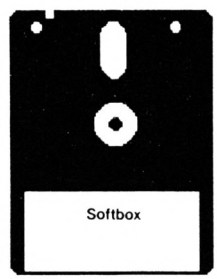
Unterschrift _____ **SPI/7**

SCHNEIDER CPC-WELT
ABO-SERVICE SPI/7
POSTFACH 1161
D-8044
UNTERSCHLEISSHEIM

PROGRAMMSERVICE

Hiermit bestelle ich in Kenntnis Ihrer Verkaufsbedingungen die Listings dieses Heftes auf

☐ Kassetten (50 DM) ☐ Diskette (50 DM)



Ich zahle (Zutreffendes bitte ankreuzen!):
per beigefügtem Scheck ☐ Sein ☐
gegen Bankabbuchung am Versandtag ☐

Meine Bank (mit Ortsname) _____
Meine Kontonummer _____
Meine Bankleitzahl _____

(steht auf jedem Bankauszug)
Nachname _____
PLZ/Ort _____ **SPI/7**
Name _____
Str./Nr. _____

Verkaufsbedingungen: Lieferung nur gegen Vorkasse oder Bankabbuchung. Keine Nachnahme. Umtausch bei Nichtfunktionieren.

Unterschrift _____
Bitte ausschneiden und einsenden an

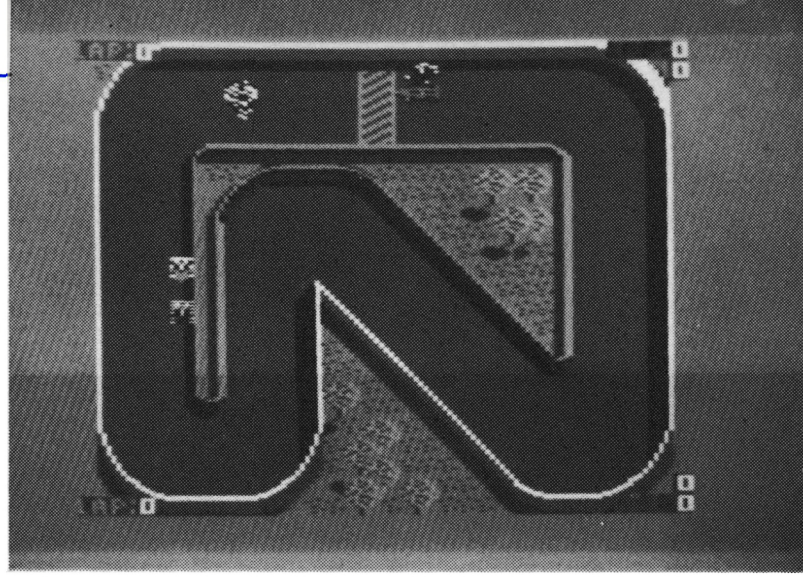
SCHNEIDER CPC-WELT
SOFTWARE-SERVICE SPI/7
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

HIGHSCORE



Das Spiele-Magazin

Hot Wheels



Schon wieder haben wir ein Autorennspiel in den Händen, das x-te! Diesmal hat es aber weder mit schneller 30er-Grafik aufzuwarten noch mit einem hervorragenden Autosprite. Heute betrachten wir das ganze Spektakel einmal aus der Vogelperspektive. Von oben sieht alles ganz anders aus – auch die Autos.

Das neue Rennspiel heißt Super Sprint und wird von Activision Software vertrieben. In ihm gibt es acht verschiedene, frei anwählbare Rennstrecken, die alle ihre eigenen Tücken und Gefahren bergen: Sich öffnende und schließende Tore, Unterführungen und Sprungrampen.

Gespielt werden kann einzeln oder zu zweit. Natürlich macht es zu zweit viel mehr Spaß, da sich dann oft eine richtige Straßenschlacht entwickelt.

Nach dem Laden erhält der Spieler die Möglichkeit, grafisch zwischen dem Ein- und Zweispielermodus zu wählen. Ist dies geschehen, so winkt ihm der Fahrer eines Rennwagens freundlich mit erhobenem Daumen zu. Daraufhin kann man sich eine von acht Rennstrecken aussuchen.
Anschließend geht es ab auf die Piste. Hier macht man die ersten Erfahrungen, wie schwer es ist, einen Rennwagen über eine kurvenreiche Strecke zu steuern.

Die Grafik der Rennpiste ist Mittelmaß. Sie ist nicht sehr farbenfroh und beschränkt sich größtenteils auf Grautöne. Durch eine gelungene Schattur wird aber ein gewisser Tiefeneffekt erzeugt, der auch Mauern und Abhänge gut erkennen läßt. Die Fahrzeuge selbst sind zwar nicht besonders gelungen, aber noch als Rennwagen zu erkennen. Bloß in der Kurve hüpfen sie regelrecht; dort ist die Animation recht schwach.

Der Sound besteht aus Tönen, die wohl Fahrgeräusche simulieren sollen, die allerdings nach längerem Spielen an den Nerven zerren. Besonders mit einem zweiten Spieler lassen sich gut Turniere am Bildschirm austragen.
Die Fahrzeuge können mit dem Joystick oder der Tastatur gesteuert werden.

Fazit

Super Sprint gehört sicher nicht zur Elite der Rennspiele. Durch seine vielen kleinen Extras, die kein anderes Rennspiel zu bieten hat, macht es aber Spaß, und das ist eigentlich das Wichtigste. Leute, die ein lustiges und fetziges Spiel suchen – und vielleicht noch einen Spielpartner haben – solltes es sich einmal ansehen. Es kostet 49,95 Mark. *mn*

Titel:	**Super Sprint**
Getestet:	**Schneider CPC**
Umsetzungen:	**C64**

Im Test: ☒ Diskette ☐ Kassette

Preis (DM): 49.95

☒ Joystick ☒ Tastatur ☐ Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Die Thundercats schlagen zu!

Die Programmierer von Elite waren diesmal schneller als das deutsche Fernsehen. Auf den Monitoren des CPC flimmern ab nun die Abenteuer des mächtigen Katzenvolkes, der Thundercats. Regie führt der Spieler.

In den USA läuft schon seit einiger Zeit eine Zeichentrickserie, in der es um ein Katzenvolk auf fernen Planeten geht. Die Mischung von Fantasy und Science Fiction stimmt, und die Serie wurde zum Renner des Kinderprogrammes vieler Fernsehstationen. Es ist abzusehen, daß die Filme, Comics und Puppen bald auch Europa erobern werden. Eine wahre Schwemme (denken Sie an He-Man) ist zu befürchten. Das erste Produkt aus der Vermarktungsstrategie liegt bereits vor: Thundercats als Computerspiel für den CPC.

Sollten Sie Katzenliebhaber sein, dann vergessen Sie alle positiven Eigenschaften Ihres Haustieres. Das Thundercat-Team hat eher menschliche Züge, an die beliebten Schmusetiere erinnert nur der Löwenkopf. Es darf also nach bester Barbarenart gekämpft werden.

Der allzu böse Mum-Ra überfällt das friedliche Katzenvolk und bringt sich in den Besitz eines magischen Steines. Lion-O (man beachte den Vornamen!) aus der Thundercat-Mannschaft war zufällig nicht anwesend,

was ihm zwar das Leben rettete, ihn als Hüter des Steines aber verpflichtet, das gute Stück wieder herbei zu holen. Es dient nämlich nicht nur der Verschönerung tapferer Stammesmitglieder, sondern verleiht dem Schwert Thundera (was man vielleicht mit „Donnerlüttchen" übersetzen könnte) magische Kräfte.

Die Geschichte haben Sie so oder ähnlich sicherlich schon oft gehört, zuletzt bei Barbarian.

Thundercats ist ein Spiel für Leute, die gleich zur Sache kommen wollen, und das Menü ist entsprechend kurz. Nach einem sehr guten Titelbild ist die Entscheidung, ob sie mit dem Joystick oder der Tastatur spielen wollen, fällig. Letzteres ist nicht zu empfehlen, da die unglückliche Belegung nicht zu ändern ist und der „Fingerknoten" droht.

Sodann darf losgelaufen und zugeschlagen werden. Mit einem Schwert gilt es, sich der finsteren Gesellschaft zu entledigen, von der der brave Held dauernd angegriffen wird. Lassen Sie sich dabei nicht von der Statur der Gegner täuschen. Gerade die Zwergkatzen unterlaufen den Schwerthieb, es sei denn, Sie bücken sich, bevor Sie zuschlagen.

Es ist eigentlich sehr einfach, Hindernisse zu überspringen, vorausgesetzt, der aufmerksame Kämpfer bemerkt sie rechtzeitig. Oft ist ein Neustart erforderlich, weil die fünf Leben in irgendeinem Sumpf verloren gegangen oder in eine andere Falle geraten sind.

Damit es nicht gar zu langweilig wird – immer nur töten oder getötet werden kann ganz schön ermüdend sein – dürfen Sie sich auch noch um ein Punktekonto kümmern. Schlagen Sie mit dem Schwert an giftige Pilze oder Bäume, dann wird Ihnen ein Bonus zuerkannt. Gleiches gilt, wenn Sie ein Level in der vorgegebenen Zeit abarbeiten oder sich eines Baumhauses (so sieht es jedenfalls aus) bemächtiten.

Auch vom Spiel her stellt Thundercats keine großen Anforderungen an den Denker. Ein Action-Game eben, bei dem es nur auf gute Reaktionsfähigkeit ankommt. Die Grafik versöhnt jedoch etwas mit dieser Einfallslosigkeit. Besser als vermutet stellen sich die einzelnen Szenen dar. Im Hintergrund ist immer die fantastische Landschaft eines fernen Planeten zu sehen. Das Kämpfer-Sprite ist relativ groß und detailreich, bewegt sich aber trotzdem ruckfrei. Ebenso unauffällig und seidenweich scrollt der Bildschirm während der Laufbewegung. Die Darstellung der Gegner, der Hindernisse oder der Bonusobjekte kann nicht kritisiert werden. Die Grafiker von Elite scheinen besser zu sein als die Ideenlieferanten für die Story.

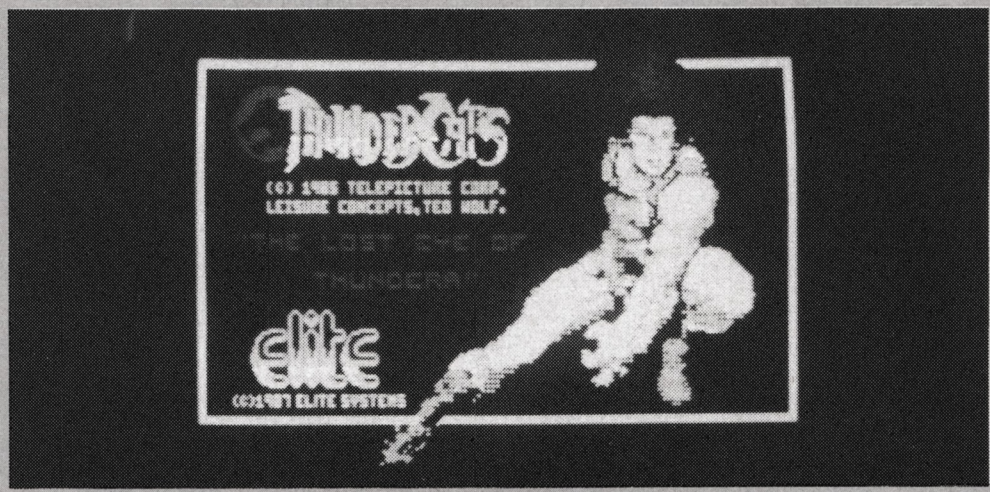
Der Sound fällt zumindest nicht auf. Während des Einstellungsmenüs gibt es eine kleine Melodie, recht nett anzuhören. Sonst sind ohnehin nur noch die entsprechenden Geräusche zu hören.

Negativ – und dies gleich in mehrfacher Hinsicht – ist die Spielanleitung. Wichtige Angaben, etwa was gesammelt werden soll, fehlen. Deshalb die Vermutung, daß es sich bei oben erwähntem Bonus um ein Baumhaus handeln könnte. Auch der Stil der Anleitung läßt zu wünschen übrig. Da ist die Rede vom Kill-Bonus, und wer nicht weiter weiß, dem wird empfohlen, über den Feind zu springen und ihn „von hinten fertig zu machen". Keine sanfte Tour für ein Programm, das eigentlich gar nicht so brutal ist.

Fazit

Gute Grafik alleine kann noch nicht überzeugen. Selbst wenn man sich an die relativ simplen Originalgeschichten der Fernsehserie gehalten hätte, wären die Spielideen besser ausgefallen. Elite wagte einen Frühstart, noch läuft in der Bundesrepublik das Thundercats-Geschäft nicht so richtig. Vielleicht wäre es besser gewesen, etwas mehr Zeit zu investieren und mit einer besseren Geschichte auf den Markt zu kommen. Liebhabern des Genres kann das Spiel aufgrund des technischen Ablaufes (sprich: der einwandfreien Programmierung) allerdings empfohlen werden. *GS*

Vergessen Sie Ihren Schmusetiger - Thundercats lieben den Kampf



Titel:	Thundercats
Getestet:	Schneider CPC
Umsetzungen:	C64

Im Test: ☒ ☒ ☐

Preis (DM): 44.95

☒ Joystick
☒ Tastatur
☐ Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Wasserski mit dem CPC

Feuchte Disziplinen auf dem Computer: Weitsprung, Slalom und Figuren. Fahren Sie Wasserski wie Patrice Martin!

Nach Fußball, Tennis, Schwimmen und olympischen Disziplinen kann man jetzt endlich auch auf seinem Computer eine Sportart ausüben, welche sich seit der Erfindung des Motorbootes einer immer größer werdenden Beliebtheit erfreut: Wasserskifahren. Infogrames hat diesen Sport für den Schneider CPC und den Atari ST umgesetzt. Als Vorbild hat man sich übrigens einen sehr bekannten Wasserskifahrer ausgesucht: Patrice Martin, der seit Jahren alle begehrten Preise einheimst und Rekorde um Rekorde macht. Wer ihn noch nicht kennt, lernt ihn spätestens auf dem Cover der Verpackung kennen. Außerdem kann man ihn noch einmal während des Ladens auf dem Monitor erkennen.

Nach einem erfolgreichen Ladevorgang zeigen sich auf dem Monitor mehrere kleine Bildchen, welche man als Icons bezeichnen könnte. Sie stehen für folgende Menüpunkte: Anzahl der Spieler, einzelne Disziplin oder komplettes Spiel. Teilnehmen können bis zu drei Spieler. Mit dem Joystick wird ein bestimmtes Icon ausgewählt und mit dem Feuerknopf gestartet. Es gibt übrigens leider nur drei Disziplinen.

Die erste Disziplin nennt sich Weitsprung. Dieser Wettkampf erfolgt in zwei Phasen, nämlich dem Anlauf zur Sprungschanze mit anschließendem Sprung und die hoffentlich gelungene Landung auf dem Wasser. Während der Flugphase muß die Haltung der Spielfigur mit dem Joystick ausbalanciert werden, was sich wesentlich einfacher anhört als es in Wirklichkeit ist! Der Spieler hat hier drei Versuche, gewertet wird nur der beste.

Die zweite Disziplin wird in der Anleitung schlicht und einfach als „Figuren" bezeichnet. Es handelt sich um einen zweiteiligen Wettkampf mit zwei Läufen von je zwanzig Sekunden. Der Spieler muß möglichst viele Punkte sammeln, indem er ein Maximum an Figuren aneinanderreiht. Diese können im Kielwasser oder auf den Seitenwellen des Motorbootes vollzogen werden.

Um zu möglichst kunstvollen Figuren zu kommen, bedient man sich wiederum des Joysticks. Eine Spezialität von Patrice Martin, der Salto, läßt sich nachmachen.

Einige Figuren können mit einem Step durchgeführt werden: Der Wasserskifahrer steigt während der Drehung mit einem Bein über das Zugseil.

Die dritte und letzte Disziplin findet auf einem großen Gewässer statt und nennt sich Slalom. Hier gilt es, einen Parcours zwischen jeweils sechs Bojen links und rechts zu fahren. Auch diese Aufgabe ist nicht einfach.

Das Hauptmenü und alle Disziplinen werden von einer flotten Musik begleitet. Beim Atari ST existieren sogar verschiedene Musikstükke. Auch gute Toneffekte wie Meeresrauschen fehlen nicht. Über die Grafik ließen sich wahre Lobeshymnen singen. Selten hat man solch eine Qualität, unterstützt durch viele kleine, liebevoll gestaltete Details, auf dem Schneider CPC gesehen. Auch die Grafikfähigkeiten des ST werden voll und ganz unterstützt. Wer sich vor dem Kauf davon überzeugen will, sollte einen kurzen Blick auf die Rückseite der Verpakkung werfen, auf der einige Bildschirmfotos abgedruckt sind.

FAZIT:
LEIDER NUR MITTELMÄSSIGES SPIEL

Nach all dem Lob ein paar Kritikpunkte, die sich stark auf die Motivation von Championship Water Skiing auswirken: Es geht um die Disziplinen. Sie sind zwar alle gut gelungen, aber erstens sind sie zu leicht und zweitens sind es zu wenig. Für einen etwas geübteren Computerfreak dürfte es kein Problem sein, die drei Sportarten mit vielen Punkten zu überstehen. Die durchschnittliche Spieldauer beträgt laut Cover nur eine knappe Stunde, was wirklich nicht untertrieben ist. Unter dem Aspekt, daß ein Atari ST-Besitzer ganze sechzig Mark hinblättern muß, lohnt sich der Kauf nicht.



Dieses Problem hätte Infogrames eventuell durch den Einbau von weiteren Disziplinen beheben können. Auf die Spieldauer und damit auf die Motivation hätte sich das positiv ausgewirkt. Aber dennoch könnte das Spiel schon allein wegen der guten Grafik und des ausgezeichneten Sounds ein Hit werden.

TB

Titel:	**Championship Water Skiing**
Getestet:	**Schneider CPC**
Umsetzungen:	**Atatri ST**

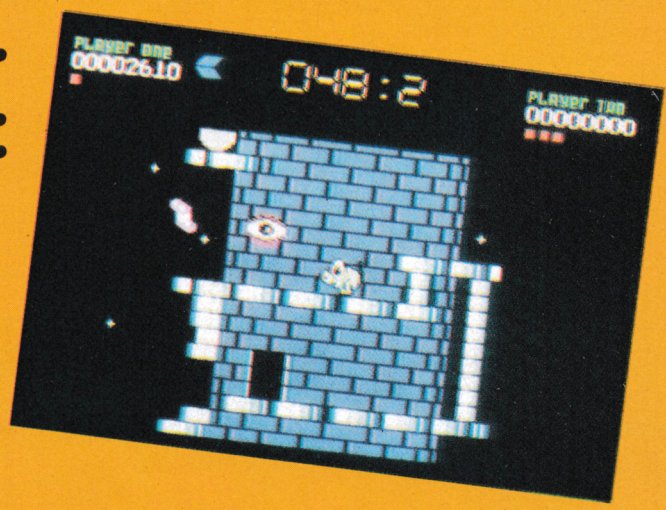
Im Test: ☒ 💾 ☐ 📼

Preis (DM): 39,95

☒ **Joystick** ☒ **Tastatur** ☐ **Maus**

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Abenteuer in einer Wasserwelt



Auf dem Planeten Nebulus geht es nicht ganz mit rechten Dingen zu. Irgend jemand hat es sich zu seinem Hobby gemacht, riesige Türme im Meer zu bauen – und das ohne Genehmigung der zuständigen Behörden. Sie werden in Ihrem Büro bei Destructo Inc., auf Deutsch etwa: Zerstörer & Co, aus dem wohlverdienten Mittagsschlaf gerissen. Das Telefon klingelt, am anderen Ende der Leitung ist Ihr Boß. Er erwähnt einen Zerstörungsauftrag, den Sie übernehmen sollen.

Sie sind gerade dabei, den Hörer in den Papierkorb zu werfen, als Sie das Wort „Gehaltserhöhung" hören. Erst jetzt wird's für Sie interessant. Als er dann noch von dem neuen Mini-U-Boot Mk. 7 spricht, sind Sie vollends hingerissen. So hingerissen, daß Sie die Aufzählung der Gefahren total überhören.

Ziel dieses Spiels ist es, Ihre Spielfigur mit dem U-Boot zu den Türmen zu steuern und sie zu sprengen. Keine leichte Aufgabe, wie Sie schnell bemerken werden.

Den Reiz des neuen Games von Hewson machen ohne Zweifel die hervorragenden Grafikeffekte aus. Auch die Spielidee ist nicht von schlechten Eltern.

Während des Ladevorgangs erfreut Sie ein nicht gerade überwältigendes Ladebild.

Es ist glücklicherweise das Schlechteste am ganzen Spiel. Nachdem das Game komplett im Speicher steht, erscheint der Titelschirm in Kombination mit einem guten und fetzigen Musikstück. Hier können Sie zwischen dem Ein- oder Zweispielermodus und Sound on/off wählen. Jetzt geht's ins Spiel. Sie haben bereits den ersten Turm erreicht und können sich sofort an die wahrlich destruktive Aufgabe machen. Ihr Opfer macht es Ihnen allerdings alles andere als leicht.

Ihre Aufgabe ist es jetzt, den Turm zu besteigen. Dazu sind an der Außenwand Plattformen, kleine Durchgänge und Aufzüge angebracht. Leider sind Sie aber nicht der einzige im Turm. Sämtliche Arten von Aliens, angefangen vom einfachen Ball, bis hin zum hüpfenden Auge ist alles vorhanden, was das Spielerherz begehrt. Allesamt scheinen sie etwas gegen einen zu haben, der versucht, ihre Behausung in die Luft zu jagen . . .

Sie beginnen im untersten Stockwerk eines Turmes. Bis Sie ganz oben sind, gibt es eine Menge zu tun.

Im eigentlichen Sinne ist „Nebulus" nichts anderes als ein Hüpf-, Kletter- und Plattformspiel. Allerdings eines der Sonderklasse.

HERVORRAGENDE GRAFIKEFFEKTE

Der Spieler steuert ein niedliches, jedoch für irdische Augen etwas ungewöhnlich anmutendes Sprite. Seine Animation und Farbgebung ist hervorragend gelungen – wie ausnahmslos auch die der anderen Sprites.

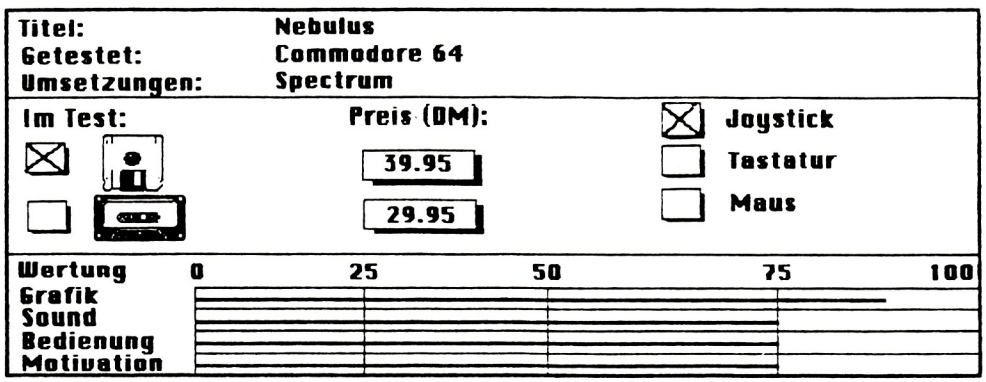
Absolute Spitzenklasse ist allerdings die Grafik des Turms. Kaum einer hätte je gedacht, daß eine solch geschickte Farbgebung und – vor allem – Animation auf dem CBM 64 möglich wäre. Geht das Spielersprite um den Turm herum, so bewegt sich nicht die Figur, sondern das Gebäude scrollt dreidimensional, also mit einer gewissen Verzerrung bei den Rundungen des Bauwerks. Der Anblick ist faszinierend. Durch die geschickte Farbgebung der Ziegelsteine kommt ein Schattierungseffekt hinzu.

Gespielt werden kann bei der Commodore-Version nur über den Joystick. Dieser wird vollkommen verzögerungsfrei und präzise abgefragt und in Bewegungen umgesetzt. Das ist bei „Nebulus" auch sehr not-

wendig, denn sonst hätte man bei dieser Masse an Aliens und Fallen kaum eine Chance, das Ziel zu erreichen, zumal das Spiel sowieso schon sehr schwer ist. Hier sollte auch erwähnt werden, daß Sie nicht so ohne weiteres eines Ihrer drei Bildschirmleben verlieren können. Bei Berührung mit einem anderen Sprite fällt Ihre Spielfigur nur wieder einige Stockwerke nach unten, und muß von dort weitermachen. Berühren Sie jedoch die Wasseroberfläche, so geht eines der drei Leben verloren.
Die Motivation erreicht absolute Spitzenwerte. Jeder, der einmal angefangen hat zu spielen, kommt so schnell nicht mehr vom Monitor weg. Auch nach längerer Zeit flaut der Spielwitz nicht ab, da sich der Spieler immer wieder an der hervorragenden Grafik erfreuen kann. So läßt sich „Nebulus" auch ideal als Grafikdemo verwenden, das so manchen Amigabesitzer vor Neid erblassen läßt.

FAZIT:

Ein Fan von Action- und Geschicklichkeitsspielen muß Nebulus einfach haben. Keiner wird den Kauf dieses Games je bereuen. Die Programmierer haben sich viel Mühe gegeben, aus dem C64 soviel wie möglich herauszuholen – und es ist ihnen offensichtlich auch gelungen. Nebulus ist ein Spiel, das jeder auch nach längerer Zeit wieder gern in den Computer laden wird.
Vertrieb: Ariolasoft, 4830 Gütersloh.



Das Imperium schlägt zurück!

Die futuristische Comicserie fand beim Leser so großen Anklang, daß sogar ein Kinofilm gedreht wurde, welcher aber einen weniger großen Erfolg erlangte. Doch die Schöpfer der sagenhaften Weltraumherren ließen sich dadurch nicht weiter beeindrucken und gaben Gremlin Graphics die Rechte für eine Computerversion.



Seit langer Zeit schon herrscht im Universum ein ständiger Kampf zwischen dem Guten und dem Bösen. Besonders bezeichnend hierfür sind He-Man und Skeletor, die sich wahrscheinlich schon als Kinder nicht ausstehen konnten. Um seine Macht auszuweiten, hat Skeletor den guten He-Man eingesperrt. Nach vielen vergeblichen Versuchen gelang diesem jedoch die Flucht. Zu seiner Überraschung befindet er sich aber nicht auf seinem Heimatplaneten Greyskulle, sondern im modernen Amerika. Um nun in seine Heimat zurückzukehren, muß He-Man acht magische Akkorde sammeln, mit welchen der kosmische Schlüssel aktiviert werden kann. Skeletor setzt natürlich alles dahinter, um dies zu verhindern!
Wie Sie sicher schon erkannt haben, besteht die Aufgabe des Spielers, der natürlich die Rolle des He-Man übernimmt, darin, die acht Akkorde zu finden. Dafür hat man ganze zehn Minuten Zeit und fünf Leben zur Verfügung. Außerdem ist He-Man natürlich mit einem Lasergewehr bewaffnet, dem nie die Energie ausgeht!
Schon nach wenigen Metern erkennt man, daß Skeletor nicht zu unterschätzen ist, denn überall tauchen wie aus dem Nichts bewaffnete Universumskrieger auf, die natürlich sofort das Feuer eröffnen. Jeder Treffer kostet He-Man einen mehr oder weniger großen Teil seiner Energie.

Ist diese komplett aufgebraucht, ist man um ein Leben ärmer. Zum Glück liegen auch ab und zu ein

paar Schwerter herum, mit denen man seinen Energievorrat wieder auffrischen kann. Leider wird einem dadurch das Spiel etwas zu leicht gemacht.
Der Spieler betrachtet das gesamte Spielfeld aus der Vogelperspektive. Neben den Spielfiguren zeigt sich auf dem Monitor noch eine relativ magere Grafik, die nur aus den Begrenzungen der Straße und ein paar Grabsteinen besteht. Hier hätte man schon ein bißchen mehr Mühe hineinstecken können. Von Zeit zu Zeit taucht ein kleines Fenster auf, in welchem sich ein Gesicht zeigt, das offensichtlich irgendeinen Kommentar abgibt. Von Sprachausgabe nicht die Spur! Am unteren Rand des Fensters scrollt zwar der „gesprochene" Text vorbei, ist aber schwerlich zu entziffern.

Von Sound ist auch nicht sonderlich viel zu hören! Außer den Geräuschen der Lasergewehre und dem „sssst" der auftauchenden Soldaten ertönt praktisch nichts aus dem Lautsprecher. Schade, denn eine flotte Begleitmusik könnte den Spieler sicherlich anfeuern und die Motivation somit erheblich steigern!

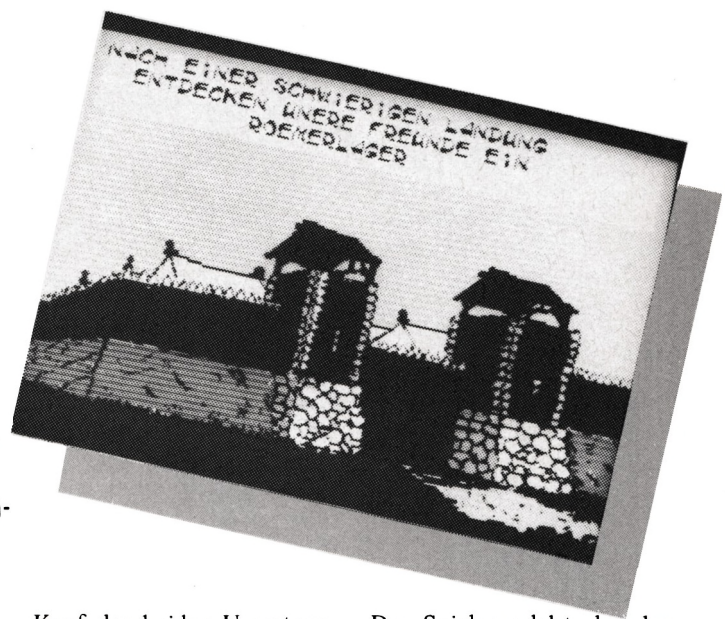
FAZIT:
KEINE GELUNGENE UMSETZUNG!

Zusammenfassend kann man über Masters of the Universe folgendes sagen: Im großen und ganzen ist das Spiel nicht schlecht gemacht, weist aber zu viele Schwachpunkte auf. Durch die vielen Leben und die endlos lange Spielzeit ist das Programm zu leicht. Auch die Grafik und der Sound tragen nicht sonderlich zur Steigerung der Atmosphäre bei. Also, das Geld lieber sparen, denn dieses Programm wird garantiert schon bald in der Schublade landen. *(TB)*

Titel:	**Masters Of The Universe**		
Getestet:	**Schneider CPC**		
Umsetzungen:	**Commodore 64 / Spectrum / Atari ST / MSX**		
Im Test:	**Preis (DM):**	☒ **Joystick**	
☐ 💾		☐ **Tastatur**	
☒ 📼	29,95	☐ **Maus**	

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Comic-Helden auf dem Vormarsch!



Richtig überlegen, um möglichst lange zu überleben, heißt die Devise bei den neuesten Abenteuern von Asterix und Blueberry. Vorschnelle Reaktionen können schlimme Folgen haben!

Neben Lucky Luke (siehe Testbericht an anderer Stelle) wurden auch die neuesten Abenteuer der legendären Comic-Helden Asterix und Blueberry durch die französische Softwarefirma Coktel Vision für viele Computersysteme umgesetzt und von Bomico ins Deutsche übersetzt. Da sich Lucky Luke vom Spielprinzip von den anderen Umsetzungen abhebt, soll diesen hier ein eigener Testbericht zugute kommen.

Während es sich bei Lucky Luke um ein reines Actionspiel handelt, wird bei Asterix und Blueberry auch der Grips des Spielers gefordert. Es muß gleich vorangesagt werden, daß sich der Kauf der beiden Umsetzungen nur dann lohnt, wenn man den gleichnamigen Comicband noch nicht kennt. Es sind dies „Asterix im Morgenland" und „Blueberry und das Gespenst mit den goldenen Kugeln". Der Ablauf der beiden Programme gleicht dem entsprechenden Comicband nämlich aufs Haar.

Der Spieler erlebt also den Ablauf der Handlung vom Monitor aus mit und kann diese teilweise durch „Was-Nun-Menüs" beeinflussen.
Ein solches Menü kann zum Beispiel so aussehen:

1. Wir wollen alle Vorbereitungen treffen;
2. Wir müssen unverzüglich abreisen.



Durch Druck auf die entsprechende Zahlentaste kann man nun eine Handlung der Personen auswählen. Im obigen Beispiel hätte die Taste 2 einen erneuten Spielstart zur Folge, denn wie kann man eine große, weite Reise antreten, ohne auch nur die geringsten Vorbereitungen getroffen zu haben? Sie sehen, auch Überlegen ist bei beiden Programmen notwendig. Vorschnelle Handlungen können schlimme Folgen nach sich ziehen. Stellen Sie sich nur vor, Sie sind kurz vor der Lösung des Spieles und stehen plötzlich wieder ganz am Anfang.

FANTASTISCHE GRAFIKEN

Auf dem Monitor zeigen sich ständig grafisch hervorragende Bilder, die denen des Comics stark ähneln. Fairerweise muß hier jedoch gesagt werden, daß die Feinheit der Grafiken computerabhängig ist. Es liegt auf der Hand, daß die Grafiken des Atari ST ein bißchen besser sind als die des Schneider CPC. Aber keine Angst, sehenswert sind sie auf allen Systemen!

Wenn der Spieler mal nicht zu einer Entscheidung aufgerufen ist, können Sie sich gemütlich mit dem Joystick in der Hand im Sessel zurücklehnen und in aller Ruhe dem Ablauf des Geschehens folgen. Per Tastendruck werden immer neue Grafiken durch Fenstertechnik auf den Monitor geholt. Ab und zu können Sie sich auch eine spezielle Figur aussuchen, deren Handlungen Sie beeinflussen möchten. Teilweise, wenn die Figur nur etwas zu sagen hat, werden Sie auch von einer wahren Redeflut überschüttet, wofür bei Asterix der Häuptling Majestix ein gutes Beispiel ist.

SPRACHAUSGABE

Auf einigen Computern ertönen die Worte parallel zur Bildschirmausgabe auch per Sprachausgabe aus dem Lautsprecher. Leider ist diese kaum verständlich und nur in Französisch gehalten. Aber es dürfte für Übersetzer auch ziemlich schwierig sein, digitalisierte Wörter ins Deutsche zu übersetzen. Bewerten wir diesen Minuspunkt also nicht zu hart.

Ab und zu, beispielsweise wenn Asterix und Obelix den Römern in die Hände gefallen sind, wechselt das gewohnte Monitorbild und Sie müssen Ihr Geschick und Ihre Reaktionsfähigkeit in einem kleinen Actionspiel beweisen. Hierbei müssen Sie einige verstreute Gegenstände in einem Labyrint aufsammeln, welches aus der Vogelperspektive zu sehen ist. Natürlich versuchen die Gegner, im obigen Beispiel also die Römer, Sie daran zu hindern. Diese müssen also auch noch irgendwie verschwinden. Nun, dieses Problem ist auch ganz einfach zu lösen. Man lockt die Römer zu Obelix, der diesen auf den Kopf springt, was bei Obelix' zarter Figur natürlich kein Vergnügen ist. □



Die Ähnlichkeit mit den echten Comics ist verblüffend.

FAZIT

Im großen und ganzen stoßen die beiden Programme auf Gefallen und Begeisterung. Allerdings weisen die beiden Programme auch einige Schwachstellen auf, welche den Spielspaß doch ziemlich mindern. So existiert beispielsweise keine Möglichkeit, den aktuellen Spielstand abzuspeichern, was zur Folge hat, daß man ziemlich lange am Computer sitzen muß, denn wer unterbricht solch ein langwieriges Spiel schon gerne, wenn er bereits ziemlich weit fortgeschritten ist? Auch ist der Preis von 45 Mark doch ziemlich hoch gegriffen. Aber wer mal ein paar Mark übrig hat und den Comicband noch nicht kennt, der sollte sich eines oder beide der Programme ruhig mal ansehen. Da das Spielprinzip aber bestimmt nicht jedermanns Sache ist, sollte man unbedingt vor dem Kauf beim Händler unbedingt mal probespielen! □

Titel:	**Asterix / Blueberry**
Getestet:	**Atari ST**
Umsetzungen:	**Schneider CPC / Commodore 64 / Amiga**

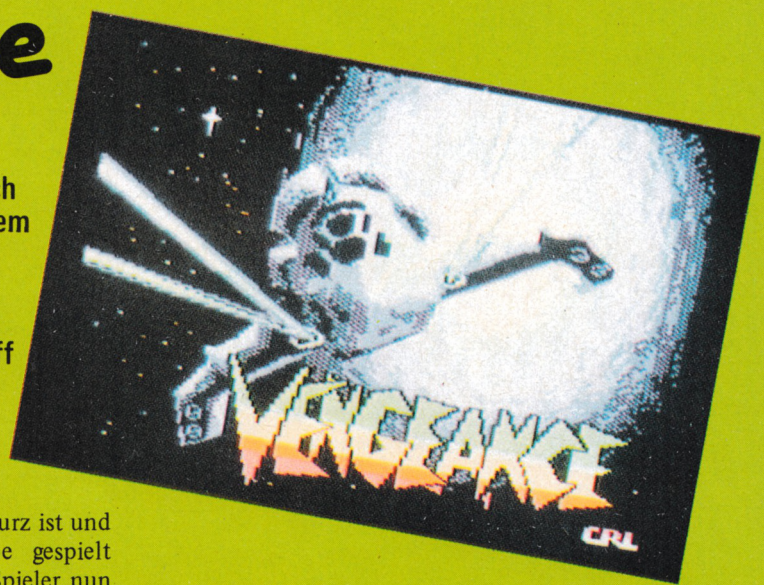
Im Test: ☒ Diskette ☐ Kassette

Preis (DM): 44,95

☒ Joystick ☒ Tastatur ☒ Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Vengeance



Ein neues Spiel nach altem Schema mit neuem Namen. So ungefähr könnte man das neue Game von CRL, "Vengeance", kurz und schmerzlos beschreiben. Ziel und einziger Sinn ist es, durch ein riesiges Alienraumschiff zu fliegen und alles über den Haufen zu ballern, das das Pech hat, Ihnen über den Weg zu laufen.

Die nicht ganz kurze Ladezeit wird zur einen Hälfte von einem sehr gut gelungenen Grafikbild überbrückt, zur anderen bekommt der Spieler nur einen gelöschten Textbildschirm zu sehen, wie ihn jeder kennt. Ist nun Vengeance komplett geladen, so steht man im Eingangsbild. Dieses zeigt eine (sehr karg ausgefallene) Highscoreliste unter einem flimmernden "Vengeance" – Schriftzug. Alles also nicht gerade berauschend. Lobenswert ist jedoch die Musik, die im Eingangsbild rauf- und runtergespielt wird. Ein gut komponiertes und fetziges kleines Sample. So fetzig, daß einem gar nicht auffällt, daß das Stück recht kurz ist und laufend das selbe gespielt wird. Falls dem Spieler nun einmal die Lust an der Musik vergangen ist, so sollte er den Feuerknopf am Joystick drücken.

Jetzt kann er nämlich direkt ins Spielgeschehen einsteigen. Und das, was da geboten wird, kommt dem Ballerspielfreak mit Sicherheit sehr bekannt vor. Ein (noch) einsames Raumschiff im (noch) leeren Weltraum. Nach einigen Sekunden beginnt der Bildschirm jedoch von oben nach unten zu scrollen, und die ersten Teile eines überdimensionalen Raumschiffs erscheinen. Von nun an heißt es, über dessen Oberfläche zu fliegen und auf alles, was sich einen bietet, zu ballern. Das geht immer so weiter – bis man entweder das Ziel erreicht hat, oder abgeschossen wurde, wobei das Letztere wahrscheinlicher ist. Während man so schön vor sich hin düst, sollte nicht übersehen werden, daß es auch acht Objekte gibt, die man nicht abknallen sollte, sondern aufsammeln. Erst wenn alle acht eingesammelt wurden, kann man sich auf dem Weg zum Ende der Raumbasis machen. Aber der Weg ist lang – ob er sich lohnt, ist fraglich.

Die Grafik bewegt sich in der gehobenen Mittelklasse. Wie üblich bei derartigen Games, betrachtet der Spieler die Szenerie aus der Vogelperspektive, also von oben. Wie schon vorhin erwähnt, scrollt die Oberfläche vom oberen Bildschirmrand zum unteren. Das Scrolling ist zwar nicht sonderlich schnell, aber sehr fein aufgelöst und butterweich. Die Farbgebung ist ebenfalls gut gelungen. Auch die Sprites – und derer gibt es viele – sind mit gut erkennbaren Farben ausgestattet und hervorragend animiert. Leider kommt es nicht selten vor, daß feindliche Raumschiffe aus heiterem Himmel direkt vor

oder hinter dem eigenen Gleiter erscheinen. Der Spieler hat da kaum noch eine Chance, auszuweichen.

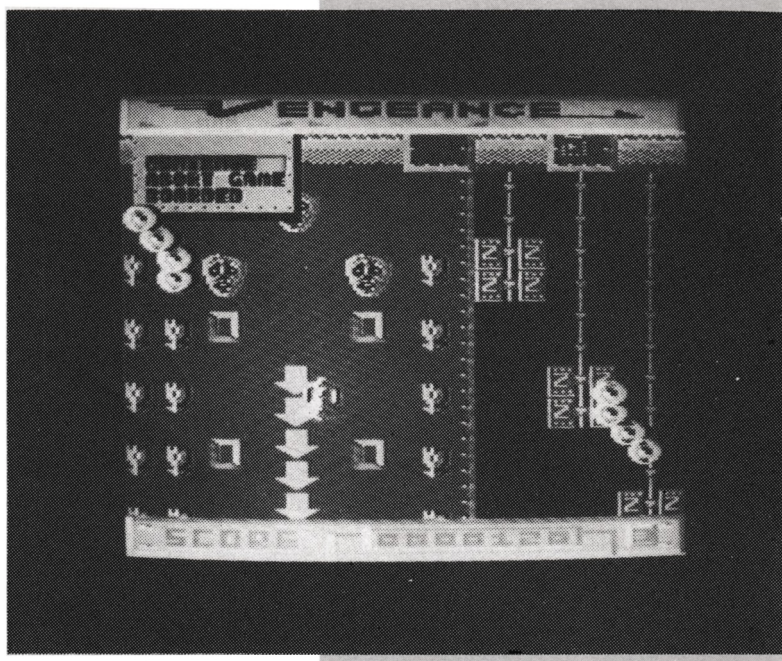
UNZUVERLÄSSIGE STEUERUNG

Die Steuerung des eigenen Raumschiffes hat leider einige Macken. Während sie oft einwandfrei funktioniert, hapert es manchmal an der Geschwindigkeit. Gerade in schwierigen Situationen, wo schnelle Reaktion verlangt ist, kann man am Joystick rütteln, soviel's einem Spaß macht. Das Raumschiff reagiert meist erst nach einer Sekunde – und da ist es sehr oft schon zu spät.
Die Motivation von „Vengeance" . . . naja. Sie ist nicht gerade als überragend zu bezeichnen. Am Anfang mag es ja Spaß machen, ständig zu ballern und nach links oder rechts zu steuern. Schon nach kurzer Zeit sackt der Spielwitz jedoch auf den Tiefpunkt. Die Folge davon ist, daß Vengeance in der Schublade landet und so schnell nicht mehr vorgeholt wird.

FAZIT

Leider ist "Vengeance" ein Spiel, das absolut nichts Neues bietet. Außer einer primitiven Bildschirmballerei wird hier nichts geboten – da kann die Story noch so gut sein! Grafisch und in Sachen Sound ist das Game gar nicht so übel, aber der Spielwitz fehlt. Und was nützt die beste Graphik wenn das Spiel selbst nichts taugt. "Vengeance" ist also nur hartgesottenen Ballerspielfans zu empfehlen.
Vertrieb: Rushware, Kaarst
Preis: Diskette 49,95 Mark.

mn



Titel:	**Vengeance**
Getestet:	**Commodore 64**
Umsetzungen:	

Im Test: **Preis (DM):** 49.95

☒ Joystick
☐ Tastatur
☐ Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Airborne Ranger

Die amerikanische Softwarefirma Microprose ist für ihre qualitativ hochwertigen militärischen Simulationen wie Gunship oder Silent Service bekannt. Diesmal scheint das Softwarehaus jedoch eine Ausnahme gemacht zu haben. Wie soll's ander sein: Kaum vom gewohnten Pfad abgewichen geht schon etwas daneben. In diesem Fall das ganze Spiel.

Militärisch ist es geblieben, aber aus der Simulation wurde ein Aktionsspiel à la Commando. Vielleicht etwas komplizierter und technisch ausgereifter, aber grafisch und in Sachen Geschwindigkeit um drei Klassen schlechter. Das Drumherum wurde etwa wie bei Gunship gestaltet. In vielen einzelnen Menüs kann sich der Spieler für eine von zwölf mehr oder weniger freundliche Missionen für seinen Fallschirmjäger entscheiden. Dazu gehören Spionageaufträge, bei denen er geheime Dokumente fotografieren muß, sowie Sabotage und Attentate auf Flugabwehr, Raketenstellungen und feindliche Abfangjäger. Auch Gefangene kann der Spieler befreien. Das Ganze hörte sich zwar recht gut und aktionsreich an, doch der Eindruck Hat der Spieler alle Parametermenüs hinter sich gebracht, so geht es ans Game. Zuerst steuert er ein Flugzeug nach links und rechts über das feindliche Lager, welches von unten nach oben über den Bildschirm scrollt (wie neu!). Währenddessen kann der Spieler einige Versorgungspakete abwerfen, bevor sein Fallschirmjäger an die Reihe kommt.
Die Grafik, die für diese Sequenzen verwendet wurde, ist nicht nennenswert. Sie scheint aus der Vorzeit der Computergeschichte zu stammen und hätte ohne weiteres auch auf einem ZX 81 realisiert werden können.

Ist der Soldat gelandet, so gelangt man in einen anderen Spielmodus. Ab jetzt wird der Jäger direkt durch die Szenerie gesteuert. Und diese erinnert sehr stark an das Spiel Commando. Der Soldat kann zwar laufen oder kriechen, wohin er will, aber im Grunde genommen hat sich nicht sehr viel geändert. Natürlich wurde das Grundkonzept noch erheblich verfeinert. So gibt es bei Airborne Ranger auch Minenfelder, lange Schützengräben, die der Jäger benutzen kann, Bunker und noch einiges mehr.

An Feinden mangelt es nicht. Der erfahrene Soldat kann zwischen vier verschiedenen Arten von Waffen umschalten. Dazu gehören ein etwas verbessertes M 16-Sturmgewehr, Handgranaten, ein Raketenwerfer und zur Not auch ein Messer und ein Sprengsatz mit Zeitzünder.

Die Grafik dieser Sequenz ist nicht überragend. Der Hintergrund zeigt eine Spur von dreidimensionaler Darstellung, die aber kaum auffällt. Ansonsten gibt es daran nicht viel zu bemängeln. Die vielen einzelnen Gegenstände wie Bunker, Bäume, Gräben und Stacheldraht sind teilweise gut gelungen, teilweise aber auch unkenntlich. Wirklich schlecht sind die Sprites der Soldaten; sowohl die der feindlichen als auch des eigenen.

Es gibt für einen waschechten Soldaten drei Arten der Fortbewegung: Gehen, Laufen und Kriechen. Die Sprites sind sehr sehr grob und ihre Animation läßt zu wünschen übrig. Auch die Farbgebung wurde alles andere als klug gewählt.

Das Ziel des Spiels ist nicht einfach zu erreichen, zumal die Motivation bald stark nachläßt. Das liegt sicher auch an der nicht gerade überragenden Geschwindigkeit. Der Hauptgrund ist aber der, daß es nichts anderes zu tun gibt, als durch die Gegend zu laufen und alles mögliche über den Haufen zu schießen.

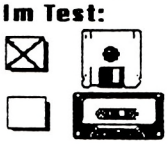
FAZIT

Airborne Ranger paßt wahrhaftig nicht in die Reihe der sehr guten Games von Microprose. Wenn es der erste und letzte Ausrutscher bleibt, so sei er ihnen verziehen. Bis auf pure Knallerlei hat das Spiel nichts zu bieten; abgesehen von den ganz gut gelungenen Menüs. Der Käufer sollte sich von der Action versprechenden Verpackung und Aufmachung und dem hohen Preis von 69 Mark nicht täuschen lassen. Der Inhalt wird dem keinesfalls gerecht.

mn

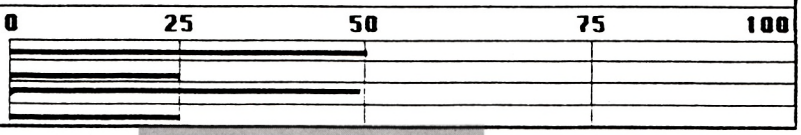
Titel: Airborne Ranger
Getestet: C64
Umsetzungen: CPC

Im Test:



Preis (DM): 69.-

- [x] Joystick
- [ ] Tastatur
- [ ] Maus



Lucky Luke

Es scheint eine neue Masche zu sein, Comicserien oder -Ausgaben auf Homecomputer zu adaptieren. Auch das Softwarehaus Coktel Vision macht hier keine Ausnahme. Der Titel "Lucky Luke" ist nur ein Ausschnitt aus einer Serie von Adaptionen, zu der auch Ausgaben der Comics "Asterix" und "Blueberry" zählen.

Im Spiel haben Sie die Aufgabe, als Lucky Luke eine Westernstadt von Räubern und Banditen zu befreien, sowie einen Zug sicher ans Ziel zu bringen. Dazu warten einige recht schwierige und komplexe Aufgaben auf Sie.

Das Game besteht aus fünf einzelnen Teilen, die jeweils extra nachgeladen werden müssen. Die erste Episode spielt in einer typischen Westernstadt aus sechs Häusern, einem Bahnhof und einer Anlegestelle für Flußdampfer. Sie betrachten die ganze Szenerie aus der Vogelperspektive – auch Lucky Luke, von dem nicht mehr zu sehen ist, als sein riesenhafter Cowboyhut und eine Andeutung von Armen. Jetzt muß es unser Held schaffen, sich Munition zu besorgen, um damit einem Banditen auf dem Flußdampfer das Handwerk zu legen. Ist das vollbracht, ist der erste Teil auch schon gelaufen und Sie können in aller Ruhe zum Bahnhof gehen und mit dem Zug die Stadt verlassen. Um an die Munition für Lucky Luke zu gelangen, müssen Sie erst zur Bank gehen und sich Geld besorgen. Überhaupt spielt die logische Reihenfolge in diesem Teil eine sehr wichtige Rolle.

Die Animation der Spielfigur ist nicht gerade lobenswert. Auch der Spielscreen läßt einiges zu wünschen übrig. Mehrere Gebäude sind durch eine unglückliche Farbgebung nur mit großer Mühe zu erkennen. Die Hauptursache hierfür ist sicher, daß das Spiel fast komplett in BASIC geschrieben wurde, was natürlich enorm auf Kosten der Geschwindigkeit, sowohl bei der Abfrage des Joysticks, als auch bei der Bewegung der Spielfiguren, geht.

Kommen wir zur zweiten Episode:

In ihr heißt es, eine Schießerei mit schier unendlich vielen Banditen zu überleben. Zwanzig davon sollen von Ihnen ins Jenseits befördert werden. Dazu müssen Sie Ihren Bildschirmhelden



vor einer Häuserfront hin- und hersteuern und ballern, was das Zeug hält. Ist das geschafft, spielt sich das selbe nocheinmal ab. Diesmal im Inneren eines Saloons. Diesen Level zu schaffen, ist für den Spieler entweder ein Ding der Unmöglichkeit, oder er macht mit der Episode schon nach einer Minute ein Ende – wenn er den Trick kennt. Dieser ist leider nicht sehr schwer zu entdecken. Durch die primitive Spielidee sind hier Spannung und Spielwitz nicht gerade im Übermaß vorhanden.

Die Grafik von "Lucky Luke" ist recht gut gelungen. Jedoch wurde sie des öfteren durch eine mißlungene Farbgebung verstümmelt. Besitzer eines Grünmonitors sollten das Spiel also lieber auf diesem spielen. Hier kommt die eigentliche Grafik besser zur Geltung.

Ganz anders gibt sich die dritte Episode, welche eine Art Action-Adventure darstellt. Hier haben Sie die Aufgabe, eine entwendete Nitroglyzerinreserve (daher der Name des Games) zu finden und hinter einem Felsen zu plazieren, welcher die Zugstrecke blockiert. Dazu müssen Sie Lucky Luke von Bildschirm zu Bildschirm führen und über Leitern und Abgründe lotsen. Der Teil ist sehr lang und mit mehreren logischen Rätseln ausgestattet, daher ist er nicht einfach zu schaffen und nimmt einige Zeit in Anspruch. Der Bildschirmaufbau benötigt bei einem Screenwechsel ein wenig Zeit, was den Spielablauf aber kaum behindert, da es sich hier sowieso nicht um ein schnelles Actionspiel, sondern mehr um ein Denkspiel handelt. Auf Grafik wurde vermutlich nicht viel Wert gelegt, da diese auch nicht sonderlich ausgefallen ist. Das Spielersprite wurde unserer Ansicht nach etwas zu klein konstruiert und einige Gegenstände sind durch die Farbwahl wieder einmal kaum zu erkennen. Auch die Animation kann nicht gerade gelobt werden, wofür die Hauptschuld aber der Programmiersprache BASIC zuzuschreiben ist. Trotz dieser grafischen Schwächen birgt der Teil doch einige Herausforderungen an den Spieler, die ihn eine Weile an den Bildschirm fesseln können.

MATHEMATISCHE PROBLEME IM VIERTEN TEIL

Erst, wenn dieser Level vollendet ist, gelangen Sie in die vierte Episode. Hier geht es darum, aus einem Wirrwarr von Weichen wieder ein System herzustellen, das der Zug passieren kann. Das dürfte die schwerste Aufgabe des ganzen Spiels sein und eine kleine Herausforderung für Mathematiker. Der Spieler wird hier mit Sicherheit längere Zeit beschäftigt sein, bevor er diesen Level erfolgreich abschließen kann. Die Grafik ist hier gut gelungen und recht übersichtlich, was bei den hier gestellten Problemen auch absolut notwendig ist. Die Steuerung mit dem Joystick ist jedoch langsam und träge, was hier nach längerer Spielzeit auf Kosten der Nerven des Spielers geht.

FAZIT

"Lucky Luke" ist kein übermäßig gutes Spiel und hinkt programmtechnisch seinen Artgenossen in der Zeit hinterher. Trotzdem ist das Game besonders Personen zu empfehlen, die sich gerne einmal nach der Arbeit bei einer kniffligen Denkaufgabe am Monitor entspannen. In Sachen Rätsel hat "Lucky Luke" nämlich eine ganze Menge zu bieten. Ausgesprochene Actionsfans sollten lieber die Hände davon lassen.

Vertrieb: Bomico, Frankfurt. *mn*

Titel:	**Lucky Luke**
Getestet:	**Commodore 64**
Umsetzungen:	**Schneider CPC / Atari ST**

Im Test: ☒ 💾 ☐ 💽

Preis (DM): 49,95

☒ **Joystick**
☐ **Tastatur**
☐ **Maus**

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Kurzberichte!

In letzter Zeit waren die populärsten Softwarefirmen wieder einmal recht rege. Es erschien ein ganzer Haufen von Games für alle möglichen Homecomputer-Systeme. Einige davon wollen wir kurz vorstellen.

JINXTER

Mit dem neuen Adventure Jinxter setzt Magnetic Scrolls seine Reihe von Abenteuerspielen für gehobene Ansprüche fort. Es ist mit dreißig Bildern der gleichen hervorragenden Grafik ausgestattet, wie seine Vorgänger The Pawn und The Guild of Thieves.

Das Adventure weicht jedoch in einem gravierend von den üblichen Abenteuerspiel-Regeln ab: Der Held, und das sind Sie, kann nicht sterben. Jede lebensgefährliche Situation wird durch Glücksfälle wieder in Wohlgefallen aufgelöst. Sie brauchen also nicht ständig um Ihr Leben zu bangen.

Ort der Handlung ist Aquitania, wo es noch Hexen gibt. Da diese ehrenwerten Damen früher viel Unheil anrichteten, beschloß ein guter Zauberer, ein Armband zu schaffen, das dem Treiben Einhalt gebot. Gesagt – getan. Leider hatte die Oberhexe Jannedor dagegen etwas einzuwenden, klaute das Band und zerlegte es in seine Einzelteile, die sie im ganzen Land versteckte.

Jetzt war Aquitania wieder von Pech und Unheil verfolgt. Also muß ein wagemutiger Held her, der die Teile des Bandes wiederfindet, sonst ist das Land verloren. Diesen Helden verkörpert, wie sollte es anders sein, der Spieler.

Jinxter hat einen Parser, der sich ohne weiteres mit einem Infocomparser messen kann. Das Spiel sprachlich auf einem sehr hohen Niveau und zögert nicht, den Spieler und auch andere Adventures durch den Kakao zu ziehen. Es ist erhältlich für die Systeme Atari ST, Amiga, IBM PC und C64 und kostet etwa 80 Mark.

COSMIC CAUSEWAY: TRAILBLAZER II

Aus dem Hause Gremlin kommt eine Fortsetzung zum schon recht betagten, aber guten Trailblazer: Cosmic Causeway. Das Spiel ist im Grunde genauso aufgebaut wie sein Vorgänger. Der Spieler hat die Aufgabe, einen Ball über eine dreidimensionale Rennstrecke mit vielen Hindernissen zu steuern.

Die Strecke ist zwar schnurgerade, wird aber rasend schnell bewegt. An Hindernissen mangelt es auch nicht. Zu den einfachen Löchern im Bodenbelag bei Trailblazer kommen Mauern und verschiedene Monster, die einem das Leben schwer machen, hinzu. Das Spiel gibt es nur auf dem Commodore 64 für etwa 45 Mark für die Diskette.

THE GREAT GIANA SISTERS FÜR AMIGA UND ST

Das Geschicklichkeitsspiel The Great Giana Sisters, das bisher nur die Commodore-64-Besitzer auskosten durften, gibt es jetzt auch für die 16-Bitter Amiga und Atari ST.

Der Spieler steuert eine Heldin, die durch einen Traum in eine Fantasiewelt verschlagen wurde. Aufgabe ist es, sie wieder in unsere Welt zurückzuführen. Das geht jedoch nur mit Hilfe eines wundersamen Diamanten, der erst gefunden werden muß.

Das Spiel hat eine geschickte und flotte Steuerung und wartet mit niedlicher und hervorragender Grafik auf. Auch der Sound läßt kaum Wünsche offen. Ein fetziger Song im Discostil bildet die Intromusik.

SCARY MONSTERS: VOM FILM IN DEN COMPUTER

Scary Monsters nennt man die aus den Filmen hinlänglich bekannten sympathischen Wesen wie Frankenstein oder Dracula. Ihr Gefolge besteht aus nicht minder erschreckenden Kreaturen wie Zombies, Mutanten und einigen weiteren Monstern.

Im Spiel verkörpern Sie den Helden Harry Jones, der auf Graves Island, dem Geburtsort obengenannter Wesen, gestrandet ist. Aufgabe ist es, einem verrückten Professor, dem Erschaffer der Scary Monsters, das Handwerk zu legen und nebenbei noch einigen dieser Kreaturen das Lebenslicht auszublasen. Und wenn Sie schon in Aktion sind, können Sie auch gleich Ihre Freundin von der Insel befreien.

Dazu stehen Ihnen verschiedene Zaubersprüche nebst vielen anderen Waffen zur Verfügung.

Das Spiel ist ein Arcade-Adventure und sollte laut Hersteller hervorragende



Grafiken und viele besondere Extras vorweisen können. Es wird nur für den Commodore 64 zum Preis von etwa 40 Mark für Diskette und 25 Mark für Kassette bei Ariolasoft erhältlich sein.

SIDEWIZE: IM WELTRAUM NICHTS NEUES

Ein neues Actionspiel nach alter Manier programmierte das Softwarehaus Firebird. Die Story können Sie getrost vergessen, denn es geht nur wieder um Krieg und bösartige außerirdische Invasoren.
Sie verkörpern den Weltraumhelden Dynar, der als einziger den Mut hat, den Aliens, nur mit einem Düsenrucksack und einer Laserpistole bewaffnet, entgegenzutreten.
Zur Konfrontation kommt es im leeren Weltraum. Dynar fliegt Scharen von Invarsoren entgegen und muß auf alles ballern, was sich bewegt. Natürlich gibt es verschiedene Bonusgegenstände, die seine Ausrüstung aufbessern.
Ansonsten hat Sidewize nichts Neues zu bieten. Level für Level gilt es, die gleichen Sprites abzuschießen. So wird das Game sehr schnell eintönig und landet bald in der Schublade.
Es ist nur für den C64 für etwa 40 Mark erhältlich.

MASK

Gremlin trägt mit Mask ebenfalls seinen Teil zu den Arcade-Adventures bei.
Auch hier hat der Spieler wieder einmal gegen machtbesessene Außerirdische zu kämpfen, die die Weltherrschaft erringen wollen.

Mask spielt sich an vier Orten ab, wo jeweils zwei Agenten mit Hilfe eines Scanners gerettet werden sollen. Um den Scanner zu aktivieren, müssen Sie jedoch zuerst einen passenden Schlüssel finden.
Die bösartigen Aliens, genannt Venom, bleiben nicht untätig und versuchen, Sie daran zu hindern.

Außer einer sehr einfallslosen Hintergrundgeschichte und einer dreidimensionalen Grafikdarstellung hat Mask nicht Neues vorzuweisen.
Das Spiel ist für den CPC und für den C64 zum Preis von etwa 45 Mark für Diskette erhältlich.

FAZIT

Im Bereich Spiele tut sich auch auf dem Markt der Achtbitter wie C64 und Schneider CPC noch einiges. Leider droht das Niveau der Actionspiele, wie bei Sidewize, wieder etwas zu sinken. Auch Action-Adventures gibt es mittlerweile in Hülle und Fülle, so daß es gut wäre, wenn sich die Softwarefirmen etwas von diesem Sektor wegbewegen würden.
Allerdings werden sowohl Geschicklichkeitsspiele, wie Giana Sisters, als auch Abenteuerspiele immer komplexer und programmtechnisch anspruchsvoller. Hoffen wir, daß das auch in Zukunft so bleibt. *mn*

Ein Poke für alle Fälle

Hallo, liebe Spielefreaks und Joystickkartisten! Diesmal gibts wieder Tips, Pokes und Lösungen für viele Spiele. Von Hacker bis Samantha Fox Strip Poker ist alles vertreten. Ein besonderes Schmankerl haben wir für die CPC-Besitzer: Tips mit Karte zu Hacker II. Im großen und ganzen müßten diese auch für alle anderen Computerversionen des Hacker-Spieles gelten. Schickt uns Pokes, Tips, Karten und Lösungen zu einem x-beliebigen Spiel oder Adventure auf jedem x-beliebigen Computersystem, denn allein von Eurer Mitarbeit leben die Players Pages. Es gibt viel zu tun – poken wir's an.

HACKER II

Thomas Oversberg aus Mechernich hat für die Players Pages das Programm Hacker II auseinandergenommen und dabei einige Dinge entdeckt, die sicher auch alle anderen Hacker interessieren dürften. Thomas selbst hat die Schneider-Version. Thomas ist zufällig auch der Gewinner des monatlichen Orginalspieles. Herzlichen Glückwunsch!

Der Wachmann schaut nur nach vorne und zur Seite. Man kann hinter ihm herfahren, solange er nicht an einer Kreuzung abbiegt. Dann aber sollte man sich schnell in einen Raum verziehen, weil er nämlich kurz nach hinten guckt. Der Wachmensch kommt nicht an allen Kameras vorbei, kann die Gänge aber meistens einsehen. Der Roboter sollte möglichst nie am Tresor vorbeifahren, da dieser von vier Kameras bewacht wird. Die Aktenschränke sind in folgender Reihenfolge zu öffnen: 31, 37, 33 und 35. Andernfalls wird Alarm ausgelöst.

Um zum Tresor zu gelangen, hält man sich zunächst in Raum 35 auf und schaltet das Videoband ein. Die Monitore werden auf das Videobild des Tresors eingestellt. Sobald der Wachmann am Tresor erscheint, muß man Raum 35 in Richtung Osten verlassen und den letzten Monitor auf das Bild des Tresors einstellen. Wenn der Wachmann bei Kamera 1 angelangt ist, fährt man nach Norden, bis es nicht mehr weiter geht (auf Motorgeräusch achten). Dort angelangt, zwei Schritte nach Westen gehen und den MRU nach Norden hin ausrichten. Schon ist man im Tresor angelangt. Die nun entwendende Akte trägt den Namen „Köder". Jetzt den Tresor verlassen und nach Osten fahren, das Live-Bild auf Kamera 18 stellen und den Kontrollmonitor und das TGS wieder einstellen.

Bei Kamera 18 angekommen, muß man nach Süden fahren, bis der Motor aufhört zu brummen. Dann zehn Schritte nach Westen gehen (auf TGS

achten). Jetzt ist das Ziel erreicht. Nun nur noch den MUR auf Süden stellen, dorthin drei Schritte machen und man ist am Ausgang. Noch ein wichtiger Nachtrag: Das TGS-Bild fällt nach dem Öffnen von Aktenschrank 37 aus. Die Richtungsanzeige ist aber noch vorhanden. Ab hier muß man nach Gehör fahren. Deshalb sollte man sich gut mit dem Plan vertraut machen.

Noch ein Extra-Bonbon der Redaktion: Gibt man am Anfang als Passwort den Begriff "COVER" ein, dann befindet man sich gleich am Schluß des Spieles und kann sich für „hervorragende Arbeit" ein Lob ausdrucken lassen.

FIRETRAP

Die wenigen Räume von Firetrap (Schneider CPC) kann man relativ schnell durchspielen, indem man immer an der Hauskante entlang geradeaus nach oben klettert. Wenn rettungsbedürftige Hausbewohner auf dem Weg sind, kann man diese gleich erlösen. Andere Personen oder Hunde sollte man außer acht lassen. Dies hört sich zwar sehr makaber an, trägt aber wesentlich zum schnelleren Beenden von Firetrap bei.

SAMANTHA FOX STRIP POKER

Die Wanze für Samantha Fox Strip Poker hat vor allem bei den Besitzer eines CPC6128 großen Wirbel verursacht. Besitzer dieses CPC-Typs konnten nämlich gar nichts damit anfangen, da der 6128 ständig mit der unhöflichen Meldung „Memory full" abbrach. Also aufgepaßt: Für den CPC6128 wird die Wanze folgendermaßen benutzt:

```
1 MEMORY &A2FF:LOAD"SAM-
  STRIP.BIN":'Wanze
2 OPENOUT"d":MEMORY &1FF:
  LOAD"SAMANTHA.P GM",
  &200:'Samantha
3 call &a300:'starten und staunen!
```

LIGHTFORCE

Ein kleines Programm zu Lightforce für den Schneider CPC, welches die beiden Einzelfiles von Kassette lädt und anschließend ein bißchen herumpoked. Die Folge: Unendlich viele Raumschiffe.

```
10 REM LIGHTFORCE-POKER
20 openout"d":memory &5DC-1:
   print"Bitte warten . . .":load"!
   LIGHTFORCE"
30 poke 1565,201:call &5EF
40 load"!LF",&1CB8:poke &79DA,
   &00
50 print chr$(7):call &642
```

DIE ERBSCHAFT

Das Spiel „Die Erbschaft" von Infogrames ist zwar nicht mehr das aktuellste, aber viele wissen noch nicht, wie sie die Million in Las Vegas gewinnen können. Sind wir doch mal ehrlich: Wer hat das Programm schon vollständig gelöst? Die folgenden Tips zum dritten Teil sollen Ihnen das Gewinnen erleichtern.

In den dritten Teil gelangt man mit dem Codewort OLAAGAKA. Jetzt gilt es, sich erstmal den Grundstock für die Million zu schaffen. Also auf zu Laon! Bevor man bei ihm nicht mindestens drei- bis vierhunderttausend Dollar gemacht hat, ist es sinnlos, sich an anderen Vergnü-



gungsorten herumzutreiben. Immer nur eine Kugel nehmen, das ist sicherer. Dann auf zum Anwalt. Die wertvollen Figuren und Vasen auf seinem Regal kann man stehlen und beim guten Laon zu Geld machen. Aber Vorsicht: Immer nur klauen, wenn der Anwalt die Augen ganz geschlossen hat, sonst ist man dran.

Als nächstes begibt man sich zum Roulette, wo man am besten auf Farben, Pair, Unpair, Mauque und Passe setzt. Wer es unbedingt mit Zahlen versuchen will, dem sei die 17 ans Herz gelegt. Auch beim Spielautomat sollte man sein Glück versuchen, es lohnt sich bestimmt.

Haben Sie gewußt, daß man in der Peepshow auch Geld kassieren kann? Das geht so: Man betritt das Lokal und wartet, bis die Bardame auf der Bühne steht. Dann bewegt man den Cursor auf den Körper der Frau und ist um ein paar tausend Dollar reicher. Allerdings funktioniert das nur einmal.

Um seine Gewinnchancen beim Würfeln zu erhöhen, muß man nur recht freundlich zu den Mitmenschen sein. Wer nämlich am Flughafen (2. Teil) dem Bettler außer dem Finderlohn noch ein paar Dollar extra zugesteckt hat, lernt hier dessen Bruder kennen, der mit Rat und Tat zur Seite steht. Seltsam, wie schnell sich die Gewinnchancen erhöhen lassen.

LUCKY LUKE

Für alle Lucky-Luke-Spieler, die an diesem Game verzweifeln, hier einige lebensrettende Tips für die ersten beiden Levels: Im ersten Level sollte man unbedingt eine bestimmte logische Reihenfolge beim Besuch der einzelnen Stationen beachten: Zuerst muß Lucky zur Bank rechts oben, das ist das Gebäude links neben dem Gleis, gehen. Hat der Bankier seine Bitte an ihn ausgesprochen, so führt man seine Figur zum Waffengeschäft in der Mitte des Bildschirms und besorgt sich Munition, um später den Banditen am Schiffssteg zu vertreiben. Ist das erledigt, geht's zum Bahnhof. Erst jetzt kann Lucky mit dem Zug das erste Bild verlassen.

Der zweite Level ist ein reines Ballerspiel. Hier tut der Spieler gut daran, sich vor einem der Halunken zu postieren und wie wild auf ihn zu schießen. Ist er getroffen, taucht sofort ein neuer an derselben Stelle auf. Denn kann man auch gleich umnieten. Die Chancen, selbst getroffen zu werden, sind bei dieser Taktik nicht sonderlich hoch. Im Saloon spielt sich die gleiche Geschichte ab. Für den dritten Level werden noch Tips gesucht. Wer welche hat, sollte sie uns schicken.

WER WEISS MEHR?

Zu guter Letzt wieder unsere Problemecke: Schmerzhafte Sorgen haben viele Spieler bei Sidewalk (CPC, C64 und ST). Jedesmal, wenn man einen Passanten höflich nach seinem Motorrad fragt und anschließend um die Ecke biegt, taucht irgendein Schlägertyp auf und vermöbelt den armen Spieler. Deshalb zwei Fragen:

1. Gibt es eine Kampfstrategie?
2. Wo ist das Motorrad zu finden?
3. Hat jemand schon eine Karte gemalt?

Zu Masters of the Universe: Wo sind die acht Chords verteilt? Gibt es auch einen Cheatmode oder ähnliches? Noch eine Frage zur US-Version von Aliens: Wer kennt hier die genaue Lösung?

TB/mn





Fortsetzung von Seite 70

gaben von der Tastatur entgegen, sobald das letzte Zeichen der Eingabedatei gelesen wurde. Auch bei GET sind die Optionen ECHO und NOECHO möglich. Die Angabe von SYSTEM hat jedoch eine gänzlich andere Wirkung als bei PUT. Sie veranlaßt, daß auch das auszuführende Programm aus der Eingabedatei gelesen wird. Dies ist allerdings nur dann von Nutzen, wenn mehrere Programme, die nur Standardeingaben verlangen, der Reihe nach abgearbeitet werden sollen.

KOMPLEXES BETRIEBSSYSTEM

Warum bietet nur CP/M Plus diese Möglichkeiten – abgesehen von SUBMIT, das auch in der Version 2.2 existiert? Die Antwort ist einfach und schwierig zugleich. Selbstverständlich entsteht ein so komplexes Programm wie das Betriebssystem CP/M nicht von heute auf morgen. Es wächst mit den Bedürfnissen der Anwender und nicht zuletzt mit den durch die Hardware gegebenen Voraussetzungen. Das ist auch der Grund dafür, weshalb nur Besitzer eines CPC6128 diese Möglichkeiten ausnutzen können. Doch wenn Sie MS-DOS-Anhänger ein wenig demotivieren wollen, warum versuchen Sie es nicht einmal mit einem dezenten Hinweis auf ältere Versionen dieses Betriebssystems?
Der Vergleich von CP/M 2.0 und MS-DOS 3.21 entspricht nämlich der Gegenüberstellung eines Oldtimers mit einem Formel-Eins-Wagen.
Und zu guter Letzt sollte auch nicht vergessen werden, daß CP/M „nur" ein Acht-Bit-Betriebssystem ist und ihm somit durch die Hardware erhebliche Einschränkungen auferlegt werden. *AE* □

Rund um den Computer

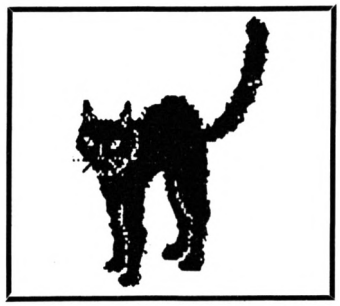
Das weltweite Computernetz von IBM, zu dem rund 300 000 ordentlich angemeldete Benutzer Zugang haben, bekam kurz vor Weihnachten 1987 Besuch vom Weihnachtsmann. Auf dem Bildschirm erschienen plötzlich die besten Wünsche zum bevorstehenden Fest, verbunden mit der hübschen Grafik eines Weihnachtsbaumes. Ob-

SCHÖNE BESCHERUNG BEI IBM

wohl sich die weitere Drohung „Alle Daten werden nun gelöscht" als makabrer Spaß entpuppte, herrschte helle Aufregung.
Einerseits wäre es sicherlich möglich gewesen, die Daten über Kunden und Lieferungen zu löschen, andererseits haust der Virus nun im ganzen System. Zwar soll *Christmas Exec* unschädlich gemacht worden sein, aber niemand ist sicher, ob sämtliche Viren gefaßt wurden. □

WIE COMMODORE ZUR KATZE KAM

Einen Spaß wollte sich ein kinderloses Lehrerehepaar in Korbach, Hessen, erlauben und deckte damit einen Skandal auf: Felix, Kater der Familie, wurde auf den Bögen der Volkszählung zum Familienmitglied erklärt. Da man amtlicherseits bei Haushaltsmitgliedern nicht mehr nachfragt, ob es sich um Tiere oder Menschen handelt, hätte die Sache eigentlich gutgehen müssen. Ganz so anonym waren die Daten jedoch nicht. Die Meldebehörde fragte schnell nach, um welchen Zuwachs es sich da handelt. Der Spaß konnte erklärt werden, und da anscheinend die Rechtslage etwas unklar ist, hatte er keine Folgen.
Bis dann plötzlich Commodore nicht auf das Tier verzichten wollte. Der Firma, auf Kundenjagd, kam der Zwölfjährige gerade recht und lud ihn zu einer schicken Fete ein. Zu spät, denn mittlerweile war der Kater verstorben und es bleibt auch fraglich, ob er zu Lebzeiten großes



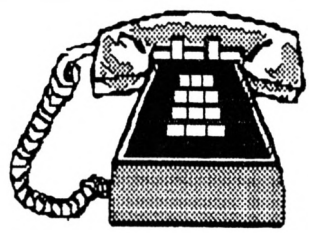
Interesse an der EDV hatte. Interessant wäre der Besuch sicherlich gewesen, denn vielleicht hätte man erfahren, wie denn Commodore an die anonymen Daten der Volkszählung kam. □

BRILLEN FÜR ENTWICKLUNGSLÄNDER

Die Idee gibt es schon länger, sie läßt sich nun jedoch etwas einfacher durchführen: Die abgelegten Brillen – weil unmodisch oder nicht mehr richtig angepaßt – deutscher Wohlstandsbürger konnten bislang der Dritten Welt gespendet werden. Was hierzulande ausgezeichnet klappt, nämlich die Beratung durch einen Augenarzt, fehlte aber oft in Afrika. Dort hätten die eingehenden Sehhilfen erst vermessen und dem richtigen Patienten übergeben werden müssen. In der Praxis war es jedoch oft ein Selbstbedienungsladen.
Ein Garchinger Augenarzt sorgte endlich dafür, daß die richtige Brille an den richtigen Bedürftigen kommt. Sein wesentliches Hilfsmittel ist ein Computer. Hier sind alle Sehhilfen gespeichert, die als Spende zur Verfügung stehen. Meldet ein Land einen Bedarf, so kann das Richtige herausgesucht und gezielt verschickt werden. Es steht nicht mehr zu befürchten, daß eine wahllos aus der Sendung herausgefischte Brille Kopfschmerzen verursacht und wieder weggeworfen wird. □

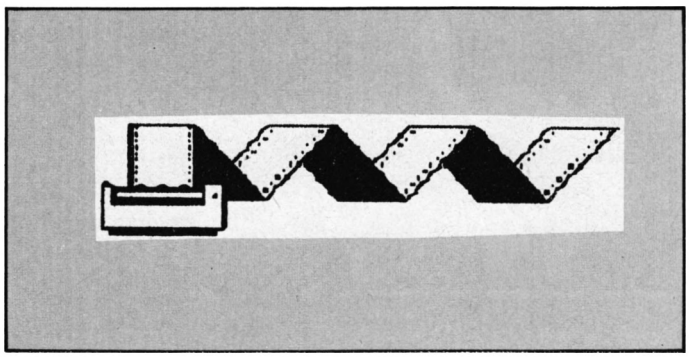
COMPUTER IN DER ANÄSTHESIE

High-Tech in der Medizin ist schon längst nichts Neues mehr. Patienten werden sich in Zukunft aber an weitere Computer, die zu ihrem Wohl im Operationssaal eingesetzt werden, gewöhnen müssen. Den Anästhesie-Ärzten, deren Aufgabe es ist, dafür zu sorgen, daß der Patient während der Operation schläft, danach aber wieder aufwacht, steht ein elektronischer Helfer zur Verfügung. Bislang befindet sich in einer Münchner Privatklinik eine Gerätekombination in Erprobung, mit der Blutdruck, Herzfrequenz und Sauerstoffsättigung laufend gemessen und aufgezeichnet werden. Die Ärzte können sich voll und ganz auf die Narkose konzentrieren.
Daß dabei auch Hemmschwellen besonderer Art zu überwinden sind, hat man schon erkannt. Bislang wurde nämlich über die Operation ein Narkoseprotokoll angefertigt, bei dem so manche Meßkurve beschönigt wurde. Kollege Computer ist da unbestechlich, das Protokoll weist Fehler in der Behandlung zweifelsfrei nach. □



MINI-DATEI

Das Datei-verwaltungs-programm für alle CPCs



Unser Leser Markus Wildi hat ein Dateiprogramm geschrieben, das sich mit einigen professionellen Programmen dieser Art durchaus vergleichen lassen kann. Abtippen lohnt sich!

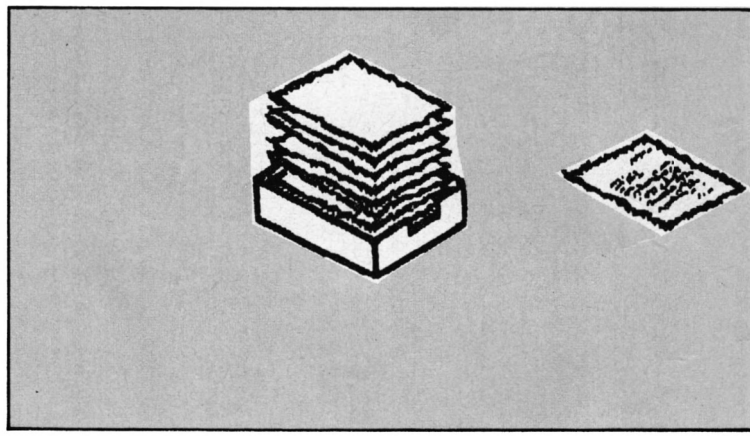
Die mit „Mini-Datei" angelegten Daten können von keinem anderen Dateiverwaltungsprogramm geladen werden!
Bevor Sie das Programm starten, sollten Sie eine leere, im Data-Format formatierte Diskette bereit halten. Sie werden Sie später beim Abspeichern brauchen.
Um Ihnen die einzelnen Menüpunkte und Funktionen zu erklären, erstellen wir eine Adressenverwaltung. Zuerst stehen Ihnen drei Hauptmenüs zur Auswahl:
1. Programm
2. Diskette
3. Sonstiges.

Um eine Datei zu erstellen, in unserem Fall eine Adressenverwaltung, wählen Sie als erstes Menü 3 Sonstiges. Es erscheint ein Hilfsmenü mit folgenden Funktionen:
– Datei laden
– Datei anlegen
– Funktionstasten
– Datum
– Farben
– Eingabemaske
– Druckmaske
– Drucker
– Feldlänge
– Feldanzahl
– Ende.

Wählen Sie nun (mit Cursortasten + Copy) die Funktion – Eingabemaske –. Es wird die Eingabemaske einer Programmverwaltung angezeigt. Da wir jedoch eine Adressenverwaltung erstellen möchten, ändern Sie die Eingabemaske wie folgt:

Programmverwaltung	– Adressenverwaltung
1. Programmname	1. Anrede
2. Hersteller	2. Name
3. Datenträger	3. Vorname
4. Preis	4. Straße/Nummer
5. Bestellnummer	5. Plz/Ort
6. Computer	6. Telefon
7. Art	7. Beruf
8. Monitor	8. Bemerkung
9. Steuerung	9. –
10. Bewertung	10. –

Es stehen Ihnen also zehn Eingabefelder zur Verfügung. Da wir nicht, wie in der Programmverwaltung, zehn Felder benötigen, sondern nur acht, wählen Sie nun die Funktion – Feldanzahl –.
Bestimmen Sie mit den Cursortasten die Feldanzahl. In unserem Fall wählen Sie die Anzahn von acht Feldern.
Eingabe beendet? (ESC)
Da Sie wieder in Menü 3 sind, wählen Sie nochmals die Funktion – Eingabemaske –. Sie sehen jetzt, daß sich das Programm auf die acht Felder beschränkt hat.



(ESC) verläßt Funktion und kehrt zu Menü 3 zurück.
Die Funktion – Feldlänge – ist genau so zu bedienen wie die Funktion – Feldanzahl –, jedoch kann hier die Feldlänge von jedem einzelnen Feld bestimmt werden.
(ESC) wenn Eingabe beendet!

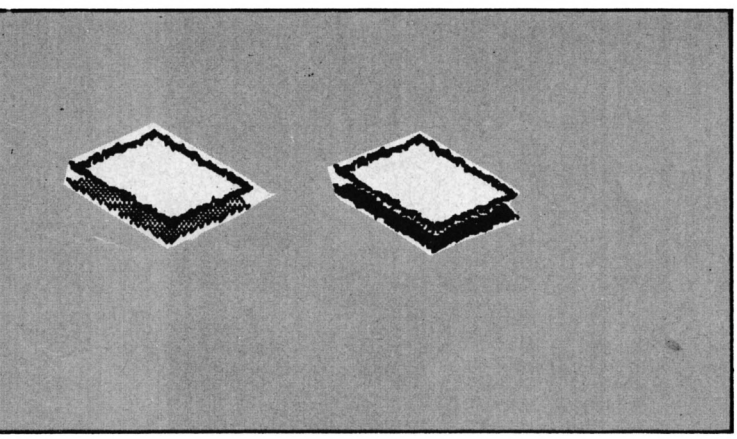
Druckmaske:

Mit dieser Funktion können Sie die Reihenfolge der ersten fünf Felder für den Listen- oder Einzeldruck bestimmen. Mehr dazu später.
Bevor Sie nun Adressen eingeben, sollten Sie eine Datei eröffnen beziehungsweise anlegen. Wählen Sie dazu die Funktion – Datei anlegen –. Geben Sie den Namen Ihrer Datei ein, zum Beispiel: Adressen.Dat, und wählen Sie anschließend Anlegen.
Achtung! Beachten Sie die Anweisung:
Bitte legen Sie eine leere Datendiskette ein . . .
und drücken Sie die Enter-Taste, da sonst beim Anlegen einer beschriebenen Diskette vorhandene Daten gelöscht werden können! Pro angelegte Datei benötigen Sie eine getrennte Datendiskette!
Der Rechner ermittelt die maximalen Datensätze, die auf einer Diskettenseite Platz haben, von selbst. Sobald die Datei angelegt ist, kehrt das Programm zu Menü 3 zurück.
Wenn Sie Ihre Datei später einmal laden möchten, so wählen Sie die Funktion – Datei laden –.

Geben Sie ein: Adressen.Dat und wählen Sie anschließend – Laden –.
Jetzt können Sie Adressen eingeben. Wählen Sie dazu Menü 1 Programm . Es erscheint wieder ein Hilfsmenü mit folgenden Funktionen:
– Eingeben
– Drucken
– Ändern
– Löschen
– Sortieren
– Suchen
– Weiter
– Zurück
– Nummer.

Eingeben:

Es erscheint die von Ihnen erstellte Eingabemaske. Geben Sie nun Ihre persönlichen Daten ein. Möchten Sie weitere Adressen eingeben, so wählen Sie die Funktion – Weiter –. Um Ihnen das Eintippen der Adressen zu erleichtern, stehen Ihnen die Funktionstasten zur Verfügung. (Menü 3: Funktion – Funktionstasten –).



Zurück/Nummer:

Mit – Zurück – blättern Sie eine Adresse zurück. Wählen Sie – Nummer –, so können Sie eine bestimmte Adresse aus dem gesamten Datensatz abrufen.

Suchen:

Wieder erscheint die komplette Eingabemaske. Fahren Sie mit den Cursortasten zum Beispiel auf Name und drücken Sie Copy. Tippen Sie nun den zu suchenden Namen ein, zum Beispiel Huber. Wenn Sie danach die Taste "S" drücken, wird der Rechner alle Adressen suchen, die Huber enthalten. Möchten Sie jedoch alle Adressen, die Huber heißen und in der Südstraße wohnen, so geben Sie beim Feld Straße einfach Südstraße ein . . . Hat der Rechner die gesuchte Adresse gefunden, so meldet er sich mit einem weiterem Hilfsmenü:

Weiter – Menü – Drucken:

Haben Sie sich für –Weiter– entschieden, so sucht der Rechner nach weiteren möglichen Adressen. Bei – Drucken – wird die gefundene Adresse ausgedruckt. (Falls Ihr Drucker nicht empfangsbereit sein sollte, so wählen Sie Menü 3: Funktion – Drucker –.) Bei – Menü – wird der Suchvorgang gestoppt und ins Menü 1 zurückgesprungen.

Sortieren:

Wieder wird die Eingabemaske angezeigt. Wählen Sie mittels Cursortasten + Copy, nach welchem Kriterium sortiert werden soll. Eine weitere Frage erscheint, bei der Sie angeben müssen, zwischen welchen Adressennummern ausgelesen werden soll. Achtung: Die Sortierroutine kann, je nach Anzahl der Datensätze, einige Zeit brauchen; haben Sie Geduld! Für eine volle Diskette werden 24 Stunden und mehr benötigt!

Drucken:

Nun möchten Sie sicher Ihre Daten zu Papier bringen. Wählen Sie die Funktion – Drucken –. Sie können nun die augenblickliche Adresse (Einzeldruck), oder eine Liste (Liste) ausdrucken. Bei der Liste wird noch zusätzlich gefragt, welche Adressen gedruckt werden sollen.

Ändern:

Steuern Sie mit den Tasten (Tab) und (Caps Lock) auf das zu ändernde Feld. Wählen Sie danach mit den Cursortasten + Copy die Funktion – Ändern –, um das gewählte Feld zu korrigieren.

Löschen:

Die augenblickliche Adresse wird gelöscht.

Nun zu Menü 2, dem Diskettenmenü: (ESC, dann Menü 2 wählen):
– Löschen
– Umbenennen
– Starten
– Laden
– Katalog
– User
– Laufwerk.

Löschen:

Es wird nach dem zu löschenden File gefragt.

Umbenennen:

Zuerst wird nach dem alten, dann nach dem neuen Filenamen gefragt. Sollte das gefragte File nicht auf Diskette vorhanden sein, so wird ins Menü zurückgesprungen.

Starten/Laden:

Das gesuchte Programm wird von Diskette gestartet beziehungsweise geladen.
Wichtig: Sollte das gesuchte File nicht auf Diskette vorhanden sein, so wird das Programm unterbrochen, und Sie müssen „Mini-Datei" von neuem starten.

Katalog:

Der Disketteninhalt wird angezeigt.

User:

Die Usernummer (0 bis 16) kann mit den Cursortasten bestimmt werden.

Laufwerk:

Bei dieser Funktion kann zwischen Laufwerk A und B gewechselt werden. Es ist also das Arbeiten mit zwei Laufwerken möglich. *Markus Wildi*□

```
100 '*****************************  <2234>
110 '*   STARTPROGRAMM FUER     *  <2229>
120 '*         MINIDATEI        *  <22F6>
130 '*            VON           *  <2275>
140 '*       MARKUS WILDI       *  <2276>
150 '*           FUER           *  <2228>
160 '*   SCHNEIDER CPC-WELT     *  <2247>
170 '*    CPC 464/664/6128   JE*  <2256>
180 '*****************************  <22D4>
1000 CALL &BBFF:CALL &BB00          <0ED4>
1010 POKE &BDEE,&C9                 <0D54>
1020 MODE 2:BORDER 0:INK 0,0:INK 1
,18                                 <159B>
1030 SYMBOL AFTER 256:MEMORY &8D87
-1                                  <115F>
1040 LOAD"MC1",&A000                <0FB4>
1050 CALL &A000                     <09E8>
1060 LOAD"MC2",&A690                <0FF9>
1070 IF PEEK(6)=&7B THEN GOSUB 126
0 ELSE IF PEEK(6)=&91 THEN GOSUB 1
310                                 <25AA>
1080 SYMBOL AFTER 64                <092E>
1090 SYMBOL 64,60,64,60,66,60,2,60
                                    <1C1C>
1100 SYMBOL 123,198,0,120,12,124,2
04,118,0                            <1E3D>
1110 SYMBOL 125,198,0,102,102,102,
102,62                              <1C45>
1120 SYMBOL 124,198,0,60,102,102,1
02,60                               <1C1F>
1130 SYMBOL 255,255,85,85,85,85,85
,255,255                            <201F>
1140 SYMBOL 126,120,198,198,252,19
8,198,248,192                       <20A2>
1150 SYMBOL 91,219,60,102,102,126,
102,102,0                           <1F96>
1160 SYMBOL 93,198,0,198,198,198,1
98,60,0                             <1E66>
1170 SYMBOL 92,198,56,198,198,198,
198,124,0                           <1F96>
1180 KEY DEF 28,1,123,91:KEY DEF 2
6,1,125,93:KEY DEF 29,1,124,92      <2B44>
1190 KEY DEF 25,1,126,45:KEY DEF 7
1,1,121,89:KEY DEF 43,1,122,90      <2B75>
1200 KEY DEF 25,1,126,63:KEY DEF 2
4,1,94,96:KEY DEF 22,1,60,62        <2BF4>
1210 KEY DEF 31,1,46,58:KEY DEF 19
,1,35,39:KEY DEF 18,1,13,1,30       <2DA8>
1220 KEY DEF 57,1,51,64:KEY DEF 39
,1,44,59:KEY DEF 32,1,48,61         <2BDF>
1230 KEY DEF 17,1,43,42:KEY DEF 41
,1,55,47:KEY DEF 9,1,224,248        <2A0F>
1240 KEY DEF 79,1,127,250:KEY DEF
30,1,45,95:KEY DEF 66,0,0,0,0       <2BA0>
1250 RUN"mdatei"                    <0E67>
1260 POKE &A6A1,&1:POKE &A6A2,&C:P
OKE &A6AA,&1B                       <1F21>
1270 POKE &A6B8,&1:POKE &A6B9,&C:P
OKE &A6C1,&1B                       <1F42>
1280 POKE &A6D2,&1:POKE &A6D3,&C:P
OKE &A6DB,&1B                       <1FB2>
1290 POKE &A6EA,&4E:POKE &A6E6,&E0
:POKE &A6ED,&97:POKE &A6F2,&E0      <28A0>
1300 RETURN                         <0654>
1310 POKE &A6A1,&5:POKE &A6A2,&C:P
OKE &A6AA,&1F                       <1F24>
1320 POKE &A6B8,&5:POKE &A6B9,&C:P
OKE &A6C1,&1F                       <1F48>
1330 POKE &A6D2,&5:POKE &A6D3,&C:P
OKE &A6DB,&1F                       <1FB6>
1340 POKE &A6EA,&52:POKE &A6E6,&E4
:POKE &A6ED,&9B:POKE &A6F2,&E4      <28E5>
1350 RETURN                         <06B8>

100 '*****************************  <2234>
110 '* MINIDATEI-HAUPTPROGRAMM *  <22B4>
120 '*            VON          *  <2261>
130 '*       MARKUS WILDI      *  <2262>
140 '*           FUER          *  <2214>
150 '*     SCHNEIDER CPC-WELT   *  <22E3>
160 '*      CPC 464/664/6128  JE*  <222B>
170 '*****************************  <22C0>
1000 /TAPE:OPENOUT"D":MEMORY HIMEM
-1:CLOSEOUT:/DISC                   <1FA2>
1010 DIM mo$(11)                    <0F75>
1020 x0=23:y1=8:y2=13:y3=8:y4=10:y
5=9:y6=9:y7=14:y8=8:y9=15:z9=1:y10
=8:z10=1:y11=10:y12=18:y13=12:y14=
15:x15=36:y16=8                     <AE2D>
1030 s$=STRING$(80,154):recr$=SPAC
E$(250):datei$="PROGRAMM":in$=CHR$
(24)                                <43D3>
1040 nummer=1:stift=18:ta=29:mo=9:
jahr=1986:nummermax=706             <4A31>
1050 feldl=250:felda=10:lo=5:datum
$="29. OKTOBER   1986"              <422C>
1060 FOR a=0 TO 9:fu$(a)=CHR$(48+a
)+SPACE$(9):feldl(a)=25:NEXT        <4065>
1070 FOR a=0 TO 4:druck(a)=a:NEXT   <242B>
1080 RESTORE 1200                   <09DC>
1090 FOR a=0 TO 9:READ a$:k$(a)=a$
+SPACE$(14-LEN(a$)):NEXT            <3609>
1100 FOR a=0 TO 8:READ a$:p$(a)=a$
+SPACE$(9-LEN(a$)):NEXT             <358A>
1110 FOR a=0 TO 2:READ a$:m$(a)=a$
:NEXT                               <2680>
1120 FOR a=0 TO 6:READ a$:disc$(a)
=a$+SPACE$(10-LEN(a$)):NEXT         <3905>
1130 FOR a=0 TO 10:READ a$:so$(a)=
a$+SPACE$(15-LEN(a$)):NEXT          <38BC>
1140 FOR a=0 TO 2:READ a$:fa$(a)=a
$+SPACE$(6-LEN(a$))+":":NEXT        <3ACA>
1150 FOR a=0 TO 2:READ a$:da$(a)=a
$+SPACE$(11-LEN(a$))+":":NEXT       <3BC6>
1160 FOR a=0 TO 1:READ a$:dl$(a)=a
$+SPACE$(11-LEN(a$))+":":NEXT       <3BCF>
```

```
1170 FOR a=0 TO 1:READ a$:drucker$
(a)=a$+SPACE$(8-LEN(a$)):NEXT      <3B7F>
1180 FOR a=0 TO 11:READ a$:mo$(a)=
a$+SPACE$(9-LEN(a$)):NEXT          <370F>
1190 FOR a=0 TO 2:READ a$:sum$(a)=
a$++SPACE$(7-LEN(a$)):NEXT         <384D>
1200 DATA "PROGRAMMNAME","HERSTELL
ER","DATENTRIGER","PREIS","BESTELL
NUMMER","COMPUTER","ART","MONITOR"
,"STEUERUNG","BEWERTUNG"           <7B65>
1210 DATA "EINGEBEN","DRUCKEN","[N
DERN","L\SCHEN","SORTIEREN","SUCHE
N","WEITER","ZURICK","NUMMER"      <5E52>
1220 DATA "PROGRAMM","DISKETTE","S
ONSTIGES"                          <2839>
1230 DATA "L\SCHEN","UMBENENNEN","
STARTEN","LADEN","CATALOG","USER",
"LAUFWERK"                         <4BF2>
1240 DATA "DATEI LADEN","DATEI ANL
EGEN","FUNKTIONSTASTEN","DATUM","F
ARBEN","EINGABEMASKE","DRUCKMASKE"
,"DRUCKER","FELDLINGE","FELDANZAHL
","ENDE"                           <8DBB>
1250 DATA "PAPIER","RAND","STIFT"  <1E05>
1260 DATA "DATEINAME","DATENSITZE"
,"ANLEGEN"                         <2910>
1270 DATA "DATEINAME","LADEN"      <1A2C>
1280 DATA "NLQ 401","DMP 2000"     <1B20>
1290 DATA "JANUAR","FEBRUAR","MIRZ
","APRIL","MAI","JUNI","JULI","AUG
UST","SEPTEMBER","OKTOBER","NOVEMB
ER","DEZEMBER"                     <71E5>
1300 DATA "WEITER","MENI","DRUCKEN
"                                  <20FF>
1310 MODE 2                        <07EE>
1320 PRINT s$" (c) BY MARKUS WILDI
 JUN.      M I N I - D A T E I"SPC
(11)datum$s$                       <4F39>
1330 LOCATE 23,5:PRINT"PROGRAMM
   DISKETTE      SONSTIGES"        <334C>
1340 WINDOW#2,12,56,10,24          <14F6>
1350 GOSUB 6700                    <092F>
1360 INK 1,stift                   <10CF>
1370 LOCATE x0,5:PRINT in$m$(z0)in
$                                  <2472>
1380 FOR w=1 TO 10:NEXT            <115C>
1390 IF INKEY(1)=0 AND x0<49 THEN
x0=x0+14:z0=z0+1:LOCATE x0-14,5:PR
INT m$(z0-1):GOTO 1370             <5358>
1400 IF INKEY(8)=0 AND x0>23 THEN
x0=x0-14:z0=z0-1:LOCATE x0+14,5:PR
INT m$(z0+1):GOTO 1370             <5366>
1410 IF INKEY(9)=0 THEN 1430       <116F>
1420 GOTO 1390                     <0991>
1430 IF x0=23 THEN i=13:GOSUB 7330
:GOTO 1470                         <2089>
1440 IF x0=37 THEN i=9:GOSUB 7330:
GOTO 3390                          <1F53>
1450 IF x0=51 THEN i=9:GOSUB 7330:
GOTO 3960                          <1F98>
1460 GOTO 1380                     <0991>
1470 IF open=0 THEN GOSUB 6890:GOT
O 1380                             <19D4>
1480 IF feldl<>fl OR felda<>fa THE
N feldl=fl:felda=fa:GOSUB 6700     <46CF>
1490 FOR a=0 TO felda-1            <1799>
1500 IF feldl(a)<>feldlsi(a)THEN f
eldl(a)=feldlsi(a)                 <4540>
1510 NEXT                          <0693>
1520 recr$=SPACE$(fl)              <16A7>
1530 WINDOW#2,63,77,7,24           <1368>
1540 MOVE 502,296:DRAW 592,296:DRA
W 592,134:DRAW 502,134:DRAW 502,29
6                                  <2FAC>
1550 FOR a=0 TO 8:LOCATE 65,8+a:PR
INT p$(a):NEXT                     <27B0>
1560 IF z2>felda THEN z2=felda:y2=
8+felda                            <34D8>
1570 nu=(nummer-1)*(feldl+2):/RECR
EAD,@nu,@recr$:GOSUB 6760          <44A3>
":
LOCATE 14,y2:PRINT in$k$(z2)":"in$
                                   <4559>
1590 '                             <077D>
1600 IF INKEY(2)=0 AND y1<16 THEN
y1=y1+1:z1=z1+1:LOCATE 65,y1-1:PRI
NT p$(z1-1):GOTO 1580              <520A>
1610 IF INKEY(0)=0 AND y1>8 THEN y
1=y1-1:z1=z1-1:LOCATE 65,y1+1:PRIN
T p$(z1+1):GOTO 1580               <5177>
1620 IF INKEY(70)=0 AND y2<13+feld
a-1 THEN y2=y2+1:z2=z2+1:LOCATE 14
,y2-1:PRINT k$(z2-1)":":GOTO 1580  <6187>
1630 IF INKEY(68)=0 AND y2>13 THEN
 y2=y2-1:z2=z2-1:LOCATE 14,y2+1:PR
INT k$(z2+1)":":GOTO 1580          <5680>
1640 IF INKEY(9)=0 THEN i=9:GOSUB
7330:ON z1+1 GOSUB 1670,1790,2220,
2300,2350,2720,3180,3240,3300      <468A>
1650 IF INKEY(66)=0 THEN CLS#2:CLO
SEIN:GOTO 1390                     <198E>
1660 GOTO 1590                     <09BC>
1670 LOCATE 14,y2:PRINT k$(z2)":":
dat$=""                            <28C7>
1680 FOR a=0 TO felda-1:LOCATE 30,
13+a:PRINT SPACE$(25):NEXT         <2DEA>
1690 FOR a=0 TO felda-1            <172A>
1700 LOCATE 14,13+a:PRINT in$k$(a)
":"in$SPC(feldl(a)+1)              <3C81>
1710 LOCATE 30,13+a:laenge=feldl(a
):mini=32:maxi=126:GOSUB 6540      <449A>
1720 dat$=dat$+e$+SPACE$(feldl(a)-
LEN(e$)):d$(a)=e$+SPACE$(feldl(a)-
LEN(e$))                           <5F90>
1730 LOCATE 14,13+a:PRINT k$(a)":"
                                   <1FF0>
```

```
1740 NEXT                                  (0660)
1750 nu=(nummer-1)*(feldl+2)               (2509)
1760 /RECWRITE,@nu,@dat$                   (1E04)
1770 LOCATE 14,y2:PRINT in$k$(z2)"
:"in$                                      (289C)
1780 RETURN                                (0615)
1790 IF druckon=0 THEN GOSUB 7340:
RETURN                                     (19D4)
1800 WINDOW#3,33,47,10,13:CALL &A6
90,3,&9310:CLS#3                           (2361)
1810 MOVE 264,248:DRAW 368,248:DRA
W 368,200:DRAW 264,200:DRAW 264,24
8                                          (2C99)
1820 LOCATE 35,11:PRINT in$"EINZEL
DRUCK"in$:e=2                              (2BB2)
1830 LOCATE 35,12:PRINT"LISTE"             (1499)
1840 FOR w=1 TO 10:NEXT w                  (15CD)
1850 IF INKEY(2)=0 THEN LOCATE 35,
11:PRINT"EINZELDRUCK":LOCATE 35,12
:PRINT in$"LISTE      "in$:e=1             (4AF6)
1860 IF INKEY(0)=0 THEN LOCATE 35,
12:PRINT"LISTE      ":LOCATE 35,11
:PRINT in$"EINZELDRUCK"in$:e=2             (4A48)
1870 IF INKEY(9)=0 THEN IF e=1 THE
N GOSUB 1900:CALL &A693,4,&953A EL
SE von=nummer:bis=von:such=1:GOSUB
 1960                                      (542D)
1880 IF INKEY(66)=0 THEN CALL &A69
3,3,&9310:i=66:GOSUB 7330:RETURN           (2830)
1890 GOTO 1840                             (095D)
1900 WINDOW#4,28,58,22,24:CALL &A6
90,4,&953A:CLS#4                           (23D6)
1910 MOVE 222,54:DRAW 460,54:DRAW
460,26:DRAW 222,26:DRAW 222,54             (29DF)
1920 LOCATE 30,23:PRINT"DRUCKEN VO
N ";:laenge=5:mini=48:maxi=57:GOSU
B 6540                                     (4328)
1930 von=VAL(e$):IF von>nummermax
OR von=0 THEN LOCATE 42,23:PRINT S
PC(5):GOTO 1920                            (4433)
1940 LOCATE 42,23:PRINT USING"####
#";von;:PRINT" BIS ";:GOSUB 6540           (2C0B)
1950 bis=VAL(e$):IF bis<von OR bis
>nummermax THEN LOCATE 52,23:PRINT
 SPC(5):GOTO 1940                          (4938)
1960 such=1:feldldr=0:FOR a=0 TO l
o-1:feldldr=feldldr+feldl(druck(a)
):NEXT                                     (5C68)
1970 IF e=2 THEN 2080                      (10AD)
1980 PRINT#8,"PROGRAMM: MINI-DATEI
     DATEI: "datei$".DAT      AUSDR
UCK: "datum$"      NUMMERN:"von"-"b
is                                         (6EAE)
1990 PRINT#8,"--------------------
     -------"STRING$(LEN(datei$+".
DAT"),"-")"      ----------"STRING$
(LEN(datum$),"-")"      "STRING$(10
+LEN(STR$(von))+LEN(STR$(bis)),"-"
)                                          (9A61)
2000 PRINT#8                               (0814)
2010 IF lo=1 THEN b=0:GOTO 2060 EL
SE bi=lo-2                                 (2874)
2020 FOR b=0 TO bi                         (1273)
2030 IF feldl(b)<=LEN(k$(druck(b))
)THEN PRINT#8,LEFT$(k$(druck(b)),f
eldl(b))":";:GOTO 2050                     (62F3)
2040 PRINT#8,k$(druck(b))SPACE$(fe
ldl(b)-LEN(k$(druck(b))))":";              (4CA7)
2050 NEXT                                  (06CE)
2060 IF feldl(b)<=LEN(d$(druck(b+1
)))THEN PRINT#8,LEFT$(k$(druck(b))
,feldl(b))ELSE PRINT#8,k$(druck(b)
)                                          (7576)
2070 PRINT#8,STRING$(feldldr+4,"="
)                                          (1D7C)
2080 FOR a=von TO bis                      (182B)
2090 IF INKEY(66)=0 THEN GOSUB 703
0:IF jn=1 THEN RETURN                      (1E60)
2100 nu=(a-1)*(feldl+2):/RECREAD,@
nu,@recr$                                  (3AA8)
2110 GOSUB 6760                            (09FD)
2120 IF lo=1 THEN d=0:GOTO 2160            (1946)
2130 FOR d=0 TO bi                         (1259)
2140 PRINT#8,d$(druck(d))SPACE$(fe
ldl(druck(d))-LEN(d$(druck(d))))":
";                                         (56F8)
2150 NEXT                                  (0697)
2160 PRINT#8,d$(druck(d))SPACE$(fe
ldl(druck(d))-LEN(d$(druck(d))))           (524C)
2170 PRINT#8,STRING$(feldldr+4,"-"
)                                          (1D46)
2180 IF INKEY(66)=0 THEN GOSUB 703
0:IF jn=0 THEN CALL &A693,4,&953A:
RETURN                                     (2940)
2190 NEXT                                  (06E7)
2200 such=0                                (0EDC)
2210 RETURN                                (0672)
2220 LOCATE 30,y2:laenge=feldl(z2)
:mini=32:maxi=126:GOSUB 6540               (43BF)
2230 IF e$=""THEN LOCATE 30,y2:PRI
NT d$(z2):RETURN                           (2672)
2240 d$(z2)=e$+SPACE$(feldl(z2)-LE
N(e$))                                     (3237)
2250 LOCATE 30,y2:PRINT d$(z2)             (1B77)
2260 dat$="":FOR a=0 TO felda-1:da
t$=dat$+d$(a):NEXT                         (3C19)
2270 nu=(nummer-1)*(feldl+2)               (251B)
2280 /RECWRITE,@nu,@dat$                   (1E17)
2290 RETURN                                (0613)
2300 FOR a=0 TO felda-1:LOCATE 30,
13+a:PRINT SPACE$(feldl(a)):d$(a)=
"":NEXT                                    (4759)
2310 dat$=SPACE$(feldl)                    (18DE)
2320 nu=(nummer-1)*(feldl+2)               (2580)
2330 /RECWRITE,@nu,@dat$                   (1E7C)
2340 RETURN                                (0678)
```

```
2350 WINDOW#3,4,20,7,8+felda:CALL
&A690,3,&8F1F:CLS#3                    <294E>
2360 MOVE 30,298:DRAW 154,298:DRAW
 154,280-(felda*16):DRAW 30,280-(f
elda*16):DRAW 30,298                   <48F2>
2370 FOR a=0 TO felda-1:LOCATE 6,8
+a:PRINT k$(a):NEXT                    <2F99>
2380 IF z3>felda THEN z3=felda:y3=
8+felda                                <34BB>
2390 LOCATE 6,y3:PRINT in$k$(z3)in
$CHR$(8)" "                            <2C87>
2400 FOR w=1 TO 10:NEXT                <1158>
2410 IF INKEY(2)=0 AND y3<7+felda
THEN y3=y3+1:z3=z3+1:LOCATE 6,y3-1
:PRINT k$(z3-1):GOTO 2390              <5975>
2420 IF INKEY(0)=0 AND y3>8 THEN y
3=y3-1:z3=z3-1:LOCATE 6,y3+1:PRINT
 k$(z3+1):GOTO 2390                    <5095>
2430 IF INKEY(9)=0 THEN 2460           <11AB>
2440 IF INKEY(66)=0 THEN CALL &A69
3,3,&8F1F:i=66:GOSUB 7330:RETURN       <2826>
2450 GOTO 2400                         <0941>
2460 WINDOW#4,23,55,18,20:CALL &A6
90,4,&957F:CLS#4                       <23A6>
2470 MOVE 182,118:DRAW 434,118:DRA
W 434,90:DRAW 182,90:DRAW 182,118      <29D7>
2480 LOCATE 25,19:PRINT"SORTIEREN
VON ";:laenge=5:mini=48:maxi=57:GO
SUB 6540                               <4596>
2490 von=VAL(e$):IF von>nummermax
OR von=0 THEN LOCATE 39,19:PRINT S
PC(5):GOTO 2480                        <44AB>
2500 LOCATE 39,19:PRINT USING"####
#";von;:PRINT" BIS ";:GOSUB 6540       <2C7A>
2510 bis=VAL(e$):IF bis<von OR bis
>nummermax THEN LOCATE 49,19:PRINT
 SPC(5):GOTO 2500                      <49A6>
2520 LOCATE 49,19:PRINT USING"####
#";bis                                 <1C52>
2530 mi=1:FOR a=0 TO z3-1:mi=mi+fe
ldl(a):NEXT                            <3946>
2540 GOSUB 6940                        <0903>
2550 FOR a=von TO bis                  <18D7>
2560 nu1=(a-1)*(feldl+2):dat1$=SPA
CE$(feldl):/RECREAD,@nu1,@dat1$        <513B>
2570 FOR b=a+1 TO bis                  <18B1>
2580 IF INKEY(66)=0 THEN GOSUB 712
0:IF jn=0 THEN 2660                    <20F4>
2590 nu2=(b-1)*(feldl+2):dat2$=SPA
CE$(feldl):/RECREAD,@nu2,@dat2$        <51B5>
2600 IF UPPER$(MID$(dat1$,mi,feldl
(z3)))<UPPER$(MID$(dat2$,mi,feldl(
z3)))THEN 2630                         <5364>
2610 /RECWRITE,@nu2,@dat1$             <2085>
2620 dat1$=dat2$                       <1446>
2630 NEXT b                            <0AEB>
2640 /RECWRITE,@nu1,@dat1$             <20A9>
2650 NEXT a                            <0A0E>
2660 CLS#5                             <08E0>
2670 CALL &A693,4,&957F                <0F6E>
2680 CALL &A693,3,&8F1F                <0F29>
2690 IF nummer>bis THEN 2710           <1ABA>
2700 GOSUB 3190                        <09D2>
2710 RETURN                            <065C>
2720 WINDOW#3,22,66,14,15+felda:CA
LL &A690,3,&8F1F:CLS#3                 <2C00>
2730 MOVE 174,184:DRAW 524,184:DRA
W 524,168-(felda*16):DRAW 174,168-
(felda*16):DRAW 174,184                <45B4>
2740 FOR a=0 TO felda-1:LOCATE 24,
15+a:PRINT k$(a)": "s$(a):NEXT         <3F6D>
2750 IF z14>felda THEN z14=felda:y
14=14+felda                            <3812>
2760 LOCATE 24,y14:PRINT in$k$(z14
)":"in$                                <2A3D>
2770 FOR w=1 TO 10:NEXT                <113C>
2780 IF INKEY(2)=0 AND y14<14+feld
a THEN y14=y14+1:z14=z14+1:LOCATE
24,y14-1:PRINT k$(z14-1)":":GOTO 2
760                                    <65DE>
2790 IF INKEY(0)=0 AND y14>15 THEN
 y14=y14-1:z14=z14-1:LOCATE 24,y14
+1:PRINT k$(z14+1)":":GOTO 2760        <5C5C>
2800 IF INKEY(9)=0 THEN GOSUB 2840
                                       <122E>
2810 IF INKEY(66)=0 THEN CALL &A69
3,3,&8F1F:i=66:GOSUB 7330:RETURN       <280A>
2820 IF INKEY(60)=0 THEN 2870          <125B>
2830 GOTO 2770                         <09D3>
2840 LOCATE 40,y14:laenge=feldl(z1
4):mini=32:maxi=126:GOSUB 6540         <453A>
2850 LOCATE 40,y14:s$(z14)=e$:PRIN
T s$(z14)+SPACE$(feldl(z14)-LEN(s$
(z14)))                                <5518>
2860 RETURN                            <068A>
2870 CALL &A693,3,&8F1F                <0FA5>
2880 GOSUB 6980                        <09EB>
2890 FOR b=1 TO nummermax              <19E6>
2900 nu=(b-1)*(feldl+2)                <20BF>
2910 /RECREAD,@nu,@recr$:such=1:GO
SUB 6760                               <2D57>
2920 ok=0                              <0C33>
2930 FOR c=0 TO felda-1                <17DF>
2940 IF INKEY(66)=0 THEN GOSUB 702
0:IF jn=0 THEN 3000 ELSE GOSUB 698
0                                      <263F>
2950 IF s$(c)=""THEN ok=ok+1:GOTO
2970                                   <26DC>
2960 IF INSTR(UPPER$(d$(c)),UPPER$
(s$(c)))<>0 THEN ok=ok+1               <378E>
2970 NEXT                              <0602>
2980 IF ok=felda THEN 3010             <18EB>
2990 NEXT                              <0629>
3000 CALL &A693,5,&8F1F:such=0:GOS
UB 3190:RETURN                         <2059>
3010 WINDOW#4,34,60,6,10:CALL &A69
```

```
3,5,&8F1F                                <1E16>
3020 MOVE 270,312:DRAW 474,312:DRA
W 474,248:DRAW 270,248:DRAW 270,31
2                                        <2F88>
3030 LOCATE 38,7:PRINT"DATENSATZ G
EFUNDEN!"                                <2189>
3040 LOCATE 36,9:PRINT"WEITER /MEN
]   /DRUCKEN"                            <25A3>
3050 LOCATE 22,11:PRINT USING"####
#";b                                     <1AAB>
3060 GOSUB 6790                          <095D>
3070 LOCATE x15,9:PRINT in$sum$(z1
5)in$                                    <28E6>
3080 FOR w=1 TO 10:NEXT w                <1582>
3090 IF INKEY(1)=0 AND z15<2 THEN
x15=x15+8:z15=z15+1:LOCATE x15-8,9
:PRINT sum$(z15-1):GOTO 3070             <592A>
3100 IF INKEY(8)=0 AND z15>0 THEN
x15=x15-8:z15=z15-1:LOCATE x15+8,9
:PRINT sum$(z15+1):GOTO 3070             <593C>
3110 IF INKEY(9)=0 THEN ON z15+1 G
OTO 3130,3140,3150                       <23D3>
3120 GOTO 3080                           <09D2>
3130 CLS#4:GOSUB 6980:GOTO 2990          <12E3>
3140 CLS#4:such=0:nummer=b:i=9:GOS
UB 7330:RETURN                           <2F35>
3150 IF druckon=0 THEN GOSUB 7340:
GOTO 3090                                <1CB8>
3160 bi=lo-2                             <1205>
3170 von=b:bis=von:e=2:GOSUB 1960:
GOTO 3080                                <2F34>
3180 IF nummer<nummermax THEN numm
er=nummer+1 ELSE RETURN                  <35D9>
3190 LOCATE 22,11:PRINT USING"####
#";nummer                                <1F7D>
3200 nu=(nummer-1)*(feldl+2)             <2562>
3210 /RECREAD,@nu,@recr$                 <1E92>
3220 GOSUB 6760                          <09AC>
3230 RETURN                              <066E>
3240 IF nummer>1 THEN nummer=numme
r-1 ELSE RETURN                          <2AC2>
3250 LOCATE 22,11:PRINT USING"####
#";nummer                                <1FF5>
3260 nu=(nummer-1)*(feldl+2)             <25DA>
3270 /RECREAD,@nu,@recr$                 <1E0A>
3280 GOSUB 6760                          <0925>
3290 RETURN                              <06E6>
3300 LOCATE 14,11:PRINT in$"NUMMER
:"in$"     "                             <263F>
3310 LOCATE 22,11:laenge=5:mini=48
:maxi=57:GOSUB 6540                      <3223>
3320 nummer=VAL(e$)                      <176B>
3330 IF nummer>nummermax OR nummer
=0 THEN 3300                             <2C71>
3340 LOCATE 14,11:PRINT"NUMMER: "U
SING"#####";nummer                       <293E>
3350 nu=(nummer-1)*(feldl+2)             <2590>
3360 /RECREAD,@nu,@recr$                 <1EC0>
3370 GOSUB 6760                          <09DA>
3380 RETURN                              <069C>
3390 WINDOW#1,3,33,6,24:CALL &A690
,1,&8D87:CLS#1                           <214C>
3400 WINDOW#3,58,73,8,21                 <1319>
3410 MOVE 20,296:DRAW 260,296:DRAW
 260,24:DRAW 20,24:DRAW 20,296           <2C5C>
3420 MOVE 20,264:DRAW 260,264            <156D>
3430 LOCATE 5,8:PRINT"USER: "USING
"##";user;:PRINT"        LAUFWERK:
 "CHR$(65+drive)                         <46DE>
3440 WINDOW 5,32,10,23:CLS:/DIR:WI
NDOW 1,80,1,25                           <23B0>
3450 MOVE 462,280:DRAW 580,280:DRA
W 580,72:DRAW 462,72:DRAW 462,280        <2FD6>
3460 MOVE 462,152:DRAW 580,152           <1410>
3470 FOR a=0 TO 6:LOCATE 60,9+a:PR
INT disc$(a):NEXT                        <2AB2>
3480 LOCATE 60,y5:PRINT in$disc$(z
5)in$                                    <281A>
3490 FOR w=1 TO 10:NEXT                  <11E0>
3500 IF INKEY(2)=0 AND y5<15 THEN
y5=y5+1:z5=z5+1:LOCATE 60,y5-1:PRI
NT disc$(z5-1):GOTO 3480                 <5539>
3510 IF INKEY(0)=0 AND y5>9 THEN y
5=y5-1:z5=z5-1:LOCATE 60,y5+1:PRIN
T disc$(z5+1):GOTO 3480                  <541A>
3520 IF INKEY(9)=0 THEN 3550             <1156>
3530 IF INKEY(66)=0 THEN CLS#3:CAL
L &A693,1,&8D87:GOTO 1390                <222E>
3540 GOTO 3490                           <09EB>
3550 ON z5+1 GOTO 3560,3630,3730,3
780,3830,3850,3890                       <29E8>
3560 WINDOW 60,72,18,19                  <110A>
3570 LOCATE 1,1:PRINT"NAME:"             <12DB>
3580 LOCATE 1,2:mini=32:maxi=122:l
aenge=12:GOSUB 6540                      <319F>
3590 IF e$=""OR INSTR(e$,".")>9 TH
EN 3610                                  <20E9>
3600 /ERA,@e$                            <1086>
3610 CLS:WINDOW 1,80,1,25                <11D4>
3620 GOTO 3830                           <0932>
3630 WINDOW 60,72,17,20                  <117A>
3640 LOCATE 1,1:PRINT"ALTER NAME:"
                                         <186E>
3650 LOCATE 1,2:mini=32:maxi=122:l
aenge=12:GOSUB 6540:file$=e$             <3E7B>
3660 IF e$=""OR INSTR(e$,".")>9 TH
EN 3710                                  <2091>
3670 LOCATE 1,3:PRINT"NEUER NAME:"
                                         <1867>
3680 LOCATE 1,4:GOSUB 6540               <0E7F>
3690 IF e$=""OR INSTR(e$,".")>9 TH
EN 3710                                  <20CD>
3700 /REN,@e$,@file$                     <19EB>
3710 CLS:WINDOW 1,80,1,25                <119D>
3720 GOTO 3830                           <09FA>
3730 WINDOW 60,72,18,19                  <115F>
```

```
3740 LOCATE 1,1:PRINT"NAME:"          <1230>
3750 LOCATE 1,2:mini=32:maxi=122:l
aenge=12:GOSUB 6540                   <31F5>
3760 IF e$=""OR INSTR(e$,".")>9 TH
EN CLS:WINDOW 1,80,1,25:GOTO 3830     <2E86>
3770 RUN e$                           <0A28>
3780 WINDOW 60,72,18,19               <11C3>
3790 LOCATE 1,1:PRINT"NAME:"          <1294>
3800 LOCATE 1,2:mini=32:maxi=122:l
aenge=12:GOSUB 6540                   <3158>
3810 IF e$=""OR INSTR(e$,".")>9 TH
EN CLS:WINDOW 1,80,1,25:GOTO 3830     <2EEA>
3820 LOAD e$                          <0A05>
3830 WINDOW 5,32,10,23:CLS:/DIR:WI
NDOW 1,80,1,25                        <23BC>
3840 GOTO 3490                        <0946>
3850 user=user+1:IF user>15 THEN u
ser=0                                 <2C6A>
3860 LOCATE 11,8:PRINT USING"##";u
ser                                   <1927>
3870 /USER,user                       <13DC>
3880 GOTO 3490                        <0996>
3890 drive=drive+1:IF drive>1 THEN
 drive=0                              <2FD1>
3900 LOCATE 31,8:PRINT CHR$(65+dri
ve)                                   <1B6F>
3910 drive$=CHR$(65+drive)            <1D68>
3920 WINDOW 60,72,18,19               <11DB>
3930 /DRIVE,@drive$                   <16D8>
3940 CLS:WINDOW 1,80,1,25             <116A>
3950 GOTO 3830                        <09C7>
3960 WINDOW#1,48,66,8,20:CALL &A69
0,1,&8D87:CLS#1                       <2284>
3970 MOVE 382,278:DRAW 520,278:DRA
W 520,88:DRAW 382,88:DRAW 382,278     <2F04>
3980 FOR a=0 TO 10:LOCATE 50,9+a:P
RINT so$(a):NEXT                      <294D>
3990 LOCATE 50,y6:PRINT in$so$(z6)
in$                                   <265E>
4000 FOR w=1 TO 10:NEXT               <11DE>
4010 IF INKEY(2)=0 AND y6<19 THEN
y6=y6+1:z6=z6+1:LOCATE 50,y6-1:PRI
NT so$(z6-1):GOTO 3990                <53E3>
4020 IF INKEY(0)=0 AND y6>9 THEN y
6=y6-1:z6=z6-1:LOCATE 50,y6+1:PRIN
T so$(z6+1):GOTO 3990                 <52D9>
4030 IF INKEY(9)=0 THEN i=9:GOSUB
7330:ON z6+1 GOTO 4060,4530,5030,5
180,5470,5680,5840,5990,6150,6330,
6450                                  <4E66>
4040 IF INKEY(66)=0 THEN CALL &A69
3,1,&8D87:GOTO 1390                   <1E3E>
4050 GOTO 4000                        <09E1>
4060 WINDOW#2,14,42,17,20:CALL &A6
90,2,&95D7:CLS#2                      <2318>
4070 MOVE 108,134:DRAW 328,134:DRA
W 328,88:DRAW 108,88:DRAW 108,134     <29AF>
4080 LOCATE 16,18:PRINT dl$(0)" "d
ateis".DAT"                           <2678>
4090 LOCATE 16,19:PRINT dl$(1)        <15B5>
4100 LOCATE 16,y12:PRINT in$dl$(z1
2)in$                                 <28F0>
4110 FOR w=1 TO 10:NEXT               <11BA>
4120 IF INKEY(2)=0 AND y12<19 THEN
 y12=y12+1:z12=z12+1:LOCATE 16,y12
-1:PRINT dl$(z12-1):GOTO 4100         <5AB7>
4130 IF INKEY(0)=0 AND y12>18 THEN
 y12=y12-1:z12=z12-1:LOCATE 16,y12
+1:PRINT dl$(z12+1):GOTO 4100         <5A17>
4140 IF INKEY(9)=0 THEN ON z12+1 G
OTO 4170,4220                         <1F40>
4150 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:GOTO 400
0                                     <2BA0>
4160 GOTO 4110                        <0937>
4170 LOCATE 29,y12:laenge=8:mini=4
8:maxi=122:GOSUB 6540                 <369D>
4180 IF e$=""THEN 4200                <1128>
4190 datei$=UPPER$(e$)+SPACE$(8-LE
N(e$))                                <253C>
4200 LOCATE 29,y12:PRINT datei$       <19FA>
4210 GOTO 4110                        <099B>
4220 IF datei$=SPACE$(8)THEN 4600     <1896>
4230 GOSUB 7210                       <09B2>
4240 LOCATE 29,y12:PRINT"OK."         <1659>
4250 CLOSEIN                          <0667>
4260 OPENIN datei$+".PRG"             <1544>
4270 FOR a=0 TO 9:INPUT#9,fu$(a):N
EXT                                   <2036>
4280 FOR a=0 TO 9:INPUT#9,k$(a):NE
XT                                    <1F71>
4290 FOR a=0 TO 4:INPUT#9,druck(a)
:NEXT                                 <23B4>
4300 FOR a=0 TO 9:INPUT#9,feldl(a)
:feldlsi(a)=feldl(a):NEXT             <434C>
4310 INPUT#9,papier                   <127F>
4320 INPUT#9,rand                     <1022>
4330 INPUT#9,stift                    <11C6>
4340 INPUT#9,datum$                   <11ED>
4350 INPUT#9,ta                       <0EF3>
4360 INPUT#9,mo                       <0E08>
4370 INPUT#9,jahr                     <108E>
4380 INPUT#9,nummermax                <1527>
4390 INPUT#9,feldl:fl=feldl           <2047>
4400 INPUT#9,felda:fa=felda           <20B4>
4410 CLOSEIN                          <06A7>
4420 OPENIN datei$+".DAT"             <1554>
4430 CALL &A693,2,&95D7               <0F73>
4440 open=1                           <0EC8>
4450 IF felda<5 THEN lo=felda         <1F3F>
4460 BORDER rand:INK 0,papier:INK
1,stift                               <269E>
4470 LOCATE 62,2:PRINT datum$         <14EA>
4480 FOR a=0 TO 9:KEY a,fu$(a):NEX
T                                     <22E4>
4490 FOR a=felda TO 9:k$(a)="":NEX
```

```
T                                          <2598>
4500 nummer=1                              <10DB>
4510 GOSUB 6700                            <09EB>
4520 GOTO 3960                             <094E>
4530 WINDOW#2,11,39,9,13:CALL &A69
0,2,&95D7:CLS#2                            <2220>
4540 nummermax=INT(178000/(feldl+2
))                                         <2920>
4550 MOVE 84,262:DRAW 304,262:DRAW
 304,200:DRAW 84,200:DRAW 84,262           <2CF8>
4560 LOCATE 13,10:PRINT da$(0)" "d
atei$".DAT"                                <26B8>
4570 LOCATE 13,11:PRINT da$(1)" "U
SING"#####";nummermax                      <2D0E>
4580 LOCATE 13,12:PRINT da$(2)             <1580>
4590 LOCATE 13,y11:PRINT in$da$(z1
1)in$                                      <2808>
4600 FOR w=1 TO 10:NEXT                    <118F>
4610 IF INKEY(2)=0 AND y11<12 THEN
 y11=y11+1:z11=z11+1:LOCATE 13,y11
-1:PRINT da$(z11-1):GOTO 4590              <5A77>
4620 IF INKEY(0)=0 AND y11>10 THEN
 y11=y11-1:z11=z11-1:LOCATE 13,y11
+1:PRINT da$(z11+1):GOTO 4590              <5A89>
4630 IF INKEY(9)=0 THEN ON z11+1 G
OTO 4660,4640,4710                         <23D5>
4640 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:GOTO 401
0                                          <2BCA>
4650 GOTO 4600                             <095E>
4660 LOCATE 26,y11:laenge=8:mini=4
8:maxi=122:GOSUB 6540                      <3678>
4670 IF e$=""THEN 4690                     <1157>
4680 datei$=UPPER$(e$)+SPACE$(8-LE
N(e$))                                     <2512>
4690 LOCATE 26,y11:PRINT datei$            <19C8>
4700 GOTO 4600                             <09C2>
4710 IF datei$=""THEN 4600                 <159E>
4720 GOSUB 7270                            <0962>
4730 LOCATE 26,y11:PRINT"OK."              <1632>
4740 CLOSEOUT                              <0641>
4750 OPENOUT datei$+".PRG"                 <1518>
4760 FOR a=0 TO 9:PRINT#9,fu$(a):N
EXT                                        <20EE>
4770 FOR a=0 TO felda-1:PRINT#9,k$
(a):NEXT:FOR a=felda TO 9:k$(a)=ST
RING$(14,32):PRINT#9,k$(a):NEXT            <5F8C>
4780 FOR a=0 TO 4:PRINT#9,druck(a)
:NEXT                                      <23D7>
4790 FOR a=0 TO 9:PRINT#9,feldl(a)
:feldlsi(a)=feldl(a):NEXT                  <43AB>
4800 PRINT#9,papier                        <1284>
4810 PRINT#9,rand                          <10C5>
4820 PRINT#9,stift                         <1124>
4830 PRINT#9,datum$                        <113B>
4840 PRINT#9,ta                            <0EB6>
4850 PRINT#9,mo                            <0ECA>
4860 PRINT#9,jahr                          <102F>
4870 PRINT#9,nummermax                     <157F>
4880 PRINT#9,feldl:fl=feldl                <2074>
4890 PRINT#9,felda:fa=felda                <2036>
4900 CLOSEOUT                              <0681>
4910 OPENOUT datei$+".DAT"                 <1528>
4920 FOR a=1 TO nummermax                  <1904>
4930 LOCATE 33,11:PRINT USING"####
#";a                                       <1A2F>
4940 PRINT#9,SPACE$(fl)                    <12FF>
4950 NEXT                                  <0682>
4960 PRINT#9,SPACE$(128)SPACE$(128
)                                          <155E>
4970 CLOSEOUT                              <060D>
4980 open=1                                <0E02>
4990 CLOSEIN                               <0632>
5000 OPENIN datei$+".DAT"                  <15DE>
5010 CALL &A693,2,&95D7                    <0FFD>
5020 GOTO 4010                             <09C9>
5030 WINDOW#2,2,18,13,24:CALL &A69
0,2,&95D7:CLS#2                            <2273>
5040 MOVE 8,198:DRAW 138,198:DRAW
138,26:DRAW 8,26:DRAW 8,198                <2454>
5050 FOR a=0 TO 9:LOCATE 3,14+a:PR
INT USING"F#: ";a;:PRINT fu$(a):NE
XT                                         <375B>
5060 LOCATE 3,y7:PRINT in$;:PRINT
USING"F#:";z7;:PRINT in$                   <2BCC>
5070 FOR w=1 TO 10:NEXT                    <113D>
5080 IF INKEY(2)=0 AND y7<23 THEN
y7=y7+1:z7=z7+1:LOCATE 3,y7-1:PRIN
T USING"F#:";z7-1:GOTO 5060                <5219>
5090 IF INKEY(0)=0 AND y7>14 THEN
y7=y7-1:z7=z7-1:LOCATE 3,y7+1:PRIN
T USING"F#:";z7+1:GOTO 5060                <5205>
5100 IF INKEY(9)=0 THEN GOSUB 5130
                                           <12EA>
5110 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:GOTO 400
0                                          <2B23>
5120 GOTO 5070                             <09C4>
5130 LOCATE 7,y7:laenge=10:mini=32
:maxi=126:GOSUB 6540                       <3589>
5140 IF e$=""THEN e$=fu$(z7)               <1F3A>
5150 fu$(z7)=e$:LOCATE 7,y7:PRINT
fu$(z7)+SPACE$(10-LEN(e$))                 <3DF6>
5160 KEY z7,fu$(z7)                        <188F>
5170 RETURN                                <069F>
5180 WINDOW#2,18,52,21,24:CALL &A6
90,2,&95D7:CLS#2                           <23E6>
5190 MOVE 142,72:DRAW 408,72:DRAW
408,24:DRAW 142,24:DRAW 142,72             <29BB>
5200 LOCATE 20,22:PRINT"ALTES DATU
M: "datum$                                 <2424>
5210 LOCATE 20,23:PRINT"NEUES DATU
M:"                                        <1B04>
5220 LOCATE 33,23:PRINT USING"##";
ta                                         <1852>
5230 FOR w=1 TO 30:NEXT                    <11C2>
```

```
5240 IF INKEY(1)=0 AND ta<31 THEN
ta=ta+1:GOTO 5220                    <29BF>
5250 IF INKEY(8)=0 AND ta>1 THEN t
a=ta-1:GOTO 5220                     <28A6>
5260 IF INKEY(9)=0 THEN 5280         <111F>
5270 GOTO 5230                       <09F0>
5280 LOCATE 33,23:PRINT STRING$(3-
LEN(STR$(ta)),"0");MID$(STR$(ta),2
,2);"."                              <39DD>
5290 i=9:GOSUB 7330                  <10F9>
5300 LOCATE 37,23:PRINT mo$(mo)      <1923>
5310 FOR w=1 TO 30:NEXT              <1163>
5320 IF INKEY(1)=0 AND mo<11 THEN
mo=mo+1:GOTO 5300                    <295F>
5330 IF INKEY(8)=0 AND mo>0 THEN m
o=mo-1:GOTO 5300                     <2820>
5340 IF INKEY(9)=0 THEN i=9:GOSUB
7330:GOTO 5360                       <1E60>
5350 GOTO 5310                       <0911>
5360 LOCATE 47,23:PRINT USING"####
";jahr                               <1CEB>
5370 FOR w=1 TO 30:NEXT              <11DB>
5380 IF INKEY(1)=0 AND jahr<9999 T
HEN jahr=jahr+1:GOTO 5360            <30EB>
5390 IF INKEY(8)=0 AND jahr>1986 T
HEN jahr=jahr-1:GOTO 5360            <30C6>
5400 IF INKEY(9)=0 THEN 5420         <1199>
5410 GOTO 5370                       <096F>
5420 datum$=STRING$(3-LEN(STR$(ta)
),"0")+MID$(STR$(ta),2,2)+". "+mo$
(mo)+" "+MID$(STR$(jahr),2,4)        <5FA4>
5430 LOCATE 62,2:PRINT datum$        <146E>
5440 CALL &A693,2,&95D7              <0F5B>
5450 i=9:GOSUB 7330                  <103A>
5460 GOTO 4000                       <09EC>
5470 WINDOW#2,6,20,7,11:CALL &A690
,2,&95D7:CLS#2                       <21F7>
5480 MOVE 48,296:DRAW 146,296:DRAW
 146,230:DRAW 48,230:DRAW 48,296     <2A7C>
5490 LOCATE 8,8:PRINT fa$(0)USING"
 ##";papier                          <23AA>
5500 LOCATE 8,9:PRINT fa$(1)USING"
 ##";rand                            <21AD>
5510 LOCATE 8,10:PRINT fa$(2)USING
" ##";stift
                                     <2350>
"                                    <21EF>
5530 FOR w=1 TO 10:NEXT              <11D8>
5540 IF INKEY(2)=0 AND y8<10 THEN
y8=y8+1:z8=z8+1:LOCATE 8,y8-1:PRIN
T fa$(z8-1):GOTO 5520                <5201>
5550 IF INKEY(0)=0 AND y8>8 THEN y
8=y8-1:z8=z8-1:LOCATE 8,y8+1:PRINT
 fa$(z8+1):GOTO 5520                 <51F1>
5560 IF INKEY(1)=0 THEN GOSUB 5600
                                     <1217>
5570 IF INKEY(8)=0 THEN GOSUB 5640
                                     <128C>
5580 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:GOTO 400
0                                    <2BD2>
5590 GOTO 5530                       <09D7>
5600 IF z8=0 AND papier<26 THEN pa
pier=papier+1:INK 0,papier:LOCATE
15,8:PRINT USING" ##";papier         <55C3>
5610 IF z8=1 AND rand<26 THEN rand
=rand+1:BORDER rand:LOCATE 15,9:PR
INT USING" ##";rand                  <493B>
5620 IF z8=2 AND stift<26 THEN sti
ft=stift+1:INK 1,stift:LOCATE 15,1
0:PRINT USING" ##";stift             <51F8>
5630 RETURN                          <0638>
5640 IF z8=0 AND papier>0 THEN pap
ier=papier-1:INK 0,papier:LOCATE 1
5,8:PRINT USING" ##";papier          <54CD>
5650 IF z8=1 AND rand>0 THEN rand=
rand-1:BORDER rand:LOCATE 15,9:PRI
NT USING" ##";rand                   <4861>
5660 IF z8=2 AND stift>0 THEN stif
t=stift-1:INK 1,stift:LOCATE 15,10
:PRINT USING" ##";stift              <506F>
5670 RETURN                          <0689>
5680 WINDOW#2,22,42,14,14+felda+1:
CALL &A690,2,&95D7:CLS#2             <2E40>
5690 MOVE 176,184:DRAW 328,184:DRA
W 328,168-(felda*16):DRAW 176,168-
(felda*16):DRAW 176,184              <450D>
5700 FOR a=0 TO felda-1:LOCATE 24,
15+a:PRINT USING"##";a+1;:PRINT".
"k$(a):NEXT                          <4418>
5710 IF z9>felda-1 THEN z9=felda:y
9=14+felda                           <37A5>
5720 LOCATE 24,y9:PRINT in$USING"#
#";z9;:PRINT"."in$                   <2B5C>
5730 FOR w=1 TO 10:NEXT              <1169>
5740 IF INKEY(2)=0 AND y9<14+felda
 THEN y9=y9+1:z9=z9+1:LOCATE 24,y9
-1:PRINT USING"##";z9-1;:PRINT".":
GOTO 5720                            <613C>
5750 IF INKEY(0)=0 AND y9>15 THEN
y9=y9-1:z9=z9-1:LOCATE 24,y9+1:PRI
NT USING"##";z9+1;:PRINT".":GOTO 5
720                                  <58EB>
5760 IF INKEY(9)=0 THEN GOSUB 5790
                                     <12BB>
5770 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:IF ae=1
THEN FOR a=0 TO felda-1:LOCATE 14,
13+a:PRINT k$(a)": ":NEXT:GOTO 400
0 ELSE 4000                          <6A98>
5780 GOTO 5730                       <0996>
5790 LOCATE 28,y9:laenge=13:mini=3
2:maxi=126:GOSUB 6540                <3647>
5800 IF e$=""THEN LOCATE 28,y9:PRI
NT k$(z9-1):RETURN                   <28E9>
5810 ae=1                            <0CB4>
5820 k$(z9-1)=e$+SPACE$(14-LEN(e$)
```

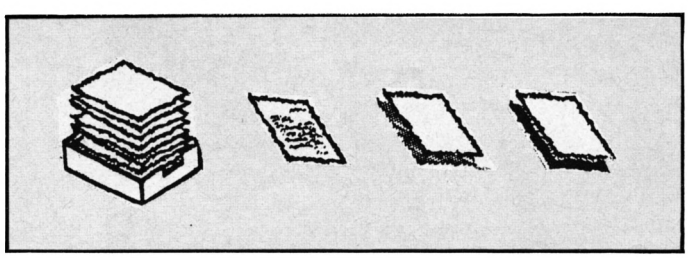
```
):LOCATE 28,y9:PRINT k$(z9-1)      <40C6>
5830 RETURN                        <06C8>
5840 IF felda<5 THEN lo=felda ELSE
 lo=5                              <28EA>
5850 IF z10>felda THEN z10=felda:y
10=7+felda                         <37E9>
5860 WINDOW#2,10,31,7,8+lo:CALL &A
690,2,&95D7:CLS#2                  <2787>
5870 MOVE 78,294:DRAW 236,294:DRAW
 236,282-(lo*16):DRAW 78,282-(lo*1
6):DRAW 78,294                     <4274>
5880 FOR a=0 TO lo-1:LOCATE 12,8+a
:PRINT USING"##";a+1;:PRINT". "k$(
druck(a)):NEXT                     <4ABD>
5890 LOCATE 12,y10:PRINT in$USING"
##";z10;:PRINT"."in$               <2D84>
5900 FOR w=1 TO 10:NEXT            <11BD>
5910 IF INKEY(2)=0 AND y10<7+lo TH
EN y10=y10+1:z10=z10+1:LOCATE 12,y
10-1:PRINT USING"##";z10-1;:PRINT"
.":GOTO 5890                       <6448>
5920 IF INKEY(0)=0 AND y10>8 THEN
y10=y10-1:z10=z10-1:LOCATE 12,y10+
1:PRINT USING"##";z10+1;:PRINT".":
GOTO 5890                          <5EA3>
5930 IF INKEY(1)=0 AND druck(y10-8
)<felda-1 THEN druck(y10-8)=druck(
y10-8)+1:GOTO 5970                 <5854>
5940 IF INKEY(8)=0 AND druck(y10-8
)>0 THEN druck(y10-8)=druck(y10-8)
-1:GOTO 5970                       <4F6B>
5950 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:GOTO 400
0                                  <2BB7>
5960 GOTO 5900                     <095A>
5970 LOCATE 16,y10:PRINT k$(druck(
y10-8))                            <2930>
5980 GOTO 5900                     <0982>
5990 WINDOW#2,30,42,11,14:CALL &A6
90,2,&95D7:CLS#2                   <23E2>
6000 MOVE 238,230:DRAW 320,230:DRA
W 320,184:DRAW 238,184:DRAW 238,23
0                                  <2971>
6010 LOCATE 32,12:PRINT drucker$(0
):LOCATE 32,13:PRINT drucker$(1)   <300B>
6020 LOCATE 32,y13:PRINT in$drucke
r$(z13)in$                         <2D0A>
6030 FOR w=1 TO 10:NEXT            <11C2>
6040 IF INKEY(2)=0 AND y13<13 THEN
 y13=y13+1:z13=z13+1:LOCATE 32,y13
-1:PRINT drucker$(z13-1):GOTO 6020
                                   <5FDF>
6050 IF INKEY(0)=0 AND y13>12 THEN
 y13=y13-1:z13=z13-1:LOCATE 32,y13
+1:PRINT drucker$(z13+1):GOTO 6020
                                   <5F74>
6060 IF INKEY(9)=0 THEN ON z13+1 G
OTO 6090,6090                      <1F68>
6070 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:GOTO 400
0                                  <2BA8>
6080 GOTO 6030                     <0955>
6090 WIDTH 142                     <0827>
6100 IF INP(&F500)=90 THEN GOSUB 6
840:IF dr=1 THEN GOTO 6030         <236D>
6110 PRINT#8,CHR$(15)              <0FA9>
6120 PRINT#8,CHR$(27)CHR$(120)CHR$
(0)CHR$(27)CHR$(83)CHR$(0)         <2B2A>
6130 PRINT#8,CHR$(27)CHR$(72)CHR$(
27)CHR$(51)CHR$(16)                <27A7>
6140 CALL &A693,2,&95D7::druckon=1
:GOTO 4000                         <22BF>
6150 WINDOW#2,8,29,7,8+felda:CALL
&A690,2,&95D7:CLS#2                <29EA>
6160 MOVE 60,296:DRAW 228,296:DRAW
 228,280-felda*16:DRAW 60,280-feld
a*16:DRAW 60,296                   <4436>
6170 FOR a=0 TO felda-1:LOCATE 10,
8+a:PRINT k$(a)": "USING"##";feldl
(a):NEXT                           <489C>
6180 WINDOW#3,12,25,22,23:CALL &A6
90,3,&9E17                         <1FDF>
6190 WINDOW#4,1,25,22,25:CLS#4     <1769>
6200 MOVE 8,56:DRAW 192,56:DRAW 19
2,8:DRAW 8,8:DRAW 8,56             <22C3>
6210 LOCATE 3,23:PRINT"MAXIMUM  :
25 ZEICHEN"                        <237A>
6220 LOCATE 3,24:PRINT"MINIMUM  :
 1 ZEICHEN"                        <23D2>
6230 IF z16>felda-1 THEN z16=felda
-1:y16=7+felda                     <3BB9>
6240 LOCATE 26,y16:PRINT USING"##"
;feldl(z16)                        <2752>
6250 LOCATE 10,y16:PRINT in$k$(z16
)":"in$                            <2A9B>
6260 FOR w=1 TO 10:NEXT            <118F>
6270 IF INKEY(2)=0 AND y16<7+felda
 THEN y16=y16+1:z16=z16+1:LOCATE 1
0,y16-1:PRINT k$(z16-1)":":GOTO 62
50                                 <64EC>
6280 IF INKEY(0)=0 AND y16>8 THEN
y16=y16-1:z16=z16-1:LOCATE 10,y16+
1:PRINT k$(z16+1)":":GOTO 6250     <5B5F>
6290 IF INKEY(1)=0 AND feldl(z16)<
25 THEN feldl(z16)=feldl(z16)+1:GO
TO 6240                            <4AC1>
6300 IF INKEY(8)=0 AND feldl(z16)>
1 THEN feldl(z16)=feldl(z16)-1:GOT
O 6240                             <495E>
6310 IF INKEY(66)=0 THEN feldl=0:F
OR a=0 TO felda-1:feldl=feldl+feld
l(a):NEXT:CALL &A693,2,&95D7:CLS#4
:CALL &A693,3,&9E17:i=66:GOSUB 733
0:GOTO 4010                        <7B69>
6320 GOTO 6260                     <0969>
6330 WINDOW#2,9,34,9,13:CALL &A690
```

```
,2,&95D7:CLS#2                          <21AA>
6340 MOVE 70,264:DRAW 258,264:DRAW
 258,198:DRAW 70,198:DRAW 70,264        <2C5D>
6350 fda=felda                          <140D>
6360 LOCATE 11,10:PRINT"MAXIMUM
: 10 FELDER"                            <24C7>
6370 LOCATE 11,11:PRINT"MINIMUM
:  1 FELD"                              <220B>
6380 LOCATE 11,12:PRINT in$"FELDAN
ZAHL:"in$"    FELDER"                   <3004>
6390 LOCATE 23,12:PRINT USING"##";
felda                                   <1BC5>
6400 FOR w=1 TO 30:NEXT                 <11EC>
6410 IF INKEY(1)=0 AND felda<10 TH
EN felda=felda+1:GOTO 6390              <320A>
6420 IF INKEY(8)=0 AND felda>1 THE
N felda=felda-1:GOTO 6390               <3136>
6430 IF INKEY(66)=0 THEN feldl=0:F
OR a=0 TO felda-1:feldl=feldl+feld
l(a):NEXT:CALL &A693,2,&95D7:i=66:
GOSUB 7330:IF felda<>fda THEN GOSU
B 6700:GOTO 3960 ELSE 4010              <870C>
6440 GOTO 6400                          <09C3>
6450 WINDOW#2,8,32,14,16:CALL &A69
0,2,&95D7:CLS#2                         <2297>
6460 MOVE 62,182:DRAW 250,182:DRAW
 250,152:DRAW 62,152:DRAW 62,182        <274F>
6470 LOCATE 10,15:PRINT in$"WARMST
ART"in$" / KALTSTART":jn=0              <388E>
6480 FOR w=1 TO 10:NEXT w               <1520>
6490 IF INKEY(8)=0 THEN LOCATE 10,
15:PRINT in$"WARMSTART"in$" / KALT
START":jn=0                             <41AF>
6500 IF INKEY(1)=0 THEN LOCATE 10,
15:PRINT"WARMSTART / "in$"KALTSTAR
T"in$:jn=1                              <4174>
6510 IF INKEY(66)=0 THEN CALL &A69
3,2,&95D7:i=66:GOSUB 7330:GOTO 400
0                                       <2B1A>
6520 IF INKEY(9)=0 THEN IF jn=1 TH
EN CALL 0 ELSE i=9:GOSUB 7330:GOTO
 1310                                   <2BDF>
6530 GOTO 6480                          <09F2>
6540 e$="":l=0:CALL &BB81               <1816>
6550 WHILE INKEY$<>"":WEND              <0DC2>
6560 IF PEEK(6)=&80 THEN POKE &B4E
8,0 ELSE POKE &B632,0                   <1EDB>
6570 a$=INKEY$:IF a$=""THEN 6570        <19B4>
6580 IF a$=CHR$(13)THEN CALL &BB84
:RETURN                                 <1887>
6590 IF a$=CHR$(32)THEN 6650            <1521>
6600 IF a$<>CHR$(127)THEN 6630          <15F3>
6610 IF LEN(e$)>=1 THEN PRINT CHR$
(8)CHR$(16);                            <1E9C>
6620 IF LEN(e$)>=1 THEN e$=LEFT$(e
$,LEN(e$)-1):l=l-1                      <3540>
6630 IF a$<CHR$(mini)THEN 6570          <1AE6>
6640 IF a$>CHR$(maxi)THEN 6570          <1AB4>
6650 l=l+1:IF l>laenge THEN l=laen
ge:GOTO 6570                            <3430>
6660 e$=e$+a$                           <13D8>
6670 PRINT a$;                          <0BCE>
6680 IF l=laenge THEN 6570              <186E>
6690 GOTO 6570                          <0905>
6700 LOCATE 1,7:PRINT CHR$(20)          <1106>
6710 MOVE 94,216:DRAW 94,248:DRAW
216,248:DRAW 216,216                    <20D9>
6720 MOVE 94,216:DRAW 444,216:DRAW
 444,200-(felda*16):DRAW 94,200-(f
elda*16):DRAW 94,216                    <4504>
6730 LOCATE 14,11:PRINT"NUMMER: "U
SING"#####";nummer                      <29C7>
6740 FOR a=0 TO felda-1:LOCATE 14,
13+a:PRINT k$(a)": ":NEXT               <3527>
6750 RETURN                             <06FD>
6760 mitte=0                            <0F9C>
6770 FOR c=0 TO felda-1:d$(c)=MID$
(recr$,1+mitte,feldl(c)):mitte=mit
te+feldl(c):NEXT                        <6A5D>
6780 IF such=1 THEN RETURN              <11E7>
6790 WINDOW 30,54,13,12+felda:CLS:
WINDOW 1,80,1,25                        <27B8>
6800 FOR a=0 TO felda-1                 <172A>
6810 LOCATE 30,13+a:PRINT d$(a)         <1CD9>
6820 NEXT                               <0624>
6830 RETURN                             <069C>
6840 WINDOW#3,6,42,22,24:CALL &A69
0,3,&9C87:CLS#3                         <226B>
6850 MOVE 48,56:DRAW 328,56:DRAW 3
28,24:DRAW 48,24:DRAW 48,56             <29B9>
6860 LOCATE 8,23:PRINT"DRUCKER IST
 NICHT EMPFANGSBEREIT!"                 <2F8F>
6870 FOR a=1 TO 600:IF INP(&F500)=
26 THEN CALL &A693,3,&9C87:RETURN
ELSE NEXT                               <2C3F>
6880 CALL &A693,3,&9C87:dr=1:RETUR
N                                       <19F4>
6890 WINDOW#1,2,19,5,7                  <116E>
6900 MOVE 16,328:DRAW 128,328:DRAW
 128,296:DRAW 16,296:DRAW 16,328        <2CC3>
6910 LOCATE 4,6:PRINT"DATEI LADEN!
"                                       <19F2>
6920 FOR w=1 TO 1500:NEXT               <124C>
6930 CLS#1:RETURN                       <0A62>
6940 WINDOW#5,61,76,22,24               <14B5>
6950 MOVE 486,54:DRAW 606,54:DRAW
606,24:DRAW 486,24:DRAW 486,54          <2CCA>
6960 LOCATE 63,23:PRINT"BITTE WART
EN!"                                    <1C9E>
6970 RETURN                             <06B6>
6980 WINDOW#5,5,21,20,22:CALL &A69
0,5,&8F1F:CLS#5                         <22BF>
6990 MOVE 38,88:DRAW 162,88:DRAW 1
62,56:DRAW 38,56:DRAW 38,88             <27FF>
7000 LOCATE 7,21:PRINT"BITTE WARTE
```

```
N!"                                          <1BC1>
7010 RETURN                                  <0606>
7020 CALL &A693,5,&8F1F                      <0FA6>
7030 WINDOW#6,49,63,19,23:CALL &A6
90,6,&90B7:CLS#6                             <238B>
7040 MOVE 392,104:DRAW 496,104:DRA
W 496,38:DRAW 392,38:DRAW 392,104 <2CCC>
7050 LOCATE 51,20:PRINT"UNTERBRUCH
!"                                           <1A9A>
7060 LOCATE 51,22:PRINT"JA   / "in
$"NEIN"in$:jn=1                              <2E8D>
7070 FOR w=1 TO 10:NEXT                      <11E6>
7080 IF INKEY(8)=0 THEN LOCATE 51,
22:PRINT in$"JA  "in$" / NEIN":jn=
0                                            <37CB>
7090 IF INKEY(1)=0 THEN LOCATE 51,
22:PRINT"JA   / "in$"NEIN"in$:jn=1
                                             <3795>
7100 IF INKEY(9)=0 THEN CALL &A693
,6,&90B7:i=9:GOSUB 7330:RETURN               <26CE>
7110 GOTO 7070                               <09F5>
7120 WINDOW#6,31,45,9,13:CLS#5:CAL
L &A690,6,&9867:CLS#6                        <26F2>
7130 MOVE 248,264:DRAW 352,264:DRA
W 352,198:DRAW 248,198:DRAW 248,26
4                                            <2C1F>
7140 LOCATE 33,10:PRINT"UNTERBRUCH
!"                                           <1AB5>
7150 LOCATE 33,12:PRINT"JA   / "in
$"NEIN"in$:jn=1                              <2E16>
7160 FOR w=1 TO 10:NEXT                      <1199>
7170 IF INKEY(8)=0 THEN LOCATE 33,
12:PRINT in$"JA  "in$" / NEIN":jn=
0                                            <37B5>
7180 IF INKEY(1)=0 THEN LOCATE 33,
12:PRINT"JA   / "in$"NEIN"in$:jn=1
                                             <3736>
7190 IF INKEY(9)=0 THEN CALL &A693
,6,&9867:i=9:GOSUB 7330:GOSUB 6940
:RETURN                                      <2B7F>
7200 GOTO 7160                               <097E>
7210 WINDOW#3,8,42,7,11::CALL &A69
0,3,&9A5F:CLS#3                              <229E>
7220 MOVE 64,296:DRAW 328,296:DRAW
 328,232:DRAW 64,232:DRAW 64,296  <2CE5>
7230 LOCATE 10,8:PRINT"BITTE DATEN
DISKETTE EINLEGEN..."                        <2D18>
7240 LOCATE 12,10:PRINT"UND DIE EN
TER-TASTE DRJCKEN"                           <2AA3>
7250 WHILE INKEY$<>CHR$(13):WEND  <115A>
7260 CALL &A693,3,&9A5F:RETURN    <1171>
7270 WINDOW#3,14,49,20,24:CALL &A6
90,3,&9A5F:CLS#3                             <23DF>
7280 MOVE 110,88:DRAW 386,88:DRAW
386,24:DRAW 110,24:DRAW 110,88    <2943>
7290 LOCATE 16,21:PRINT"BITTE LEER
E DISKETTE EINLEGEN..."                      <2FC7>
7300 LOCATE 18,23:PRINT"UND DIE EN
TER-TASTE DRJCKEN"                           <2AE7>
7310 WHILE INKEY$<>CHR$(13):WEND  <11D2>
7320 CALL &A693,3,&9A5F:RETURN    <11E9>
7330 WHILE INKEY(i)<>-1:WEND:RETUR
N                                            <153F>
7340 WINDOW#3,1,26,16,18:CALL &A69
0,3,&8F1F:CLS#3                              <22B3>
7350 MOVE 8,152:DRAW 200,152:DRAW
200,120:DRAW 8,120:DRAW 8,152     <24B8>
7360 LOCATE 3,17:PRINT"DRUCKER INI
TIALISIERT?"                                 <24BC>
7370 FOR w=1 TO 1500:NEXT w                  <163F>
7380 CALL &A693,3,&8F1F                      <0FF3>
7390 RETURN                                  <06FE>

100 '*****************************  <2234>
110 '*        MCLOADER 1         *  <222F>
120 '*    ERZEUGT >>MC1.BIN<<    *  <22E4>
130 '*      FUER MINIDATEI       *  <22C4>
140 '*           VON             *  <22A7>
150 '*      MARKUS WILDI         *  <22DA>
160 '*           FUER            *  <223C>
170 '*    SCHNEIDER CPC-WELT     *  <225B>
180 '*     CPC 464/664/6128   JE*  <226A>
190 '*****************************  <22E8>
640 MEMORY &9FFF                             <098A>
650 a=&A000:e=&A180:zb=1000:e=e+1 <2CE0>
660 FOR i=a TO e:IF i=e THEN SAVE"
mc1.bin",b,&A000,&180:END                    <39E2>
670 READ d$:POKE i,VAL("&"+d$)               <1D5A>
730 IF i<e THEN NEXT i                       <1526>
1001 DATA 01,0A,A0,21,22,A0,CD,D1 <1E3A>
1002 DATA BC,C9,12,A0,C3,CF,A0,C3 <1ED8>
1003 DATA 0D,A1,52,45,43,52,45,41 <1E90>
1004 DATA C4,52,45,43,57,52,49,54 <1E65>
1005 DATA C5,00,00,00,00,00,00,00 <1EC3>
1006 DATA 00,00,00,00,00,00,00,00 <1E82>
1007 DATA 00,C3,CF,A0,C3,0D,A1,FE <1E03>
1008 DATA 02,C2,B4,A0,DD,6E,02,DD <1E56>
1009 DATA 66,03,2B,7E,FE,04,C2,B4 <1E5C>
1010 DATA A0,23,11,26,A0,01,05,00 <1EEE>
1011 DATA ED,B0,DD,6E,00,DD,66,01 <1E6E>
1012 DATA 2B,7E,FE,02,C2,B4,A0,23 <1EEE>
1013 DATA 11,2E,A0,01,03,00,ED,B0 <1EB2>
1014 DATA 3A,2A,A0,FE,81,30,0D,21 <1E1B>
1015 DATA 00,00,22,2B,A0,3E,00,32 <1E54>
1016 DATA 2D,A0,37,C9,FE,98,D2,B4 <1EE1>
1017 DATA A0,21,29,A0,CB,FE,3A,2A <1E8A>
1018 DATA A0,FE,97,28,0F,21,2A,A0 <1EBB>
1019 DATA 34,2B,CB,3E,2B,CB,1E,2B <1E62>
1020 DATA CB,1E,18,EA,2A,28,A0,22 <1E90>
1021 DATA 2B,A0,3A,27,A0,CB,3F,32 <1EF0>
1022 DATA 2D,A0,3A,2E,A0,FE,00,CA <1E68>
1023 DATA B4,A0,37,C9,37,3F,C9,2A <1EAD>
1024 DATA 2B,A0,23,22,2B,A0,11,29 <1E14>
1025 DATA 00,2A,7D,BE,19,3A,2B,A0 <1E3E>
1026 DATA 77,23,3A,2C,A0,77,C9,CD <1E9B>
1027 DATA 37,A0,D0,CD,DC,A0,CD,F8 <1E0D>
1028 DATA A0,ED,B0,C9,CD,BE,A0,21 <1E5A>
```

```
1029 DATA 80,A1,CD,5A,A1,CD,B7,A0 (1ED3)
1030 DATA 21,00,A2,CD,5A,A1,CD,B7 (1E3C)
1031 DATA A0,21,80,A2,CD,5A,A1,C9 (1E06)
1032 DATA 3A,2D,A0,5F,16,00,21,80 (1E3E)
1033 DATA A1,19,ED,5B,2F,A0,3A,2E (1E15)
1034 DATA A0,4F,06,00,C9,CD,37,A0 (1E63)
1035 DATA D0,CD,BE,A0,21,80,A1,CD (1E28)
1036 DATA 5A,A1,CD,F8,A0,EB,ED,B0 (1ED4)
1037 DATA CD,BE,A0,21,80,A1,CD,61 (1EDA)
1038 DATA A1,CD,B7,A0,21,00,A2,CD (1E2F)
1039 DATA 5A,A1,CD,F8,A0,EB,ED,B0 (1EDA)
1040 DATA CD,BE,A0,21,00,A2,CD,61 (1E65)
1041 DATA A1,CD,B7,A0,21,80,A2,CD (1E25)
1042 DATA 5A,A1,CD,F8,A0,EB,ED,B0 (1EE0)
1043 DATA CD,BE,A0,21,80,A2,CD,61 (1E0A)
1044 DATA A1,C9,DF,5E,A1,C9,92,D3 (1EB9)
1045 DATA 07,DF,65,A1,C9,68,A1,07 (1E5E)
1046 DATA E5,D5,C5,E5,11,08,00,CD (1E6D)
1047 DATA 98,CA,CD,10,D4,D2,A9,D3 (1E88)
1048 DATA EB,E3,CD,F3,D9,C3,A6,D3 (1E9B)
1049 DATA 00                      (090A)
100 '*************************** (2234)
110 '*        MCLOADER 2         * (2227)
120 '*    ERZEUGT >>MC2.BIN<<    * (22B4)
130 '*      FUER MINIDATEI       * (22C4)
140 '*           VON             * (22A7)
150 '*       MARKUS WILDI        * (228A)
160 '*           FUER            * (223C)
170 '*    SCHNEIDER CPC-WELT     * (225B)
180 '*     CPC 464/664/6128   JE* (226A)
190 '*************************** (22E8)
650 a=&A690:e=&A6F7:zb=1000:e=e+1 (2C14)
660 FOR i=a TO e:IF i=e THEN SAVE"
mc2.bin",b,&A690,&67:END          (3916)
670 READ d$:POKE i,VAL("&"+d$)    (1D5A)
730 IF i<e THEN NEXT i            (1526)
1001 DATA EF,99,A6,EF,B0,A6,EF,C7 (1EB5)
1002 DATA A6,CD,E1,A6,D5,E5,7E,02 (1EDA)
1003 DATA CD,F9,0B,03,15,20,F7,E1 (1EFD)
1004 DATA D1,CD,13,0C,1D,20,ED,C9 (1EBA)
1005 DATA CD,E1,A6,D5,E5,0A,77,CD (1E24)
1006 DATA F9,0B,03,15,20,F7,E1,D1 (1ECB)
1007 DATA CD,13,0C,1D,20,ED,C9,CD (1E1C)
1008 DATA E1,A6,D5,E5,0A,5F,7E,02 (1EF7)
1009 DATA 73,CD,F9,0B,03,15,20,F4 (1E48)
1010 DATA E1,D1,CD,13,0C,1D,20,EA (1E4A)
1011 DATA C9,D5,DD,7E,02,CD,E8,10 (1E1B)
1012 DATA F5,CD,56,12,CD,95,0B,F1 (1E79)
1013 DATA E5,CD,E8,10,E1,C1,C9,00 (1EB0)
```



MOTORWAY

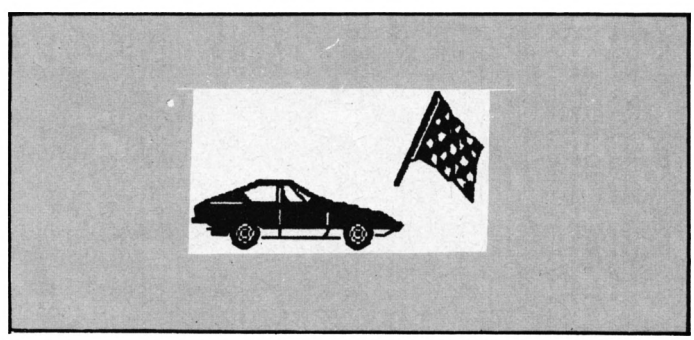
Vollgas ist angesagt

Dieses Spiel, eine Rennsimulation, wird sicher allen Autofreaks unter unseren Lesern gefallen. Sie sollten sich jedoch diese Beschreibung genau durchlesen, um mit der Tipparbeit keine Schwierigkeiten zu haben. Wie immer ist auch dieses Programm in der lauffähigen Version auf unserer Softbox enthalten.

Mit Hilfe des Programms >MOTORDAT.ERZ< wird die Datei >MOTORDAT.< erzeugt. Sie enthält die Werte für die zu symbolisierenden Zeichen.
Nach dem Starten des Programms >MOTORBIL.ERZ< entsteht das Titelbild für das Vorprogramm >MOTORVOR.BAS<, welches unter dem Namen >MOTORBIL.BIN< automatisch abgespeichert wird. Das Erzeugungsprogramm >MOTORWAD.ERZ< stellt eine Datei >MOTORWAY.DAT< her, die die Eintragungstabelle enthält. Alle drei Programme müssen nur einmal vor dem ersten Spiel gestartet werden. Danach sind sie überflüssig, es sei denn, durch das Programm >MOTORAD.ERZ< soll, da die Rennstrecke geändert wurde, wieder eine neue Eintragungstabelle erstellt werden.

Start des Spiels:

>MOTORWAY< läuft nur auf dem CPC 6128. Da die



zweite 64 kByte-Bank genutzt wird, muß vor dem Start das Programm >BANKMAN< geladen werden.
Nun kann das Vorprogramm >MOTORVOR< gestartet werden. Hierbei gelten folgende Tastenbelegungen:
a) >1< und >2< Lautstärkeregulation der Musik während des Titelbildes
b) >RETURN< Musik aus/an
c) >Leertaste< Hauptprogramm starten.
Nachdem Sie die Leertaste gedrückt haben, wird das Hauptprogramm >MOTORWAY.BAS< ab Zeile 80

Bitte lesen Sie weiter auf Seite 119

```
10 '*****************************  <2397>
11 '*   MOTORWAY-VORPROGRAMM    *  <23EB>
12 '*           VON             *  <2351>
13 '*       HOLGER HAUBER       *  <2365>
14 '*           FUER            *  <23FC>
15 '*    SCHNEIDER CPC-WELT     *  <23E3>
20 '*         CPC 6128       JE*  <23B3>
30 '*****************************  <23BF>
40 '                                <075B>
50 MODE 0:INK 0,0:INK 12,0:BORDER
0:PAPER 0:PEN 1                     <1BEC>
60 ON ERROR GOTO 810                <0B0A>
70 INK 1,15:INK 2,6:INK 3,24:TAG    <17F8>
80 MOVE 116,200,2,1:PRINT"Please w
ait !";                             <2187>
90 MOVE 112,196,1,1:PRINT"Please w
ait !";                             <2106>
100 DIM d(50,10):DIM g(50),sou1(48
),sou2(44),lan(44),i1(12):SYMBOL A
FTER 32:n=1                         <5462>
110 ENV 1,5,3,4,5,-3,8:ENV 2,3,5,1
,1,0,10,2,-1,9,13,-1,20             <3497>
120 RESTORE 910                     <094A>
130 FOR i=1 TO 48:READ sou1(i):NEX
T i                                 <249B>
140 FOR i=1 TO 43:READ sou2(i),lan
(i):NEXT i                          <3110>
150 OPENIN"motordat"                <1071>
160 WHILE NOT EOF                   <09F2>
170 INPUT#9,g(n),d(n,1),d(n,2),d(n
,3),d(n,4),d(n,5),d(n,6),d(n,7),d(
n,8)                                <7B2C>
180 n=n+1                           <1044>
190 WEND                            <06D2>
200 CLOSEIN                         <06B3>
210 FOR i=1 TO n-1                  <138C>
220 SYMBOL g(i),d(i,1),d(i,2),d(i,
3),d(i,4),d(i,5),d(i,6),d(i,7),d(i
,8)                                 <7806>
230 NEXT i                          <0A3D>
240 FOR i=0 TO 232 STEP 4           <1160>
250 MOVE i+88,204,0,0:DRAW i+88,18
0:MOVE i*-1+552,204,0,0:DRAW i*-1+
552,180                             <4492>
260 NEXT i:CLS                      <0C00>
270 FOR i=0 TO 15:INK i,0:NEXT i    <1D59>
280 LOAD"motorbil"                  <10DA>
290 /SCREENSWAP,1,2                 <15D0>
300 CLS                             <0683>
310 m(1)=1:m(2)=2:m(3)=4:m(4)=8     <2C19>
320 MODE 0:PRINT CHR$(23);CHR$(3);
:TAG                                <1800>
330 FOR p=1 TO 4                    <0E9C>
340 GRAPHICS PEN m(p),1             <1398>
350 LOCATE#1,1,25:PRINT#1,CHR$(48+
p);                                 <1EEF>
360 FOR x=0 TO 7                    <0ED6>
370 FOR y=0 TO 14 STEP 2            <11E0>
380 BORDER INT(RND*25)              <0FE8>
390 IF TEST(x*4,y)=0 THEN 420       <1B3B>
400 MOVE(x+6)*32,(y+6)*16:PRINT CH
R$(143);                            <266A>
410 MOVE(x+6)*32,(y+7)*16:PRINT CH
R$(143);                            <263F>
420 NEXT y,x,p                      <1405>
430 BORDER 0                        <07AA>
440 FOR i=1 TO 500:NEXT i           <16B2>
450 LOCATE#1,1,25:PRINT#1," ";      <1631>
460 FOR p=4 TO 1 STEP-1             <1134>
470 FOR i=0 TO 15                   <0F8B>
480 IF(i AND m(p))=0 THEN INK i,0
ELSE INK i,26                       <2B26>
490 NEXT i:FOR k=1 TO 1000:NEXT k:
NEXT p                              <22C1>
500 CLS:FOR i=1 TO 4:GRAPHICS PEN
i,0:NEXT i                          <1F40>
510 i1(1)=26:i1(2)=24:i1(3)=15:i1(
4)=6:i1(5)=4:i1(6)=8:i1(7)=2:i1(8)
=1:i1(9)=13:i1(10)=7:i1(11)=16      <84EF>
520 ii1(1)=10:ii1(2)=12:ii1(3)=18   <2BD8>
530 FOR i=0 TO 15:INK i,0:NEXT i    <1D62>
540 /SCREENSWAP,1,2                 <15C5>
550 onoff=1:n=48:n2=44:ton=12:ton2
=12                                 <35B1>
560 ON SQ(1)GOSUB 830               <0D9B>
570 ON SQ(2)GOSUB 870               <0D71>
580 EVERY 12,3 GOSUB 720            <0ED5>
590 RESTORE 680:ORIGIN 0,0,128,512
,50,0:TAG:FRAME                     <1ED0>
600 READ lau$:IF lau$="*"THEN lau$
="":FOR i=0 TO 9:READ a:lau$=lau$+
CHR$(a):NEXT i ELSE IF lau$="ENDE"
THEN 590                            <639E>
610 lau$=lau$+CHR$(32)              <1902>
620 FOR i=512 TO 128-(LEN(lau$)-1)
*32 STEP-4                          <26B3>
630 MOVE i,34,13,0:PRINT lau$;:NEX
T i                                 <21BA>
640 READ lau$:IF lau$="*"THEN lau$
="":FOR i=0 TO 9:READ a:lau$=lau$+
CHR$(a):NEXT i ELSE IF lau$="ENDE"
THEN 590                            <63EE>
650 lau$=CHR$(32)+lau$              <19B1>
660 FOR i=128-LEN(lau$)*32 TO 488
STEP 4                              <216D>
670 MOVE i,34,13,0:PRINT lau$;:NEX
T i:GOTO 600                        <26D1>
680 DATA copyright,by the,*,32,205
,206,207,208,209,206,210,211,32,an
d,Holger,Hauber,in 1986/87,*****    <6259>
690 DATA special,greetings,to the,
manager of,*,212,213,214,215,215,2
16,217,214,218,215,OLIVER,ABRECHT,
()()()                              <6A61>
700 DATA *,45,205,206,207,208,209,
206,210,211,45,Gustav,Stresemann,r
```

```
oad 53,7530,Pforzheim,GERMANY,/\/\
/\                                  <66C5>
710 DATA ENDE                       <0BA0>
720 FOR k=1 TO 11:INK k,i1(k):NEXT
 k                                  <270F>
730 FOR m=11 TO 1 STEP-1:i1(m+1)=i
1(m):NEXT m:i1(1)=i1(12)            <45D4>
740 FOR k1=1 TO 3:INK k1+12,ii1(k1
):NEXT k1                           <2E54>
750 FOR m1=3 TO 1 STEP-1:ii1(m1+1)
=ii1(m1):NEXT m1:ii1(1)=ii1(4)      <4B71>
760 IF INKEY(47)=0 OR JOY(0)=16 TH
EN CHAIN"motorway",80               <261A>
770 IF INKEY(64)=0 AND ton<>0 THEN
 ton2=ton2-1:IF ton2<8 THEN ton2=8
:ton=ton2 ELSE ton=ton2             <5D4A>
780 IF INKEY(65)=0 AND ton<>0 THEN
 ton2=ton2+1:IF ton2>14 THEN ton2=
14:ton=ton2 ELSE ton=ton2           <5F52>
790 IF INKEY(18)=0 THEN onoff=onof
f XOR 1:IF onoff=1 THEN ON SQ(1)GO
SUB 830:ON SQ(2)GOSUB 870           <4052>
800 RETURN                          <066A>
810 MODE 2:PEN 1:PAPER 0:BORDER 1:
INK 1,24:INK 0,1:CLEAR INPUT:PRINT
"Fehler ";ERR;" in Zeile ";ERL      <3BC6>
820 END                             <06CD>
830 n=n+1:IF n>48 THEN n=1:n2=1:to
n=INT(RND*2)*ton2                   <4086>
840 SOUND 1,sou1(n),25,9,0,1        <1EA1>
850 IF onoff=1 THEN ON SQ(1)GOSUB
830                                 <19D2>
860 RETURN                          <06E2>
870 n2=n2+1:IF n2>43 THEN n2=1:ton
=0                                  <2DAB>
880 SOUND 2,sou2(n2),lan(n2),ton,0
,2:SOUND 4,sou2(n2)+1,lan(n2),ton,
0,1                                 <5C0E>
890 IF onoff=1 THEN ON SQ(2)GOSUB
870                                 <19DC>
900 RETURN                          <0631>
910 DATA 239,213,239,213,239,213,2
39,213,239,213,239,213,239,213,239
,213                                <46BC>
920 DATA 179,159,179,159,179,159,1
79,159,239,213,239,213,239,213,239
,213                                <46DE>
930 DATA 159,142,159,142,179,159,1
79,159,239,213,239,213,239,213,239
,213                                <463A>
940 DATA 142,12.5,142,12.5,106,25,
113,25,142,25,179,12.5,179,12.5,14
2,25,159,25,169,12.5,213,25,213,12
.5,213,100,0,75                     <733C>
950 DATA 142,12.5,142,12.5,142,25,
159,25,169,25,213,12.5,213,12.5,14
2,25,159,25,169,12.5,213,25,213,12
.5,213,100,0,75                     <735C>
960 DATA 142,12.5,142,12.5,113,12.
5,142,12.5,119,12.5,119,12.5,142,5
0,142,12.5,142,12.5,142,12.5,159,3
7.5,169,12.5,213,12.5,213,100,0,75
                                    <86EC>

10 '*****************************  <2397>
11 '*  MOTORWAY-HAUPTPROGRAMM  *   <2351>
12 '* ABSPEICHERN ALS<MOTORWAY>*   <23C7>
13 '*           VON             *  <2353>
14 '*       HOLGER HAUBE        *  <23AE>
15 '*          FUER             *  <2370>
20 '*    SCHNEIDER CPC-WELT     *  <23ED>
30 '*        CPC 6128           *  <23E9>
35 '*****************************  <23C9>
38 '                                <0757>
40 ' *****                          <0D6F>
50 ' SYMBOLE                        <0F97>
60 ' *****                          <0D97>
70 RUN"motorvor"                    <101F>
80 MODE 0                           <0744>
90 FOR i=0 TO 15:INK i,0:NEXT i:BO
RDER 1                              <20F6>
100 SYMBOL AFTER 160                <0900>
110 SYMBOL 160,&X1101110,&X101010,
&X101010,&X1001010,&X1101110,0,0,0
                                    <22A2>
120 SYMBOL 161,&X1101110,&X1001010
,&X1101010,&X1101010,&X1101110,0,0
,0                                  <22A9>
130 SYMBOL 162,&X10110110,&X101101
10,&X10110110,&X10110110,&X1011011
0,0,0,0                             <221E>
140 SYMBOL 163,&X10100110,&X101001
10,&X10110110,&X10010110,&X1001011
0,0,0,0                             <2238>
150 SYMBOL 164,&X10111011,&X101010
11,&X10010011,&X10101011,&X1011101
1,0,0,0                             <22AF>
160 SYMBOL 165,&X11011011,&X100101
1,&X1001011,&X10010011,&X11011011,
0,0,0                               <22F6>
170 SYMBOL 167,0,0,&X10000000,&X10
000000,&X10101101,&X11001010,&X101
01010,0                             <2289>
180 SYMBOL 168,0,&X1000,&X1010,&X1
0010,&X10011,&X10100010,&X10100010
,0                                  <249C>
190 SYMBOL 169,0,0,0,0,&X10000000,
&X10000000,&X10000000,0             <1E97>
200 SYMBOL 170,&X11110,&X10010,&X1
0010,&X10010,&X100100,&X100100,&X1
00100,&X111100                      <287E>
210 SYMBOL 171,&X10,&X10,&X10,&X10
,&X100,&X100,&X100,&X100            <2802>
220 SYMBOL 172,&X11110,&X10,&X10,&
X111110,&X100000,&X100000,&X100000
,&X111100                           <28BB>
```

```
230 SYMBOL 173,&X11110,&X10,&X10,&
X11110,&X100,&X100,&X100,&X111100 <285D>
240 SYMBOL 174,&X10010,&X10010,&X1
0010,&X11110,&X100,&X100,&X100,&X1
00                                <28CE>
250 SYMBOL 175,&X11110,&X10000,&X1
0000,&X11100,&X100,&X100,&X100,&X1
11100                             <281A>
260 SYMBOL 176,&X11110,&X10000,&X1
0000,&X11100,&X100100,&X100100,&X1
00100,&X111100                    <2844>
270 SYMBOL 177,&X11110,&X10010,&X1
0010,&X10010,&X100,&X100,&X100,&X1
00                                <284C>
280 SYMBOL 178,&X11110,&X10010,&X1
0010,&X11110,&X100100,&X100100,&X1
00100,&X111100                    <2820>
290 SYMBOL 179,&X11110,&X10010,&X1
0010,&X11110,&X100,&X100,&X100,&X1
11100                             <28FC>
300 SYMBOL 180,0,0,&X1110,&X100,&X
100,&X100,&X100,0                 <22BF>
310 SYMBOL 181,0,0,&X10101011,&X10
111010,&X10101011,&X10101010,&X101
01011,0                           <2236>
320 SYMBOL 182,0,&X100000,&X100000
0,&X1001010,&X11101010,&X1001010,&
X1001010,&X1000100                <262D>
330 SYMBOL 183,0,&X1000,&X1000,&X1
001000,&X10101000,&X11101000,&X100
01000,&X1100100                   <2644>
340 SYMBOL 184,0,&X1000000,&X10000
00,&X11100100,&X1001010,&X1001110,
&X1001000,&X100110                <264E>
350 SYMBOL 185,0,0,0,&X11010000,&X
10101000,&X10101000,&X10101000,&X1
0101010                           <221B>
360 SYMBOL 186,0,0,&X110110,&X1000
100,&X1010110,&X1010100,&X110110,0
                                  <2291>
370 SYMBOL 187,0,0,&X11101100,&X10
101010,&X11101100,&X10101010,&X101
01010,0                           <224B>
380 SYMBOL 188,0,0,0,0,0,&X1000000
,&X1000000,0                      <1CD5>
390 SYMBOL 189,0,&X1010101,&X10101
11,&X1100101,&X1010101,&X1010101,0
,0                                <2283>
400 SYMBOL 190,&X1000000,&X1010000
0,&X10100000,&X10100000,&X10100000
,&X10100000,&X1000000,0           <26F4>
410 SYMBOL 191,&X100000,&X1100000,
&X10100000,&X100000,&X100000,&X100
000,&X100000,0                    <2653>
420 SYMBOL 192,&X1000000,&X1010000
0,&X100000,&X100000,&X1000000,&X10
000000,&X11100000,0               <26DB>
430 SYMBOL 193,&X11000000,&X100000
,&X100000,&X1000000,&X100000,&X100
000,&X11000000,0                  <26D3>
440 SYMBOL 194,&X1000000,&X1000000
,&X10000000,&X10100000,&X11100000,
&X100000,&X100000,0               <2667>
450 SYMBOL 195,&X11100000,&X100000
00,&X10000000,&X11000000,&X100000,
&X100000,&X11000000,0             <2639>
460 SYMBOL 196,&X1100000,&X1000000
0,&X10000000,&X11000000,&X10100000
,&X10100000,&X1000000,0           <2609>
470 SYMBOL 197,&X11100000,&X100000
,&X100000,&X1000000,&X1000000,&X10
000000,&X10000000,0               <2601>
480 SYMBOL 198,&X1000000,&X1010000
0,&X10100000,&X1000000,&X10100000,
&X10100000,&X1000000,0            <26FE>
490 SYMBOL 199,&X1000000,&X1010000
0,&X10100000,&X11100000,&X100000,&X
10100000,&X1000000,0              <26A8>
500 SYMBOL 200,&X11101010,&X100101
0,&X1001010,&X1001110,0,0,0,0     <2071>
510 SYMBOL 201,&X11001010,&X101011
10,&X11001110,&X10101010,0,0,0,0  <20C7>
520 SYMBOL 202,&X11,&X1111,&X11111
,&X111111,&X1111111,&X1111111,&X11
111111,&X11111111                 <2857>
530 SYMBOL 203,&X11000000,&X111100
00,&X11111000,&X11111100,&X1111111
0,&X11111110,&X11111111,&X11111111
                                  <2833>
540 SYMBOL 204,&X111,&X1111,&X1111
,&X1111,&X11111,&X11111,&X11111,&X
111111                            <28DE>
550 SYMBOL 205,&X11110000,&X111110
00,&X11111000,&X11111000,&X1111010
0,&X11110100,&X11110100,&X11101100
                                  <2856>
560 SYMBOL 206,&X1111,&X11111,&X11
111,&X11111,&X101111,&X101111,&X10
1111,&X110111                     <283D>
570 SYMBOL 207,&X11100000,&X111100
00,&X11110000,&X11110000,&X1111100
0,&X11111000,&X11111000,&X11111100
                                  <289F>
580 SYMBOL 208,0,0,&X11001010,&X10
011011,&X11011011,&X10011011,&X110
10010,0                           <2281>
590 DIM x2(32),y2(32),ii2(32),einm
(35),eins(35),ein$(35)            <471D>
600 OPENIN"motorway.dat":FOR k=1 T
O 35:INPUT#9,einm(k),eins(k),ein$(
k):NEXT k:CLOSEIN                 <5491>
610 DEG:ORIGIN 0,0                <0B41>
620 PAPER 0:WINDOW 1,20,1,19:WINDO
W#1,1,20,20,25:PAPER#1,10:CLS#1   <2C87>
630 GOSUB 4930                    <091F>
640 ON ERROR GOTO 7470            <0B1B>
```

```
650 GOSUB 760                                   (09B2)
660 /SCREENCOPY,2,1                             (1562)
670 GOSUB 1380                                  (0942)
680 /SCREENCOPY,3,1                             (159A)
690 GOSUB 2080                                  (095C)
700 /SCREENCOPY,4,1                             (15D2)
710 GOSUB 2710                                  (0938)
720 /SCREENCOPY,5,1                             (150B)
730 GOSUB 3400                                  (09FB)
740 q3(1)=3:q3(2)=2:q3(4)=4:q3(5)=
5:q3(3)=1                                       (3B0D)
750 GOTO 3910                                   (093A)
760 ' *****                                     (0D11)
770 ' BILD 1                                    (0E8F)
780 ' *****                                     (0D3A)
790 CLS                                         (0659)
800 MOVE 0,96,1:DRAW 88,96:MOVE 54
8,96:DRAW 640,96                                (2385)
810 MOVE 0,120,1:DRAW 640,132'124               (19B9)
820 MOVE 0,122,2:DRAW 640,134'126               (194F)
830 MOVE 0,150,2:DRAW 640,168'156               (193F)
840 MOVE 0,152,3:DRAW 640,170'158               (19E2)
850 MOVE 0,177,3:DRAW 640,201'185               (194D)
860 MOVE 0,179,4:DRAW 640,203'187               (19E4)
870 MOVE 0,197,4:DRAW 640,227'207               (196C)
880 MOVE 0,199,1:DRAW 640,229'209               (191C)
890 MOVE 0,213,1:DRAW 640,249'225               (19CF)
900 MOVE 0,215,2:DRAW 640,251'227               (1989)
910 MOVE 0,226,2:DRAW 640,268'240               (1A00)
920 MOVE 0,228,3:DRAW 640,270'242               (1A0E)
930 MOVE 0,236,3:DRAW 640,284'252               (1A7F)
940 MOVE 0,238,4:DRAW 640,286'254               (1ABD)
950 MOVE 0,245,4:DRAW 640,299'263               (1AC6)
960 MOVE 0,247,1:DRAW 640,301'265               (1AE3)
970 MOVE 0,250,1:DRAW 640,310'270               (1ADA)
980 MOVE 0,252,2:DRAW 640,312'272               (1A58)
990 MOVE 0,256,2:DRAW 640,322'278               (1B9F)
1000 MOVE 0,258,3:DRAW 640,324'280
                                                (1B78)
1010 MOVE 0,258,3:DRAW 640,330'282
                                                (1B9A)
1020 MOVE 0,260,4:DRAW 640,332'284
                                                (1BB9)
1030 MOVE 0,262,4:DRAW 640,340'288
                                                (1B31)
1040 MOVE 0,264,1:DRAW 640,342'290
                                                (1B50)
1050 MOVE 0,264,1:DRAW 640,348'292
                                                (1B76)
1060 MOVE 88,96,7:DRAW 548,96                   (1544)
1070 MOVE 1,100:FILL 1                          (0D64)
1080 MOVE 1,140:FILL 2                          (0D85)
1090 MOVE 1,170:FILL 3                          (0D61)
1100 MOVE 1,180:FILL 4                          (0DBA)
1110 MOVE 1,205:FILL 1                          (0DC5)
1120 MOVE 1,220:FILL 2                          (0DBF)
1130 MOVE 1,230:FILL 3                          (0D15)
1140 MOVE 1,242:FILL 4                          (0DB0)
1150 MOVE 320,276:FILL 1                        (104D)
1160 MOVE 320,284:FILL 2                        (1067)
1170 MOVE 320,300:FILL 4                        (1087)
1180 MOVE 180,282:FILL 4                        (0F61)
1190 RESTORE 1210                               (0908)
1200 GOSUB 4810                                 (09D1)
1210 DATA -134,351,339,95,140,7                 (1CB4)
1220 DATA -134,-692,1045,15,45,7                (1D35)
1230 MOVE 320,100:FILL 7                        (0F11)
1240 GOSUB 4810                                 (0920)
1250 DATA -482,871,886,130,150,0                (1DF0)
1260 DATA -474,-2007,2404,15,30,0               (1E5A)
1270 MOVE 1,150:FILL 0:MOVE 639,15
0:FILL 0                                        (186B)
1280 MOVE 0,300,6:DRAW 640,300,6                (186D)
1290 MOVE 320,350:FILL 6                        (10CB)
1300 PLOT 248,280,2                             (0E9B)
1310 MOVE 244,282,3:DRAW 236,284                (16C1)
1320 MOVE 232,286,4:DRAW 226,288                (16FA)
1330 MOVE 222,290,1:DRAW 214,292                (16A7)
1340 MOVE 210,294,2:DRAW 202,296                (1630)
1350 MOVE 194,296,2:DRAW 190,290                (1662)
1360 MOVE 194,296,2:DRAW 194,292                (1685)
1370 RETURN                                     (06E0)
1380 ' *****                                    (0DEC)
1390 ' BILD 2                                   (0E6E)
1400 ' *****                                    (0D13)
1410 CLS                                        (0634)
1420 MOVE 0,96,1:DRAW 88,96:MOVE 5
44,96:DRAW 640,96                               (2351)
1430 MOVE 0,122,1:DRAW 640,140'124
                                                (1994)
1440 MOVE 0,124,2:DRAW 640,142'126
                                                (19CA)
1450 MOVE 0,152,2:DRAW 640,180'156
                                                (1924)
1460 MOVE 0,154,3:DRAW 640,182'158
                                                (1949)
1470 MOVE 0,180,3:DRAW 640,217'185
                                                (1938)
1480 MOVE 0,182,4:DRAW 640,219'187
                                                (19C3)
1490 MOVE 0,201,4:DRAW 640,247'207
                                                (1942)
1500 MOVE 0,203,1:DRAW 640,249'209
                                                (1982)
1510 MOVE 0,218,1:DRAW 640,273'225
                                                (1A5D)
1520 MOVE 0,220,2:DRAW 640,275'227
                                                (1AD8)
1530 MOVE 0,232,2:DRAW 640,296'240
                                                (1AE5)
1540 MOVE 0,234,3:DRAW 640,298'242
                                                (1AFD)
1550 MOVE 0,242,3:DRAW 640,316'252
                                                (1A83)
1560 MOVE 0,244,4:DRAW 640,318'254
                                                (1A77)
```

```
1570 MOVE 0,252,4:DRAW 640,335'263                    <1A0D>
1580 MOVE 0,254,1:DRAW 640,337'265                    <1A3C>
1590 MOVE 0,258,1:DRAW 640,350'270                    <1B46>
1600 MOVE 0,260,2:DRAW 640,352'272                    <1B4F>
1610 MOVE 0,265,2:DRAW 640,366'278                    <1BD7>
1620 MOVE 0,267,3:DRAW 640,368'280                    <1BC6>
1630 MOVE 0,268,3:DRAW 640,378'282                    <1B09>
1640 MOVE 0,270,4:DRAW 640,380'284                    <1B53>
1650 MOVE 0,272,4:DRAW 640,392'288                    <1B93>
1660 MOVE 0,274,1:DRAW 640,394'290                    <1B70>
1670 MOVE 0,275,1:DRAW 640,404'292                    <1B8E>
1680 MOVE 88,96,7:DRAW 544,96          <1522>
1690 MOVE 639,130:FILL 1               <0FD7>
1700 MOVE 639,150:FILL 2               <0F70>
1710 MOVE 639,190:FILL 3               <0F93>
1720 MOVE 639,230:FILL 4               <0FAA>
1730 MOVE 639,260:FILL 1               <10D0>
1740 MOVE 639,280:FILL 2               <10EF>
1750 MOVE 639,300:FILL 3               <100B>
1760 MOVE 639,330:FILL 4               <106A>
1770 MOVE 639,340:FILL 1               <10F4>
1780 MOVE 639,360:FILL 2               <100E>
1790 MOVE 639,370:FILL 3               <10AC>
1800 MOVE 80,280:FILL 3                <0F2A>
1810 MOVE 88,282:FILL 3                <0FD6>
1820 MOVE 639,385:FILL 4               <10B1>
1830 MOVE 80,286:FILL 4                <0FEB>
1840 RESTORE 1860                      <0980>
1850 GOSUB 4810                        <09E7>
1860 DATA 3,194,130,35,140,7           <1997>
1870 DATA 3,-535,837,5,45,7            <1816>
1880 MOVE 88,96,7:DRAW 544,96          <15B4>
1890 MOVE 320,100:FILL 7               <0F3B>
1900 GOSUB 4810                        <094C>
1910 DATA -173,314,249,90,135,0        <1C36>
1920 DATA -165,-1450,1767,5,30,0       <1D5D>
1930 MOVE 1,150:FILL 0:MOVE 639,15
0:FILL 0                               <1895>
1940 MOVE 0,300,6:DRAW 640,300,6       <1898>
1950 MOVE 320,350:FILL 6               <10F6>
1960 MOVE 260,258,3:DRAW 236,266       <1729>
1970 MOVE 232,268,4:DRAW 200,276       <16F0>
1980 MOVE 196,276,1:DRAW 176,280       <16ED>
1990 MOVE 172,282,2:DRAW 152,286       <1622>
2000 MOVE 148,288,3:DRAW 136,290       <16BD>
2010 MOVE 132,290,4:DRAW 120,292       <1691>
2020 MOVE 116,292,1:DRAW 104,294       <1637>
2030 PLOT 80,292,1                     <0EA6>
2040 MOVE 76,292,2:DRAW 76,296         <1664>
2050 MOVE 76,296,2:DRAW 80,294         <16E9>
2060 MOVE 88,296,2:DRAW 100,296        <161F>
2070 RETURN                            <065B>
2080 ' *****                           <0D67>
2090 ' BILD 3                          <0EEC>
2100 ' *****                           <0D8F>
2110 CLS                               <06AE>
2120 MOVE 0,96,1:DRAW 88,96:MOVE 5
48,96:DRAW 640,96                      <23DA>
2130 MOVE 640,120,1:DRAW 0,132'124                    <1907>
2140 MOVE 640,122,2:DRAW 0,134'126                    <19C5>
2150 MOVE 640,150,2:DRAW 0,168'156                    <194D>
2160 MOVE 640,152,3:DRAW 0,170'158                    <198F>
2170 MOVE 640,177,3:DRAW 0,201'185                    <1903>
2180 MOVE 640,179,4:DRAW 0,203'187                    <1948>
2190 MOVE 640,197,4:DRAW 0,227'207                    <195B>
2200 MOVE 640,199,1:DRAW 0,229'209                    <190F>
2210 MOVE 640,213,1:DRAW 0,249'225                    <1910>
2220 MOVE 640,215,2:DRAW 0,251'227                    <1962>
2230 MOVE 640,226,2:DRAW 0,268'240                    <1A4E>
2240 MOVE 640,228,3:DRAW 0,270'242                    <1A92>
2250 MOVE 640,236,3:DRAW 0,284'252                    <1ADC>
2260 MOVE 640,238,4:DRAW 0,286'254                    <1A45>
2270 MOVE 640,245,4:DRAW 0,299'263                    <1A7F>
2280 MOVE 640,247,1:DRAW 0,301'265                    <1AD0>
2290 MOVE 640,250,1:DRAW 0,310'270                    <1AD0>
2300 MOVE 640,252,2:DRAW 0,312'272                    <1A55>
2310 MOVE 640,256,2:DRAW 0,322'278                    <1B57>
2320 MOVE 640,258,3:DRAW 0,324'280                    <1B90>
2330 MOVE 640,258,3:DRAW 0,330'282                    <1BB2>
2340 MOVE 640,260,4:DRAW 0,332'284                    <1B03>
2350 MOVE 640,262,4:DRAW 0,340'288                    <1B64>
2360 MOVE 640,264,1:DRAW 0,342'290                    <1B97>
```

```
2370 MOVE 640,264,1:DRAW 0,348'292
                                        (1BB5)
2380 MOVE 88,96,7:DRAW 548,96           (1599)
2390 MOVE 639,100:FILL 1                (0F8E)
2400 MOVE 639,140:FILL 2                (0FA9)
2410 MOVE 639,170:FILL 3                (0F85)
2420 MOVE 639,180:FILL 4                (0FE0)
2430 MOVE 639,205:FILL 1                (0F0C)
2440 MOVE 639,220:FILL 2                (0F03)
2450 MOVE 639,230:FILL 3                (0F5D)
2460 MOVE 639,242:FILL 4                (0FF8)
2470 MOVE 320,276:FILL 1                (10A2)
2480 MOVE 320,284:FILL 2                (10BC)
2490 MOVE 320,298:FILL 4                (105A)
2500 MOVE 460,282:FILL 4                (109B)
2510 MOVE 448,284:FILL 4                (1002)
2520 RESTORE 2540                       (0917)
2530 GOSUB 4810                         (0939)
2540 DATA 774,-692,1045,315,345,7       (1ED0)
2550 DATA 774,351,339,220,265,7         (1C85)
2560 MOVE 320,100:FILL 7                (0F7B)
2570 GOSUB 4810                         (098A)
2580 DATA 1114,-2007,2404,330,345,
0                                       (2098)
2590 DATA 1122,871,886,210,230,0        (1DFC)
2600 MOVE 1,150:FILL 0:MOVE 639,15
0:FILL 0                                (18D5)
2610 MOVE 0,300,6:DRAW 640,300,6        (18D6)
2620 MOVE 320,350:FILL 6                (1035)
2630 PLOT 388,280,2                     (0F3E)
2640 MOVE 392,282,3:DRAW 400,284        (180A)
2650 MOVE 404,286,4:DRAW 412,288        (1876)
2660 MOVE 416,290,1:DRAW 424,292        (1840)
2670 MOVE 428,294,2:DRAW 436,296        (1837)
2680 MOVE 444,296,2:DRAW 448,290        (185E)
2690 MOVE 444,296,2:DRAW 444,292        (1890)
2700 RETURN                             (0648)
2710 ' *****                            (0D54)
2720 ' BILD 4                           (0EDF)
2730 ' *****                            (0D7C)
2740 CLS                                (069D)
2750 MOVE 0,96,1:DRAW 92,96:MOVE 5
48,96:DRAW 640,96                       (233E)
2760 MOVE 640,122,1:DRAW 0,140'124
                                        (19F7)
2770 MOVE 640,124,2:DRAW 0,142'126
                                        (19B1)
2780 MOVE 640,152,2:DRAW 0,180'156
                                        (1935)
2790 MOVE 640,154,3:DRAW 0,182'158
                                        (1983)
2800 MOVE 640,180,3:DRAW 0,217'185
                                        (196E)
2810 MOVE 640,182,4:DRAW 0,219'187
                                        (19AA)
2820 MOVE 640,201,4:DRAW 0,247'207
                                        (193D)
2830 MOVE 640,203,1:DRAW 0,249'209
                                        (19F6)
2840 MOVE 640,218,1:DRAW 0,273'225
                                        (1AC6)
2850 MOVE 640,220,2:DRAW 0,275'227
                                        (1A29)
2860 MOVE 640,232,2:DRAW 0,296'240
                                        (1A4C)
2870 MOVE 640,234,3:DRAW 0,298'242
                                        (1AA8)
2880 MOVE 640,242,3:DRAW 0,316'252
                                        (1AED)
2890 MOVE 640,244,4:DRAW 0,318'254
                                        (1A39)
2900 MOVE 640,252,4:DRAW 0,335'263
                                        (1A8C)
2910 MOVE 640,254,1:DRAW 0,337'265
                                        (1AE3)
2920 MOVE 640,258,1:DRAW 0,350'270
                                        (1BD3)
2930 MOVE 640,260,2:DRAW 0,352'272
                                        (1B34)
2940 MOVE 640,265,2:DRAW 0,366'278
                                        (1B89)
2950 MOVE 640,267,3:DRAW 0,368'280
                                        (1BBB)
2960 MOVE 640,268,3:DRAW 0,378'282
                                        (1BDB)
2970 MOVE 640,270,4:DRAW 0,380'284
                                        (1B45)
2980 MOVE 640,272,4:DRAW 0,392'288
                                        (1B89)
2990 MOVE 640,274,1:DRAW 0,394'290
                                        (1BBE)
3000 MOVE 640,275,1:DRAW 0,404'292
                                        (1BDE)
3010 MOVE 92,96,7:DRAW 548,96           (1594)
3020 MOVE 1,130:FILL 1                  (0D76)
3030 MOVE 1,150:FILL 2                  (0D0A)
3040 MOVE 1,190:FILL 3                  (0D0E)
3050 MOVE 1,230:FILL 4                  (0D1F)
3060 MOVE 1,260:FILL 1                  (0E58)
3070 MOVE 1,280:FILL 2                  (0E75)
3080 MOVE 1,300:FILL 3                  (0E93)
3090 MOVE 1,330:FILL 4                  (0E33)
3100 MOVE 1,340:FILL 1                  (0EBD)
3110 MOVE 1,360:FILL 2                  (0EDA)
3120 MOVE 1,370:FILL 3                  (0E75)
3130 MOVE 560,280:FILL 3                (1097)
3140 MOVE 552,282:FILL 3                (100B)
3150 MOVE 1,385:FILL 4                  (0E79)
3160 MOVE 556,286:FILL 4                (1049)
3170 RESTORE 3190                       (098F)
3180 GOSUB 4810                         (0951)
3190 DATA 637,-535,837,315,355,7        (1D1F)
3200 DATA 637,194,130,220,325,7         (1CF8)
3210 MOVE 320,100:FILL 7                (0F90)
3220 GOSUB 4810                         (09A1)
3230 DATA 805,-1450,1767,330,355,0
                                        (1F41)
```

```
3240 DATA 813,314,249,225,270,0   (1C51)
3250 MOVE 1,150:FILL 0:MOVE 639,15
0:FILL 0                          (18EA)
3260 MOVE 0,300,6:DRAW 640,300,6  (18ED)
3270 MOVE 320,350:FILL 6          (104A)
3280 MOVE 376,258,3:DRAW 400,266  (18B6)
3290 MOVE 404,268,4:DRAW 436,276  (1807)
3300 MOVE 440,276,1:DRAW 460,280  (1863)
3310 MOVE 464,282,2:DRAW 484,286  (189A)
3320 MOVE 488,288,3:DRAW 500,290  (18A1)
3330 MOVE 504,290,4:DRAW 516,292  (18E8)
3340 MOVE 520,292,1:DRAW 532,294  (18CF)
3350 PLOT 560,292,1               (0FB8)
3360 MOVE 564,292,2:DRAW 564,296  (183B)
3370 MOVE 564,296,2:DRAW 560,294  (1836)
3380 MOVE 548,296,2:DRAW 536,296  (1818)
3390 RETURN                       (06B0)
3400 ' *****                      (0DBC)
3410 ' BILD 5                     (0E4A)
3420 ' *****                      (0DE4)
3430 CLS                          (0603)
3440 MOVE 0,300,6:DRAW 640,300    (169B)
3450 MOVE 0,96,1:DRAW 84,96:MOVE 5
52,96:DRAW 640,96                 (23C6)
3460 MOVE 0,124,1:DRAW 116,124:MOV
E 520,124:DRAW 640,124            (230C)
3470 MOVE 1,126,2:DRAW 120,126:MOV
E 516,126:DRAW 639,126            (235A)
3480 MOVE 56,156,2:DRAW 152,156:MO
VE 484,156:DRAW 584,156           (243B)
3490 MOVE 60,158,3:DRAW 156,158:MO
VE 480,158:DRAW 580,158           (2481)
3500 MOVE 108,185,3:DRAW 184,185:M
OVE 452,185:DRAW 532,185          (24A6)
3510 MOVE 112,187,4:DRAW 188,187:M
OVE 448,187:DRAW 528,187          (24F9)
3520 MOVE 148,207,4:DRAW 212,207:M
OVE 428,207:DRAW 492,207          (2495)
3530 MOVE 152,209,1:DRAW 212,209:M
OVE 424,209:DRAW 488,209          (2433)
3540 MOVE 180,225,1:DRAW 232,225:M
OVE 408,225:DRAW 460,225          (24FD)
3550 MOVE 184,227,2:DRAW 236,227:M
OVE 404,227:DRAW 456,227          (249B)
3560 MOVE 208,240,2:DRAW 252,240:M
OVE 388,240:DRAW 432,240          (24C5)
3570 MOVE 212,242,3:DRAW 252,242:M
OVE 384,242:DRAW 428,242          (2401)
3580 MOVE 232,252,3:DRAW 264,252:M
OVE 376,252:DRAW 408,252          (25CE)
3590 MOVE 236,254,4:DRAW 268,254:M
OVE 372,254:DRAW 404,254          (251D)
3600 MOVE 252,263,4:DRAW 276,263:M
OVE 364,263:DRAW 388,263          (29C7)
3610 MOVE 252,265,1:DRAW 280,265:M
OVE 360,265:DRAW 388,265          (290E)
3620 MOVE 264,270,1:DRAW 284,270:M
OVE 356,270:DRAW 376,270          (2AB6)
3630 MOVE 268,272,2:DRAW 288,272:M
OVE 352,272:DRAW 372,272          (2A3A)
3640 MOVE 280,278,2:DRAW 296,278:M
OVE 344,278:DRAW 360,278          (2A65)
3650 MOVE 284,280,3:DRAW 296,280:M
OVE 344,280:DRAW 356,280          (2A03)
3660 MOVE 288,282,3:DRAW 300,282:M
OVE 340,282:DRAW 352,282          (2ABC)
3670 MOVE 292,284,4:DRAW 300,284:M
OVE 340,284:DRAW 348,284          (2A03)
3680 MOVE 296,288,4:DRAW 308,288:M
OVE 332,288:DRAW 344,288          (2A2E)
3690 MOVE 300,290,1:DRAW 308,290:M
OVE 332,290:DRAW 340,290          (2A93)
3700 MOVE 304,292,1:DRAW 312,292:M
OVE 328,292:DRAW 336,292          (2ADF)
3710 ' *****                      (0D28)
3720 MOVE 312,294,2:DRAW 320,300:M
OVE 328,294:DRAW 320,300          (2A37)
3730 MOVE 308,294,2:DRAW 320,300:M
OVE 332,294:DRAW 320,300          (2AEE)
3740 MOVE 86,96,7:DRAW 320,298:DRA
W 550,96:DRAW 86,96               (25E4)
3750 MOVE 0,126,7:DRAW 316,300:MOV
E 324,300:DRAW 640,126            (265D)
3760 ' *****                      (0D8C)
3770 MOVE 50,100:FILL 1:MOVE 590,1
00:FILL 1                         (19DA)
3780 MOVE 100,140:FILL 2:MOVE 540,
140:FILL 2                        (1986)
3790 MOVE 130,170:FILL 3:MOVE 510,
170:FILL 3                        (1930)
3800 MOVE 160,200:FILL 4:MOVE 480,
200:FILL 4                        (190B)
3810 MOVE 200,220:FILL 1:MOVE 440,
220:FILL 1                        (1931)
3820 MOVE 220,230:FILL 2:MOVE 420,
230:FILL 2                        (1936)
3830 MOVE 240,245:FILL 3:MOVE 400,
245:FILL 3                        (1998)
3840 MOVE 260,260:FILL 4:MOVE 380,
260:FILL 4                        (1CD2)
3850 MOVE 270,268:FILL 1:MOVE 370,
268:FILL 1                        (1C91)
3860 MOVE 285,274:FILL 2:MOVE 355,
274:FILL 2                        (1CE3)
3870 MOVE 300,286:FILL 4:MOVE 342,
286:FILL 4                        (1CC2)
3880 MOVE 320,100:FILL 7          (0FD0)
3890 MOVE 320,350:FILL 6          (1026)
3900 RETURN                       (06AE)
3910 ' *****                      (0DBA)
3920 ' ZEIGER AUFBAU              (1592)
3930 ' *****                      (0DE2)
3940 INK 0,12:INK 1,26:INK 2,26:IN
K 3,6:INK 4,6:INK 5,7:INK 6,2:INK
7,13:INK 8,8:INK 9,0:INK 10,4:INK
11,15:INK 12,24:INK 13,26:INK 14,1
```

```
3:INK 15,26                                    (633D)
3950 ORIGIN 0,0,0,640,0,400                    (15D0)
3960 ORIGIN 320,30:MOVE 0,0,4,1:DR
AW 68*SIN(252),44*COS(252)                     (2A9F)
3970 ORIGIN 468,20:MOVE 44*SIN(400
),44*COS(400),11,1:DRAW 58*SIN(400
),58*COS(400)                                  (3FF4)
3980 ORIGIN 580,20:MOVE 44*SIN(320
),44*COS(320),11,1:DRAW 58*SIN(320
),58*COS(320)                                  (3FFE)
3990 ORIGIN 0,0:TAG:MOVE 16,40,6,1
:PRINT CHR$(129);:TAGOFF                       (219F)
4000 PAPER#1,10:PEN#1,8,0:LOCATE#1
,3,2:PRINT#1,CHR$(170)CHR$(170)"."
CHR$(170)CHR$(170)                             (3B61)
4010 TAG:FOR i=458 TO 490 STEP 16:
MOVE i,22,1,1:PRINT CHR$(190);:NEX
T i:MOVE 506,22:PRINT CHR$(188);:F
OR i=522 TO 538 STEP 16:MOVE i,22,
1,1:PRINT CHR$(190);:NEXT i:TAGOFF
                                               (736A)
4020 TAG:MOVE 80,24,8,1:PRINT CHR$
(143);:TAGOFF                                  (1C39)
4030 ORIGIN 0,112:TAG:MOVE 320-32,
0,2,1:PRINT CHR$(202)CHR$(203);:MO
VE 320-96,0,14,1:PRINT CHR$(204)CH
R$(205);:MOVE 320+32,0:PRINT CHR$(
206)CHR$(207);:TAGOFF                          (62A8)
4040 ORIGIN 0,0:PLOT 320,10,4,1:PL
OT 324,10,4,1                                  (2152)
4050 RESTORE 4760                              (099A)
4060 IF INKEY(47)<>0 AND JOY(0)<>1
6 THEN 4060                                    (1BBF)
4070 ' *****                                   (0DF9)
4080 ' SCHLEIFE                                (1008)
4090 ' *****                                   (0D22)
4100 v=0:w=0:g=1:t=0:b=40:h=0:s=0:
s0=400:x=0:a=0:ue=100:lr=1/50000:a
0=25:bv=0.2:hc=2/15000:kc=2/45000:
ab=10:t=TIME/300:t0=t:ws=0:i(1)=26
:i(2)=26:i(3)=6:i(4)=6:dts=32000               (F46A)
4110 lk=1:lrk=0.5:hrk=10000:kw=0.2
:lp=0:le=0:le1=0:l=0:q1=3                      (5A8E)
4120 v1=v:w1=w:g1=g:t1=t:b1=b:h1=h
:s1=s:lp1=lp                                   (5E53)
4130 IF dts>s0/(v+0.0125)+1.5 THEN
 dts=s0/(v+0.0125)+1.5:AFTER dts G
OSUB 4560                                      (5506)
4140 a=0:x=0                                   (1241)
4150 dt=MIN(TIME/300-t,1.8):t=TIME
/300                                           (2D60)
4160 IF INKEY(0)=0 OR JOY(0)=1 THE
N x=1:a=a0*2^(1-g)*(15*w/20000-w*w
/6250000+0.25)*(1-ABS((75-h)/150))
:b=b-bv*dt                                     (8325)
4170 IF INKEY(2)=0 OR JOY(0)=2 THE
N x=-1:a=-ab                                   (2973)
4180 IF INKEY(64)=0 AND v<50 THEN
g=1:GOSUB 4520                                 (22F3)
4190 IF INKEY(65)=0 AND v<100 THEN
 g=2:GOSUB 4520                                (22B4)
4200 IF INKEY(57)=0 AND v<200 THEN
 g=3:GOSUB 4520                                (2203)
4210 IF INKEY(56)=0 THEN g=4:GOSUB
 4520                                          (1AB0)
4220 IF INKEY(8)=0 OR JOY(0)=4 THE
N le=le-lk*dt                                  (2DE2)
4230 IF INKEY(1)=0 OR JOY(0)=8 THE
N le=le+lk*dt                                  (2DB9)
4240 le=le*(1-lrk*dt)                          (2196)
4250 IF ABS(le*v*v)>hrk THEN le=SG
N(le)*hrk/v/v:SOUND 1,40,dt*100                (50D3)
4260 l=l+(le-kw*(q1-3))*dt*v/100               (34A9)
4270 a=a-lr*v*v                                (1E4A)
4280 v=v+a*dt                                  (195C)
4290 IF v<0 THEN v=0                           (13B5)
4300 IF v>245 THEN v=245                       (155F)
4310 s=s+v*dt/3600                             (1D4D)
4320 w=v*2^(1-g)*ue:IF w>5000 THEN
 w=5000                                        (3263)
4330 h=h*(1-v*kc)+hc*w*(x+1)                   (3114)
4340 IF b<0 THEN GOTO 6060                     (113B)
4350 IF h>100 THEN GOTO 6060                   (123F)
4360 ORIGIN 320,30:MOVE 0,0,4,1:DR
AW 68*SIN(252+v1*0.84),44*COS(252+
v1*0.84)                                       (4467)
4370 MOVE 0,0,4,1:DRAW 68*SIN(252+
v*0.84),44*COS(252+v*0.84)                     (3A84)
4380 ORIGIN 320,-300:MOVE 60*le1,S
QR(96100-3600*le1*le1),4,1:DRAW 60
*le1+4,SQR(96100-3600*le1*le1),4,1
:MOVE 60*le,SQR(96100-3600*le*le),
4,1:DRAW 60*le+4,SQR(96100-3600*le
*le),4,1:le1=le                                (C9B3)
4390 ORIGIN 468,20:sb=SIN(320+b1*2
):cb=COS(320+b1*2):MOVE 44*sb,44*c
b,11,1:DRAW 58*sb,58*cb                        (6333)
4400 sb=SIN(320+b*2):cb=COS(320+b*
2):MOVE 44*sb,44*cb,11,1:DRAW 58*s
b,58*cb                                        (5997)
4410 IF h>50 THEN ORIGIN 580,20:sh
=SIN(240+h1*1.6):ch=COS(240+h1*1.6
):MOVE 44*sh,44*ch,11,1:DRAW 58*sh
,58*ch:sh=SIN(240+h*1.6):ch=COS(24
0+h*1.6):MOVE 44*sh,44*ch,11,1:DRA
W 58*sh,58*ch                                  (D11C)
4420 ORIGIN 0,0:TAG:MOVE 16+w1*0.0
36,40,6,1:PRINT CHR$(129);:MOVE 16
+w*0.036,40:PRINT CHR$(129);:TAGOF
F                                              (4A1B)
4430 t1=t-t0:tm=INT(t1/60):ts=INT(
t1)MOD 60:PAPER#1,10:PEN#1,8,0:LOC
ATE#1,3,2:PRINT#1,CHR$(INT(tm/10)+
170)CHR$((tm MOD 10)+170)"."CHR$(I
NT(ts/10)+170)CHR$((INT(ts)MOD 10)
+170)                                          (A63F)
```

```
4440 TAG:FOR i=3 TO 1 STEP-1:MOVE(
i*-1+4)*16+442,22,1,1:PRINT CHR$(1
90+INT(s1/10^(i-1))MOD 10);:MOVE(i
*-1+4)*16+442,22:PRINT CHR$(190+IN
T(s/10^(i-1))MOD 10);:NEXT i        <8EC1>
4450 FOR i=2 TO 1 STEP-1:MOVE(i*-1
+3)*16+506,22,1,1:PRINT CHR$(190+I
NT(s1/10^(i-3))MOD 10);:MOVE(i*-1+
3)*16+506,22:PRINT CHR$(190+INT(s/
10^(i-3))MOD 10);:NEXT i            <8C9F>
4460 lp=(l-kw)*dt*v+lp              <28D2>
4470 llp=lp/20:llp1=lp1/20:IF llp>
138 OR llp<-138 THEN GOTO 6060      <41AA>
4480 ORIGIN 0,112:MOVE 320-96+llp1
,0,14,1:PRINT CHR$(204)CHR$(205);:
MOVE 320+32+llp1,0:PRINT CHR$(206)
CHR$(207);:MOVE 320-32+llp1,0,2,1:
PRINT CHR$(202)CHR$(203);           <762F>
4490 MOVE 320-32+llp,0,2,1:PRINT C
HR$(202)CHR$(203);:MOVE 320-96+llp
,0,14,1:PRINT CHR$(204)CHR$(205);:
MOVE 320+32+llp,0:PRINT CHR$(206)C
HR$(207);:TAGOFF                    <6F00>
4500 ws=ws+v*dt/3600:IF ws>0.4 THE
N ws=0:GOSUB 4630                   <3A6A>
4510 GOTO 4120                      <0943>
4520 ORIGIN 0,0:TAG:MOVE 44+g1*36,
24,8,1:PRINT CHR$(143);:MOVE 44+g*
36,24:PRINT CHR$(143);:TAGOFF:RETU
RN                                  <44F9>
4530 ' *****                        <0D93>
4540 ' RANDWECHSEL                  <130D>
4550 ' *****                        <0DBB>
4560 d=i(1):i(1)=i(2):i(2)=i(3):i(
3)=i(4):i(4)=d                      <4E26>
4570 INK 1,i(1):INK 2,i(2):INK 3,i
(3):INK 4,i(4)                      <30C1>
4580 dts=s0/(v+0.0125)+1.5:AFTER d
ts GOSUB 4560                       <322D>
4590 RETURN                         <0614>
4600 ' *****                        <0D20>
4610 ' BILDWECHSEL                  <133E>
4620 ' *****                        <0D49>
4630 q2=q1                          <10AC>
4640 READ q1                        <0BC1>
4650 IF q1=q2 THEN 4740             <156C>
4660 IF q1=10 THEN q1=q2:GOTO 6210
                                    <1F9E>
4670 GOSUB 5950                     <09AE>
4680 DI                             <0611>
4690 FOR j=1 TO 4:INK j,13:NEXT j   <1D81>
4700 |SCREENSWAP,1,q3(q1)           <20E5>
4710 EI                             <0650>
4720 GOSUB 5950                     <0913>
4730 INK 1,i(1):INK 2,i(2):INK 3,i
(3):INK 4,i(4)                      <3003>
4740 q3(q2)=q3(q1)                  <1E1C>
4750 RETURN                         <0654>
4760 DATA 4,5,4,3,2,1,2,3,2,3,4,5,
4,3,2,3,4,3,4,5,4,3,10              <3515>
4770 RETURN                         <067C>
4780 ' *****                        <0D88>
4790 ' KREISBOGEN                   <1208>
4800 ' *****                        <0DB0>
4810 FOR i=1 TO 2                   <0EE2>
4820 FOR j=1 TO 6:READ a(j):NEXT j
                                    <20FF>
4830 ORIGIN a(1),a(2)               <1531>
4840 MOVE a(3)*SIN(a(4)),a(3)*COS(
a(4)),a(6)                          <3511>
4850 FOR j=a(4)TO a(5)STEP 5        <1C23>
4860 DRAW a(3)*SIN(j),a(3)*COS(j)   <2736>
4870 NEXT j,i                       <0F1E>
4880 ORIGIN 0,0                     <09BC>
4890 RETURN                         <066E>
4900 ' *****                        <0D7A>
4910 ' ANZEIGE AUFBAU               <1674>
4920 ' *****                        <0DA2>
4930 MOVE 0,2,12:DRAW 0,94:DRAW 63
6,94:DRAW 636,2:DRAW 0,2            <26E7>
4940 ORIGIN 320,30                  <0CA3>
4950 FOR i=1 TO 2                   <0EFC>
4960 MOVE 0,64+i,9                  <118D>
4970 FOR j=0 TO 360 STEP 5          <12B7>
4980 DRAW(88+i)*SIN(j),(64+i)*COS(
j)                                  <2BFC>
4990 NEXT j,i                       <0F0E>
5000 MOVE 0,0:FILL 11               <0D89>
5010 MOVE 0,4,9                     <0B55>
5020 FOR j=0 TO 360 STEP 5          <121C>
5030 DRAW 5*SIN(j),4*COS(j)         <1B7A>
5040 NEXT j                         <0AE7>
5050 MOVE 0,0:FILL 9                <0C1C>
5060 FOR i=64 TO 60 STEP-2          <13EB>
5070 MOVE 0,i,9                     <0E7B>
5080 FOR j=0 TO 360 STEP 36         <1323>
5090 PLOT(i+24)*SIN(j),i*COS(j)     <2651>
5100 NEXT j,i                       <0FE9>
5110 FOR i=64 TO 52 STEP-2          <134F>
5120 MOVE(i+24),0,7                 <130D>
5130 FOR j=90 TO 450 STEP 36        <14FB>
5140 PLOT(i+24)*SIN(j),i*COS(j)     <26B6>
5150 NEXT j,i                       <0F4F>
5160 PEN#1,9,1:LOCATE#1,10,5:PRINT
#1,CHR$(167)CHR$(168)CHR$(169)      <2C77>
5170 TAG                            <06BB>
5180 MOVE-71,4,2,1:PRINT CHR$(160)
;                                   <18BE>
5190 MOVE-59,28,2,1:PRINT CHR$(161
);                                  <19C2>
5200 MOVE-32,43,2,1:PRINT CHR$(162
);                                  <196D>
5210 MOVE 8,43,2,1:PRINT CHR$(163)
;                                   <1763>
5220 MOVE 26,28,2,1:PRINT CHR$(164
);                                  <185A>
```

```
5230 MOVE 43,4,2,1:PRINT CHR$(165)
;                                   (174A)
5240 ORIGIN 320,-300                (0E8D)
5250 MOVE-SQR(315^2-300^2),300,9,0
                                    (1E6A)
5260 FOR j=-SQR(315^2-300^2)TO SQR
(315^2-300^2)STEP 10:DRAW j,SQR(31
5^2-j^2):NEXT j                    (4B3B)
5270 MOVE-SQR(305^2-300^2),300,10:
FOR j=-SQR(305^2-300^2)TO SQR(305^
2-300^2)STEP 5:DRAW j,SQR(305^2-j^
2):NEXT j                           (63E5)
5280 MOVE 0,310:FILL 9:MOVE-90,301
:FILL 9:MOVE 90,301:FILL 9          (25F5)
5290 MOVE-SQR(314^2-300^2),300,7:F
OR j=-SQR(314^2-300^2)TO SQR(314^2
-300^2)STEP 5:DRAW j,SQR(314^2-j^2
):NEXT j                            (62D5)
5300 TAGOFF:ORIGIN 0,0              (0B16)
5310 MOVE 12,90,10,0:DRAW 232,90:D
RAW 224,54:DRAW 4,54:DRAW 12,90     (2B13)
5320 MOVE 16,88,8,0:DRAW 228,88:DR
AW 220,56:DRAW 8,56:DRAW 16,88      (2A98)
5330 MOVE 200,70:FILL 10            (0F4A)
5340 PEN#1,7,1:LOCATE#1,1,2:PRINT#
1,CHR$(180)CHR$(181)                (2555)
5350 FOR i=468 TO 580 STEP 112      (154B)
5360 ORIGIN i,20                    (0D6B)
5370 MOVE 70*SIN(315),70*COS(315),
5,0                                 (1F24)
5380 FOR j=315 TO 405 STEP 5        (14D8)
5390 DRAW 70*SIN(j),70*COS(j)       (1D95)
5400 NEXT j                         (0ABA)
5410 MOVE 40*SIN(315),40*COS(315)   (1B1C)
5420 FOR j=315 TO 405 STEP 5        (1427)
5430 DRAW 40*SIN(j),40*COS(j)       (1DA5)
5440 NEXT j                         (0A0A)
5450 MOVE 70*SIN(315),70*COS(315):
DRAW 40*SIN(315),40*COS(315)        (329A)
5460 MOVE 70*SIN(45),70*COS(45):DR
AW 40*SIN(45),40*COS(45)            (2ED8)
5470 ORIGIN 0,0:MOVE i,65:FILL 12   (160A)
5480 NEXT i                         (0A55)
5490 ORIGIN 468,20                  (0CF0)
5500 FOR i=345 TO 390 STEP 15       (1532)
5510 MOVE 66*SIN(i),66*COS(i),10:D
RAW 62*SIN(i),62*COS(i)             (3961)
5520 NEXT i                         (0AA5)
5530 MOVE 70*SIN(330),70*COS(330),
5:DRAW 40*SIN(330),40*COS(330)      (3437)
5540 MOVE 50*SIN(320),50*COS(320):
FILL 5                              (1E9F)
5550 ORIGIN 580,20                  (0C63)
5560 FOR i=333 TO 369 STEP 18       (15C3)
5570 MOVE 66*SIN(i),66*COS(i),10:D
RAW 62*SIN(i),62*COS(i)             (39D9)
5580 NEXT i                         (0A1E)
5590 MOVE 70*SIN(387),70*COS(387),
5:DRAW 40*SIN(387),40*COS(387)      (34AF)
5600 MOVE 50*SIN(400),50*COS(400):
FILL 5                              (1E76)
5610 ORIGIN 0,0:TAG                 (0B56)
5620 MOVE 442,44,11,0:PRINT CHR$(1
43)CHR$(143);:MOVE 444,46,2,1:PRIN
T CHR$(182)CHR$(183);               (3B16)
5630 MOVE 554,44,11,0:PRINT CHR$(1
43)CHR$(143);:MOVE 556,46,2,1:PRIN
T CHR$(184)CHR$(185);               (3B2C)
5640 TAGOFF                         (066D)
5650 MOVE 8,8,11,0:DRAW 220,8       (149F)
5660 MOVE 8,26,11:DRAW 220,26       (1474)
5670 MOVE 8,8,11:DRAW 8,26          (127A)
5680 MOVE 220,8:DRAW 220,26         (1196)
5690 MOVE 100,16:FILL 6             (0E7E)
5700 MOVE 184,8,11:DRAW 184,26      (1474)
5710 MOVE 148,8,11:DRAW 148,26      (1414)
5720 MOVE 112,8,11:DRAW 112,26      (14B3)
5730 MOVE 76,8,11:DRAW 76,26        (1438)
5740 TAG                            (0630)
5750 MOVE 12,24,13,1:PRINT CHR$(18
6)CHR$(187);                        (1F75)
5760 MOVE 80,24,11:PRINT"1";:MOVE
116,24:PRINT"2";:MOVE 152,24:PRINT
"3";:MOVE 188,24:PRINT"4";          (3B90)
5770 TAGOFF                         (0670)
5780 MOVE 438,24,13,0:DRAW 616,24   (195E)
5790 MOVE 438,8,13,0:DRAW 616,8     (17FD)
5800 MOVE 438,24,13,0:DRAW 438,8    (1857)
5810 MOVE 616,24,13,0:DRAW 616,8    (18D7)
5820 MOVE 500,20:FILL 8             (0F82)
5830 TAG                            (06E4)
5840 MOVE 568,22,1,1:PRINT CHR$(18
9);                                 (1984)
5850 TAGOFF                         (0610)
5860 MOVE 12,42,5,0:DRAW 216,42:DR
AW 216,32:DRAW 12,32:DRAW 12,42     (2B88)
5870 MOVE 82,42,5,0:DRAW 82,52:DRA
W 146,52:DRAW 146,42:MOVE 114,48:F
ILL 5                               (2ECA)
5880 TAG:MOVE 86,50,9,1:PRINT CHR$
(200)CHR$(201);                     (2072)
5890 PEN,0:TAGOFF:MOVE 0,0,,0       (1293)
5900 ORIGIN 0,0,0,640,400,96        (1639)
5910 RETURN                         (066A)
5920 ' *****                        (0D76)
5930 ' ZEIGER                       (0E8C)
5940 ' *****                        (0D9E)
5950 ORIGIN 320,30:MOVE 0,0,4,1:DR
AW 68*SIN(252+v*0.84),44*COS(252+v
*0.84)                              (4285)
5960 ORIGIN 468,20:sb=SIN(320+b*2)
:cb=COS(320+b*2):MOVE 44*sb,44*cb,
11,1:DRAW 58*sb,58*cb               (6114)
5970 IF h>50 THEN ORIGIN 580,20:sh
=SIN(240+h*1.6):ch=COS(240+h*1.6):
MOVE 44*sh,44*ch,11,1:DRAW 58*sh,5
```

```
8*ch ELSE ORIGIN 580,20:MOVE 44*SI
N(320),44*COS(320),11,1:DRAW 58*SI
N(320),58*COS(320)                    <AE38>
5980 ORIGIN 0,0:TAG:MOVE 16+w*0.03
6,40,6,1:PRINT CHR$(129);:TAGOFF      <2D32>
5990 PAPER#1,10:PEN#1,8,0:LOCATE#1
,3,2:PRINT#1,"      "                 <27AB>
6000 TAG:FOR j=3 TO 1 STEP-1:MOVE(
j*-1+4)*16+442,22,1,1:PRINT CHR$(1
90+INT(s/10^(j-1))MOD 10);:NEXT j:
MOVE 506,22:PRINT CHR$(188);          <6683>
6010 FOR j=2 TO 1 STEP-1:MOVE(j*-1
+3)*16+506,22,1,1:PRINT CHR$(190+I
NT(s/10^(j-3))MOD 10);:NEXT j         <5362>
6020 llp=lp/20:ORIGIN 0,112:MOVE 3
20-32+llp,0,2,1:PRINT CHR$(202)CHR
$(203);:MOVE 320-96+llp,0,14,1:PRI
NT CHR$(204)CHR$(205);:MOVE 320+32
+llp,0:PRINT CHR$(206)CHR$(207);:T
AGOFF                                 <8519>
6030 ORIGIN 320,-300:MOVE 60*le1,S
QR(96100-3600*le1*le1),4,1:DRAW 60
*le1+4,SQR(96100-3600*le1*le1),4,1
                                      <688B>
6040 ORIGIN 0,0:TAG:MOVE 44+g*36,2
4,8,1:PRINT CHR$(143);:TAGOFF         <297E>
6050 RETURN                           <0681>
6060 ORIGIN 0,0:WINDOW#2,6,15,10,1
2:PAPER#2,9:CLS#2                     <2211>
6070 MOVE 160,208,11,0:DRAW 480,20
8:DRAW 480,256:DRAW 160,256:DRAW 1
60,208                                <3086>
6080 TAG:MOVE 180,238,2,1:PRINT"GA
ME OVER";:TAGOFF                      <2156>
6090 IF INKEY(47)<>0 THEN 6090        <121E>
6100 AFTER 1 GOSUB 6200               <0BBA>
6110 GOTO 6720                        <092F>
6120 FOR j=0 TO 15:INK j,0:NEXT j     <1D0E>
6130 GOSUB 5920                       <092E>
6140 ORIGIN 0,0,0,640,400,96:MOVE
0,0,0,0                               <1F0B>
6150 ON q1 GOSUB 1380,760,3400,208
0,2710                                <1F19>
6160 IF q1<>3 THEN/SCREENSWAP,1,q3
(3)                                   <25E2>
6170 q4=q3(3)                         <13BF>
6180 q3(3)=1:q3(q1)=q4                <22D9>
6190 GOTO 3910                        <09D0>
6200 RETURN                           <06AF>
6210 ORIGIN 0,0:WINDOW#2,8,13,10,1
2:PAPER#2,9:CLS#2                     <22A6>
6220 MOVE 224,208,11,0:DRAW 416,20
8:DRAW 416,256:DRAW 224,256:DRAW 2
24,208                                <30E2>
6230 TAG:MOVE 246,238,2,1:PRINT"RE
ADY";:TAGOFF                          <1D4D>
6240 IF INKEY(47)<>0 THEN 6240        <12F8>
6250 ' *****                          <0D0B>
6260 ' EINTRAGUNGSLISTE               <183F>
6270 ' *****                          <0D32>
6280 AFTER 1 GOSUB 6200               <0B23>
6290 IF einm(35)*60+eins(35)<tm*60
+ts THEN 6720                         <3380>
6300 TAG:ORIGIN 0,0                   <0B7B>
6310 PAPER 9:CLS:RESTORE 7010         <0E3D>
6320 INK 0,1:INK 1,25:INK 2,24:INK
 3,15:INK 4,6:INK 5,7:INK 6,2:INK
7,13:INK 8,8:INK 9,0:INK 10,4:INK
11,15:INK 12,24:INK 13,26:INK 14,1
3:INK 15,26                           <639A>
6330 FOR j=1 TO 32                    <0F4F>
6340 READ x2(j),y2(j),ii2(j)          <2A32>
6350 NEXT j                           <0A29>
6360 FOR j=1 TO 26                    <0F73>
6370 MOVE x2(j),y2(j),14,1:PRINT C
HR$(143);:MOVE x2(j),y2(j),ii2(j),
1:PRINT CHR$(j+64);                   <612C>
6380 NEXT j                           <0A65>
6390 FOR j=1 TO 7                     <0E95>
6400 READ x3,y3,n3:MOVE x3,y3,4,1:
PRINT CHR$(n3);                       <349E>
6410 NEXT j                           <0AA2>
6420 FOR j=27 TO 32                   <10A6>
6430 MOVE x2(j),y2(j),14,1:PRINT C
HR$(143);:READ n2:MOVE x2(j),y2(j)
,ii2(j),1:PRINT CHR$(n2);             <6629>
6440 NEXT j                           <0ADE>
6450 TAGOFF                           <06C4>
6460 MOVE x2(1)-4,y2(1)+2,6,0:DRAW
 x2(1)+32,y2(1)+2:DRAW x2(1)+32,y2
(1)-16:DRAW x2(1)-4,y2(1)-16:DRAW
x2(1)-4,y2(1)+2                       <7F24>
6470 MOVE 376,232,6,0:DRAW 616,232
:DRAW 616,200:DRAW 376,200:DRAW 37
6,232:MOVE 400,220:FILL 0             <3B68>
6480 m2=1:m21=1:pp2=12:FOR k=1 TO
7:p$(k)="":NEXT k                     <3DFE>
6490 m21=m2                           <111D>
6500 IF INKEY(1)=0 OR JOY(0)=8 THE
N m2=m2+1:IF m2=33 THEN m2=1:GOTO
6540 ELSE GOTO 6540                   <4029>
6510 IF INKEY(8)=0 OR JOY(0)=4 THE
N m2=m2-1:IF m2=0 THEN m2=32:GOTO
6540 ELSE GOTO 6540                   <4099>
6520 IF INKEY(47)=0 OR JOY(0)=16 T
HEN PEN 6:PAPER 0:GOTO 6580           <2269>
6530 GOTO 6490                        <0941>
6540 MOVE x2(m21)-4,y2(m21)+2,9,0:
DRAW x2(m21)+32,y2(m21)+2:DRAW x2(
m21)+32,y2(m21)-16:DRAW x2(m21)-4,
y2(m21)-16:DRAW x2(m21)-4,y2(m21)+
2                                     <B12F>
6550 MOVE x2(m2)-4,y2(m2)+2,6,0:DR
AW x2(m2)+32,y2(m2)+2:DRAW x2(m2)+
32,y2(m2)-16:DRAW x2(m2)-4,y2(m2)-
16:DRAW x2(m2)-4,y2(m2)+2             <A721>
```

```
6560 FOR j=1 TO 100:NEXT j                    <1542>
6570 GOTO 6490                                <0991>
6580 IF m2<27 THEN IF pp2<19 THEN
pp2=pp2+1:LOCATE pp2,12:PRINT CHR$
(m2+64):p$(pp2-12)=CHR$(m2+64):FOR
 j=1 TO 150:NEXT j                            <7052>
6590 IF m2=27 THEN IF pp2<19 THEN
pp2=pp2+1:LOCATE pp2,12:PRINT CHR$
(228):p$(pp2-12)=CHR$(228):FOR j=1
 TO 150:NEXT j                                <6409>
6600 IF m2=28 THEN IF pp2<19 THEN
pp2=pp2+1:LOCATE pp2,12:PRINT CHR$
(42):p$(pp2-12)=CHR$(42):FOR j=1 T
O 150:NEXT j                                  <6450>
6610 IF m2=29 THEN IF pp2<19 THEN
pp2=pp2+1:LOCATE pp2,12:PRINT CHR$
(46):p$(pp2-12)=CHR$(46):FOR j=1 T
O 150:NEXT j                                  <64C4>
6620 IF m2=30 THEN IF pp2<19 THEN
pp2=pp2+1:p$(pp2-12)=CHR$(32):FOR
j=1 TO 150:NEXT j                             <5155>
6630 IF m2=31 THEN IF pp2>12 THEN
LOCATE pp2,12:PRINT CHR$(32):p$(pp
2-12)=CHR$(32):pp2=pp2-1:FOR j=1 T
O 150:NEXT                                    <60E8>
6640 IF m2=32 THEN GOTO 6660                  <137F>
6650 GOTO 6490                                <0932>
6660 PAPER 9:CLS                              <091D>
6670 FOR j=35 TO 0 STEP-1                     <12AE>
6680 IF einm(j)*60+eins(j)>tm*60+t
s THEN 6690 ELSE 6700                         <3C0C>
6690 NEXT j                                   <0AD3>
6700 FOR k=34 TO j+1 STEP-1:einm(k
+1)=einm(k):eins(k+1)=eins(k):ein$
(k+1)=ein$(k):NEXT k                          <7535>
6710 einm(j+1)=tm:eins(j+1)=ts:ein
$(j+1)="":FOR k=1 TO 7:ein$(j+1)=e
in$(j+1)+p$(k):NEXT k                         <7B7D>
6720 ' *****                                  <0DB9>
6730 ' TOP 35                                 <0EC4>
6740 ' *****                                  <0DE1>
6750 PAPER 9:CLS                              <09D1>
6760 INK 0,16:INK 1,19:INK 2,18:IN
K 3,9:INK 4,6:INK 5,7:INK 6,2:INK
7,13:INK 8,8:INK 9,0:INK 10,4:INK
11,15:INK 12,24:INK 13,26:INK 14,1
3:INK 15,26                                   <635F>
6770 RESTORE 7410:PEN 0,0:FOR k=3
TO 5:FOR kk=5 TO 14:READ chr:LOCAT
E kk,k:PRINT CHR$(chr):NEXT kk,k              <5085>
6780 z1=0:z2=0                                <1453>
6790 TAG:ORIGIN 0,0                           <0B51>
6800 FOR k=1 TO 7:MOVE k*24-16,((k
*-1)+8)*25+120,11,1:PRINT USING"##
";k;                                          <40FC>
6810 MOVE k*24+44,((k*-1)+8)*25+12
0,10:PRINT CHR$(41);                          <3035>
6820 MOVE k*24+88,((k*-1)+8)*25+12
0,13:PRINT ein$(k);                           <36F2>
6830 MOVE k*24+322,((k*-1)+8)*25+1
20,10:PRINT USING"##";einm(k);                <3E65>
6840 MOVE k*24+374,((k*-1)+8)*25+1
20,8:PRINT CHR$(46);                          <30C4>
6850 MOVE k*24+398,((k*-1)+8)*25+1
20,11:PRINT USING"##";eins(k);:IF
eins(k)<10 THEN MOVE k*24+398,((k*
-1)+8)*25+120,11:PRINT CHR$(48);              <7D5F>
6860 NEXT k:TAGOFF                            <0CE2>
6870 IF INKEY(0)=0 OR JOY(0)=1 THE
N IF z2>0 THEN z1=z2:z2=z2-7:GOTO
6920                                          <3D42>
6880 IF INKEY(2)=0 OR JOY(0)=2 THE
N IF z2<28 THEN z1=z2:z2=z2+7:GOTO
 6920                                         <3EA5>
6890 IF INKEY(47)=0 OR JOY(0)=16 T
HEN GOTO 6990                                 <1C9B>
6900 IF INKEY(58)=0 THEN GOTO 7440
                                              <13F5>
6910 GOTO 6870                                <0923>
6920 TAG:FOR k=1 TO 7:MOVE k*24-16
,((k*-1)+8)*25+120,11,1:PRINT USIN
G"##";k+z1;:MOVE k*24-16,((k*-1)+8
)*25+120,11,1:PRINT USING"##";k+z2
;                                             <80F3>
6930 MOVE k*24+88,((k*-1)+8)*25+12
0,13:PRINT ein$(k+z1);:MOVE k*24+8
8,((k*-1)+8)*25+120,13:PRINT ein$(
k+z2);                                        <74D8>
6940 MOVE k*24+322,((k*-1)+8)*25+1
20,10:PRINT USING"##";einm(k+z1);:
MOVE k*24+322,((k*-1)+8)*25+120,10
:PRINT USING"##";einm(k+z2);                  <8423>
6950 MOVE k*24+398,((k*-1)+8)*25+1
20,11:PRINT USING"##";eins(k+z1);:
IF eins(k+z1)<10 THEN MOVE k*24+39
8,((k*-1)+8)*25+120,11:PRINT CHR$(
48);                                          <8939>
6960 MOVE k*24+398,((k*-1)+8)*25+1
20,11:PRINT USING"##";eins(k+z2);:
IF eins(k+z2)<10 THEN MOVE k*24+39
8,((k*-1)+8)*25+120,11:PRINT CHR$(
48);                                          <891A>
6970 NEXT k                                   <0A08>
6980 GOTO 6870                                <09B0>
6990 PAPER 0                                  <07C5>
7000 GOTO 6120                                <095B>
7010 DATA 32,168,6                            <0FA1>
7020 DATA 56,192,5                            <0F48>
7030 DATA 76,216,4                            <0FED>
7040 DATA 84,240,3                            <0FFD>
7050 DATA 92,264,2                            <0F8C>
7060 DATA 104,288,3                           <1077>
7070 DATA 112,312,4                           <1094>
7080 DATA 120,336,5                           <102B>
7090 DATA 128,360,6                           <1087>
7100 DATA 144,384,5                           <1017>
```

```
7110 DATA 172,364,4                    (10F3)
7120 DATA 180,340,3                    (107F)
7130 DATA 184,316,2                    (1093)
7140 DATA 184,292,3                    (1028)
7150 DATA 200,268,4                    (10EF)
7160 DATA 228,288,5                    (109C)
7170 DATA 240,312,6                    (10AC)
7180 DATA 252,336,5                    (1045)
7190 DATA 264,360,4                    (1051)
7200 DATA 292,380,3                    (108E)
7210 DATA 312,360,2                    (1069)
7220 DATA 316,336,3                    (108D)
7230 DATA 316,312,4                    (1021)
7240 DATA 312,288,5                    (1030)
7250 DATA 308,264,6                    (10D5)
7260 DATA 300,240,5                    (1008)
7270 DATA 292,216,4                    (106C)
7280 DATA 292,192,3                    (10EC)
7290 DATA 296,168,2                    (10F8)
7300 DATA 316,148,3                    (10E9)
7310 DATA 352,156,4                    (1000)
7320 DATA 372,176,5                    (1045)
7330 DATA 368,316,79                   (11BD)
7340 DATA 400,320,84                   (116D)
7350 DATA 432,328,79                   (1198)
7360 DATA 464,336,82                   (1168)
7370 DATA 496,340,87                   (115C)
7380 DATA 528,336,65                   (1188)
7390 DATA 560,332,89                   (113C)
7400 DATA 228,42,46,32,242,208         (1BB1)
7410 DATA 158,150,156,150,156,32,1
46,156,150,152                         (2DA5)
7420 DATA 149,149,149,151,153,32,3
2,157,147,156                          (2C71)
7430 DATA 145,147,153,145,32,32,14
6,153,146,153                          (2CA7)
7440 |ERA,"motorway.dat"               (1908)
7450 OPENOUT"motorway.dat":FOR k=1
 TO 35:WRITE#9,einm(k),eins(k),ein
$(k):NEXT k:CLOSEOUT                   (545E)
7460 GOSUB 7480:END                    (0BEF)
7470 GOSUB 7480:PRINT"Fehler ";ERR
;" in Zeile ";ERL:END                  (28B3)
7480 MODE 2:PEN 1:PAPER 0:INK 1,24
:INK 0,1:BORDER 1:CLEAR INPUT:RETU
RN                                     (2003)
```

```
10 '****************************    (2397)
11 '*MOTORDATERZEUGUNGSPROGRAMM*    (235C)
12 '*           VON              *    (2351)
13 '*      HOLGER HAUBER         *    (2365)
14 '*          FUER              *    (236E)
15 '*    SCHNEIDER CPC-WELT      *    (23E3)
20 '*        CPC 6128         JE*    (2399)
30 '****************************    (23BF)
40 '                                   (075B)
50 OPENOUT"motordat"                   (1015)
60 FOR i=1 TO 14                       (0F73)
70 READ a,b,c,d,e,f,g,h,i              (32EF)
80 WRITE#9,a,b,c,d,e,f,g,h,i           (35A0)
90 NEXT i                              (0A24)
100 CLOSEOUT                           (06EE)
110 DATA 205,230,68,68,92,92,68,68
,230                                   (24F4)
120 DATA 206,24,122,66,66,66,66,94
,24                                    (23A1)
130 DATA 207,112,32,32,32,32,34,46
,110                                   (24D7)
140 DATA 208,124,0,16,16,16,16,124
,124                                   (2496)
150 DATA 209,46,110,64,60,6,2,118,
116                                    (23A2)
160 DATA 210,118,54,32,32,44,32,32
,112                                   (24A8)
170 DATA 211,124,124,0,16,16,16,16
,56                                    (2390)
180 DATA 212,60,54,112,96,60,2,66,
60                                     (224D)
190 DATA 213,0,0,60,50,112,96,98,6
0                                      (210C)
200 DATA 214,0,0,60,50,114,98,98,6
0                                      (2124)
210 DATA 215,24,24,62,24,48,48,54,
28                                     (2220)
220 DATA 216,24,0,24,24,56,48,54,2
8                                      (213E)
230 DATA 217,0,0,62,48,48,28,2,124
                                       (20A3)
240 DATA 218,14,27,24,24,56,48,124
,48                                    (231C)
```

```
10 '****************************    (2397)
11 '*    ERZEUGUNGSPROGRAMM     *    (23C7)
12 '*          FUER             *    (236A)
13 '*       MOTORWAY.DAT        *    (2305)
14 '*          VON              *    (2391)
15 '*       HOLGER HAUBE        *    (23B0)
16 '*    SCHNEIDER CPC-WELT     *    (23E5)
20 '*        CPC 6128         JE*    (23AE)
30 '****************************    (23BF)
40 '                                   (075B)
50 OPENOUT"motorway.dat"               (14C9)
60 READ einm,eins,ein$                 (1CC2)
70 FOR i=1 TO 35                       (0FDB)
80 WRITE#9,einm,eins,ein$              (1FFC)
90 NEXT i                              (0A24)
100 CLOSEOUT                           (06EE)
110 END                                (063F)
120 DATA 19,59,"......."               (168F)
```

```
10 '*****************************  (2397)
11 '*MOTORBILERZEUGUNGSPROGRAMM*  (235C)
12 '*           VON             *  (238D)
13 '*      HOLGER HAUBE         *  (2311)
14 '*           FUER            *  (236E)
15 '*     SCHNEIDER AKTIV       *  (239A)
20 '*         CPC 6128        JE*  (2399)
30 '*****************************  (23BF)
40 '                               (075B)
50 MODE 0:INK 0,1:INK 12,0:BORDER
0:PAPER 0:PEN 1                    (1BE6)
60 SYMBOL AFTER 32                 (09B3)
70 SYMBOL 223,0,0,102,0,0,102,0,0  (1A02)
80 SYMBOL 224,0,0,24,0,0,102,0,0   (1A23)
90 SYMBOL 225,0,0,96,0,0,96,0,0    (1A80)
100 SYMBOL 226,0,0,60,0,0,24,0,0   (1A9E)
110 SYMBOL 227,0,0,62,0,0,96,0,0   (1A07)
120 SYMBOL 228,0,0,24,0,0,102,0,0  (1AB3)
130 SYMBOL 229,0,0,126,0,0,96,0,0  (1AA4)
140 SYMBOL 230,0,0,126,0,0,24,0,0  (1AB6)
150 SYMBOL 231,102,0,0,102,0,0,0,0
                                   (1A46)
160 SYMBOL 232,102,0,0,24,0,0,0,0  (1AC8)
170 SYMBOL 233,96,0,0,126,0,0,0,0  (1A01)
180 SYMBOL 234,24,0,0,60,0,0,0,0   (1A89)
190 SYMBOL 235,6,0,0,124,0,0,0,0   (19B0)
200 SYMBOL 236,102,0,0,24,0,0,0,0  (1AE3)
210 SYMBOL 237,96,0,0,96,0,0,0,0   (1A1B)
220 SYMBOL 238,24,0,0,24,0,0,0,0   (1A8E)
230 SYMBOL 239,0,0,0,102,102,0,126
,126                               (1C04)
240 SYMBOL 240,0,0,0,60,102,0,102,
102                                (1C0A)
250 SYMBOL 241,0,0,0,96,96,0,96,96
                                   (1C89)
260 SYMBOL 242,0,0,0,60,24,0,24,24
                                   (1C3D)
270 SYMBOL 243,0,0,0,126,96,0,124,
62                                 (1C6F)
280 SYMBOL 244,0,0,0,60,102,0,102,
102                                (1C7E)
290 SYMBOL 245,0,0,0,126,96,0,124,
124                                (1C0F)
300 SYMBOL 246,0,0,0,126,24,0,24,2
4                                  (1C4F)
310 SYMBOL 247,0,102,102,0,0,0,0,0
                                   (1A2C)
320 SYMBOL 248,0,102,60,0,0,0,0,0  (1ACD)
330 SYMBOL 249,0,96,126,0,0,0,0,0  (1ACE)
340 SYMBOL 250,0,24,60,0,0,0,0,0   (1AEA)
350 SYMBOL 251,0,6,126,0,0,0,0,0   (1945)
360 SYMBOL 252,0,102,60,0,0,0,0,0  (1A63)
370 SYMBOL 253,0,96,96,0,0,0,0,0   (1A4B)
380 SYMBOL 254,0,24,24,0,0,0,0,0   (1AC8)
390 ORIGIN-100,-500,0,640,340,50   (1BF5)
400 DEG                            (0668)
410 r=1                            (0B8D)
420 FOR i=5 TO 60 STEP 0.8         (163C)
430 MOVE 0,0,r:DRAW 1200*SIN(i-1),
1200*COS(i-0.8)                    (32CE)
440 MOVE 0,0,r:DRAW 1200*SIN(i),12
00*COS(i)                          (2981)
450 MOVE 855*SIN(i)-8,855*COS(i):F
ILL r                              (2753)
460 r=r+1:IF r>11 THEN r=1         (2035)
470 NEXT i                         (0A1E)
480 ORIGIN 0,0                     (094A)
490 FOR i=1 TO 61                  (0F8C)
500 READ x,y,x1,y1                 (1B77)
510 MOVE x,y-20,12,0:DRAW x1,y1-20
                                   (2759)
520 NEXT i                         (0A83)
530 DATA   4,120, 32,304 , 32,304,
 64,304 , 64,304, 72,248 , 72,248,
 84,300                            (4997)
540 DATA  84,300,112,300 ,112,300,
124,116 ,124,116, 92,120 , 92,120,
 92,248                            (4917)
550 DATA  92,248, 84,180 , 84,180,
 60,184 , 60,184, 52,248 , 52,248,
 40,120                            (4908)
560 DATA  40,120,  4,120 ,216,120,
212,296 ,212,296,168,300 ,168,300,
172,328                            (4974)
570 DATA 172,328,284,320 ,284,320,
280,288 ,280,288,244,292 ,244,292,
244,120                            (49B3)
580 DATA 244,120,216,120 ,320,116,
324,292 ,364,212,392,116 ,392,116,
364,120                            (49F0)
590 DATA 364,120,348,188 ,348,188,
348,116 ,348,116,320,116 ,352,236,
352,268                            (499E)
600 DATA 392,116,388,304 ,388,304,
420,304 ,420,304,416,184 ,416,184,
428,236                            (49D1)
610 DATA 428,236,448,236 ,448,236,
456,188 ,456,188,456,304 ,456,304,
484,304                            (4927)
620 DATA 484,304,480,116 ,480,116,
452,120 ,452,120,436,176 ,436,176,
420,116                            (4923)
630 DATA 420,116,392,116 ,480,116,
504,292 ,504,292,536,296 ,536,296,
564,116                            (49A9)
640 DATA 564,116,536,116 ,536,116,
528,172 ,528,172,516,172 ,516,172,
512,116                            (49A5)
650 DATA 512,116,480,116 ,520,204,
520,240 ,520,240,524,204 ,524,204,
520,204                            (49C6)
660 DATA 564,120,580,200 ,580,200,
544,296 ,544,296,568,304 ,568,304,
592,248                            (4931)
670 DATA 592,248,596,300 ,596,300,
628,296 ,628,296,596,116 ,596,116,
```

```
564,120                                 <49E1>
680 DATA 328,292,356,292                <16F3>
690 FOR i=1 TO 6                        <0EB2>
700 READ x,y,rx,ry,e                    <20B9>
710 ORIGIN x,y-20                       <1228>
720 MOVE 0,ry                           <0D5D>
730 FOR a=0 TO e STEP 5                 <1363>
740 DRAW rx*SIN(a),ry*COS(a)            <23F9>
750 NEXT a,i                            <0F58>
760 DATA 164,212,44,88,360              <18DB>
770 DATA 164,212,16,44,360              <180E>
780 DATA 284,206,40,84,360              <18C0>
790 DATA 284,206,16,40,360              <18E1>
800 DATA 356,252,8,16,180               <1792>
810 DATA 356,252,32,40,160              <18C0>
820 ORIGIN 0,-20                        <0B17>
830 MOVE 1,80:FILL 12:MOVE 164,212
:FILL 12:MOVE 284,206:FILL 12:MOVE
 360,252:FILL 12:MOVE 200,280:FILL
 12:MOVE 248,272:FILL 12:MOVE 380,
200:FILL 12                             <5584>
840 DIM i1(12):i1(1)=26:i1(2)=24:i
1(3)=15:i1(4)=6:i1(5)=4:i1(6)=8:i1
(7)=2:i1(8)=1:i1(9)=13:i1(10)=7:i1
(11)=16                                 <8FDB>
850 ORIGIN 0,0,0,640,400,0              <15BF>
860 FOR j=398 TO 0 STEP-398             <1558>
870 FOR i=1 TO 640 STEP 88              <131F>
880 FOR k=0 TO 10                       <0FAD>
890 PLOT i+k*8,j,k+1,0:PLOT(i+k*8)
+4,j,k+1,0                              <3EEF>
900 NEXT k,i,j                          <1450>
910 FOR j=340 TO 50 STEP-290            <1614>
920 FOR i=1 TO 640 STEP 132             <1336>
930 FOR k=0 TO 10                       <0F12>
940 PLOT i+k*12,j,k*-1+11,0:PLOT(i
+k*12)+4,j,k*-1+11,0:PLOT(i+k*12)+
8,j,k*-1+11,0                           <6C70>
950 NEXT k,i,j                          <14B4>
960 RESTORE 1020:TAG:pln=13             <1505>
970 FOR i=7 TO 1 STEP-1                 <11A5>
980 FOR j=1 TO 64                       <0FEC>
990 READ pl:IF pl=1 THEN PLOT j*4+
192,i*2+346,pln,0                       <3372>
1000 NEXT j:pln=pln+1:IF pln>15 TH
EN pln=13                               <2F0E>
1010 NEXT i                             <0A58>
1020 DATA 0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,
0                                       <867A>
1030 DATA 1,1,0,1,1,1,0,0,0,1,1,0,
1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,
1,0,0,0,0,1,1,1,1,0,0,1,1,0,1,1,1,
0,0,0,0,1,1,0,0,0,0,0,0,1,1,1,1,0,
0                                       <86B6>
1040 DATA 0,1,1,0,0,1,1,0,0,0,1,1,
0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,
0,0,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,
1,0,0,1,1,1,1,1,0,0,0,1,1,0,0,0,0,
0                                       <86A9>
1050 DATA 0,1,1,0,0,1,1,0,0,0,1,1,
0,0,0,0,0,1,1,1,1,1,1,0,0,1,1,0,0,
0,0,0,0,1,1,1,1,1,1,0,0,1,1,0,0,1,
1,0,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,
0                                       <8620>
1060 DATA 0,1,1,1,1,1,0,0,0,0,1,1,
0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,1,
1,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,1,
1,0,0,0,1,1,0,0,0,0,0,0,1,1,1,1,0,
0                                       <8610>
1070 DATA 0,1,1,0,0,0,0,0,0,0,1,1,
0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,
1,1,0,0,1,1,0,0,0,0,0,0,1,1,0,0,1,
1,0,0,0,1,1,0,1,1,0,0,0,0,0,1,1,
0                                       <8662>
1080 DATA 1,1,1,1,0,0,0,0,0,1,1,1,
1,0,0,0,0,0,1,1,1,1,0,0,0,1,1,1,1,
1,0,0,0,0,1,1,1,1,0,0,0,1,1,0,0,1,
1,0,0,0,0,1,1,1,0,0,0,1,1,1,1,1,0,
0                                       <868E>
1090 ii1(1)=10:ii1(2)=12:ii1(3)=18
                                        <2B4F>
1100 FOR j=50 TO 590 STEP 540           <1502>
1110 ORIGIN j,370:DEG:pln=13            <1A7D>
1120 FOR i=10 TO 370 STEP 20            <148D>
1130 MOVE 0,0,pln:DRAW 35*SIN(i),2
0*COS(i)                                <2925>
1140 pln=pln+1:IF pln>15 THEN pln=
13                                      <291B>
1150 NEXT i,j                           <0F83>
1160 FOR j=80 TO 600 STEP 520           <1554>
1170 ORIGIN j,25:DEG:pln=13             <1940>
1180 MOVE 0,23                          <0A80>
1190 FOR i=0 TO 360 STEP 20             <1369>
1200 DRAW 21*SIN(i),23*COS(i),pln       <2486>
1210 pln=pln+1:IF pln>15 THEN pln=
13                                      <29A7>
1220 NEXT i,j                           <0F0E>
1230 FOR j=40 TO 560 STEP 520           <15B0>
1240 ORIGIN j,25:DEG:pln=13             <19CC>
1250 MOVE 0,23                          <0A0D>
1260 FOR i=360 TO 0 STEP-20             <14DF>
1270 DRAW 21*SIN(i),23*COS(i),pln       <2413>
1280 pln=pln+1:IF pln>15 THEN pln=
13                                      <2933>
1290 NEXT i,j                           <0F9B>
1300 ORIGIN 80,25:DEG:pln=13            <17B8>
1310 MOVE 0,15                          <0A66>
1320 FOR i=360 TO 0 STEP-40             <14A7>
1330 DRAW 15*SIN(i),15*COS(i),pln       <2451>
1340 pln=pln+1:IF pln>15 THEN pln=
13                                      <29AB>
1350 NEXT i                             <0A02>
1360 ORIGIN 560,25:DEG:pln=13           <1838>
```

```
1370 MOVE 0,15                          <0ADE>
1380 FOR i=0 TO 360 STEP 40             <1336>
1390 DRAW 15*SIN(i),15*COS(i),pln       <24C9>
1400 pln=pln+1:IF pln>15 THEN pln=
13                                      <2924>
1410 NEXT i                             <0A79>
1420 ORIGIN 0,0:TAG:FOR i=1 TO 8        <152B>
1430 MOVE i*32+160,390,13,1:PRINT
CHR$(i+222);:NEXT i                     <2D37>
1440 FOR i=1 TO 8                       <0E99>
1450 MOVE i*32+160,374,13,1:PRINT
CHR$(i+230);:NEXT i                     <2DD8>
1460 FOR i=1 TO 8                       <0EC1>
1470 MOVE i*32+160,390,14,1:PRINT
CHR$(i+238);:NEXT i                     <2DCE>
1480 FOR i=1 TO 8                       <0EE9>
1490 MOVE i*32+160,374,14,1:PRINT
CHR$(i+246);:NEXT i                     <2D66>
1500 TAGOFF                             <0603>
1510 SAVE"motorbil",B,&C000,&4000       <1D1F>
1520 MODE 2:PEN 1:PAPER 0:INK 0,1:
INK 1,24:BORDER 1                       <1B77>
```

Vollgas

Fortsetzung von Seite 103

gestartet. Nun wird die Datei >MOTORWAY.DAT< mit der Eintragungstabelle nachgeladen.
Danach baut der Computer das Armaturenbrett und die fünf Bilder mit den verschiedenen Straßenstellungen, von denen vier in der zweiten Bank abgelegt werden und eine im Bildschirmbereich der ersten Bank bleibt, auf. Während dieser Wartezeit, die etwa 65 Sekunden beträgt, sind alle 16 PENs schwarz gefüllt. Sobald das Bild erscheint, kann durch das Drücken der Leertaste die Fahrsimulation beginnen. Hierbei gelten die folgenden Joystickbelegungen:
>aufwärts< Beschleunigung des Fahrzeugs
>abwärts< Abbremsen des Fahrzeugs
>links< und >rechts< Steuern
>1<, >2<, >3< und >4< Gangschaltung.
Wenn die Meldung >GAME OVER< erscheint, so sind Sie entweder auf den rotweißen Rand gefahren, es ist kein Benzin mehr vorhanden oder die Motortemperatur ist zu hoch. Um zu vermeiden, daß die Temperatur zu schnell ansteigt, sollten Sie nicht zu lange in niedrigen Gängen fahren, sondern rechtzeitig einen höheren Gang einlegen.
Auch ständiges Abbremsen und wieder Beschleunigen sollte vermieden werden, um immer genügend Benzin im Tank zu behalten. Nur so kann es Ihnen gelingen, die rund zehn Kilometer lange Rennstrecke durchzustehen.
Haben Sie die Strecke in einer relativ guten Zeit zurückgelegt, dürfen Sie Ihren Namen in die Tabelle eintragen. Um die Eintragung abzuschließen, fahren Sie mit dem Cursor auf das Feld >END< und drücken den Joy-Button oder die Leertaste. Damit gelangen Sie zu der Top 35 Eintragungstabelle.
Allerdings werden Grünmonitor-Besitzer Schwierigkeiten mit der Farbgebung in der Eintragungstabelle haben. Wenn Sie keinen Farbmonitor besitzen, ist es am besten, die entsprechenden Ink- und Pen-Befehle anzupassen.
Nach dem Drücken der Leertaste muß eines der fünf Bilder, auf das die Meldung >GAME OVER< oder >READY< geprintet wurde, neu aufgebaut werden, bevor die nächste Fahrsimulation beginnen kann. Fehler, die zum Beispiel an der Ganganzeigetafel durch gleichzeitiges Drücken der Gangtasten an den Bildern entstanden sind, können nur, da die anderen vier Straßenstellungen nicht nochmals frisch aufgebaut wurden, durch Unterbrechen des Programms und einen Neustart mit >RUN 80< behoben werden.
Sollte die vorgegebene Rennstrecke mit der Zeit zu langweilig werden, so kann die Data-Zeile in der Programmzeile 4760, die die Werte für die verschiedenen Straßenstellungen enthält, beliebig verändert werden. Dabei stehen die Werte für:

1 = stark linke Kurve	4 = leicht rechte Kurve
2 = leicht linke Kurve	5 = stark rechte Kurve
3 = gerade Strecke	10 = ** ENDE **

Nun hoffen wir, daß Sie viel Spaß mit dieser Fahrsimulation haben werden. Gelingt es Ihnen, unsere Rekordzeit von 3 min und 42 sec zu brechen? *JE* □

AMSTRAD-CLUB LUXEMBURG

Der Luxemburger User-Club (früher: Schneider Master Club) sucht weitere Mitglieder im In- und Ausland. Der Club befaßt sich mit den CPC 464, 664 und 6128. Zum Angebot gehört unter anderem eine Clubzeitung, die alle drei Monate erscheint, eine Club-Bibliothek und ein Mitglieder-Service. Ein Infoblatt wird auf Anforderung zugeschickt. Dieser Aufruf gilt auch für andere Computerclubs, die an Kontakten interessiert sind.
Meldet Euch bei:
Alain Theisen
Rue Nic Biever 18
L-4807 Rodange
oder bei:
Serge Linckels
Op Echelter 2
L-8366 Hagen □

AMSTRAD-CLUB SCHWEIZ

Auch in der Schweiz gibt es einen Amstrad- bzw. Schneider-CPC-Club. Zum Angebot des Vereins gehören regelmäßige Clubtreffs und das Club-Info ebenso wie ein Reparaturservice und Public-Domain-Software. Für Clubmitglieder gibt es Einkaufsvergünstigungen bis zu fünfzehn Prozent. Zum Angebot für Hardwarebastler gehört eine Anzahl erprobter Bauanleitungen und Bausätze.
Die Kontaktadresse:
Klaus-Dieter Preiss
Im Hof 20
CH-8637 Laupen/
Wald ZH
Telefon: 951302 □

AMSTRAD-CLUB INTERNATIONAL

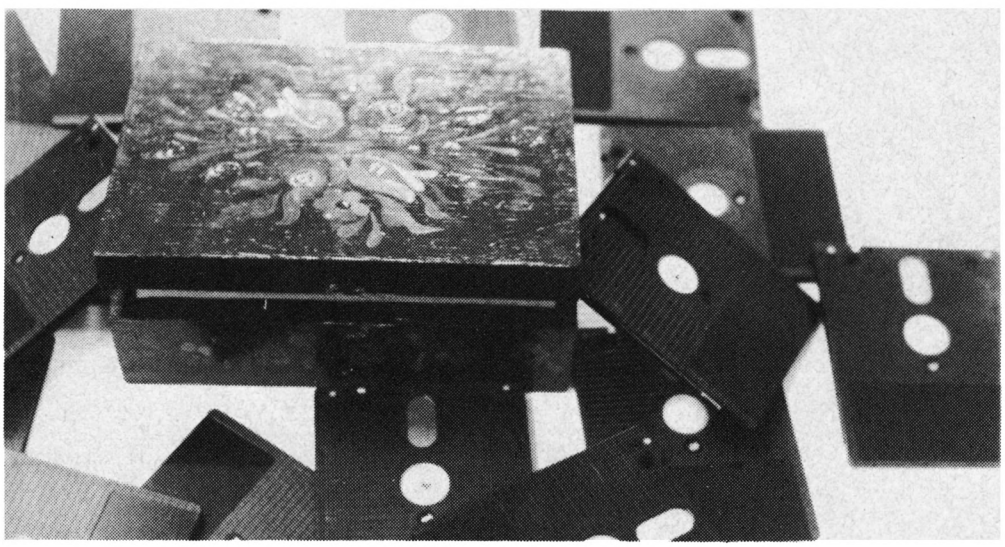
WACCI, Jeff Walkers europaweiter Amstrad- und Schneider-Club sucht noch Mitglieder, auch in Österreich und der Schweiz. Angeboten werden Hardware, Software, Freesoft, Public Domain, ein monatlicher Newsletter und vieles mehr. Informationen sind erhältlich gegen drei Achtzig-Pfennig-Marken; aus der Schweiz und Österreich ist kein Rückporto erforderlich. Dort ist auch die Einrichtung weiterer Landesstellen geplant. Interessenten melden sich bei:
WACCI BRD
Enzianstraße 10
7464 Schömberg □

DRIVE A: DISC FULL ...

Ohne Ärger

Ab jetzt gibt es keinen Ärger mehr mit der allzu frühen "Disc Full"-Meldung bei Drei-Zoll-Disketten. Jeder CPC-Besitzer kann die Kapazität pro Disc-Seite um 30 KByte erweitern.

Welcher Schneider-Freak kennt nicht die Meldung "Disc Full" auf seiner sündhaft teuren Drei-Zoll-Diskette? 178 KByte sind eben doch noch nicht die Welt. Doch jetzt eilt Hilfe herbei: 208 KByte pro Diskettenseite lassen sich mit dem abgedruckten Programm von jedem CPC-Besitzer frei nutzen. Das sind somit pro Drei-Zoll-Diskette 60 KByte mehr. So etwas wirkt sich auch positiv auf Ihren Geldbeutel aus, sei er nun klein oder groß.
Mit Dutzenden von RSX-Befehlen brauchen die zusätzlichen Diskettenkapazitäten nicht angesprochen zu werden. Die ganz normalen BASIC-Befehle wie SAVE, LOAD oder CAT genügen. Alles andere macht das Maschinensprache-Programm. Es erkennt, ob es sich um eine normale CP/M- oder datenformatierte Diskette handelt oder ob Sie eine für 208 KByte vorbereitete verwenden.



Mit unserem Programm kann die Drei-Zoll-Diskette auf 208 KByte erweitert werden. Kein Ärger mehr mit "Disc Full".

Sie müssen nur Ihre Leerdisketten entsprechend formatieren. Aber auch das ist kein Problem, denn unser Formatierungsprogramm ist aufgrund seiner bequemen Menüsteuerung einfach zu bedienen.
Tippen Sie als erstes das abgedruckte Programm ab oder nutzen Sie unseren Diskettenservice. Speichern Sie das Formatierungsprogramm als BASIC-Programm mit der Befehlsfolge SAVE"FORMAT" ab. Falls sich beim Eintippen ein Fehler in den Datenzeilen eingeschlichen haben sollte, wird Ihnen das mitgeteilt. Es ist daher wichtig, das BASIC-Programm vollständig einzugeben, da sonst eine detaillierte Angabe über die Programmzeile, in der der Fehler aufgetreten ist, nicht korrekt erfolgen kann.
Mit Hilfe des Programms können Sie jetzt Ihre Leerdisketten auf 208 KByte pro Seite formatieren. Danach wird auf ihnen automatisch die Routine DOS 208 abgespeichert. Somit brauchen Sie später das Programm nicht lange zu suchen, da es sich auf jeder neu formatierten Diskette befindet.
Das DOS-208-Programm benötigt zudem nur 2 KByte auf Ihrer Disk. Falls Sie jedoch DOS 208 nicht gespeichert haben möchten, können Sie es problemlos mit ERA,"DOS208.BIN" löschen. Außerdem wird ein File namens "*==> DOS.208 0K" in das Inhaltsverzeichnis der Diskette eingetragen. Durch diesen Vermerk können Sie bei einem CATALOG leicht erkennen, ob es sich um eine normale Daten- oder CP/M-Diskette handelt oder um eine für 208 KByte vorbereitete Diskette. Ist eine 208-KByte-Diskette eingelegt, muß die DOS208-Routine aktiv sein. Dies erkennen Sie daran, daß bei einem CATALOG die Meldung "DOS 208 aktiv" erscheint.
Um eine Diskette vorzubereiten, verfahren Sie wie folgt:

- Starten Sie das abgedruckte Programm mit RUN.
- Als erstes müssen Sie angeben, in welchem Laufwerk Sie Ihre Leerdiskette formatieren möchten.
- Nachdem Sie zwischen Laufwerk A und B gewählt haben, legen Sie Ihre Leerdiskette ein und drücken die Taste "F".

Jetzt formatiert das Programm Ihre Leerdiskette auf 42 Spuren mit je 10 Sektoren à 512 Byte. Im normalen Daten- und CP/M-Format werden nur 40 Spuren à neun Sektoren genutzt, so daß Sie 30 KByte pro Diskettenseite mehr zur Verfügung haben.
Nach dem Formatieren speichert das Programm automatisch die Routine DOS208 ab. Bei dem abschließenden CATALOG erkennen Sie, daß sich jetzt auf Ihrer Diskette ein freier Speicherplatz von 207 K-Byte befindet, den Sie mit den üblichen BASIC-Befehlen LOAD, SAVE, OPENOUT und OPENIN frei nutzen können.

Die Initialisierung der DOS-208-Routine nach dem Einschalten des Computers geschieht auf folgende Art:

- Legen Sie eine Diskette ein, auf der sich die durch das Programm "FORMAT" abgespeicherte Routine DOS208.BIN befindet.
- Laden Sie das Maschinensprache-Programm mit LOAD"DOS208"
- Geben Sie den Befehl CALL &B000 ein, und Sie sehen die Einschaltmeldung von DOS208. Beachten Sie, daß diese Befehlsreihenfolge absichtlich so gewählt worden ist und kein BASIC-Lader für den

Lesen Sie bitte weiter auf Seite 122

```
10 '*****************************  (2397)
15 '*         DOS 208           *  (23EB)
20 '*           VON             *  (239D)
25 '*      JUERGEN ROHLJE       *  (23C1)
30 '*          FUER             *  (234F)
35 '*    SCHNEIDER CPC WELT     *  (23CF)
40 '*     CPC 464/664/6128    JE*  (23FA)
45 '*****************************  (23DD)
50 REM                             (067B)
60 DATA &F3,&08,&D9,&F5,&7D,&32,&F
E,&B0, 1318                        (2C31)
70 DATA &FE,&4C,&28,&0D,&FE,&61,&2
8,&09, 783                         (2BCB)
80 DATA &FE,&70,&28,&05,&F1,&D9,&0
8,&FB, 1128                        (2CA7)
90 DATA &C9,&F1,&D9,&08,&FB,&E5,&C
5,&D5, 1557                        (2CFF)
100 DATA &F5,&3A,&FE,&B0,&FE,&70,&
20,&06, 1137                       (2C24)
110 DATA &21,&58,&B0,&CD,&0E,&B0,&
3A,&7D, 875                        (2BDD)
120 DATA &BE,&DF,&55,&B0,&3A,&4C,&
BE,&CB, 1201                       (2C78)
130 DATA &6F,&28,&3B,&2A,&42,&BE,&
ED,&4B, 820                        (2BBE)
140 DATA &7D,&BE,&0A,&FE,&00,&20,&
05,&22, 650                        (2B89)
150 DATA &FE,&B0,&18,&07,&11,&40,&
00,&19, 567                        (2B93)
160 DATA &22,&FE,&B0,&CD,&B9,&AF,&
21,&F4, 1306                       (2CBD)
170 DATA &B0,&3A,&EB,&B0,&BE,&20,&
0B,&2A, 920                        (2BD7)
180 DATA &FE,&B0,&11,&18,&00,&19,&
36,&00, 550                        (2B39)
190 DATA &18,&0C,&ED,&5B,&FE,&B0,&
01,&19, 820                        (2B1B)
200 DATA &00,&21,&E1,&AF,&ED,&B0,&
F1,&D1, 1296                       (2C2E)
210 DATA &C1,&E1,&C9,&2A,&FE,&B0,&
11,&16, 1130                       (2CB9)
220 DATA &00,&19,&56,&2A,&7D,&BE,&
5E,&21, 595                        (2B60)
230 DATA &EA,&B0,&E5,&DF,&FA,&AF,&
E1,&36, 1566                       (2CB0)
240 DATA &00,&06,&0A,&E5,&C5,&DF,&
FD,&AF, 1093                       (2C97)
250 DATA &C1,&E1,&D0,&3A,&51,&BE,&
23,&77, 1109                       (2C4F)
260 DATA &10,&F1,&C9,&28,&00,&03,&
07,&00, 508                        (2BDC)
270 DATA &D1,&00,&3F,&00,&C0,&00,&
10,&00, 480                        (2B49)
280 DATA &00,&00,&C1,&0A,&20,&32,&
E5,&02, 516                        (2BCF)
290 DATA &04,&00,&00,&FF,&63,&C7,&
07,&5D, 657                        (2B58)
300 DATA &C5,&07,&21,&C3,&3E,&22,&
7F,&BE, 845                        (2B5F)
310 DATA &3E,&AF,&32,&81,&BE,&21,&
18,&B0, 839                        (2BB3)
320 DATA &7E,&FE,&FF,&C8,&CD,&5A,&
BB,&23, 1352                       (2CF5)
330 DATA &18,&F6,&07,&0C,&18,&20,&
44,&4F, 492                        (2BF2)
340 DATA &53,&20,&32,&30,&38,&20,&
69,&73, 521                        (2B42)
350 DATA &74,&20,&69,&6E,&69,&74,&
69,&61, 786                        (2B5B)
360 DATA &6C,&69,&73,&69,&65,&72,&
74,&20, 796                        (2BA1)
370 DATA &18,&0A,&0D,&0A,&A4,&20,&
62,&79, 472                        (2BFF)
380 DATA &20,&4A,&75,&65,&72,&67,&
65,&6E, 752                        (2B00)
390 DATA &20,&52,&6F,&68,&6C,&6A,&
65,&2C, 688                        (2BF2)
400 DATA &20,&31,&39,&38,&37,&0A,&
FF,&30, 562                        (2B61)
410 DATA &C6,&07,&0A,&44,&4F,&53,&
20,&32, 527                        (2B49)
420 DATA &30,&38,&20,&61,&6B,&74,&
69,&76, 679                        (2B2F)
430 DATA &0A,&FF, 265              (1343)
440 dat=0:sz=0:dz=60               (1E83)
450 FOR adr=&AF3E TO &B067         (14AA)
460 READ byte:dat=dat+1            (1D5F)
470 sz=sz+byte                     (1894)
480 POKE adr,byte                  (147F)
490 IF dat<8 AND adr<&B067 THEN 53
0                                  (1DC1)
500 READ chksum                    (0F5A)
510 IF chksum<>sz THEN PRINT"Fehle
r in Zeile ";dz                    (2FFD)
520 dz=dz+10:sz=0:dat=0            (2433)
530 NEXT adr                       (0C56)
540 REM                            (0651)
550 REM Programmteil zum Formatier
en auf 208 KB                      (2E66)
560 REM Nach der Formatierung wird
 die Routine DOS208                (3486)
570 REM automatisch auf der eingel
egten Diskette abgespeichert       (3DD7)
580 REM                            (06A1)
590 POKE &BE7F,&C9                 (0DEC)
600 RESTORE 1060:POKE &BDEE,&C9    (1212)
610 GOSUB 1100                     (0907)
620 MODE 2:PRINT"Waehlen Sie das D
rive, indem Sie formatieren moecht
en":PRINT                          (4298)
630 PRINT" Drive A = Taste 'A'
     Drive B = Taste 'B'"          (388B)
640 a$=UPPER$(INKEY$)              (1016)
650 IF a$="A"THEN POKE &A8A0,10:PO
KE &A8A2,41:POKE &B0C8,0:GOTO 680  (2A1D)
660 IF a$="B"THEN POKE &A8E0,10:PO
```

```
KE &A8E2,41:POKE &B0C8,1:GOTO 680 <2A86>
670 GOTO 640                      <0937>
680 PRINT:PRINT"Bitte legen Sie di
e Leerdiskette in Drive ";a$:     <3A4D>
690 PRINT:PRINT"und druecken Sie d
ie Taste 'F' zum formatieren"     <38B8>
700 b$=UPPER$(INKEY$):IF b$="F"THE
N GOTO 710 ELSE GOTO 700          <2541>
710 PRINT:PRINT"Bitte warten Sie e
inen Moment ";                    <2915>
720 FOR track=0 TO 41             <13E5>
730 POKE &B0CA,track              <12E7>
740 x=&B0D7                       <0DF6>
750 FOR n=0 TO 9                  <0E68>
760 POKE x+n*4,track              <1AB4>
770 NEXT n                        <0A8C>
780 PRINT".";                     <0A28>
790 IF TRACK<39 THEN CALL &B0C7 EL
SE FOR H=1 TO 7:CALL &B0C7:NEXT H <2C25>
800 NEXT track                    <0E40>
810 PRINT:IF a$="A"THEN POKE &A700
,0 ELSE POKE &A700,1              <1F32>
820 REM                           <0682>
830 REM Eintrag des Hinweises "* =
==> Dos 208" ins Inhaltsverzeichni
s                                 <44B5>
840 REM                           <06AA>
850 DATA &1E,&00,&16,&00,&0E,&C1,&
21,&B0,&A9,&DF,&D4,&B0,&C9,&4E,&C6
,&07                              <4617>
860 RESTORE 850                   <0933>
870 FOR adr=&B0C7 TO &B0D6        <142C>
880 READ byte:POKE adr,byte:NEXT A
DR                                <25C7>
890 FOR adr=&A9B0 TO &ABAF:POKE ad
r,&E5:NEXT adr                    <2833>
900 FOR adr=&A9B0 TO &A9CF:POKE ad
r,0:NEXT adr                      <26E5>
910 name$="*==> DOS208":IF a$="A"T
HEN POKE &B0C8,0 ELSE POKE &B0C8,1
                                  <3316>
920 FOR adr=&A9B1 TO &A9BB:POKE ad
r,ASC(MID$(name$,adr-&A9B0,1)):NEX
T adr                             <40BF>
930 POKE &A9B9,&B2:CALL &B0C7:CALL
 &B000                            <173D>
940 REM                           <0671>
950 REM Speichern der Routine Dos
208 auf die Diskette              <351B>
960 REM                           <0699>
970 PRINT:PRINT"Speichern der Rout
ine DOS 208 !":SAVE"dos208",b,&AF3
E,&12A                            <4011>
980 PRINT:PRINT"Die Diskette ist j
etzt fuer 208 KB vorbereitet"     <38BC>
990 CAT:PRINT                     <080E>
1000 PRINT:PRINT"Moechten Sie noch
 eine Diskette formatieren      (J
/N)"                              <40C7>
1010 a$=UPPER$(INKEY$):IF a$="J"TH
EN RUN 540 ELSE IF a$="N"THEN CLS:
POKE &BDEE,195:NEW                <365D>
1020 GOTO 1010                    <098C>
1030 REM                          <0627>
1040 REM Routine zum Formatieren d
er Diskette                       <2B96>
1050 REM                          <064F>
1060 DATA &1E,&00,&16,&00,&0E,&C1,
&21,&D7,&B0,&DF,&D4,&B0,&C9,&52,&C
6,&07, 1782                       <4C25>
1070 DATA &00,&00,&C1,&02,&00,&00,
&C6,&02,&00,&00,&C2,&02,&00,&00,&C
7,&02, 792                        <4BB0>
1080 DATA &00,&00,&C3,&02,&00,&00,
&C8,&02,&00,&00,&C4,&02,&00,&00,&C
9,&02, 800                        <4B0B>
1090 DATA &00,&00,&C5,&02,&00,&00,
&CA,&02, 403                      <2BB9>
1100 dat=0:sz=0:dz=1060           <1F20>
1110 FOR adr=&B0C7 TO &B0FE       <144C>
1120 READ byte:dat=dat+1          <1D8B>
1130 sz=sz+byte                   <18BE>
1140 POKE adr,byte                <14AB>
1150 IF dat<16 AND adr<&B0FF THEN
1190                              <1EB1>
1160 READ chksum                  <0F84>
1170 IF chksum<>sz THEN PRINT"Fehl
er in Zeile ";dz:END              <3197>
1180 dz=dz+10:sz=0:dat=0          <245C>
1190 NEXT adr                     <0C81>
1200 RETURN                       <068A>
```

Fortsetzung von Seite 120

Maschinensprache-Teil verwendet wurde. Dieses Vorgehen gewährleistet, daß DOS208 zu jedem Zeitpunkt geladen und aktiviert werden kann, ohne daß Daten verloren gehen.
Falls Sie das Programm DOS208 auf eine Diskette kopieren möchten, auf der sich schon Programme befinden, benutzen Sie folgende Befehle:

- Diskette einlegen, die für 208 KByte durch das abgedruckte Formatierungsprogramm vorbereitet worden ist.
- LOAD"DOS208.BIN":CALL &B000
- Diskette im CP/M- oder Data-Format einlegen.
- SAVE"DOS208",b,&AF3E,&12A

Sie können auch wieder eine CP/M- oder Datendiskette einlegen und sie bearbeiten, als wäre nichts geschiehen. Das Programm DOS208 verwaltet selbständig, falls eine für 208 KByte vorbereitete Diskette eingelegt worden ist, die zusätzlichen 30 KByte. DOS208 läuft sowohl mit einem als auch mit zwei Laufwerken. Sie können sogar einen CONTROL-SHIFT-RESET auslösen. Wenn Sie CALL &B000 eingeben, wird Ihr DOS208 wieder aktiviert.

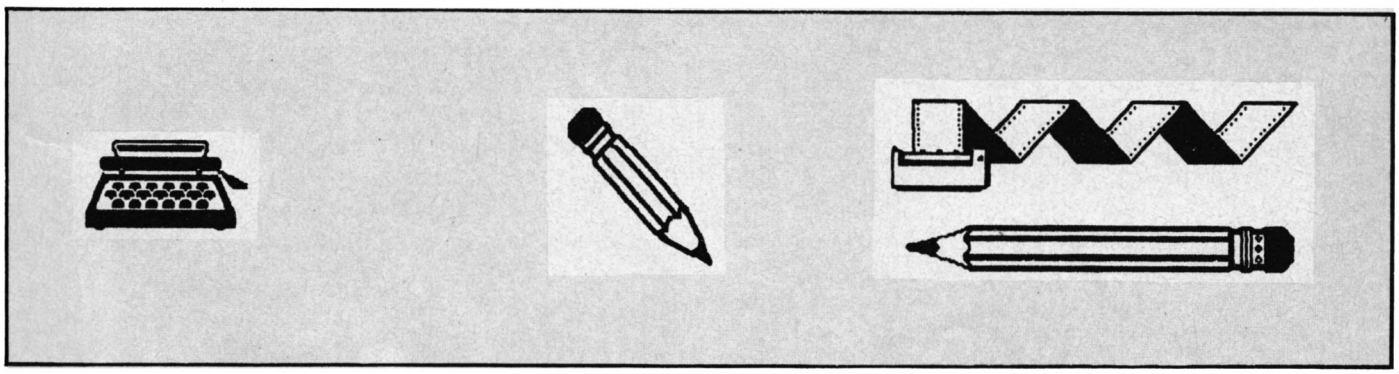
Jürgen Rohlje□

WORDPROCESSOR

Texten für Jedermann

In diesem Heft stellen wir Ihnen ein Textprogramm unseres Lesers Steffen Klaer vor, das zum Schreiben kleinerer Texte für den Hausgebrauch durchaus geeignet ist. Es stellt keine besonders hohen Anforderungen durch komplizierte Tastenbelegungen.

Es handelt sich um ein Textverarbeitungsprogramm, bei dem Sie zwar nicht über einen Fullscreen-Editor verfügen, die Zeilen aber einzeln editieren können. Das Programm ist auf den DMP 2000 abgestimmt, kann aber, falls nötig, leicht an andere Drucker angepaßt werden.
Beim Entwickeln des Programmes wurde auf Übersichtlichkeit besonderer Wert gelegt. Dadurch sind Änderungen im Programm leicht durchzuführen. Jede Subroutine bildet somit eine logische Einheit. Das Programm ist ausführlich REMarked. Alle GOSUBs und GOTOs sind mit einer Angabe versehen, die den Sprung erläutern.
Wichtig: Ist ein Drucker am System angeschlossen,



so muß dieser auf ON LINE stehen, weil der Computer sonst in einer Endlosschleife hängt.
Nach dem Start des Programms werden Sie gefragt, ob die deutsche Tastaturbelegung gewünscht wird. Ist dies der Fall, dann sieht die Belegung hinterher so aus:

ä .. eckige Klammer auf
Ä .. geschweifte Klammer auf
Ö .. inverser Schrägstrich
ö .. senkrechter Strich
ü .. eckige Klammer zu
Ü .. geschweifte Klammer zu
Z .. mit Y vertauscht
z .. mit y vertauscht
ß .. CTRL 2
§ .. Klammeraffe,

Danach erscheint das Hauptmenü mit den Optionen:

1. Text einladen
2. Text eingeben
3. Text ändern
4. Text sehen
5. Druckoptionen eingeben
6. Text ausdrucken
7. Text abspeichern
8. Quit.

Die einzelnen Routinen haben folgende Funktionen:

1. Text einladen:

Diese Option lädt eine ASCII-Datei mit vorgegebenem Namen in das Array text$().

2. Text eingeben:

Es ist jetzt möglich, einen Text einzugeben, der jedoch nicht mehr als 1000 Zeilen umfassen darf (es sei denn, Sie haben vorher die entsprechende Konstante verändert). Sie können den gegenwärtigen Text löschen oder weitere Zeilen anfügen. Die erscheinende Zeilennummer dient nur der Orientierung; sie wird nicht mit ausgedruckt.
Wenn Sie diesen Menüpunkt wieder verlassen wollen, schreiben Sie als Text einfach in die nächste Zeile die Buchstaben ETX (für End of TeXt), dann kommen Sie wieder ins Menü zurück.

3. Text ändern:

Geben Sie die gewünschte Zeilennummer ein, erscheint die alte Zeile und darunter die Nummer der neuen. Beim Verändern empfiehlt es sich, den COPY-Cursor zu verwenden.

4. Text sehen:

Dieses Unterprogramm schreibt den eingegebenen Text auf den Bildschirm. Es kann mit ESC angehalten und mit einer anderen Taste fortgesetzt werden.

5. Druckoptionen eingeben:

Wenn Sie diese Routine anwählen, verzweigt sich das Programm zu einem weiteren Menü mit den Möglichkeiten, die gegenwärtigen Optionen zu sehen, sie zu verändern oder die Routine zu verlassen. Bei den Druckoptionen wird folgendes geboten: Eingabe der Seitenlänge, der Zeichen pro Zeile, der linken Randbegrenzung, der zu überspringenden Zeilen am Seitenende und des Schriftbildes.
Alle Angaben beziehen sich auf den Drucker. Wurde am Anfang die deutsche Tastatur gewählt, so überträgt sich dies auch automatisch auf ihn.

6. Text ausdrucken:

Die Option gibt den Text auf den Drucker aus. Dieser muß auf ON LINE geschaltet sein.

Lesen Sie bitte weiter auf Seite 128

```
1 '****************************** (2506)
2 '*       WORDPRECESSOR          * (25FA)
3 '*            VON               * (2538)
4 '*       STEFFEN KLAER          * (25FE)
5 '*            FUER              * (2558)
6 '*    SCHNEIDER CPC-WELT        * (2528)
7 '*       CPC 664/6128        JE* (254E)
8 '******************************.(2514)
10 REM steffen klaer               (145C)
20 REM 11-oct-86                   (1024)
30 REM 4710 luedinghausen 2        (1B90)
40 REM CONSTanten section          (197A)
50 LET deutsch$=CHR$(2):LET cr$=CH
R$(13):LET etx$="ETX":LET esc$=CHR
$(27):LET ff$=CHR$(12):LET filelen
gth%=12:LET maxspalten%=73:LET max
zeilen%=1000:LET r0%=10:LET r1%=30
                                   (9971)
60 REM PROGRAMM Wordprocessor vers
ion 4.0                            (29F0)
70 MODE 2:DIM text$(maxzeilen%)    (1E5B)
80 GOSUB 120:REM Initialize        (16D9)
90 GOSUB 300:REM AuswahlMenue      (181F)
100 CLEAR:MODE 2                   (0957)
110 END                            (063F)
120 REM SUBROUTINE Initialize      (1C47)
130 DEG:CLS:ORIGIN 50,350:MOVE 60,
10:DRAW 60,30:DRAW 0,30            (2402)
140 FOR lcv%=90 TO 270 STEP 20:DRA
W r1%*COS(lcv%),r1%*SIN(lcv%):NEXT
 lcv%                              (41B2)
150 DRAW 60,-30:DRAW 60,-10:DRAW 0
,-10                               (1BDE)
160 FOR lcv%=270 TO 90 STEP-20:DRA
W r0%*COS(lcv%),r0%*SIN(lcv%):NEXT
 lcv%                              (425E)
170 DRAW 60,10:ORIGIN 185,310:MOVE
-60,10:DRAW-60,30:DRAW 0,30        (295D)
180 FOR lcv%=0 TO 180 STEP 20:DRAW
 r1%*SIN(lcv%),r1%*COS(lcv%):NEXT
lcv%                               (3F4A)
190 DRAW-60,-30:DRAW-60,-10:DRAW 0
,-10                               (1D90)
200 FOR lcv%=180 TO 0 STEP-20:DRAW
 r0%*SIN(lcv%),r0%*COS(lcv%):NEXT
lcv%                               (4098)
210 DRAW-60,10:MOVE-164,40:FILL 1:
ORIGIN 0,0                         (1C45)
220 RAD:LOCATE 1,9:PRINT"Schneider
 Computer Division"                (2AB5)
230 LOCATE 30,1:PRINT"PROGRAMM Wor
dprocessor Version 4.0"            (30CB)
240 LOCATE 30,2:PRINT"------------
----------------------"            (301C)
250 LOCATE 1,13:INPUT"Ist ein Druc
ker an das System angeschlossen (J
/N) ";drucker$                     (4B4A)
260 LET drucker$=UPPER$(drucker$):
IF drucker$<>"N"AND drucker$<>"J"T
HEN GOTO 250:REM eingabe wiederhol
en                                 (5903)
270 LOCATE 1,16:INPUT"Moechten Sie
 deutsche Tastaturbelegung (J/N) "
;qwertz$                           (451C)
280 LET qwertz$=UPPER$(qwertz$):IF
 qwertz$="N"THEN GOTO 290 ELSE IF
qwertz$="J"THEN GOSUB 460 ELSE IF
qwertz$<>"N"AND qwertz$<>"J"THEN G
OTO 270:REM eingabe wiederholen    (7F16)
290 RETURN                         (066C)
300 REM SUBROUTINE AuswahlMenue    (1EB2)
310 CLS:WINDOW#0,1,80,4,25:WINDOW#
1,2,79,2,2:MOVE 0,360:DRAW 639,360
:DRAW 639,392:DRAW 0,392:DRAW 0,36
0                                  (4821)
320 PRINT#1,"Function:";TAB(39)"Wo
rkfile:"                           (256C)
330 LOCATE#1,11,1:PRINT#1,"Menue";
SPC(23)                            (1F45)
340 IF workfile$=""THEN LET workfi
le$="no file"                      (2B3C)
350 LOCATE#1,49,1:PRINT#1,workfile
$;SPC(filelength%-LEN(workfile$))  (3EBA)
360 CLS:PEN 0:PAPER 1:PRINT"MENUE"
:PAPER 0:PEN 1                     (1BD8)
370 PRINT:PRINT:PRINT:PRINT"1. Tex
t einladen":PRINT:PRINT"2. Text ei
ngeben":PRINT:PRINT"3. Text aender
n":PRINT:PRINT"4. Text sehen"      (5CF4)
380 PRINT:PRINT"5. Druckoptionen e
ingeben":PRINT:PRINT"6. Text ausdr
ucken":PRINT:PRINT"7. Text abspeic
hern":PRINT:PRINT"8. Quit":PRINT:P
RINT:INPUT"Bitte waehlen Sie (1..8
) : ",wahl$                        (8C94)
390 LET wahl%=VAL(wahl$):IF wahl%<
1 OR wahl%>8 THEN GOTO 340 ELSE IF
 wahl%=8 THEN GOTO 420:REM sicherh
eitsabfrage                        (59F1)
400 ON wahl%GOSUB 530,690,970,1080
,1140,1230,1290:REM zur option ver
zweigen                            (4142)
410 GOTO 330:REM menue wiederholen
                                   (1D11)
420 LOCATE#1,11,1:PRINT#1,"Quit";S
PC(24)                             (1E73)
430 CLS:INPUT"Sind Sie sicher (J/N
) ";jn$                            (2675)
440 LET jn$=UPPER$(jn$):IF jn$="N"
THEN GOTO 330 ELSE IF jn$="J"THEN
GOTO 450 ELSE GOTO 430:REM subrout
ine verlassen oder eingabe wiederh
olen                               (6C77)
450 RETURN                         (06AB)
460 REM SUBROUTINE Qwertz          (181D)
470 SYMBOL AFTER 63                (0964)
```

```
480 SYMBOL 64,&3E,&60,&7C,&66,&3E,
&6,&6,&7C:SYMBOL 91,&66,&18,&3C,&6
6,&7E,&66,&66,&0:SYMBOL 92,&66,&3C
,&66,&66,&66,&66,&3C,&0:SYMBOL 93,
&66,&0,&66,&66,&66,&66,&3C,&0      <94E3>
490 SYMBOL 123,&66,&0,&3C,&6,&3E,&
66,&3E,&0:SYMBOL 124,&66,&0,&3C,&6
6,&66,&66,&3C,&0:SYMBOL 125,&66,&0
,&0,&66,&66,&66,&3E,&0:SYMBOL 126,
&3C,&66,&66,&7C,&66,&66,&7C,&60    <94E3>
500 KEY DEF 71,1,121,89,25:KEY DEF
 43,1,122,90,26:KEY DEF 22,1,92:KE
Y DEF 24,1,94,26:KEY DEF 26,1,64,1
24:KEY DEF 17,1,91,123:KEY DEF 19,
1,93,125                           <62E2>
510 IF drucker$="J"THEN PRINT#8,es
c$;"R";deutsch$;                   <2FBC>
520 RETURN                         <0639>
530 REM SUBROUTINE LadeFile        <1ABD>
540 LOCATE#1,11,1:PRINT#1,"Text ei
nladen";SPC(15)                    <2727>
550 CLS:INPUT"Moechten Sie das Dir
ectory sehen (J/N) ";jn$           <37D2>
560 LET jn$=UPPER$(jn$):IF jn$="N"
THEN GOTO 590 ELSE IF jn$="J"THEN
CAT ELSE GOTO 550                  <3972>
570 PRINT:PRINT"Press <cr> to cont
inue"                              <2028>
580 IF INKEY$<>cr$THEN GOTO 580:RE
M tastatur abfragen                <2724>
590 CLS:PRINT"Workfile Name eingeb
en, das Format ist filename.ext ."
:PRINT:PRINT TAB(20)"-- > ________
____":LOCATE 25,3:INPUT"",workfile
$                                  <711C>
600 LET workfile$=UPPER$(workfile$
):IF LEN(workfile$)<1 OR LEN(workf
ile$)>12 THEN GOTO 590:REM eingabe
 wiederholen                       <6275>
610 ON ERROR GOTO 1380:REM Error1  <1477>
620 LET nzeilen%=0                 <1261>
630 OPENIN workfile$               <1121>
640 WHILE NOT EOF:LINE INPUT#9,zei
le$:LET nzeilen%=nzeilen%+1:LET te
xt$(nzeilen%)=zeile$:WEND          <50EC>
650 CLOSEIN                        <0639>
660 ON ERROR GOTO 0                <06FD>
670 LOCATE#1,49,1:PRINT#1,workfile
$;SPC(filelength%-LEN(workfile$)+1
0)                                 <4105>
680 RETURN                         <0678>
690 REM SUBROUTINE TypeText        <1AE7>
700 LOCATE#1,11,1:PRINT#1,"Text ei
ngeben";SPC(15)                    <27FF>
710 CLS:INPUT"Alten Text loeschen
(J/N) ";jn$                        <2AC1>
720 LET jn$=UPPER$(jn$):IF jn$="N"
THEN GOTO 800 ELSE IF jn$="J"THEN
GOTO 730 ELSE GOTO 710:REM array l
oeschen oder eingabe wiederholen   <667A>
730 LET workfile$="no file"        <1B72>
740 LOCATE#1,49,1:PRINT#1,workfile
$;SPC(filelength%-LEN(workfile$)+1
0)                                 <4191>
750 FOR lcv%=0 TO nzeilen%:LET tex
t$(lcv%)="":NEXT lcv%              <350E>
760 LET nzeilen%=0                 <1278>
770 CLS:PRINT"Workfile Name eingeb
en, das Format ist filename.ext ."
:PRINT:PRINT TAB(20)"-- > ________
____":LOCATE 25,3:INPUT"",workfile
$                                  <7186>
780 LET workfile$=UPPER$(workfile$
):IF LEN(workfile$)<1 OR LEN(workf
ile$)>12 THEN GOTO 770:REM eingabe
 wiederholen                       <6233>
790 LOCATE#1,49,1:PRINT#1,workfile
$;SPC(filelength%-LEN(workfile$))  <3E2C>
800 CLS:PRINT"Maximale Zeichenzahl
 pro Zeile : ";maxspalten%:PRINT:P
RINT etx$;" .. EndofTeXt":PRINT"
     und Rueckkehr zum Menue":PRIN
T:PRINT:PRINT"Press <cr> to contin
ue"                                <9317>
810 IF INKEY$<>cr$THEN GOTO 810:RE
M tastatur abfragen                <2722>
820 CLS                            <0695>
830 FOR lcv%=0 TO nzeilen%-1:PRINT
 lcv%+1:LOCATE 7,lcv%+1:PRINT text
$(lcv%+1):NEXT lcv%                <4CBB>
840 LET x%=POS(#0):LET y%=VPOS(#0)
                                   <1EC0>
850 PRINT nzeilen%+1               <12DB>
860 LOCATE 7,y%:LINE INPUT"",zeile
$                                  <1A83>
870 IF LEN(zeile$)<=maxspalten%THE
N GOTO 900:REM eingabe akzeptieren
                                   <3B8C>
880 LOCATE x%,y%:PRINT SPACE$(LEN(
zeile$)+80):LOCATE x%,y%           <2F6D>
890 GOTO 820:REM eingabe wiederhol
en                                 <1FBB>
900 WHILE zeile$<>etx$             <1555>
910 LET nzeilen%=nzeilen%+1:LET te
xt$(nzeilen%)=zeile$               <3BB6>
920 LET x%=POS(#0):LET y%=VPOS(#0)
                                   <1E5F>
930 PRINT nzeilen%+1:LOCATE 7,y%:L
INE INPUT"",zeile$:IF LEN(zeile$)<
=maxspalten%THEN GOTO 950          <4926>
940 LOCATE x%,y%:PRINT SPACE$(LEN(
zeile$)+80):LOCATE x%,y%:GOTO 2140
:REM eingabe wiederholen           <4AFD>
950 WEND                           <06C5>
960 RETURN                         <06A9>
970 REM SUBROUTINE EditText        <1A7B>
```

```
980 LOCATE#1,11,1:PRINT#1,"Text ae
ndern";SPC(16)                    <26BE>
990 CLS:PRINT"Bitte Editierzeile a
ngeben, das Format ist XXXX, Berei
ch 1..";nzeilen%;".":PRINT:PRINT T
AB(20)"-- > ____":LOCATE 25,3:INPU
T"",edzeile$                      <7EB4>
1000 LET edzeile%=VAL(edzeile$):IF
 edzeile%<1 OR edzeile%>nzeilen%TH
EN GOTO 990:REM eingabe wiederhole
n                                 <5E79>
1010 LOCATE 1,14:PRINT edzeile%:LO
CATE 7,14:PRINT text$(edzeile%):LO
CATE 7,14:LINE INPUT"",zeile$:LET
zeile$=""                         <522C>
1020 FOR lcv%=7 TO 73:LOCATE lcv%,
14:LET zeile$=zeile$+COPYCHR$(#0):
NEXT lcv%                         <3E22>
1030 IF LEFT$(zeile$,3)="CLR"THEN
LET zeile$=""                     <2701>
1040 LET text$(edzeile%)=zeile$   <22E1>
1050 CLS:INPUT"Noch weitere Aender
ungen (J/N) ";jn$                 <2FE5>
1060 LET jn$=UPPER$(jn$):IF jn$="N
"THEN GOTO 1070 ELSE IF jn$="J"THE
N GOTO 990 ELSE GOTO 1050:REM weit
ere aenderungen oder eingabe wiede
rholen                            <6BF3>
1070 RETURN                       <0687>
1080 REM SUBROUTINE ListText      <1AE1>
1090 LOCATE#1,11,1:PRINT#1,"Text s
ehen";SPC(18)                     <242F>
1100 CLS:FOR lcv%=1 TO nzeilen%:PR
INT lcv%;TAB(7)text$(lcv%):NEXT lc
v%                                <3F43>
1110 PRINT:PRINT:PRINT"Press <cr>
to continue"                      <221A>
1120 IF INKEY$<>cr$THEN GOTO 1120:
REM tastatur abfragen             <2764>
1130 RETURN                       <06FF>
1140 REM SUBROUTINE EnterOptions  <1EAF>
1150 LOCATE#1,11,1:PRINT#1,"Drucko
ptionen eingeben"                 <2A0A>
1160 CLS:PEN 0:PAPER 1:PRINT"Optio
nen MENUE":PAPER 0:PEN 1          <242C>
1170 PRINT:PRINT:PRINT:PRINT"1. Dr
uckoptionen sehen":PRINT:PRINT"2.
Druckoptionen aendern":PRINT:PRINT
"3. Quit":PRINT:PRINT:INPUT"Bitte
waehlen Sie (1..3) : ",wahl$      <7A69>
1180 LET wahl%=VAL(wahl$):IF wahl%
<1 OR wahl%>3 THEN GOTO 1160 ELSE
IF wahl%=3 THEN GOTO 1220:REM subr
outine verlassen                  <5B67>
1190 ON wahl%GOSUB 1420,1540:REM z
ur option verzweigen              <2D34>
1200 IF wahl%=2 AND drucker$="J"TH
EN GOSUB 1720:REM SetOptions      <3054>
1210 GOTO 1160:REM menue wiederhol
en                                <1D37>
1220 RETURN                       <06B2>
1230 REM SUBROUTINE Hardcopy      <1A72>
1240 LOCATE#1,11,1:PRINT#1,"Text a
usdrucken";SPC(13):CLS:PRINT"Press
 <cr> to print"                   <4247>
1250 IF INKEY$<>cr$THEN GOTO 1250:
REM tastatur abfragen             <27AE>
1260 FOR lcv%=1 TO nzeilen%:PRINT#
8,text$(lcv%):NEXT lcv%           <35A0>
1270 PRINT#8,ff$                  <0EEC>
1280 RETURN                       <062C>
1290 REM SUBROUTINE SaveFile      <1A68>
1300 LOCATE#1,11,1:PRINT#1,"Text a
bspeichern";SPC(12):CLS           <2C9B>
1310 ON ERROR GOTO 2330:REM Error2
                                  <14E6>
1320 OPENOUT workfile$            <11E6>
1330 FOR lcv%=1 TO nzeilen%:PRINT#
9,text$(lcv%):NEXT lcv%           <352A>
1340 CLOSEOUT                     <06A3>
1350 ON ERROR GOTO 0              <0664>
1360 LOCATE#1,62,1:PRINT#1,"saved"
                                  <1970>
1370 RETURN                       <06E0>
1380 REM SUBROUTINE Error1        <1811>
1390 PRINT:PRINT:PRINT"Press <cr>
to continue"                      <224B>
1400 IF INKEY$<>cr$THEN GOTO 1400:
REM tastatur abfragen             <277B>
1410 RESUME 550                   <093E>
1420 REM SUBROUTINE ListOptions   <1DC9>
1430 CLS                          <065C>
1440 IF laenge%=0 THEN LET laenge$
="nicht angegeben"                <2EFC>
1450 IF zeichen%=0 THEN LET zeiche
n$="nicht angegeben"              <3003>
1460 IF links%=0 THEN LET links$="
nicht angegeben"                  <2C76>
1470 IF sprung%=0 THEN LET sprung$
="nicht angegeben"                <2E6A>
1480 IF schrift$=""THEN LET schrif
t$="6"                            <23C7>
1490 ON INSTR("12345678",schrift$)
GOSUB 1810,1840,1870,1900,1930,196
0,1990,2020:REM zum schriftbild ve
rzweigen                          <5C16>
1500 GOSUB 2050:REM PrintOptions  <180F>
1510 PRINT:PRINT:PRINT"Press <cr>
to continue"                      <223C>
1520 IF INKEY$<>cr$THEN GOTO 1520:
REM tastatur abfragen             <270E>
1530 RETURN                       <0620>
1540 REM SUBROUTINE ChangeOptions <1F91>
1550 CLS:PRINT"Bitte Seitenlaenge
in Inches eingeben, das Format ist
 XX, Bereich 1..22 .":PRINT:PRINT
```

```
TAB(20)"-- > __":LOCATE 25,3:INPUT
"",laenge$                              <79E2>
1560 LET laenge%=VAL(laenge$):IF l
aenge%<1 OR laenge%>22 THEN GOTO 1
550:REM eingabe wiederholen             <52C2>
1570 CLS:PRINT"Bitte Zeichen pro D
ruckzeile eingeben, das Format ist
 XXX, Bereich 1..255 .":PRINT:PRIN
T TAB(20)"-- > ___":LOCATE 25,3:IN
PUT"",zeichen$                          <7D9A>
1580 LET zeichen%=VAL(zeichen$):IF
 zeichen%<1 OR zeichen%>255 THEN G
OTO 1570:REM eingabe wiederholen        <563B>
1590 CLS:PRINT"Bitte linke Randbeg
renzung eingeben, das Format ist X
XX, Bereich 0..255 .":PRINT:PRINT
TAB(20)"-- > ___":LOCATE 25,3:INPU
T"",links$                              <79A3>
1600 LET links%=VAL(links$):IF lin
ks%<0 OR links%>255 THEN GOTO 1590
:REM eingabe wiederholen                <4E60>
1610 CLS:PRINT"Bitte zu uebersprin
gende Zeilen am Seitenende eingebe
n, das Format ist XXX,":PRINT"Bere
ich 1..127 .":PRINT:PRINT TAB(20)"
-- > ___":LOCATE 25,4:INPUT"",spru
ng$                                     <90A3>
1620 LET sprung%=VAL(sprung$):IF s
prung%<1 OR sprung%>127 THEN GOTO
1610:REM eingabe wiederholen            <529C>
1630 CLS:PEN 0:PAPER 1:PRINT"Schri
ftbilder MENUE":PAPER 0:PEN 1           <2900>
1640 PRINT:PRINT:PRINT:PRINT"1. Do
ppeldruck",,"5. NLQ":PRINT:PRINT"2
. Elite",,,"6. NLQ - proportional"
:PRINT:PRINT"3. Fettdruck",,,"7. P
roportional":PRINT:PRINT"4. Kursiv
schrift",,"8. Standard"                 <99A4>
1650 PRINT:PRINT:INPUT"Bitte waehl
en Sie (1..8) : ",schrift$              <3279>
1660 LET schrift%=VAL(schrift$):IF
 schrift%<1 OR schrift%>8 THEN GOT
O 1630:REM eingabe wiederholen          <55D2>
1670 ON INSTR("12345678",schrift$)
GOSUB 1810,1840,1870,1900,1930,196
0,1990,2020:REM zum schriftbild ve
rzweigen                                <5C7E>
1680 GOSUB 2050:REM PrintOptions        <1879>
1690 PRINT:PRINT:INPUT"Eingabe kor
rekt (J/N) ";jn$                        <2857>
1700 LET jn$=UPPER$(jn$):IF jn$="N
"THEN GOTO 1550 ELSE IF jn$<>"J"TH
EN GOTO 1680:REM eingabe wiederhol
en                                      <4C3B>
1710 RETURN                             <0688>
1720 REM SUBROUTINE SetOptions          <1CDF>
1730 PRINT#8,esc$;"@";                  <148A>
1740 IF qwertz$="J"THEN PRINT#8,es
c$;"R";deutsch$;                        <2EBC>
1750 PRINT#8,esc$;"C";CHR$(0);CHR$
(laenge%);                              <28D7>
1760 WIDTH zeichen%                     <1047>
1770 PRINT#8,esc$;"l";CHR$(links%)
;:PRINT#8,esc$;"N";CHR$(sprung%);       <3FB8>
1780 ON INSTR("12345678",schrift$)
GOSUB 2100,2130,2160,2190,2220,225
0,2280,2310:REM zum schriftbild ve
rzweigen                                <5C91>
1790 PRINT#8                            <086F>
1800 RETURN                             <063E>
1810 REM SUBROUTINE Doppeldruck         <1D3E>
1820 LET schriftb$="Doppeldruck"        <1FB8>
1830 RETURN                             <067A>
1840 REM SUBROUTINE Elite               <174E>
1850 LET schriftb$="Elite"              <19A6>
1860 RETURN                             <06B6>
1870 REM SUBROUTINE Fettdruck           <1B2B>
1880 LET schriftb$="Fettdruck"          <1D80>
1890 RETURN                             <06F2>
1900 REM SUBROUTINE Kursivschrift       <1F2C>
1910 LET schriftb$="Kursivschrift"
                                        <2180>
1920 RETURN                             <062D>
1930 REM SUBROUTINE Nlq                 <153E>
1940 LET schriftb$="NLQ"                <1723>
1950 RETURN                             <0669>
1960 REM SUBROUTINE Nlqprop             <19E0>
1970 LET schriftb$="NLQ - proporti
onal"                                   <269C>
1980 RETURN                             <06A5>
1990 REM SUBROUTINE Prop                <1629>
2000 LET schriftb$="Proportional"       <2063>
2010 RETURN                             <06E1>
2020 REM SUBROUTINE Standard            <1A78>
2030 LET schriftb$="Standard"           <1CE2>
2040 RETURN                             <061E>
2050 REM SUBROUTINE PrintOptions        <1E49>
2060 CLS:PRINT"Seitenlaenge :",,"Z
eichen pro Zeile :":PRINT"--------
------",,"--------------------":PRI
NT:PRINT laenge$,,,zeichen$             <7273>
2070 PRINT:PRINT:PRINT"Linke Begre
nzung :",,"Uebersprung :":PRINT"--
----------------",,"-------------"
:PRINT:PRINT links$,,,sprung$           <6E4A>
2080 PRINT:PRINT:PRINT"Schriftbild
 :":PRINT"-------------":PRINT:PRI
NT schriftb$                            <397D>
2090 RETURN                             <0683>
2100 REM SUBROUTINE SetDoppeldruck
                                        <2069>
2110 PRINT#8,esc$;"G";                  <14F5>
2120 RETURN                             <06BF>
2130 REM SUBROUTINE SetElite            <1AFB>
2140 PRINT#8,esc$;"M";                  <1490>
2150 RETURN                             <06FB>
```

```
2160 REM SUBROUTINE SetFettdruck   (1EE8)
2170 PRINT#8,esc$;"E";             (144C)
2180 RETURN                        (0636)
2190 REM SUBROUTINE SetKursivschri
ft                                 (2229)
2200 PRINT#8,esc$;"4";             (1475)
2210 RETURN                        (0672)
2220 REM SUBROUTINE SetNlq         (1895)
2230 PRINT#8,esc$;"x";CHR$(1);     (1AFD)
2240 RETURN                        (06AE)
2250 REM SUBROUTINE SetNlqprop     (1C4D)
2260 PRINT#8,esc$;"x";CHR$(1);esc$
;"p";CHR$(1);                      (2B2A)
2270 RETURN                        (06EA)
2280 REM SUBROUTINE SetProp        (1992)
2290 PRINT#8,esc$;"p";CHR$(1);     (1A94)
2300 RETURN                        (0627)
2310 REM SUBROUTINE SetStandard    (1D00)
2320 RETURN                        (0650)
2330 REM SUBROUTINE Error2         (1886)
2340 PRINT:PRINT:PRINT"Press <cr>
to continue"                       (22BC)
2350 IF INKEY$<>cr$THEN GOTO 2350:
REM abfrage wiederholen            (291A)
2360 RESUME 1350                   (09BA)
```

Fortsetzung von Seite 123

7. Text abspeichern:

Hiermit wird der Text, der sich momentan im Array befindet, auf die Floppy geschickt. Der Dateiname ist die aktuelle Workfile.

8. Quit:

Mit der Wahl von Quit beenden Sie das Programm. Vorsicht! Beantworten Sie die Sicherheitsabfrage mit ja, so wird das Array automatisch gelöscht. Die Daten sind dann unwiderruflich verloren (wenn sie sich nicht auf der Floppy befinden).

```
Function: Menue                    Workfile: no file
MENUE

1. Text einladen
2. Text eingeben
3. Text aendern
4. Text sehen
5. Druckoptionen eingeben
6. Text ausdrucken
7. Text abspeichern
8. Quit

Bitte waehlen Sie (1..8) : 6
```

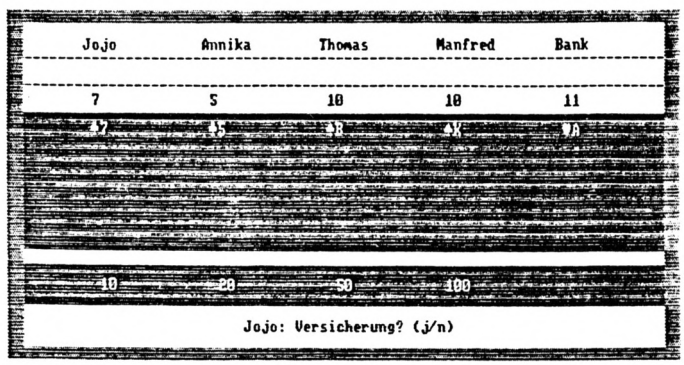
Das Programm wurde auf dem CPC 6128 erstellt. Es läuft genauso auf seinen kleineren Brüdern 464 und 664, weil auf die Verwendung von Maschinensprache verzichtet wurde.
Bei der Eingabe können alle REMs weggelassen werden; wichtig ist es jedoch, darauf zu achten, daß bei den Integervariablen das Prozentzeichen (%) angehängt wird. Dieser Variablentyp beschleunigt die Programmausführung. *Steffen Klaer*□

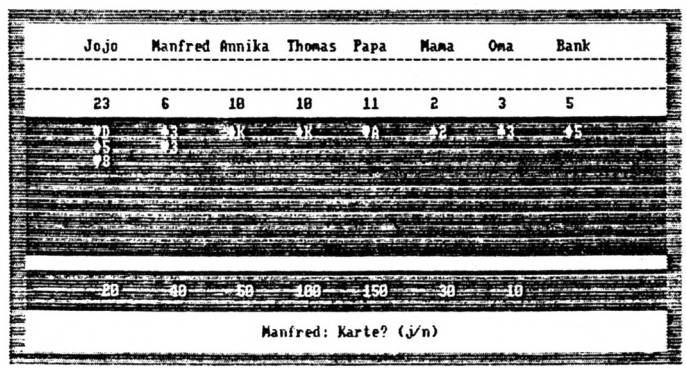
BLACKJACK

Risiko!

Blackjack wird in fast allen Spielbanken der Welt gespielt und ist dem 17 und 4 verwandt. Es werden sechs Kartenspiele zu je 52 Karten benötigt, bis zu sieben Teilnehmer können an einem Tisch spielen.

Im Gegensatz zu 17 und 4 bietet Blackjack verschiedene Kombinationen, die im Programm beschrieben sind.
Das Programm ist so aufgebaut, daß es den Spielverlauf im Casino exakt simuliert. Zum Beispiel muß der Tisch nicht voll besetzt sein; ein Spieler kann jederzeit ins Spiel einsteigen, falls noch ein Platz frei ist. Er kann auch jederzeit das Spiel wieder verlassen.
Nach dem Programmstart erscheint das Titelbild. Anschließend haben die Spieler die Wahl, die Farben zu korrigieren, die Spielregeln zu lesen oder sofort anzufangen. Bei Spielbeginn werden zuerst die Anzahl, dann die Namen der Spieler eingegeben.





Vor Ausgabe der Karten wird der Einsatz getätigt. Will ein Spieler passen, braucht er nur die Enter-Taste zu drücken.
An Spieler und Bank wird je eine Karte verteilt, danach erhält der erste Spieler sogleich die zweite Karte. Jetzt erfolgt die Abfrage nach Teilen, Verdoppeln, Ziehen usw., je nach Gegebenheit. Ist der erste Spieler bedient, kommt der nächste an die Reihe.
Ein Blackjack wird optisch und akustisch angezeigt. Nach einem Durchgang wird der neue Saldo für jeden Spieler angezeigt.
Durch Drücken der Enter-Taste wird das Spiel dann neu gestartet. Die Del-Taste ist zu drücken, wenn ein Spieler ausscheiden will und die Clr-Taste, wenn ein neuer aufgenommen werden soll.
Das Programm arbeitet bei weniger als vier Spielern im Mode 1, sonst im Mode 2. *JE*□

```
10 '*****************************  <24C3>
20 '*         BLACKJACK         *  <24EC>
30 '*            VON            *  <2407>
40 '*       R. BODINGER         *  <24B8>
50 '*           FUER            *  <247E>
55 '*     SCHNEIDER CPC-WELT     *  <24D8>
56 '*      CPC 464/664/6128   JE*  <24D6>
58 '*****************************  <2422>
60 'Spiel wird mit 6 Kartenspielen
 a 52 Karten gespielt              <3A7D>
62 'Nach Ausgabe von ca 200 Karten
 kommen abgelegte Karten           <3D9D>
64 'wieder ins Spiel. (Karten werd
en gemischt)                       <31D6>
66 '---------------------------    <24C7>
100 MODE 2:pa=1:pe=24:pe1=6:bo=10  <2A0C>
110 '--------------------------    <2322>
120 INK 0,0:INK 0,pa:INK 1,pe:INK
2,pe1:BORDER bo                    <2CFB>
130 '-----------------             <18CF>
140 '*** Titelbild ***             <188E>
150 '-----------------             <18F7>
160 MODE 1:ORIGIN 0,0:x=3:y=3      <1A2D>
170 MOVE 5,5:DRAWR 0,389:DRAWR 630
,0:DRAWR 0,-389:DRAWR-630,0        <27F8>
180 l=112:b=90:a=0:a1=0            <23E6>
190 a$="BLACKJACK"                 <15DD>
200 FOR i=1 TO 9:b$=MID$(a$,i,1)   <22DE>
210 FOR j=1 TO l:MOVE 19+a,272+(j-
a1):DRAWR b,0,1                    <3594>
220 NEXT                           <067A>
230 MOVE 111+a,272-a1:DRAWR-b-2,0,
2:DRAWR 0,l:DRAWR b+2,0:DRAWR 0,-l
                                   <3F55>
240 PAPER 1:PEN 2:LOCATE x,y:PRINT
 CHR$(227);"  ";CHR$(227)         <29CE>
250 LOCATE x,y+4:PRINT CHR$(227);"
  ";CHR$(227)                      <255C>
260 PLOT 5,5,1:TAG:MOVE 56+a,336-a
1:PRINT b$;:TAGOFF                 <29D0>
270 a=a+64:a1=a1+32:x=x+4:y=y+2:NE
XT:TAG                             <3C83>
280 MOVE 350,350:PRINT"Wie es in d
en";                               <1FF0>
290 MOVE 400,310:PRINT"Casinos";   <193A>
300 MOVE 120,110:PRINT"Europas";   <1785>
310 MOVE 70,70:PRINT"gespielt wird
";:TAGOFF                          <1F2C>
320 FOR i=1 TO 24:FOR k=1 TO 50:IN
K 0,i:NEXT:NEXT                    <266E>
330 FOR j=1 TO 1000:INK 0,i,2:NEXT
                                   <1CB4>
340 FOR i=0 TO 1000:NEXT:INK 0,pa:
FOR i=1 TO 1000:NEXT               <29FA>
350 PAPER 0:PEN 1:CLS:LOCATE 9,10:
PRINT"Farbkorrektur=Taste f,"      <2C6B>
360 LOCATE 10,12:PRINT"Spielregel=
ENTER-Taste,"                      <266C>
370 LOCATE 6,14:PRINT"ansonsten SP
ACE-Taste druecken."               <2DB1>
380 x$=INKEY$:IF x$<>CHR$(13)AND x
$<>CHR$(32)AND x$<>"f"THEN 380     <326B>
390 IF x$=CHR$(32)THEN 970         <15A1>
400 IF x$="f"THEN GOSUB 4030:GOTO
160                                <18A6>
410 '------------------            <1980>
420 '*** Spielregel ***            <19D2>
430 '------------------            <19A8>
440 MODE 2                         <071E>
450 LOCATE 16,1:PRINT"------------
--------------------------------
------                             <419E>
460 LOCATE 16,2:PRINT"Auszug der B
lackjack-Spielregel, Casino Baden
Baden.                             <4144>
470 LOCATE 16,3:PRINT"------------
--------------------------------
------                             <4105>
480 LOCATE 2,4:PRINT"Der Spieler s
pielt gegen den Croupier mit dem Z
iel, mit seinen Karten den Ge-     <599C>
490 LOCATE 2,5:PRINT"samtwert von
21 zu erreichen oder so nahe wie m
oeglich an 21 heranzukommen,       <575B>
500 LOCATE 2,6:PRINT"ohne dabei de
n Gesamtwert von 21 zu ueberschrei
ten. Asse zaehlen nach Wunsch      <587B>
510 LOCATE 2,8:PRINT"1 (eins) oder
 11 (elf), Bilder zaehlen 10 (zehn
) und alle anderen Karten ihren    <5AF9>
520 LOCATE 2,8:PRINT"normahlen Wer
t.                                 <1B78>
530 LOCATE 2,9:PRINT"Erst wenn all
e Einsaetze getaetigt sind, gibt d
er Croupier je eine Karte offen    <5A77>
540 LOCATE 2,10:PRINT"aus, zuerst
an jeden Spieler (im Uhrzeigersinn
), dann sich selbst eine. Danach   <5BA9>
550 LOCATE 2,11:PRINT"erhaelt jede
r Spieler im Gegensatz zum Croupie
r eine zweite, ebenfalls offene    <5A19>
560 LOCATE 2,12:PRINT"Karte.       <1337>
570 LOCATE 2,13:PRINT"Hat der Spie
ler BLACKJACK (ein As mit einem Bi
ld oder einer Zehn), so ist dies   <5BC5>
580 LOCATE 2,14:PRINT"die hoechste
 Paarung, die 3:2 ausbezahlt wird,
 wenn Sie nicht durch ein          <5417>
590 LOCATE 2,15:PRINT"BLACJACK des
 Croupiers ausgeglichen wird. Dies
 bedeutet dann stand-off           <53D5>
600 LOCATE 2,16:PRINT"(Punktegleic
hheit), d.h., der Satz bleibt steh
en und kann abgezogen oder ver-    <5A4F>
610 LOCATE 2,17:PRINT"aendert werd
en, da er weder gewinnt noch verli
```

```
ert. Erhaelt der Spieler keine    <5932>
620 LOCATE 2,18:PRINT"Karten, die
zusammen BLACKJACK ergeben, so ver
sucht er, so nah wie moeglich an  <5BB4>
630 LOCATE 2,19:PRINT"21 heranzuko
mmen; dazu fordert er vom Croupier
 mit den Worten ,,ziehen'' oder   <5A6E>
640 LOCATE 2,20:PRINT",,Karte'' so
 viele Karten, wie noetig sind, um
 dieses Ziel zu erreichen. Mit    <5990>
650 LOCATE 2,21:PRINT"dem Ausdruck
 ,,bleiben'' (,,danke'') gibt er z
u erkennen, dass er keine         <5466>
660 LOCATE 2,22:PRINT"weiteren Kar
ten wuenscht.                     <26CC>
670 LOCATE 2,23:PRINT"Sobald sich
alle Spieler erklaert haben, zieht
 der Croupier seine weiteren      <57C1>
680 LOCATE 2,24:PRINT"Karten nach
der Regel.                        <23FC>
690 LOCATE 30,25:PRINT"Eine Taste
druecken!"                        <2381>
700 CALL &BB06:CLS                <0BE0>
710 LOCATE 2,1:PRINT"Der Croupier
muss bis 16 oder darunter ziehen u
nd muss bei 17 oder darueber      <57E9>
720 LOCATE 2,2:PRINT"bleiben.     <1439>
730 LOCATE 2,3:PRINT"Ist der Spiel
er naeher an 21 als der Croupier,
so gewinnt er die Hoehe seines    <590F>
740 LOCATE 2,4:PRINT"Einsatzes. Is
t der Gesamtwert seiner Karten nie
driger als der des Croupiers,     <585C>
750 LOCATE 2,5:PRINT"so verliert e
r seinen Einsatz. Haben die Karten
 des Spielers den gleichen Ge-    <599E>
760 LOCATE 2,6:PRINT"samtwert wie
die Karten des Croupiers (,,stand-
off''), so kann der Spieler       <5625>
770 LOCATE 2,7:PRINT"seinen Einsat
z, der weder gewinnt noch verliert
, stehenlassen, zurueckziehen     <580B>
780 LOCATE 2,8:PRINT"oder veraende
rn. Ueberschreitet der Spieler wae
hrend des Ziehens den Gesamtwert  <5BA3>
790 LOCATE 2,9:PRINT"21, so ist se
in Satz verloren.                 <2A6A>
800 LOCATE 2,10:PRINT"Ergeben die
ersten beiden Karten des Spielers
den Gesamtwert neun, zehn oder    <5982>
810 LOCATE 2,11:PRINT"elf, kann er
 seinen urspruenglichen Einsatz ve
rdoppeln, erhaelt aber nur eine   <5AB1>
820 LOCATE 2,12:PRINT"zusaetzliche
 Karte (Double for Down).         <326F>
830 LOCATE 2,13:PRINT"Sind die bei
den ersten Karten, die der Spieler
 erhaelt, gleich, auch verschie-  <5B36>
840 LOCATE 2,14:PRINT"dener Farbe,
 so kann er sein Spiel (,,seine Ha
nd'') teilen (split). Er muss     <5895>
850 LOCATE 2,15:PRINT"dann aber de
n gleichen Einsatz noch einmal auf
 die geteilte Hand setzen und     <58F7>
860 LOCATE 2,16:PRINT"erhaelt dann
 fuer jede der beiden Haende belie
big viele Karten. Teilt der       <56AE>
870 LOCATE 2,17:PRINT"Spieler zwei
 Asse, so erhaelt er zu jedem As n
ur je eine zusaetzliche Karte.    <5970>
880 LOCATE 2,18:PRINT"In einem get
eilten Spiel ergiebt ein As mit ei
ner 10 (zehn) oder einem Bild     <58CD>
890 LOCATE 2,19:PRINT"kein  BLACKJ
ACK, sondern den Gesamtwert 21.   <38F1>
900 LOCATE 2,20:PRINT"Jeder Spiele
r kann sich in Hoehe seines Einsat
zes gegen ein BLACKJACK des       <5611>
910 LOCATE 2,21:PRINT"Croupiers ve
rsichern (wenn die erste Karte des
 Croupiers ein As ist).           <52EB>
920 LOCATE 2,22:PRINT"Die Einsaetz
e koennen nur in Zehnereinheiten g
etaetigt werden.                  <4BEB>
930 LOCATE 2,23:PRINT"Anmerkung de
s Programmautors: Der Versicherung
sbeitrag betraegt die Haelfte des <5C00>
940 LOCATE 2,24:PRINT"Einsatzes. H
ier gilt die Regel der Zehnereinhe
it nicht. Die Einsaetze sind in    <5A16>
950 LOCATE 2,25:PRINT"den meisten
Casinos auf DM 500.- beschraenkt.
 *** Eine Taste druecken! ***     <5828>
960 CALL &BB06                    <09CF>
970 MODE 1:RANDOMIZE TIME         <0B7E>
980 f=4:k=13:DIM n(f,k)'*** Karten
farbe, Kartenwert ***             <45E4>
990 LOCATE 7,10:PRINT"Anzahl der S
pieler eingeben!"                 <2AA3>
1000 LOCATE 12,13:PRINT"Maximal 7
Spieler."                         <218A>
1010 y$=INKEY$:IF y$<CHR$(49)OR y$
>CHR$(55)THEN 1010                <294D>
1020 m1=VAL(y$)                   <13F5>
1030 '--------------------        <1A59>
1040 '*** Disposition ***         <1A5E>
1050 '--------------------        <1A81>
1060 IF m1>3 THEN mo=2 ELSE mo=1'*
** MODE bestimmen ***             <36BD>
1070 MODE mo                      <0BA1>
1080 mm=(38\(m1+1))*mo:m2=mm-mo:c=
39*mo:m3=m1+1                     <4D00>
1090 WINDOW#1,1+mo,39*mo,3,3:WINDO
W#2,1+mo,39*mo,5,5:WINDOW#3,1+mo,3
9*mo,7,7                          <4FAA>
1100 WINDOW#4,1+mo,39*mo,9,17:WIND
```

```
OW#5,1+mo,39*mo,19,21:WINDOW#6,1+m
o,39*mo,23,23                          (5443)
1110 '--------------                   (15EB)
1120 '*** Rahmen ***                   (15DC)
1130 '--------------                   (1512)
1140 LOCATE 1,1:FOR i=1 TO 40*mo:P
RINT CHR$(207);:NEXT                   (259B)
1150 FOR i=2 TO 24:FOR j=1 TO mo:L
OCATE 39*mo+j,i:PRINT CHR$(207):NE
XT:NEXT                                (3D79)
1160 FOR i=2 TO 24:FOR j=1 TO mo:L
OCATE j,i:PRINT CHR$(207):NEXT:NEX
T                                      (34E8)
1170 LOCATE 1,25:FOR i=1 TO 40*mo:
PRINT CHR$(207);:NEXT                  (26CC)
1180 '----------------------------
-                                      (247E)
1190 '*** Eintragung der Spieler *
*                                      (2480)
1200 '----------------------------
-                                      (24A6)
1210 PAPER#1,0:PEN#1,1:PAPER#2,0:P
EN#2,1:PAPER#4,1:PEN#4,0:CLS#4:PAP
ER#5,1:PEN#5,0:CLS#5                   (3C1B)
1220 FOR i=273 TO 280:MOVE 16,i:DR
AW 623,i:NEXT                          (2702)
1230 name$(m3)="Bank"                  (1AC7)
1240 IF name=1 THEN GOSUB 1580:GOT
O 1300'*** bei Aenderung muessen n
icht alle Namen neu eingegeben wer
den ***                                (5F08)
1250 CLS#4:d=(c-30)/2:LOCATE#6,d,1
:PRINT#6,"Namen auf 7 Zeichen abku
erzen."                                (4917)
1260 FOR m=1 TO m1                     (122D)
1270 LOCATE#4,d,m:INPUT#4,"Name";n
ame$(m)                                (2B47)
1280 IF LEN(name$(m))>7 THEN name$
(m)=LEFT$(name$(m),7)                  (3B23)
1290 NEXT:name=1'*** Schalter bei
Aenderung ***                          (30EC)
1300 FOR m=1 TO m3                     (1286)
1310 LOCATE#1,mm*m+4*mo-mm,1:PRINT
#1,name$(m):NEXT                       (3737)
1320 LOCATE 1+mo,4:PRINT STRING$(3
8*mo,"-")                              (2106)
1330 LOCATE 1+mo,6:PRINT STRING$(3
8*mo,"-")                              (217B)
1340 '--------------------             (1AC6)
1350 '*** Spielbeginn ***              (1A02)
1360 '--------------------             (1AEE)
1370 GOSUB 1780'*** Einsatz und Au
sgabe der Ersten Karte ***             (3758)
1380 IF z(m3,1)=11 THEN GOSUB 2100
'*** Bank hat As. Blackjackversich
erung ***                              (479A)
1390 x$="":GOSUB 2190'*** Hauptpro
gramm ***                              (286B)
1400 GOSUB 2870'*** saldieren ***      (1C25)
1410 IF z$=CHR$(13)THEN 1370'*** E
NTER-Taste-naechstes Spiel ***         (3A70)
1420 IF z$=CHR$(127)THEN 1630'***
DEL-Taste-Spieleraustritt ***          (3853)
1430 '----------------------------
-                                      (2473)
1440 'Beitritt eines neuen Spieler
s, falls Platz frei                    (3613)
1450 '----------------------------
-                                      (249B)
1460 IF m3=8 THEN 1370'*** Kein Pl
atz frei ***                           (2A12)
1470 CLS#4:CLS#6:d=(c-30)/2:LOCATE
#6,d,1:PRINT#6,"Namen auf 7 Zeiche
n abkuerzen."                          (4D2D)
1480 auf$="Aufnahme eines neuen Sp
ielers."                               (2C9B)
1490 LOCATE#4,(c-LEN(auf$))/2,2:PR
INT#4,auf$                             (2954)
1500 nein$="Keine Aufnahme=ENTER"      (23A3)
1510 LOCATE#4,(c-LEN(nein$))/2,8:P
RINT#4,nein$                           (2BD5)
1520 LOCATE#4,8*mo^2,5:INPUT#4,"Na
me";name$                              (2796)
1530 IF LEN(name$)>7 THEN name$=LE
FT$(name$,7)                           (291F)
1540 IF name$=""THEN 1370              (1451)
1550 m1=m1+1:Saldo=saldo(m3):m3=m3
+1:saldo(m1)=0:saldo(m3)=saldo         (64AC)
1560 name$(m1)=name$                   (1B77)
1570 GOTO 1060                         (0971)
1580 FOR m=1 TO m3:LOCATE#2,mm*m+(
4*mo)-mm,1                             (33F9)
1590 PRINT#2,saldo(m):NEXT m:RETUR
N                                      (1F7B)
1600 '----------------------------
-                                      (24C9)
1610 '* Auscheiden eines Spielers
*                                      (2458)
1620 '----------------------------
-                                      (24F1)
1630 CLS#4:aus$="Auscheiden eines
Spielers"                              (2B87)
1640 LOCATE#4,(c-LEN(aus$))/2,2:PR
INT#4,aus$                             (291D)
1650 nein$="Kein Ausscheiden=ENTER
"                                      (25E3)
1660 LOCATE#4,(c-LEN(nein$))/2,8:P
RINT#4,nein$                           (2B03)
1670 LOCATE#4,8*mo^2,5:INPUT#4,"Na
me";name$                              (27C2)
1680 IF name$=""THEN 1370              (1468)
1690 FOR m=1 TO m3:IF name$(m)=nam
e$THEN 1720                            (2D86)
1700 NEXT:fehl$="Name nicht korrek
t":LOCATE#4,(c-LEN(fehl$))/2,10        (3FC7)
1710 PRINT#4,fehl$;CHR$(7):FOR j=1
```

```
 TO 2000:NEXT:GOTO 1630                    (294E)
1720 i=0:j=0:FOR m=1 TO m3:IF name
$(m)=name$THEN j=j+1:GOTO 1740             (48C5)
1730 i=i+1:name$(i)=name$(m):saldo
(i)=saldo(m)                               (4A20)
1740 NEXT m                                (0A1F)
1750 FOR i=1 TO m3-1:m=i:name$(m)=
name$(i):saldo(m)=saldo(i)                 (5809)
1760 NEXT i:m3=m1:m1=m1-j:GOTO 106
0                                          (2C13)
1770 '                                     (07E4)
1780 CLS#3:IF a<200 THEN 1860              (15E4)
1790 '-----------------------------
                                           (2349)
1800 'Wenn ca. 2 Drittel Karten au
sgegeben, dann Kartenpaket komplet
t                                          (4613)
1810 '-----------------------------
                                           (2372)
1820 FOR n=1 TO 6:FOR f=1 TO 4:FOR
 k=1 TO 13:n(f,k)=0:NEXT:NEXT:NEXT
:a=0                                       (4230)
1830 '---------------                     (167E)
1840 '*** Einsatz ***                      (1600)
1850 '---------------                     (16A6)
1860 CLS#5:CLS#6:a$="Einsatz Minim
um DM 10 Maximum DM 500"                   (38C2)
1870 c$="Nur Einsatz teilbar durch
 10!"                                      (2972)
1880 LOCATE#6,(c-LEN(a$))/2,1:PRIN
T#6,a$                                     (2583)
1890 FOR m=1 TO m1                         (121B)
1900 PEN#4,0:CLS#4:b$=name$(m)+":
Ihren Einsatz bitte"                       (391A)
1910 LOCATE#4,(c-LEN(b$)-4)/2,5:PR
INT#4,b$;:INPUT#4,satz(m)                  (3AE3)
1920 IF satz(m)=0 THEN satz(m)=1:G
OTO 1960                                   (2AAB)
1930 IF satz(m)<10 OR satz(m)>500
THEN LOCATE#4,(c-LEN(a$))/2,8:PRIN
T#4,a$;CHR$(7):FOR i=1 TO 1500:NEX
T:GOTO 1900                                (6297)
1940 IF satz(m)/10<>satz(m)\10 THE
N LOCATE#4,(c-LEN(c$))/2,8:PRINT#4
,c$;CHR$(7):FOR i=1 TO 1500:NEXT:G
OTO 1900                                   (61E8)
1950 GOSUB 3540                            (09EC)
1960 NEXT:CLS#4:CLS#6                      (0EEE)
1970 '-----------------------------
                                           (23B1)
1980 '*** Erste Karte ausgeben ***
                                           (23A1)
1990 '-----------------------------
                                           (23D9)
2000 i=1:FOR m=1 TO m3                     (196E)
2010 IF satz(m)=1 THEN 2060                (190B)
2020 GOSUB 3340'*** Karte ziehen *
**                                         (1FCA)
2030 z(m,i)=k:LOCATE#4,mm*m+4*mo+1
-mm,i:IF f=2 AND mo=1 OR f=3 AND m
o=1 THEN PEN#4,2 ELSE IF mo=2 OR f
=1 OR f=4 THEN PEN#4,0                     (8049)
2040 PRINT#4,CHR$(f+225);k$:s(m)=s
(m)+z(m,i)                                 (3F1D)
2050 LOCATE#3,mm*m+(4*mo)-mm,1:PRI
NT#3,s(m)                                  (3458)
2060 NEXT:RETURN                           (0856)
2070 '-----------------------------
-                                          (2477)
2080 'Gegen Blackjack der Bank ver
sichern                                    (2A90)
2090 '-----------------------------
-                                          (249F)
2100 FOR m=1 TO m1                         (12C1)
2110 IF satz(m)=1 THEN 2150                (19A7)
2120 v(m)=0:GOSUB 3480                     (1654)
2130 IF x$="n"THEN 2150                    (129F)
2140 v(m)=1                                (117B)
2150 NEXT:RETURN                           (080B)
2160 '-----------------------------
-                                          (242A)
2170 'Zweite und uebrige Karten au
sgeben                                     (29CF)
2180 '-----------------------------
-                                          (2452)
2190 FOR m=1 TO m3                         (127C)
2200 IF satz(m)=1 THEN z(m,i)=0:s(
m)=0:GOTO 2440                             (3976)
2210 FOR i=2 TO 9                          (0EB4)
2220 IF i<3 OR m=m3 OR q(m)=1 THEN
 2250                                      (288D)
2230 GOSUB 3460'*** Frage: Karte?
***                                        (203C)
2240 IF x$="n"THEN x$="":GOTO 2430
                                           (1B92)
2250 x$="":GOSUB 3340'*** Karte zi
ehen ***                                   (2701)
2260 z(m,i)=k:s(m)=s(m)+z(m,i):GOS
UB 2460'*** Wert ausgeben ***              (5B3F)
2270 IF q(m)=1 THEN 2430'*** Schal
ter. Bei Satzverdoppelung nur eine
 Karte ***                                 (4DB2)
2280 IF h(m)=0 AND z(m,1)=z(m,2)AN
D i=2 AND NOT m=m3 THEN GOSUB 2520
:GOTO 2420                                 (4924)
2290 IF i<>2 THEN 2370                     (10C1)
2300 IF s(m)=9 AND NOT m=m3 OR s(m
)=10 AND NOT m=m3 OR s(m)=11 AND N
OT m=m3 THEN GOSUB 3500                    (57C0)
2310 IF x$="j"THEN q(m)=1:satz(m)=
satz(m)*2                                  (3918)
2320 x$="":GOSUB 3540                      (1143)
2330 IF h(m)=0 AND s(m)=21 THEN GO
SUB 3570:GOTO 2360                         (2ABD)
2340 IF s(m3)=21 THEN GOSUB 3570:G
OTO 2360                                   (1E3B)
```

```
2350 GOTO 2370                                  (0989)
2360 LOCATE#3,mm*m+4*mo-mm,1:PRINT
#3,CHR$(24);s(m);CHR$(24):GOTO 244
0                                               (4583)
2370 IF s(m)>21 THEN GOSUB 3180:GO
SUB 2460                                        (1D75)
2380 IF s(m)=21 THEN 2430                       (17E3)
2390 IF s(m)>21 AND m<m3 THEN s(m)
=0:GOTO 2430                                    (30E9)
2400 IF s(m3)>21 THEN s(m3)=0:GOTO
 2450                                           (27D8)
2410 IF s(m3)>16 THEN 2450                      (1827)
2420 NEXT i                                     (0A61)
2430 j=1:IF h(m)=1 THEN GOSUB 2690
' *** h(m)=Schalter bei Kartenteil
ung ***                                         (48E4)
2440 NEXT m                                     (0A99)
2450 RETURN                                     (0653)
2460 LOCATE#4,mm*m+(4*mo)+1-mm,i:I
F f=2 AND mo=1 OR f=3 AND mo=1 THE
N PEN#4,2 ELSE IF mo=2 OR f=1 OR f
=4 THEN PEN#4,0                                 (6DCE)
2470 PRINT#4,CHR$(f+225);k$                     (1919)
2480 LOCATE#3,mm*m+(4*mo)-mm,1:PRI
NT#3,s(m):RETURN                                (36C6)
2490 '-----------------------------
-                                               (24BF)
2500 'Wenn ersten 2 Karten gleich,
 dann kann geteilt werden (split)               (44A9)
2510 '-----------------------------
-                                               (24E7)
2520 GOSUB 3490'*** Frage: Teilen?
 ***                                            (21B9)
2530 IF x$="n"AND z(m,1)=11 THEN G
OSUB 3180:GOSUB 2460'*** 2 Asse **
*                                               (388F)
2540 IF x$="n"THEN x$="":RETURN                 (18A7)
2550 v1(m)=v(m):x$=""'*** Bei Teil
ung Versicherung wie 1.Hand ***                 (5067)
2560 j=1:z1(m,j)=z(m,2):s(m)=s(m)/
2:s1(m)=s(m):z(m,2)=0:h(m)=1:i=i-1
                                                (8084)
2570 LOCATE#3,mm*m+(4*mo)-mm,1:PRI
NT#3,SPC(3)                                     (2EAE)
2580 LOCATE#3,mm*m+(4*mo)-mm,1:PRI
NT#3,s(m)                                       (347E)
2590 LOCATE#4,mm*m+(4*mo)-mm,2:PRI
NT#4,SPC(4)                                     (2E28)
2600 LOCATE#3,mm*m+(4*mo)+3-mm,1:P
RINT#3,s(m)                                     (36FF)
2610 LOCATE#4,mm*m+(4*mo)+4-mm,j:P
RINT#4,CHR$(f+225);k$                           (3F12)
2620 satz1(m)=satz(m):GOSUB 3550'*
** Satz ausgeben ***                            (3D27)
2630 IF NOT z(m,1)=11 THEN RETURN               (18F6)
2640 q(m)=1:q1(m)=1                             (1FA1)
2650 x$="":RETURN                               (0E12)
2660 '-----------------------------
-                                               (2414)
2670 'Nach Teilung Karten fuer zwe
ite Hand                                        (2B16)
2680 '-----------------------------
-                                               (243C)
2690 FOR j=2 TO 9                               (0EB6)
2700 IF j=2 OR q1(m)=1 THEN 2730                (1E1C)
2710 GOSUB 3470'Frage: Karte? *** (1CF1)
2720 IF x$="n"THEN 2830                         (1287)
2730 x$="":GOSUB 3340'*** Karte zi
ehen ***                                        (27C3)
2740 z1(m,j)=k:s1(m)=s1(m)+z1(m,j)
:LOCATE#4,mm*m+(4*mo)+4-mm,j:PRINT
#4,CHR$(f+225);k$:LOCATE#3,mm*m+(4
*mo)+3-mm,1:PRINT#3,s1(m)                       (B10F)
2750 IF q1(m)=1 THEN RETURN                     (158E)
2760 IF j<>2 THEN 2790                          (109F)
2770 IF s1(m)=9 OR s1(m)=10 OR s1(
m)=11 THEN GOSUB 3500                           (369F)
2780 IF x$="j"THEN q1(m)=1:satz1(m
)=satz1(m)*2:GOSUB 3550:GOTO 2820               (465F)
2790 IF s1(m)>21 THEN GOSUB 3260:L
OCATE#3,mm*m+(4*mo)+3-mm,1:PRINT#3
,s1(m)                                          (4CD9)
2800 IF s1(m)=21 THEN RETURN'*** K
artenabfrage entfaellt ***                      (37ED)
2810 IF s1(m)>21 THEN s1(m)=0:RETU
RN'*** ueberzogen ***                           (38BB)
2820 NEXT j                                     (0A88)
2830 x$="":RETURN                               (0E7C)
2840 '------------------                        (1805)
2850 '*** saldieren ***                         (1856)
2860 '------------------                        (182E)
2870 FOR m=1 TO m3                              (12D0)
2880 IF satz(m)=1 THEN 3010                     (199D)
2890 IF h(m)=0 AND z(m,1)+z(m,2)=2
1 AND NOT z(m3,1)+z(m3,2)=21 THEN
GOSUB 3090:GOTO 2950'*** Spieler h
at BLACKJACK nur, wenn keine Karte
nteilung ***                                    (963C)
2900 IF s(m)=0 OR s(m)<s(m3)THEN G
OSUB 3100:GOTO 2920'*** Satzverlus
t ***                                           (4837)
2910 IF s(m)>s(m3)THEN GOSUB 3120'
*** Satzgewinn ***                              (357B)
2920 IF h(m)=0 THEN 2950'*** keine
 zweite Hand ***                                (310E)
2930 IF s1(m)=0 OR s1(m)<s(m3)THEN
 GOSUB 3130:GOTO 2950'*** 2. Verlu
st ***                                          (4941)
2940 IF s1(m)>s(m3)THEN GOSUB 3140
'*** Satzgewinn ***                             (36B2)
2950 IF z(m3,1)<>11 THEN 3010'***
nichts versichert, da 1. Bankkarte
 kein As ***                                    (4E22)
2960 IF v(m)=1 THEN GOSUB 2990                  (175A)
2970 IF v1(m)=1 THEN GOSUB 2990                 (18C4)
2980 GOTO 3010                                  (0984)
```

```
2990 IF z(m3,2)=10 THEN GOSUB 3120
:RETURN'*** Bank zahlt Vers.-Praem
ie ***                                <3F1B>
3000 IF z(m3,2)<>10 THEN GOSUB 311
0:RETURN'*** Bank kassiert Versich
erung ***                             <41CE>
3010 s(m)=0:s1(m)=0:h(m)=0:satz(m)
=0:satz1(m)=0:v(m)=0:v1(m)=0:q(m)=
0:q1(m)=0:NEXT m:s(m3)=0              <9793>
3020 CLS#2:FOR m=1 TO m3:LOCATE#2,
mm*m+(4*mo)-mm,1                      <37D7>
3030 PRINT#2,saldo(m)'*** Ausgabe
der Salden ***                        <3342>
3040 NEXT m                           <0A4C>
3050 a$="Taste ENTER, DEL oder CLR
 druecken!"                           <2F4F>
3060 LOCATE#6,(c-LEN(a$))/2,1:PRIN
T#6,a$                                <25C0>
3070 z$=INKEY$:IF z$<>CHR$(13)AND
z$<>CHR$(127)AND z$<>CHR$(16)THEN
3070                                  <358B>
3080 RETURN                           <0643>
3090 saldo(m)=saldo(m)+satz(m)*1.5
:saldo(m3)=saldo(m3)-satz(m)*1.5:R
ETURN                                 <6E81>
3100 saldo(m)=saldo(m)-satz(m):sal
do(m3)=saldo(m3)+satz(m):RETURN       <604F>
3110 saldo(m)=saldo(m)-satz(m)/2:s
aldo(m3)=saldo(m3)+satz(m)/2:RETUR
N                                     <6482>
3120 saldo(m)=saldo(m)+satz(m):sal
do(m3)=saldo(m3)-satz(m):RETURN       <60F4>
3130 saldo(m)=saldo(m)-satz1(m):sa
ldo(m3)=saldo(m3)+satz1(m):RETURN     <62C1>
3140 saldo(m)=saldo(m)+satz1(m):sa
ldo(m3)=saldo(m3)-satz1(m):RETURN     <6276>
3150 '-----------------------------
-                                     <24EB>
3160 '1.Hand ueberzogen. Pruefen o
b AS dabei                            <2D90>
3170 '-----------------------------
-                                     <2412>
3180 i1=i:FOR i=1 TO i1               <1D49>
3190 IF z(m,i)=11 THEN z(m,i)=1:s(
m)=s(m)-10                            <43E0>
3200 IF s(m)<22 THEN 3220             <1744>
3210 NEXT                             <06E3>
3220 i=i1:RETURN                      <11CA>
3230 '-----------------------------
-                                     <248A>
3240 '2.Hand ueberzogen. Pruefen o
b AS dabei                            <2D93>
3250 '-----------------------------
-                                     <24B2>
3260 j1=j:FOR j=1 TO j1               <1D3D>
3270 IF z1(m,j)=11 THEN z1(m,j)=1:
s1(m)=s1(m)-10                        <4789>
3280 IF s1(m)<22 THEN 3300            <18FC>
3290 NEXT j                           <0A35>
3300 j=j1:RETURN                      <117F>
3310 '-----------------------         <1C2B>
3320 '*** Kartenausgabe ***           <1CEC>
3330 '-----------------------         <1C54>
3340 a=a+1:f=INT(RND*4)+1:LOCATE 1
,1:k=INT(RND*13)+1                    <36E9>
3350 IF n(f,k)<6 THEN n(f,k)=n(f,k
)+1:GOTO 3370                         <3EB7>
3360 a=a-1:GOTO 3340                  <1535>
3370 k$=STR$(k):k$=MID$(k$,2)         <2132>
3380 IF k$="1"THEN k$="A":k=11:GOT
O 3420                                <24DF>
3390 IF k$="11"THEN k$="B":k=10:GO
TO 3420                               <2509>
3400 IF k$="12"THEN k$="D":k=10:GO
TO 3420                               <251A>
3410 IF k$="13"THEN k$="K":k=10       <20B4>
3420 RETURN                           <06EC>
3430 '-----------------               <17E4>
3440 '*** Abfragen ***                <179B>
3450 '-----------------               <170C>
3460 frage$=name$(m)+": Karte? (j/
n)":GOTO 3510                         <317B>
3470 frage$=name$(m)+": Zweite Han
d Karte? (j/n)":GOTO 3510             <3D21>
3480 frage$=name$(m)+": Versicheru
ng? (j/n)":GOTO 3510                  <38F7>
3490 frage$=name$(m)+": Karten tei
len? (j/n)":GOTO 3510                 <39D5>
3500 frage$=name$(m)+": Einsatz ve
rdoppeln? (j/n)"                      <3906>
3510 LOCATE#6,(c-LEN(frage$))/2,1:
PRINT#6,frage$                        <2DF9>
3520 x$=INKEY$:IF x$<>"j"AND x$<>"
n"THEN 3520                           <23E0>
3530 CLS#6:RETURN                     <0A18>
3540 x$="":IF NOT m=m3 THEN LOCATE
#5,mm*m+(4*mo)-mm,2:PRINT#5,USING"
####";satz(m):RETURN                  <566A>
3550 IF NOT m=m3 THEN LOCATE#5,mm*
m+(4*mo)-mm,3:PRINT#5,USING"####";
satz1(m)                              <4D6F>
3560 RETURN                           <0603>
3570 ENV 1,1,120,1,120,-1,10:SOUND
 1,25,100,5,1:RETURN                  <2653>
4000 '-----------------------         <1E79>
4010 '*** Farbeinstellung ***         <1EB9>
4020 '-----------------------         <1EA1>
4030 MODE 1:DIM a$(26):FOR pa=0 TO
 26:READ a$(pa):NEXT:pa=1:pe=24:pe
1=6:bo=10:p=6                         <5694>
4040 INK 0,pa:INK 1,pe:INK 2,pe1:B
ORDER bo                              <27CB>
4050 LOCATE 16,2:PRINT"FARBWAHL":P
RINT                                  <184A>
4060 PRINT" e=Ende   Farben:
  Farbnummern:"                       <2D26>
```

```
4070 PRINT STRING$(39,"-")          (108B)
4080 LOCATE 1,p+6:PRINT STRING$(39
,"-")                               (1AF7)
4090 WINDOW#1,2,19,18,24:WINDOW#2,
20,39,18,24                         (23F8)
4100 LOCATE 2,p+1:PRINT"(p)apier="
a$(pa):LOCATE 26,p+1:PRINT"INK   0
,";pa                               (43E1)
4110 LOCATE 2,p+3:PRINT"(f)eder= "
a$(pe):LOCATE 26,p+3:PRINT"INK   1
,";pe                               (43A3)
4120 LOCATE 2,p+5:PRINT"(F)eder1="
a$(pe1):LOCATE 26,p+5:PRINT"INK
2,";pe1                             (45A6)
4130 LOCATE 2,p+7:PRINT"(b)order="
a$(bo):LOCATE 26,p+7:PRINT"BORDER
";bo                                (4260)
4140 PAPER#1,1:CLS#1:PAPER#2,2:CLS
#2                                  (187E)
4150 PEN#1,0:LOCATE#1,2,4:PRINT#1,
a$(pe):PEN#2,0:LOCATE#2,2,4:PRINT#
2,a$(pe1)                           (417F)
4160 x$=INKEY$:IF x$=""THEN 4160    (19CD)
4170 IF x$="p"THEN pa=pa+1:num=pa:
GOSUB 4220:pa=num:INK 0,pa:GOTO 41
00                                  (4941)
4180 IF x$="f"THEN pe=pe+1:num=pe:
GOSUB 4220:pe=num:INK 1,pe:GOTO 41
00                                  (4994)
4190 IF x$="F"THEN pe1=pe1+1:num=p
e1:GOSUB 4220:pe1=num:INK 2,pe1:GO
TO 4100                             (4EED)
4200 IF x$="b"THEN bo=bo+1:num=bo:
GOSUB 4220:bo=num:BORDER bo:GOTO 4
100                                 (47E8)
4210 IF x$="e"THEN GOTO 4240        (1330)
4220 IF num>26 THEN num=0           (181C)
4230 FOR i=1 TO 5 STEP 2:LOCATE 10
,p+i:PRINT STRING$(22," "):NEXT i:
RETURN                              (32AC)
4240 LOCATE 7,p+10:PRINT"Farbeinst
ellung OK? (j/n)"                   (2C27)
4250 x$=LOWER$(INKEY$):IF x$=""THE
N 4250 ELSE IF x$="j"THEN CLS:RETU
RN                                  (2CE5)
4260 INK 0,1:INK 1,24:BORDER 1:LOC
ATE 7,p+10:PRINT STRING$(32," "):G
OTO 4040                            (2E36)
4270 DATA SCHWARZ,BLAU,LEUCHTBLAU,
ROT,MAGENTA,MAUVE,LEUCHTROT         (3ABA)
4280 DATA PURPUR,LEUCHTMAGENTA,GRU
EN,BLAUGRUEN,HIMMELBLAU,GELB,WEISS
                                    (4122)
4290 DATA PASTELLBLAU,ORANGE,ROSA,
PASTELLMAGENTA,LEUCHTGRUEN,SEEGRUE
N                                   (423A)
4300 DATA LEUCHTBLAUGRUEN,LIMONENG
RUEN,PASTELLGRUEN,PASTELLBL.GRUEN (408B)
4310 DATA LEUCHTGELB,PASTELLGELB,L
EUCHTWEISS                          (29CA)
```

DATEIVERWALTUNG AUF DEM 6128

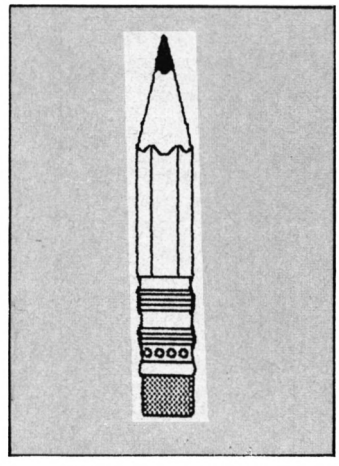
Ich besitze einen Schneider CPC 6128 und arbeite seit einem Jahr mit der Dateiverwaltung Profi-Data. Da man auf dem CPC 6128 nur mit Drei-Zoll-Disketten arbeiten kann, möchte ich gerne wissen, mit welchen Zusatzgeräten man andere Diskettenformate verwenden kann.

Vor kurzem habe ich erfahren müssen, daß auch teure Disketten oft nichts taugen. Wenn ich an einem bestimmten Punkt in meiner Dateiverwaltung angelangt war, nahm die Diskette nichts mehr auf, und ich erhielt die Meldung, die Diskette sei voll – was auch stimmte. Vorher gingen 516 Datensätze auf eine Seite; jetzt waren es nur noch 344, obwohl ich immer das gleiche machte. Nachdem ich zehn Disketten durchprobiert hatte, lief alles wieder normal ab. Ich nehme daher an, daß der Fehler bei den Disketten lag. Nun meine Frage: Kann man vor der Benutzung feststellen, ob eine Diskette defekt ist?
Ein Lob übrigens für Eure Zeitschrift CPC-WELT. Ihr solltet nur Eure Listings besser drucken; ich sitze manchmal Tage, bis ich die Fehler gefunden habe, die nur darauf beruhen, daß ich ein Zeichen nicht lesen konnte. Aber alles andere ist spitze.

Ilse Behrens,
6642 Hettlach

Ein Zweitlaufwerk für alle Schneider CPCs ist von der Firma Vortex Datensysteme, Falterstraße 51–53, 7101 Flein bei Heilbronn, erhältlich. Damit können Sie auf 5,25-Zoll-Disketten arbeiten.

Der von Ihnen geschilderte Fehler bei der Dateiverwaltung kann unserer Meinung nach kaum auf fehlerhaften Disketten beruhen, da Sie sonst eine andere Fehlermeldung erhalten hätten. Wahrscheinlich handelt es sich um einen Fehler im Programm oder um einen geringfühgigen Bedienungsfehler, der diese Folgen nach sich gezogen hat. Wenn eine Diskette fehlerhaft ist, so wird dies in der Regel bereits beim Formatieren festgestellt. Wird sie erst im Lauf der Zeit schadhaft, erkennen Sie dies daran, daß sich Dateien nicht einlesen lassen oder kein Inhaltsverzeichnis mehr angezeigt wird. Eine Meldung, die Diskette sei voll, ist sehr unwahrscheinlich – zumal bei zehn verschiedenen Disketten.
Ihr Lob für unsere Zeitschrift freut uns. Was den Druck und die Fehlerfreiheit bei Listings anbetrifft, so haben Sie sicher bemerkt, daß wir uns um eine bessere Qualität bemüht haben. Mit dem jetzt verwendeten Zeichensatz, der Verwechslungen von Zeichen weitgehend ausschließt, und unserem CPC-WELT-Checksummer, glauben wir eine sehr gute Lösung gefunden zu haben. □



DER GREIFER

Leckerbissen

Diesmal haben wir einen Leckerbissen für Sie: Ein Spiel, das uns beim Testen auf Anhieb gefallen hat. Es ist schnell, grafisch sehr gut gestaltet und bietet Ihnen zudem noch die Möglichkeit, eigene Levels nach persönlichem Geschmack zu gestalten.

Sie steuern mit dem Joystick einen Greifarm durch Gänge und sammeln dabei Punkte auf. Das hört sich leichter an als es ist, denn leider gibt es eine Vielzahl von Monstern, die Sie nicht berühren dürfen. Da gibt es solche, die sich nur horizontal bewegen, andere laufen vertikal. Außerdem treffen Sie auf eine Anzahl von Feuerstellen, die Sie ebenfalls umgehen müssen. Zum Glück verschwinden diese für einige Sekunden, bevor sie wieder aufflackern. Dann müssen Sie aber den Greiferarm schon wieder zurückgezogen ha-



ben. Dies geschieht durch Drücken des Feuerknopfes. Nach dem Start erscheint das Hauptmenü. Hier können Sie auswählen, ob Spielstufen geladen, abgespeichert oder editiert werden sollen oder ob gleich mit dem Spielen begonnen wird.

Spielstufen laden:

Bis zu drei verschiedene Spielstufen können, wenn sie vorher durch das Editormenü gestaltet worden sind, geladen werden. Jede Spielstufe kann bis zu 24 Level enthalten. In der Datei "Daten1.dat" sind in der Spielstufe eins schon die ersten drei Level definiert, so daß Sie sofort beginnen können.
Wir empfehlen Ihnen, zunächst mit diesen drei Level (in Spielstufe 1; Datei: daten1.dat) zu spielen, bevor Sie an das Gestalten von eigenen Levels herangehen, denn erst nach den ersten Spielversuchen werden Sie erkennen, wo die Schwierigkeiten liegen.
Achtung: Spielstufe 2 und 3 sind nicht definiert! Das können Sie nach eigenem Geschmack selbst durchführen. Daher diese Spielstufen beim Laden nicht aufrufen, das Programm bricht sonst mit einer Fehlermeldung ab!

Spielstufen speichern:

Zum Abspeichern eigener Levels.

Spielen:

Wird vorher der Punkt Laden nicht aufgerufen, befindet sich automatisch Spielstufe 1 im Speicher!

Editormenü:

Ein Untermenü, das keine Wünsche zum Gestalten eigener Level mehr offenläßt. Sie können Mauern definieren (in verschiedenen Farben und Formen) und Monster gestalten, wobei Sie entscheiden, ob diese sich horizontal oder vertikal bewegen. Die Bewegung der Monster endet jeweils an der nächsten Mauer. Außerdem können Sie Flammen setzen und Punkte verteilen, die dann aufgesammelt werden müssen. Der Wert der Punkte kann ebenfalls eingestellt werden.

Mauern:

Es können bis zu 54 verschiedene Mauern angewählt und gesetzt werden. Beachten Sie, daß im Programm nur einige voreingestellt sind!

Monster:

Bis zu zehn verschiedene Monster können horizontal und ebenfalls zehn vertikal gesetzt werden. Auch hier sind noch nicht alle definiert. Sie haben also selbst Gestaltungsmöglichkeiten!

Feuer:

Zum Verteilen der Flammen über das Bild.

Editor:

Zum Selbstdefinieren der Monster und der Mauern. Sie kommen danach in ein weiteres Untermenü, wo Sie völlig freie Gestaltungsmöglichkeiten haben.

Löschen:

Zum Entfernen von Monstern, Mauern usw. aus dem Bild.

Zeit:

Einstellen des Zeitlimits für die einzelnen Levels.

Startposition:

Hier legen Sie den Startpunkt und die Richtung des Greifarmes fest.

Farben:

Jedem Level können eigene Farben zugeordnet werden. Lediglich die ersten fünf sind unveränderbar.

Test:

Bei diesem Punkt können Sie das selbstgestaltete Spielfeld austesten und notfalls nochmals ändern.

Ende:

Rückkehr zum Menü.

Alle Funktionen werden mit dem Joystick aufgerufen. Ein Level ist nur dann spielbar, wenn mindestens eins

Lesen Sie bitte weiter auf Seite 145

```
100 '*************************** <2234>
110 '*       DER GREIFER         * <22E3>
120 '*           VON             * <227F>
130 '*      LOTHAR SCHOLZ        * <22AC>
140 '*           FUER            * <2214>
150 '*    SCHNEIDER CPC-WELT     * <2233>
160 '*     CPC 464/664/6128   JE* <2242>
170 '*************************** <22C0>
180 'BENOETIGT WERDEN DIE FILES (D
ATEN1.DAT) UND (GREIFER.BIN)      <405B>
190 MEMORY 19999:setcur=&A295:delc
ur=&A2DC:zeichen=&A2F9:level=&A340
:CLS:PRINT"Bitte Diskette/Kassette
 mit Datei":PRINT"GREIFER.BIN einl
egen !":PRINT:PRINT"Press any key"
:CALL &BB06:LOAD"greifer.bin"     <A9E7>
200 PRINT:PRINT"Bitte Diskette/Kas
sette mit Datei":PRINT"DATEN1.DAT
einlegen !":PRINT:PRINT"Press any
key":CALL &BB06:LOAD"daten1.dat":G
OSUB 1630                         <6F62>
210 GOTO 1320                     <09EE>
220 tes=0:EI:FOR i=41160 TO 41173:
POKE i,0:NEXT:lev=0:leb=5:GOTO 240
                                  <4413>
230 PAPER 0:PEN 1:CLS:LOCATE 5,15:
PRINT"LEVEL : ";lev+1:LOCATE 5,10:
PRINT"LIVES : ";leb:FOR i=1 TO 400
0:NEXT:CLS                        <50BF>
240 '                             <07EA>
250 adr=27900+lev*492:CALL 24999,a
dr:IF PEEK(41165)<>0 OR PEEK(41161
)=0 THEN lev=lev+1:IF lev=25 AND t
es=0 THEN 1320 ELSE IF tes=-1 THEN
 RETURN ELSE 250                  <7CB7>
260 POKE 41166,0:POKE 41170,0:POKE
 41168,31:POKE 41169,155:a=PEEK(39
708)+256*PEEK(39709):a=a-adr+40200
:GOSUB 1620:POKE 39708,l:POKE 3970
9,h:ti=PEEK(adr+491):sc=PEEK(41171
)+256*PEEK(41172)                 <B994>
270 PAPER 0:PEN 1:LOCATE 1,1:PRINT
 SPACE$(20):LOCATE 1,1:PRINT"SC:";
USING"#####";sc:LOCATE 15,1:PRINT"
TI:";USING"###";ti:EVERY 50,1 GOSU
B 340                             <5627>
280 FOR i=1 TO 40:NEXT:CALL &6291:
CALL &6520:CALL &6720:FOR i=1 TO 4
0:NEXT:CALL &6720:CALL &6380:IF PE
EK(41165)<>0 THEN 310             <4988>
290 IF PEEK(41173)<>0 THEN DI:LOCA
TE 4,1:PRINT USING"#####";PEEK(411
71)+256*PEEK(41172):EI:POKE 41173,
0:SOUND 7,478,5,10,0,0,15         <5C67>
300 GOTO 280                      <0914>
310 POKE 41173,0:a=REMAIN(1):IF PE
EK(41165)=67 THEN lev=lev+1:IF lev
=25 AND tes=0 THEN 1320 ELSE IF te
s THEN lev=lev-1:RETURN ELSE 240  <7060>
320 FOR i=1 TO 3000:NEXT:SOUND 131
,0,100,15:ON SQ(1)GOSUB 350       <2705>
330 leb=leb-1:IF leb=0 THEN GOTO 3
60 ELSE CLS:LOCATE 5,15:PRINT"LEVE
L : ";lev+1:LOCATE 5,10:PRINT"LIVE
S : ";leb:FOR i=1 TO 4000:NEXT:CLS
:GOTO 250                         <6F3A>
340 ti=ti-1:IF ti=-1 THEN POKE 411
65,255:RETURN ELSE LOCATE 18,1:PRI
NT USING"###";ti:RETURN           <40E8>
350 RETURN                        <06E4>
360 WINDOW 5,15,10,15:CLS:MOVE 144
,248:DRAWR 316,0,1:DRAWR 0,-80:DRA
WR-316,0:DRAWR 0,80:MOVE 136,252:D
RAWR 332,0:DRAWR 0,-88:DRAWR-332,0
:DRAWR 0,88:PEN 3:LOCATE 2,3:PRINT
"GAME OVER":FOR i=1 TO 4000:NEXT:I
F tes=0 THEN 1320 ELSE RETURN     <8E81>
370 a$=BIN$(a,4):b$=BIN$(b,4):c$=M
ID$(a$,4,1)+MID$(b$,4,1)+MID$(a$,2
,1)+MID$(b$,2,1)+MID$(a$,3,1)+MID$
(b$,3,1)+MID$(a$,1,1)+MID$(b$,1,1)
:c=VAL("&x"+c$):RETURN            <9E90>
380 GOSUB 1390                    <094D>
390 PAPER 0:PEN 1:MODE 0:PRINT TAB
(7);"Menue":MOVE 0,384:DRAWR 640,0
,1:adr=27900+(lev-1)*492:fadr=adr+
480:z=5:FOR i=fadr TO fadr+10:INK
z,PEEK(i):z=z+1:NEXT:CALL level,ad
r                                 <A6DA>
400 GOSUB 1440:GOTO 410           <0EEB>
410 CALL delcur:IF y=0 THEN GOSUB
460:IF w=11 THEN GOTO 1320 ELSE IF
 w=5 OR w=10 THEN 390 ELSE CALL le
vel,adr:GOTO 400                  <5639>
420 IF s<>255 THEN 430 ELSE IF y<>
24 THEN a=adr+x+(y-1)*20:IF(PEEK(a
)>67 AND PEEK(a)<148)OR(PEEK(a+20)
>67 AND PEEK(a)<148)THEN 450 ELSE
POKE a,64+mon*4:CALL zeichen,x,y,6
4+mon*4:POKE a+20,65+mon*4:CALL ze
ichen,x,y+1,65+mon*4:GOTO 450 ELSE
 450                              <EC7E>
430 a=adr+x+(y-1)*20:b=PEEK(a):IF
b>67 AND b<148 THEN IF b MOD 2=1 T
HEN POKE a-20,0:CALL zeichen,x,y-1
,0 ELSE POKE a+20,0:CALL zeichen,x
,y+1,0                            <95C3>
440 POKE a,s:CALL zeichen,x,y,s   <2A03>
450 GOSUB 1450:GOTO 410           <0E72>
460 WINDOW 5,15,3,25:PAPER 2:CLS:P
EN 1:LOCATE 3,2:PRINT"Mauer":LOCAT
E 3,4:PRINT"Monster":LOCATE 3,6:PR
INT"Feuer":LOCATE 3,8:PRINT"Punkte
":LOCATE 3,10:PRINT"Editor":LOCATE
 3,12:PRINT"Loeschen":LOCATE 3,14:
PRINT"Zeit":LOCATE 3,16:PRINT"Star
```

```
tpos."                                  (950F)
470 LOCATE 3,18:PRINT"Farben":LOCA
TE 3,20:PRINT"Test":LOCATE 3,22:PR
INT"Ende"                               (308F)
480 GOSUB 1450:CALL delcur:IF x<4
OR x>14 OR y<3 OR y>23 OR(y-3)MOD
2=1 THEN 480 ELSE w=(y-3+2)/2:ON w
 GOTO 490,540,660,670,720,1100,111
0,1160,1240,1210,1230                   (8923)
490 PAPER 3:WINDOW 8,18,7,12:CLS:P
EN 0:LOCATE 3,4:PRINT" 1.":s=1:LOC
ATE 4,2:PRINT"Mauern":mk=0:CALL ze
ichen,14,9,s                            (570F)
500 j=JOY(0):FOR i=1 TO 50:NEXT:IF
(j AND 1)=1 THEN mk=-1:s=s+1:IF s=
55 THEN s=54                            (4ED5)
510 IF(j AND 2)=2 THEN mk=-1:s=s-1
:IF s=0 THEN s=1                        (3484)
520 IF mk THEN LOCATE 3,4:PRINT US
ING"##";s:CALL zeichen,14,9,s:mk=0
                                        (3AF9)
530 IF(j AND 16)=16 THEN RETURN EL
SE 500                                  (19CA)
540 PAPER 3:PEN 0:WINDOW 8,19,7,12
:CLS:LOCATE 4,2:PRINT"Monster":LOC
ATE 2,4:PRINT"vertikal":LOCATE 2,5
:PRINT"horizontal"                      (4BC9)
550 GOSUB 1450:CALL delcur:IF y=9
AND x>6 AND x<19 THEN 560 ELSE IF
y=10 AND x>6 AND x<19 THEN 610 ELS
E 550                                   (517B)
560 CLS:LOCATE 4,2:PRINT"Monster":
LOCATE 2,5:PRINT"Nr.":a=1:LOCATE 5
,5:PRINT USING"##";a:CALL zeichen,
14,9,68:CALL zeichen,14,10,69:CALL
 zeichen,16,9,70:CALL zeichen,16,1
0,71                                    (8C35)
570 mk=0:j=JOY(0):FOR i=1 TO 150:N
EXT:IF(j AND 1)=1 THEN a=a+1:mk=-1
:IF a=11 THEN a=10                      (56B4)
580 IF(j AND 2)=2 THEN a=a-1:mk=-1
:IF a=0 THEN a=1                        (34AE)
590 IF(j AND 16)=16 THEN s=255:mon
=a:RETURN                               (28AD)
600 IF NOT mk THEN 570 ELSE LOCATE
 5,5:PRINT USING"##";a:b=64+4*a:CA
LL zeichen,14,9,b:CALL zeichen,14,
10,b+1:CALL zeichen,16,9,b+2:CALL
zeichen,16,10,b+3:GOTO 570              (9681)
610 CLS:LOCATE 4,2:PRINT"Monster":
LOCATE 2,5:PRINT"Nr.":a=1:LOCATE 5
,5:PRINT USING"##";a:CALL zeichen,
14,9,108:CALL zeichen,14,10,109:CA
LL zeichen,16,9,110:CALL zeichen,1
6,10,111                                (8C50)
620 mk=0:j=JOY(0):FOR i=1 TO 150:N
EXT:IF(j AND 1)=1 THEN a=a+1:mk=-1
:IF a=11 THEN a=10                      (5619)
630 IF(j AND 2)=2 THEN a=a-1:mk=-1
:IF a=0 THEN a=1                        (3413)
640 IF(j AND 16)=16 THEN s=255:mon
=a+10:RETURN                            (2B0C)
650 IF NOT mk THEN 620 ELSE LOCATE
 5,5:PRINT USING"##";a:b=104+4*a:C
ALL zeichen,14,9,b:CALL zeichen,14
,10,b+1:CALL zeichen,16,9,b+2:CALL
 zeichen,16,10,b+3:GOTO 620             (9601)
660 s=61:RETURN                         (0EA8)
670 PAPER 3:WINDOW 8,17,10,14:CLS:
PEN 0:LOCATE 2,2:PRINT"Punkte :":L
OCATE 5,4:PRINT" 5":a=1                 (3BA8)
680 mk=0:j=JOY(0):FOR i=1 TO 100:N
EXT:IF(j AND 1)=1 THEN a=a+1:mk=-1
:IF a=6 THEN a=5                        (54C8)
690 IF(j AND 2)=2 THEN a=a-1:mk=-1
:IF a=0 THEN a=a+1                      (39D2)
700 IF(j AND 16)=16 THEN s=62+a:RE
TURN                                    (2103)
710 IF mk THEN LOCATE 5,4:PRINT US
ING"##";a*5:GOTO 680 ELSE 680           (28F6)
720 PAPER 3:WINDOW 8,16,10,14:CLS:
PEN 0:LOCATE 2,2:PRINT"Monster":LO
CATE 2,4:PRINT"Mauer":mk=0              (3E73)
730 GOSUB 1450:CALL delcur:IF x<7
OR x>15 THEN 730 ELSE IF y<>10 AND
 y<>12 THEN 730 ELSE IF y=10 THEN
890                                     (4C20)
740 MODE 0:PAPER 0:PEN 1:FOR i=1 T
O 8:PRINT STRING$(8,144):NEXT:LOCA
TE 12,7:a$=CHR$(150)+CHR$(154)+CHR
$(156):PEN 2:PRINT a$"  "a$:a$=CHR
$(149)+" "+CHR$(149):LOCATE 12,8:P
RINT a$"  "a$:a$=CHR$(147)+CHR$(15
4)+CHR$(153):LOCATE 12,9                (9F52)
750 PRINT a$"  "a$:PEN 1:LOCATE 13
,8:PRINT CHR$(242);STRING$(4,9);CH
R$(243):FOR i=0 TO 15:LOCATE 2+i,2
0:PAPER i:PRINT" ":NEXT:PAPER 0:MO
VE 28,80:DRAWR 0,16,1:DRAWR 516,0:
DRAWR 0,-18:DRAWR-516,0                 (7D03)
760 PAPER 3:PEN 0:WINDOW 10,20,1,5
:CLS:PRINT CHR$(10);CHR$(9);"Alte
Form":PRINT CHR$(10);CHR$(9);"Nr.
1":z=1:CALL zeichen,17,3,z:PAPER 0
:PEN 1:WINDOW 1,20,1,25:LOCATE 1,1
2:PRINT"Neue":PRINT"Form:":f=1:LOC
ATE 2+f,22:PRINT CHR$(240)              (A62A)
770 PEN 2:a$=CHR$(150)+STRING$(4,1
54)+CHR$(156):b$=CHR$(149)+STRING$
(4,154)+CHR$(149):c$=CHR$(147)+STR
ING$(4,154)+CHR$(153):LOCATE 13,11
:PRINT a$:LOCATE 13,12:PRINT b$:LO
CATE 13,13:PRINT c$:LOCATE 13,14:P
RINT a$:LOCATE 13,15:PRINT b$:LOCA
TE 13,16                                (A3C7)
780 PRINT c$:PEN 1:LOCATE 14,12:PR
```

```
INT"O.K.":LOCATE 14,15:PRINT"ENDE"
:GOSUB 1440                              (30A8)
790 CALL delcur:IF x<8 AND y<8 THE
N LOCATE x+1,y+1:PEN f:PRINT CHR$(
143):PLOT 192+x*4,206-y*2,f:mk2=-1
                                         (5FEA)
800 IF y=19 AND x>0 AND x<17 THEN
LOCATE 2+f,22:PRINT " ":f=x-1:PEN 1
:LOCATE 2+f,22:PRINT CHR$(240)           (4E54)
810 IF(y=7 OR y=8 OR y=6)AND(x=16
OR x=17 OR x=18)THEN PEN 1:IF mk2
THEN LOCATE 1,1:FOR i=1 TO 8:PRINT
 STRING$(8,144):NEXT:LOCATE 7,13:m
k2=0:PRINT" "ELSE ELSE 830               (773E)
820 z=z+1:IF z=55 THEN z=54 ELSE L
OCATE 15,4:PAPER 3:PEN 0:PRINT USI
NG"##";z:PAPER 0:CALL zeichen,17,3
,z                                       (539F)
830 IF(y=7 OR y=8 OR y=6)AND(x=11
OR x=12 OR x=13)THEN PEN 1:IF mk2
THEN LOCATE 1,1:FOR i=1 TO 8:PRINT
 STRING$(8,144):NEXT:LOCATE 7,13:m
k2=0:PRINT" "ELSE ELSE 850               (7798)
840 z=z-1:IF z=0 THEN z=1 ELSE LOC
ATE 15,4:PAPER 3:PEN 0:PRINT USING
"##";z:PAPER 0:CALL zeichen,17,3,z
                                         (51A8)
850 IF(y=13 OR y=14 OR y=15)AND(x>
11 AND x<18)THEN RETURN                  (33F3)
860 IF y<5 AND x>8 THEN CALL zeich
en,6,12,z:b=1:a=1:FOR j=350 TO 336
 STEP-2:FOR i=544 TO 572 STEP 4:PE
N TEST(i,j):LOCATE b,a:PRINT CHR$(
143):b=b+1:NEXT:b=1:a=a+1:NEXT:mk2
=-1                                      (A7C9)
870 IF(y=10 OR y=11 OR y=12)AND(x>
11 AND x<18)THEN za=20000+z*32:FOR
 i=1 TO 8:FOR j=1 TO 4:a=TEST(j*64
-40,405-i*16):b=TEST(j*64-5,405-i*
16):GOSUB 370:POKE za,c:za=za+1:NE
XT:NEXT:CALL zeichen,17,3,z              (CFD6)
880 GOSUB 1450:GOTO 790                  (0EBB)
890 PAPER 0:PEN 1:WINDOW 8,19,7,12
:CLS:LOCATE 4,2:PRINT"Monster":LOC
ATE 2,4:PRINT"vertikal":LOCATE 2,5
:PRINT"horizontal"                       (4B1A)
900 GOSUB 1450:CALL delcur:IF y=9
AND x>6 AND x<19 THEN e=64 ELSE IF
 y=10 AND x>6 AND x<19 THEN e=104
ELSE 890                                 (59B1)
910 MODE 0:FOR i=1 TO 16:PRINT STR
ING$(8,144):NEXT                         (1E22)
920 FOR i=0 TO 15:LOCATE 2+i,22:PA
PER i:PRINT" ":NEXT:PAPER 0:MOVE 2
8,48:DRAWR 0,16,1:DRAWR 516,0:DRAW
R 0,-18:DRAWR-516,0:f=1:LOCATE 2+f
,24:PRINT CHR$(240)                      (6904)
930 n=1:WINDOW 10,20,1,6:PEN 1:PAP
ER 2:CLS:PRINT CHR$(10);CHR$(9);"A
lte Form":LOCATE 2,5:PRINT USING"#
#.";n:WINDOW 10,20,8,13:PEN 0:PAPE
R 3:CLS:PRINT CHR$(10);CHR$(9);"Ne
ue Form":LOCATE 2,4                      (7DDE)
940 a$=CHR$(150)+CHR$(154)+CHR$(15
6):b$=CHR$(149)+" "+CHR$(149):c$=C
HR$(147)+CHR$(154)+CHR$(153):WINDO
W 1,20,1,25:PAPER 0:PEN 2:LOCATE 1
2,15:PRINT a$;a$:LOCATE 12,16:PRIN
T b$;b$:LOCATE 12,17:PRINT c$;c$         (96B2)
950 a$=CHR$(150)+STRING$(4,154)+CH
R$(156):b$=CHR$(149)+"    "+CHR$(1
49):c$=CHR$(147)+STRING$(4,154)+CH
R$(153):LOCATE 9,18:PRINT a$;a$:LO
CATE 9,19:PRINT b$;b$:LOCATE 9,20:
PRINT c$;c$                              (89E4)
960 PEN 1:LOCATE 13,16:PRINT CHR$(
242)CHR$(9);CHR$(9);CHR$(243):LOCA
TE 10,19:PRINT"O.K.";CHR$(9);CHR$(
9);"Ende":LOCATE 15,11:PRINT" ";CH
R$(9);" ":LOCATE 15,12:PRINT" "CHR
$(9)" ":d=n*4+e:CALL zeichen,14,3,
d                                        (9B97)
961 CALL zeichen,14,4,d+1:CALL zei
chen,16,3,d+2:CALL zeichen,16,4,d+
3:mk2=0:p=1                              (5CE3)
970 GOSUB 1450:CALL delcur:IF(y=17
 OR y=18 OR y=19)AND x>13 THEN RET
URN                                      (3921)
980 IF y=21 AND x>0 AND x<17 THEN
PAPER 0:LOCATE 2+f,24:PRINT" ":f=x
-1:PEN 1:LOCATE 2+f,24:PRINT CHR$(
240)                                     (5119)
990 IF x<8 AND y<16 THEN PEN f:LOC
ATE x+1,y+1:PRINT CHR$(143):mk2=-1
:a=238-y*2:IF p=1 THEN PLOT x*4+45
0,a,f ELSE PLOT x*4+512,a,f              (7FD1)
1000 IF NOT((y=14 OR y=15 OR y=16)
AND(x=14 OR x=15 OR x=16))THEN 102
0 ELSE p=1:IF mk2=-1 THEN mk2=0:LO
CATE 1,1:PEN 1:FOR i=1 TO 16:PRINT
 STRING$(8,144):NEXT:LOCATE 15,11:
PRINT" ";CHR$(9);" ":LOCATE 15,12:
PRINT" ";CHR$(9);" "                     (A79F)
1010 n=n+1:IF n=11 THEN n=10 ELSE
d=n*4+e:CALL zeichen,14,3,d:CALL z
eichen,14,4,d+1:CALL zeichen,16,3,
d+2:CALL zeichen,16,4,d+3:LOCATE 1
1,5:PAPER 2:PRINT USING"##.";n           (A76A)
1020 IF NOT((y=14 OR y=15 OR y=16)
AND(x=11 OR x=12 OR x=13))THEN 104
0 ELSE p=1:IF mk2=-1 THEN mk2=0:LO
CATE 1,1:PEN 1:FOR i=1 TO 16:PRINT
 STRING$(8,144):NEXT:LOCATE 15,11:
PRINT" ";CHR$(9);" ":LOCATE 15,12:
PRINT" ";CHR$(9);" "                     (A7F7)
1030 n=n-1:IF n=0 THEN n=1 ELSE d=
```

```
n*4+e:CALL zeichen,14,3,d:CALL zei
chen,14,4,d+1:CALL zeichen,16,3,d+
2:CALL zeichen,16,4,d+3:LOCATE 11,
5:PAPER 2:PRINT USING"##.";n      <A5A7>
1040 IF NOT((y=17 OR y=18 OR y=19)
AND x>7 AND x<14)THEN 1070 ELSE za
=(e+n*4)*32+20000:IF p=2 THEN za=z
a+64:p=1 ELSE p=2                 <7765>
1050 FOR i=1 TO 16:FOR j=1 TO 4:a=
TEST(j*64-40,405-i*16):b=TEST(j*64
-5,405-i*16):GOSUB 370:POKE za,c:z
a=za+1:NEXT:NEXT:CALL zeichen,14,3
,d:CALL zeichen,14,4,d+1:CALL zeic
hen,16,3,d+2:CALL zeichen,16,4,d+3
                                  <D91F>
1060 LOCATE 1,1:PAPER 0:PEN 1:FOR
i=1 TO 16:PRINT STRING$(8,144):NEX
T:LOCATE 15,11:PRINT" ";CHR$(9);"
":LOCATE 15,12:PRINT" ";CHR$(9);"
"                                 <52BE>
1070 IF x=14 AND(y=3 OR y=4)THEN a
=1:b=1:FOR j=350 TO 320 STEP-2:FOR
 i=450 TO 480 STEP 4:PEN TEST(i,j)
:LOCATE a,b:PRINT CHR$(143):a=a+1:
NEXT:a=1:b=b+1:NEXT:g=0:h=1:GOSUB
1600:mk2=-1                       <AE1A>
1080 IF x=16 AND(y=3 OR y=4)THEN a
=1:b=1:FOR j=350 TO 320 STEP-2:FOR
 i=512 TO 542 STEP 4:PEN TEST(i,j)
:LOCATE a,b:PRINT CHR$(143):a=a+1:
NEXT:a=1:b=b+1:NEXT:g=2:h=3:GOSUB
1600:mk2=-1                       <AE3A>
1090 GOTO 970                     <09D8>
1100 s=0:RETURN                   <0D5D>
1110 WINDOW 10,17,11,16:PAPER 3:CL
S:PEN 0:LOCATE 2,2:PRINT"Zeit :":L
OCATE 3,4:ti=PEEK(adr+491):PRINT U
SING"###";ti                      <5097>
1120 mk=0:j=JOY(0):FOR i=1 TO 50:N
EXT:IF(j AND 1)=1 THEN ti=ti+5:mk=
-1:IF ti=260 THEN ti=255          <5BC7>
1130 IF(j AND 2)=2 THEN ti=ti-5:mk
=-1:IF ti=5 THEN ti=10            <3999>
1140 IF(j AND 16)=16 THEN POKE adr
+491,ti:RETURN                    <2661>
1150 IF mk THEN LOCATE 3,4:PRINT U
SING"###";ti:GOTO 1120 ELSE 1120  <28EB>
1160 WINDOW 3,19,12,18:PAPER 3:CLS
:PEN 0:LOCATE 3,2:PRINT"Startricht
ung":a$=CHR$(150)+CHR$(154)+CHR$(1
56):b$=CHR$(149)+" "+CHR$(149):c$=
CHR$(147)+CHR$(154)+CHR$(153):PEN
2                                 <7C5C>
1170 LOCATE 2,4:PRINT a$" "a$" "a$
" "a$:LOCATE 2,5:PRINT b$" "b$" "b
$" "b$:LOCATE 2,6:PRINT c$" "c$" "
c$" "c$:LOCATE 3,5:PRINT CHR$(240)
;STRING$(3,9);CHR$(241);STRING$(3,
9);CHR$(242);STRING$(3,9);CHR$(243
)                                 <9E67>
1180 GOSUB 1450:CALL delcur:IF y<>
14 AND y<>15 AND y<>16 THEN 1180 E
LSE IF x<3 OR x>17 OR x=6 OR x=10
OR x=14 THEN 1180                 <5DB9>
1190 IF x<6 THEN s=151 ELSE IF x>6
 AND x<10 THEN s=152 ELSE IF x>10
AND x<14 THEN s=153 ELSE s=154    <5045>
1200 RETURN                       <068A>
1210 EI:PAPER 0:PEN 1:MODE 0:FOR i
=41160 TO 41173:POKE i,0:NEXT:leb=
5                                 <3651>
1220 lev=lev-1:tes=-1:GOSUB 230:le
v=lev+1:RETURN                    <35FD>
1230 RETURN                       <06C6>
1240 PAPER 3:WINDOW 2,19,12,17:CLS
:PEN 0:LOCATE 8,2:PRINT"Farben":FO
R i=0 TO 15:LOCATE 2+i,4:PAPER i:P
RINT" ":NEXT:MOVE 61,160:DRAWR 0,1
6,0:DRAWR 516,0:DRAWR 0,-18:DRAWR-
516,0                             <6EB3>
1250 GOSUB 1450:CALL delcur:IF NOT
(y=14 AND x>6 AND x<18)THEN 1250 E
LSE a=x-2:PAPER 3:LOCATE 2+a,5:PRI
NT CHR$(240)                      <5584>
1260 WINDOW 7,18,17,20:PAPER 4:CLS
:LOCATE 2,2:PRINT"Neue":LOCATE 2,3
:PRINT"Farbe":PAPER a:LOCATE 8,2:P
RINT"    ":LOCATE 8,3:PRINT"    ":
b=0:INK a,0                       <5FB6>
1270 j=JOY(0):FOR i=1 TO 150:NEXT:
IF(j AND 1)=1 THEN b=b+1:IF b=27 T
HEN b=26 ELSE INK a,b             <51A3>
1280 IF(j AND 2)=2 THEN b=b-1:IF b
=-1 THEN b=0 ELSE INK a,b         <3862>
1290 IF(j AND 16)=16 THEN a=a-5:PO
KE fadr+a,b:RETURN ELSE 1270      <382C>
1300 GOTO 1300                    <09D7>
1310 END                          <06A3>
1320 INK 0,0:INK 1,26:INK 2,6:INK
3,24:PAPER 0:MODE 0:BORDER 0:PEN 1
:LOCATE 5,1:PRINT"Der Greifer"    <3A74>
1330 PAPER 2:WINDOW 4,16,3,22:CLS:
FOR i=0 TO 3:WINDOW 5,15,i*5+4,i*5
+6:PAPER 3:CLS:NEXT:WINDOW 1,20,1,
25:PEN 0:LOCATE 6,5:PRINT"Speicher
n":LOCATE 8,10:PRINT"Laden":LOCATE
 7,15:PRINT"Spielen":LOCATE 7,20:P
RINT"Editor"                      <8DE2>
1340 GOSUB 1440:GOTO 1360         <0E05>
1350 GOSUB 1450                   <09C8>
1360 IF x<4 OR x>14 OR y<3 OR y>20
 OR y=7 OR y=6 OR y=11 OR y=12 OR
y=17 OR y=16 THEN CALL delcur:GOTO
 1350                             <6129>
1370 w=1+(y-1)\5                  <166E>
1380 ON w GOTO 1520,1580,220,380  <1A1C>
```

```
1390 PAPER 0:PEN 1:CLS:LOCATE 7,5:
PRINT"LEVEL :":INK 4,18:PAPER 4:WI
NDOW 6,14,9,15:CLS:WINDOW 7,13,10,
14:PAPER 5:INK 5,4:CLS:PEN 1:LOCAT
E 4,3:PRINT"1":lev=1                    <5EBA>
1400 mk=0:j=JOY(0):FOR i=1 TO 100:
NEXT:IF(j AND 1)=1 THEN mk=-1:lev=
lev+1:IF lev=25 THEN lev=24             <5E89>
1410 IF(j AND 2)=2 THEN mk=-1:lev=
lev-1:IF lev=0 THEN lev=1               <3C1F>
1420 IF mk THEN LOCATE 3,3:PRINT U
SING"##";lev                            <1EB1>
1430 IF(j AND 16)=16 THEN RETURN E
LSE 1400                                <1909>
1440 x=1:y=1                            <12EB>
1450 CALL setcur,x,y                    <19C8>
1460 j=JOY(0):FOR i=1 TO 50:NEXT:I
F(j AND 2)=2 THEN mk=-1:y=y+1:IF y
=25 THEN y=24                           <4E63>
1470 IF(j AND 4)=4 THEN mk=-1:x=x-
1:IF x=-1 THEN x=0                      <35F2>
1480 IF(j AND 1)=1 THEN mk=-1:y=y-
1:IF y=-1 THEN y=0                      <35B8>
1490 IF(j AND 8)=8 THEN mk=-1:x=x+
1:IF x=20 THEN x=19                     <3682>
1500 IF mk THEN mk=0:CALL delcur:C
ALL setcur,x,y                          <3382>
1510 IF(j AND 16)=16 THEN RETURN E
LSE 1460                                <198B>
1520 PAPER 0:PEN 1:CLS:LOCATE 5,5:
PRINT"Speichern :":INK 4,18:PAPER
4:WINDOW 6,14,9,15:CLS:WINDOW 7,13
,10,14:PAPER 5:INK 5,4:CLS:PEN 1:L
OCATE 2,2:PRINT"Daten":LOCATE 4,4:
PRINT"1":d=1:GOSUB 1540:a$="daten"
+CHR$(48+d)+".dat":SAVE a$,b,20000
,20200                                  <A6CF>
1530 GOTO 1320                          <0944>
1540 j=JOY(0):FOR i=1 TO 50:NEXT:I
F(j AND 1)=1 THEN mk=-1:d=d+1:IF d
=10 THEN d=9                            <4DCC>
1550 IF(j AND 2)=2 THEN mk=-1:d=d-
1:IF d=0 THEN d=1                       <3439>
1560 IF mk THEN LOCATE 3,4:PRINT d
:mk=0                                   <1EE5>
1570 IF(j AND 16)=16 THEN RETURN E
LSE 1540                                <1983>
1580 PAPER 0:PEN 1:CLS:LOCATE 7,5:
PRINT"Laden :":INK 4,18:PAPER 4:WI
NDOW 6,14,9,15:CLS:WINDOW 7,13,10,
14:PAPER 5:INK 5,4:CLS:PEN 1:LOCAT
E 2,2:PRINT"Daten":LOCATE 4,4:PRIN
T"1":d=1:GOSUB 1540:a$="daten"+CHR
$(48+d)+".dat":LOAD a$                  <9552>
1590 GOTO 1320                          <09BD>
1600 IF p=1 THEN CALL zeichen,14,1
0,d+g:CALL zeichen,14,11,d+h ELSE
CALL zeichen,16,10,d+g:CALL zeiche
n,16,11,d+h                             <7D60>
1610 RETURN                             <06C1>
1620 h=INT(a/256):l=a-h*256:RETURN
                                        <2B47>
1630 b=0:FOR i=0 TO 25:a=40199+i*2
0:GOSUB 1620:POKE 41200+b,l:POKE 4
1201+b,h:a=40220+i*20:GOSUB 1620:P
OKE 41300+b,l:POKE 41301+b,h:b=b+2
:NEXT:RETURN                            <A089>

10 '****************************      <2397>
15 '* DATALADER 1 FUER GREIFER *      <230C>
20 '*   ERZEUGT <DATEN1.DAT>   *      <2347>
25 '*          VON             *      <236B>
30 '*      LOTHAR SCHOLZ       *      <239F>
35 '*          FUER            *      <2398>
40 '*    SCHNEIDER CPC-WELT    *      <2314>
45 '*     CPC 464/664/6128   JE*      <23AA>
50 '****************************      <23E7>
60 PAPER 0:PEN 1:INK 0,0:INK 1,25:
BORDER 0:MODE 2:LOCATE 10,10:PRINT
"Bitte warten MC-Datei  Daten1.dat
 wird erzeugt !!"                       <5762>
70 MEMORY 19999:adr=20000:FOR k=0
TO 877:READ a:IF a>0 THEN 120 ELSE
 110                                    <3788>
80 NEXT k:GOTO 130                      <0FF1>
110 a=ABS(a):READ a$:w=VAL("&"+a$)
:FOR i=1 TO a:POKE adr,w:adr=adr+1
:NEXT:zae=zae+1:GOTO 80                 <6B01>
120 FOR i=1 TO a:READ a$:w=VAL("&"
+a$):POKE adr,w:adr=adr+1:NEXT:zae
=zae+1:GOTO 80                          <5D0F>
130 SAVE"daten1.dat",b,20000,20000
                                        <1FF9>
1000 DATA -32,0,2,C3,82,-3,C3,1,82
,-3,C3,1,82,-2,C3,-4,0,-3,C3,1,82,
-3,C3,1,82,-3,C3,1,82,-4,0,-4,F0,-
4,C0,-4,F0,-4,C0,-32,F0,-4,C0,-4,F
0,-12,0,1,40,-2,C0,2                    <99DA>
1010 DATA 94,40,-3,3C,1,40,-3,3C,1
,40,-3,3C,1,40,-3,3C,1,14,-3,3C,1,
14,-3,3C,-168,0,3,C0,0,40,-5,80,-3
,C0,3,80,0,40,-3,0,1,40,-3,0,1,40,
-2,0,1,40,-2,C0                         <9455>
1020 DATA 2,0,40,-2,C0,-2,0,2,C0,8
0,-2,0,2,C0,80,-2,0,2,C0,80,-2,0,1
,40,-3,0,1,40,-14,0,3,C0,0,40,-5,8
0,-3,C0,3,80,0,40,-3,0,1,40,-3,0,1
,40,-2,0                                <8D10>
1030 DATA 1,40,-2,C0,2,0,40,-2,C0,
-2,0,2,C0,80,-2,0,2,C0,80,-2,0,2,C
0,80,-2,0,1,40,-3,0,1,40,-1319,0,2
,40,80,-2,0,2,40,80,-2,0,2,40,80,-
2,0,2,40,80                             <90AA>
1040 DATA -2,0,2,40,80,-2,0,2,40,8
0,-2,0,2,40,80,-2,0,2,40,80,-13,0,
-8,C0,-13,0,2,40,80,-2,0,2,40,80,-
```

```
2,0,3,40,80,0,-2,C0,2,80,0,-2,C0,-
15,0,2,40,80                                    <9111>
1050 DATA -2,0,2,40,80,-2,0,2,40,8
0,-2,0,1,40,-2,C0,-2,0,-2,C0,-26,0
,-2,C0,2,0,40,-2,C0,3,0,40,80,-2,0
,2,40,80,-2,0,2,40,80,-13,0,-2,C0,
-2,0,-2,C0,1                                    <91CA>
1060 DATA 80,-2,0,2,40,80,-2,0,2,4
0,80,-2,0,2,40,80,-3,0,1,8,-2,0,2,
4,8,-2,0,2,4,8C,-2,0,8,4C,8C,0,4,4
C,CC,8,4,-2,CC,4,8,0,4C,8C,-2,0,-2
,C                                              <877D>
1070 DATA -2,0,1,4,-3,0,1,C,-3,0,1
4,C,8,0,4,4C,8,0,4,CC,8,0,4,CC,8C,
-2,0,2,4C,8C,-2,0,-2,C,-6,0,-2,C0,
-2,0,2,84,48,-2,0,2,84,48,-2,0,2,8
4                                               <864B>
1080 DATA 48,-2,0,-2,C0,-14,0,-2,C
0,-2,0,2,80,40,-2,0,2,80,40,-2,0,2
,80,40,-2,0,-2,C0,-14,0,-2,C0,-2,0
,2,C4,C8,-2,0,2,C4,C8,-2,0,2,C4,C8
,-2,0,-2,C0,-14,0                               <96F2>
1090 DATA -2,C0,-2,0,-2,C0,-2,0,-2
,C0,-2,0,-2,C0,-2,0,-2,C0,-14,0,-2
,C0,-2,0,2,90,60,-2,0,2,90,60,-2,0
,2,90,60,-2,0,-2,C0,-11,0,1,8,-2,0
,2,4,8,-2,0                                     <900F>
1100 DATA 2,4,8C,-2,0,8,4C,8C,0,4,
4C,CC,8,4,-2,CC,4,8,0,4C,8C,-2,0,-
2,C,-2,0,1,4,-3,0,1,C,-3,0,14,C,8,
0,4,4C,8,0,4,CC,8,0,4,CC,8C,-2,0    <839E>
1110 DATA 2,4C,8C,-2,0,-2,C,2,0,C0
,-63,0,5,44,0,88,44,0,-3,88,3,0,4,
8C,-2,0,2,44,CC,-2,0,2,44,4C,-3,0,
1,88,-3,0,3,88,0,44,-3,CC,3,44,0,8
,-2,44                                          <8B85>
1120 DATA 10,0,8,44,4,50,F0,4,0,50
,F0,-2,0,-2,50,-2,0,-2,50,-2,0,-2,
50,-2,0,4,F0,50,A0,0,-3,88,1,0,-3,
88,3,0,4,8C,-2,0,2,44,CC,-2,0,2,44
,4C,-3,0                                        <8D4E>
1130 DATA 1,88,-3,0,1,88,-2,0,-2,C
C,16,88,44,88,8,CC,44,0,8,44,4,0,A
0,4,0,50,F0,-2,0,2,50,F0,-2,0,2,50
,F0,-3,0,1,A0,-2,0,-2,F0,1,A0,-102
5,0,1,1,-3,0                                    <9193>
1140 DATA 4,3,2,0,1,-2,B,1,0,-3,3,
4,2,1,7,3,-2,0,2,3,2,-2,0,1,54,-3,
0,1,54,-3,0,3,FC,A8,0,-3,FC,2,A8,5
4,-2,FC,-2,0,2,FC,A8,-2,0,1,54,-3,
0                                               <86FD>
1150 DATA 10,FC,A8,0,54,A8,FC,0,54
,0,54,-2,0,1,1,-3,0,4,3,2,0,1,-2,B
,1,0,-3,3,4,2,1,7,3,-2,0,2,7,2,-2,
0,1,54,-3,0,1,54,-2,0,1,FC,-2,54    <8355>
1160 DATA 1,A8,-3,54,2,0,54,-2,FC,
2,0,54,-2,FC,2,0,54,-2,FC,-2,0,2,F
C,A8,-2,0,1,54,-3,0,2,FC,A8,-10,0,
2,51,A2,-2,0,4,A2,51,0,51,-2,0,-2,
A2,-2,0,18                                      <8F03>
1170 DATA 51,A2,51,A2,51,A2,51,A2,
51,A2,51,A2,51,A2,51,A2,51,A2,-2,0
,-2,51,-2,0,4,A2,0,A2,51,-2,0,2,51
,A2,-18,0,2,51,A2,-2,0,4,B7,7B,0,5
1,-2,3F,2,A2,B7,-2,F3                           <9A9C>
1180 DATA 2,7B,B7,-2,F3,2,7B,B7,-2
,F3,2,7B,B7,-2,F3,2,7B,B7,-2,F3,2,
7B,B7,-2,F3,2,7B,51,-2,3F,4,A2,0,B
7,7B,-2,0,2,51,A2,-1034,0,2,40,80,
-2,0,2,40,80,-2,0                               <96C2>
1190 DATA -2,C0,4,0,C0,84,48,-2,C0
,4,84,48,C0,0,-2,C0,-2,0,2,40,80,-
2,0,2,40,80,-2,0,2,40,80,-2,0,-2,C
0,6,0,40,48,84,80,40,-2,C0,2,80,40
,-2,C0,6,80,40                                  <9358>
1200 DATA 48,84,80,0,-2,C0,-2,0,2,
40,80,-6,0,-2,C0,5,0,40,48,84,80,-
8,C0,5,40,48,84,80,0,-2,C0,-5,0,5,
40,80,40,80,84,-2,0,2,48,C0,-2,0,-
2,C0,-2,0,-2,C0                                 <94D9>
1210 DATA -2,0,2,C0,84,-2,0,2,48,4
0,-2,C0,4,80,0,40,80,-2,0,4,40,80,
0,40,-2,C0,2,80,84,-2,0,2,48,C0,-2
,0,-2,C0,-2,0,-2,C0,-2,0,2,C0,84,-
2,0,6,48,40                                     <9014>
1220 DATA 80,40,80,40,-2,C0,5,0,84
,C0,84,80,-3,0,1,80,-3,0,1,C0,-3,0
,1,C0,-3,0,6,80,84,C0,84,80,40,-2,
C0,-2,0,-2,C0,6,80,40,48,C0,48,40,
-3,0,1,C0,-3,0                                  <938D>
1230 DATA 1,C0,-3,0,1,40,-3,0,5,40
,48,C0,48,0,-2,C0,1,80,-39,0,54,DD
,E5,CD,FB,61,DD,E1,3A,CA,A0,FE,1,2
0,40,3A,C9,A0,FE,29,30,39,3A,C8,A0
,B7,28,33,3E,1,CD,40                            <9979>
1240 DATA A3,3E,16,32,CC,A0,3E,3D,
32,CB,A0,97,32,CD,A0,32,CE,A0,DD,6
E,0,DD,66,-2,1,169,E0,1,9,6,A,E,5,
C5,79,46,4E,E5,CD,32,BC,E1,C1,23,C
,10,F2,C9,3E,FF,32,CD                           <9AD0>
1250 DATA A0,C9,DD,6E,0,DD,66,1,97
,32,C8,A0,32,CA,A0,32,C9,A0,DD,21,
8,9D,FD,21,E8,9E,11,5C,6C,1,E0,1,7
E,FE,3F,DA,7D,62,FE,44,DA,46,62,FE
,94,DA,50,62,3A,CA,A0,3C                        <9D86>
1260 DATA 32,CA,A0,7E,32,1E,9B,FD,
77,0,FD,23,97,DD,77,0,DD,23,22,1C,
9B,23,C3,8A,62,3A,C8,A0,3C,32,C8,A
0,C3,7D,62,CB,2F,DA,7D,62,3A,C9,A0
,3C,32,C9,A0,FE,29,D2,7D,62                     <A034>
1270 DATA E5,DD,E5,E1,7D,12,13,7C,
12,13,E1,7E,FE,6C,38,4,3E,2,18,2,3
E,4,12,13,3E,1,12,13,7E,DD,77,0,97
,FD,77,0,DD,23,FD,-2,23,91,B,78,B1
,C2,19,62,C9,3A,CC,A0                           <9AF3>
1280 DATA 3C,32,CC,A0,FE,29,20,C,9
```

```
7,32,CC,A0,3A,CB,A0,4F,6,A0,18,1D,
FE,15,D8,28,11,3A,CB,A0,FE,3D,28,5
,4F,6,3D,18,C,4F,6,3E,18,7,6,3D,E,
A0,C3,F0,62,78,32,CB                    <9928>
1290 DATA A0,32,CF,A0,79,32,E1,62,
21,3,63,22,E3,62,21,8,9D,11,50,C0,
1,E0,1,7E,FE,3D,CC,3,63,-4,13,135,
23,B,78,B1,20,F0,C9,78,32,CB,A0,32
,CF,A0,79,32,E1,62,21,26                <9DBC>
1300 DATA 63,22,E3,62,18,D3,C5,3A,
CF,A0,77,E5,D5,FE,3D,28,E,FE,3E,28
,F,21,20,4E,CD,5A,63,D1,E1,C1,C9,2
1,C0,55,18,F4,21,E0,55,18,EF,C5,3A
,CF,A0,77,E5,F5,1,E0,1,9                <9D95>
1310 DATA 7E,B7,20,1E,F1,D5,FE,3D,
28,E,FE,3E,28,F,21,20,4E,CD,5A,63,
D1,E1,C1,C9,21,C0,55,18,F4,21,E0,5
5,18,EF,3E,FF,32,CD,A0,18,DB,EB,6,
8,C5,E5,6,4,1A,77,23,13                 <9C9D>
1320 DATA 10,FA,E1,1,0,8,9,C1,10,E
E,C9,-16,0,59,3A,C9,A0,DD,21,5C,6C
,47,C5,DD,66,1,DD,6E,0,DD,7E,2,FE,
1,28,17,FE,2,28,18,FE,3,28,19,CD,9
2,64,DD,23,DD,23,DD                     <9834>
1330 DATA 23,DD,23,C1,10,DC,C9,CD,
B8,64,18,EF,CD,DF,64,18,EA,CD,70,6
4,18,-2,E5,105,1,8,9D,A7,ED,42,CB,
25,CB,14,CB,25,CB,14,1,50,C0,9,EB,
E1,C9,E5,D5,21,20,4E,CD,5A              <9F2F>
1340 DATA 63,E1,1,50,0,9,EB,21,20,
4E,CD,5A,63,E1,E5,97,77,1,14,0,9,7
7,E1,C9,E5,77,1,14,0,9,3C,77,E1,3D
,6,0,4F,E5,D5,11,20,0,CD,80,A2,1,2
0,4E,9,D1,D5,E5                         <940F>
1350 DATA CD,5A,63,E1,D1,EB,1,50,0
,9,EB,1,20,0,9,CD,5A,63,E1,C9,DD,7
3,0,DD,72,-2,1,43,E0,1,D5,EB,9,7E,
1,14,0,9,46,EB,D1,B7,28,32,3E,FF,3
2,CD,A0,D5,CD,BC                        <95B8>
1360 DATA 63,7E,F5,CD,D2,63,DD,7E,
3,FE,1,28,A,3E,1,DD,77,3,F1,-2,3D,
8,18,8,3E,2,DD,77,3,F1,-2,3C,191,D
1,F5,EB,CD,BC,63,F1,CD,F1,63,C9,78
,B7,28,CF,18,C8,1,14                    <9916>
1370 DATA 0,E5,A7,ED,42,E5,1,8,9D,
A7,ED,42,E1,38,A,7E,EB,E1,B7,20,5,
CD,21,64,C9,E1,3E,4,DD,77,2,C9,1,2
8,0,E5,9,E5,1,E8,9E,A7,ED,42,E1,30
,10,7E,1,14,0,A7                        <9501>
1380 DATA ED,42,EB,E1,B7,20,5,CD,2
1,64,C9,E1,3E,3,DD,77,2,C9,FD,21,F
0,A0,2B,CD,6,65,28,17,7E,E5,1,14,0
,9,46,E1,B7,20,C,78,B7,20,8,EB,62,
6B,23,CD,21,64,C9,3E                    <9958>
1390 DATA 2,DD,77,2,C9,FD,21,54,A1
,23,CD,6,65,28,17,7E,E5,1,14,0,9,4
6,E1,B7,20,C,78,B7,20,8,EB,62,6B,2
B,CD,21,64,C9,3E,1,DD,77,2,C9,6,18
,FD,5E,0,FD,56,1                        <95DD>
1400 DATA E5,A7,ED,52,E1,C8,FD,23,
FD,23,10,EE,3E,1,B7,C9,-2,0,54,97,
32,D2,A0,CD,24,BB,F5,3A,CE,A0,FE,A
0,E1,28,1A,7C,CB,47,C2,53,65,CB,4F
,C2,75,65,CB,57,C2,95,65,CB             <A093>
1410 DATA 5F,C2,B1,65,CB,67,20,4,C
9,7C,18,F8,3E,FF,32,D2,A0,C9,2A,D0
,A0,-2,2B,32,56,2B,5E,EB,E5,1,14,0
,A7,ED,42,EB,E1,EB,E5,1,8,9D,A7,ED
,42,E1,EB,D8,3E,1,C3,CD                 <9CF4>
1420 DATA 65,2A,D0,A0,-2,2B,30,56,
2B,5E,EB,E5,1,14,0,9,EB,E1,EB,E5,1
,E8,9E,A7,ED,42,E1,EB,D0,3E,2,C3,C
D,65,2A,D0,A0,-2,2B,26,56,2B,5E,EB
,E5,2B,FD,21,F0,A0,CD,6                 <9CF6>
1430 DATA 65,E1,C8,3E,4,54,5D,1B,C
3,CD,65,2A,D0,A0,-2,2B,349,56,2B,5
E,EB,E5,23,FD,21,54,A1,CD,6,65,E1,
C8,3E,8,54,5D,13,C3,CD,65,F5,1A,B7
,28,3C,FE,3D,38,48,FE,3F,38             <A020>
1440 DATA 46,FE,44,38,B,FE,A0,28,2
C,3E,FF,32,CD,A0,18,25,E5,D5,11,5,
0,D6,3E,4F,97,47,CD,80,A2,EB,2A,D3
,A0,19,22,D3,A0,3E,FF,32,D5,A0,D1,
E1,97,12,3A,C8,A0,3D,32,C8              <9F49>
1450 DATA A0,D5,EB,1,E0,1,9,7E,EB,
D1,B7,20,4,F1,C3,27,66,F1,C9,3E,FF
,32,CD,A0,18,E7,F5,1,E0,1,E5,9,7E,
E1,C1,FE,97,CC,51,66,FE,98,CC,6F,6
6,FE,99,CC,8D,66,FE,9A                  <9BC5>
1460 DATA CC,AB,66,2A,D0,A0,73,23,
72,23,71,23,22,D0,A0,C9,78,FE,1,28
,12,FE,4,28,7,E,9A,6,3B,C3,C9,66,E
,99,6,3C,C3,C9,66,E,97,6,37,C3,C9,
66,78,FE,2,28,12,FE                     <9800>
1470 DATA 4,28,7,E,9A,6,3A,C3,C9,6
6,E,99,6,39,C3,C9,66,E,98,6,37,C3,
C9,66,78,FE,1,28,12,FE,2,28,7,E,99
,6,38,C3,C9,66,E,98,6,3B,C3,C9,66,
E,97,6,3A,C3                            <915A>
1480 DATA C9,66,78,FE,1,28,12,FE,2
,28,7,E,9A,6,38,C3,C9,66,E,98,6,3C
,C3,C9,66,E,97,6,39,C3,C9,66,C5,D5
,C5,D5,79,EB,1,E0,1,9,77,EB,D1,3A,
CE,A0,3C,32,CE,A0                       <9668>
1490 DATA D5,CD,BC,63,E1,C1,E5,C5,
D5,48,6,0,11,20,0,CD,80,A2,1,20,4E
,9,D1,CD,5A,63,C1,6,0,11,20,0,CD,8
0,A2,1,20,4E,9,D1,E5,EB,CD,BC,63,E
1,CD,5A,63,D1,C1,3E                     <982C>
1500 DATA 1,C9,-13,0,17,3A,D2,A0,B
7,C8,6,2,C5,3A,CE,A0,B7,28,43,2A,D
0,A0,-2,2B,81,56,2B,5E,22,D0,A0,EB
,E5,1,E0,1,9,97,77,E1,D5,CD,BC,63,
21,20,4E,CD,5A,63,E1,2B                 <9C02>
1510 DATA 4E,2B,56,2B,5E,EB,C5,CD,
BC,63,C1,6,0,D5,11,20,0,CD,80,A2,1
```

```
,20,4E,9,D1,CD,5A,63,3A,CE,A0,3D,3
2,CE,A0,C1,10,B3,3A,CE,A0,B7,C0,3A
,C8,A0,B7,C0,3E,43,32,CD                    <9D50>
1520 DATA A0,C9,-1240,0,44,6D,9D,3
,2,90,9D,4,1,20,9E,2,1,29,9E,2,1,3
2,9E,2,1,56,9D,2,1,5C,9D,2,1,5F,9D
,2,1,65,9D,2,1,6B,9D,2,1,71,9D,2,1
,-116,0,2                                   <8E20>
1530 DATA 1,0,-7,1,1,98,-6,1,1,0,-
3,1,7,40,0,3D,0,40,0,3D,-5,0,4,3D,
40,0,3D,-2,0,-2,40,2,1,0,-5,1,5,0,
1,0,1,0,-4,1,1,0,-4,1,1,0,-5,1      <8148>
1540 DATA 5,0,1,0,1,0,-4,1,1,0,-4,
1,1,48,-5,1,5,0,1,0,1,0,-4,1,1,48,
-4,1,16,49,3D,0,40,0,3D,0,1,0,1,0,
3D,40,0,3D,49,-4,1,1,0,-7,1,1       <80B5>
1550 DATA 0,-6,1,1,0,-4,1,1,0,-7,1
,1,0,-6,1,1,0,-4,1,1,0,-7,1,1,0,-6
,1,1,0,-4,1,-2,40,-2,0,-4,1,1,0,-4
,1,3,0,40,0,-7,1,1,0,-4,1,1         <7EA9>
1560 DATA 0,-4,1,1,0,-9,1,-11,0,-9
,1,1,0,-4,1,1,0,-4,1,1,0,-9,1,1,0,
-4,1,1,0,-4,1,1,0,-5,1,-9,0,1,6C,-
9,0,1,1,-9,0,1,6D,-9,0,2,1,3D       <806C>
1570 DATA -8,1,1,3F,-8,1,3,3D,1,40
,-8,1,1,3F,-8,1,2,3F,1,-4,40,1,3D,
-4,40,1,3F,-2,40,1,3D,-6,40,-3,1,1
,3F,-6,1,1,3F,-5,1,1,3F,-6,1,1,3F,
-6,1,1                                      <8B70>
1580 DATA 3F,-5,1,1,3F,-6,1,1,40,-
3,1,1,40,-2,1,1,3F,-2,1,1,40,-2,1,
1,40,-5,1,-3,40,1,3D,-4,40,1,3F,-3
,40,1,3D,-4,40,-22,1,12,E,9,12,9,4
,0,1A,5                                     <8CB7>
1590 DATA 2,14,0,FF,-46,2,-8,3F,1,
3D,-4,3F,-7,2,1,3F,-11,2,1,3F,-7,2
,1,3F,-11,2,1,3F,-7,2,1,3F,-11,2,1
,3F,-3,2,-7,3F,-2,3D,1,3F,-6,2,1,3
D,-3,2,1,3F                                 <90D0>
1600 DATA -8,2,1,3F,-6,2,1,3F,-3,2
,1,3F,-8,2,1,3F,-6,2,1,3F,-3,2,1,3
F,-8,2,1,3F,-6,2,1,48,-3,2,-7,3F,-
2,2,-5,3F,3,3D,3F,49,-9,2,1,3F,-9,
2,1,0                                       <8A39>
1610 DATA -9,2,3,3F,2,0,-7,2,1,0,-
9,2,2,3F,2,-2,0,-6,2,1,3D,-9,2,2,3
F,2,-9,0,-9,2,4,3F,2,0,48,-9,2,-8,
3F,5,2,0,49,0,3D,-5,0,-2,2,1,3F,-8
,2                                          <8781>
1620 DATA -2,0,-6,2,1,0,-2,2,1,3F,
-8,2,4,48,0,3F,3D,-4,0,1,3D,-2,2,1
,3F,-8,2,2,49,0,-9,2,1,3F,-8,2,-2,
0,-9,2,1,3F,-8,2,-2,0,-9,2,-8,3F,-
3,0                                         <88BC>
1630 DATA 5,3F,0,3D,0,3F,-12,2,1,9
7,-10,2,1,9,-7,0,4,1A,0,A,5A,-42,4
,-5,0,1,6C,-7,0,1,6C,-2,0,-4,4,8,0
,3D,0,3D,0,6D,0,3D,-2,0,1,3D,-2,0,
1,6D                                        <8909>
1640 DATA -2,0,-4,4,3,41,4,0,-10,4
,3,0,4,41,-4,4,4,41,4,0,4,-8,41,4,
0,48,4,41,-4,4,-3,41,-2,0,1,6C,-7,
0,3,49,4,41,-4,4,6,41,4,41,4,3D,6D
,-2,0                                       <8AAD>
1650 DATA 1,3D,-5,0,2,4,41,-4,4,-3
,41,-2,4,1,3D,-9,4,1,41,-4,4,3,41,
4,41,-10,0,-3,41,-4,4,5,41,4,41,0,
3D,-4,0,1,3D,-3,0,3,41,4,3D,-4,4,1
,41,-11,4                                   <8EAB>
1660 DATA 1,0,-2,4,1,41,-4,4,4,0,4
,41,3D,-2,0,10,3D,0,3D,0,3D,0,3D,4
1,4,41,-4,4,1,41,-3,4,1,0,-10,4,1,
41,-2,4,3,0,6C,41,-16,0,3,4,0,6D,-
17,0                                        <89B8>
1670 DATA -3,4,1,41,-5,4,5,0,4,0,4
,3D,-4,4,1,41,-4,4,10,41,4,41,0,3D
,0,3D,4,0,4,-3,3D,3,41,4,41,-4,4,1
0,41,4,41,3D,0,3D,0,4,0,4,-3,3D,3,
41                                          <8744>
1680 DATA 4,41,-4,4,1,41,-7,4,1,0,
-6,4,1,41,-4,4,-16,0,-4,4,2,0,3D,-
2,0,1,3D,-5,0,1,3D,-2,0,1,3D,-2,0,
-12,4,1,97,-29,4,5,17,D,C,12,B,-6,
0,1                                         <8868>
1690 DATA 46,-491,0,1,5A,-491,0,1,
5A,-491,0,1,5A,-491,0,1,5A,-491,0,
1,5A,-491,0,1,5A,-491,0,1,5A,-491,
0,1,5A,-491,0,1,5A,-491,0,1,5A,-49
1,0,1,5A,-491,0,1,5A,-491,0,1       <A2ED>
1700 DATA 5A,-491,0,1,5A,-491,0,1,
5A,-491,0,1,5A,-491,0,1,5A,-491,0,
1,5A,-491,0,1,5A,-491,0,1,5A,-491,
0,4,5A,11,9D,98                             <72D1>

10 '*****************************  <2397>
15 '* DATALADER 2 FUER GREIFER *  <2364>
20 '*   ERZEUGT <GREIFER.BIN>   *  <23E0>
25 '*           VON             *  <236B>
30 '*      LOTHAR SCHOLZ        *  <239F>
35 '*          FUER             *  <2398>
40 '*    SCHNEIDER CPC-WELT     *  <2314>
45 '*     CPC 464/664/6128   JE*  <23AA>
50 '*****************************  <23E7>
60 PAPER 0:PEN 1:INK 0,0:INK 1,25:
BORDER 0:MODE 2:LOCATE 10,10:PRINT
"Bitte warten MC-Datei  Greifer.bi
n wird erzeugt !!"                          <58A6>
70 MEMORY 19999:adr=41500:FOR k=0
TO 43:READ a:IF a>0 THEN 120 ELSE
110                                         <3971>
80 NEXT k:GOTO 130                          <0FF1>
110 a=ABS(a):READ a$:w=VAL("&"+a$)
:FOR i=1 TO a:POKE adr,w:adr=adr+1
:NEXT:zae=zae+1:GOTO 80                     <6B01>
120 FOR i=1 TO a:READ a$:w=VAL("&"
+a$):POKE adr,w:adr=adr+1:NEXT:zae
```

```
=zae+1:GOTO 80                                  <5D0F>
130 SAVE"greifer.bin",b,41500,500               <2372>
1000 DATA 2,0,FF,-2,0,7,55,80,0,AA
,40,0,FF,-2,0,1,FF,-2,0,7,55,80,0,
AA,40,0,FF,-3,0,7,C0,0,55,80,0,AA,
40,-2,0,3,C0,0,FF,-2,0,1,FF,-2,0,1
,FF,-2,0                                        <8DAD>
1010 DATA 1,FF,-2,0,1,FF,-2,0,1,FF
,-2,0,1,FF,-2,0,1,FF,-3,0,7,C0,0,5
5,80,0,AA,40,-2,0,3,C0,0,FF,-2,0,7
,55,80,0,AA,40,0,FF,-2,0,1,FF,-2,0
,7,55                                           <8A1A>
1020 DATA 80,0,AA,40,0,FF,-5,0,2,A
7,21,-2,0,102,3E,10,29,CB,13,CB,12
,30,4,9,30,1,13,3D,20,F2,C9,FE,2,C
0,97,47,57,DD,5E,0,DD,4E,2,CB,21,C
B,21,C5,1,50,0,CD                               <9685>
1030 DATA 80,A2,C1,9,1,0,C0,9,11,1
C,A2,22,7E,A2,CD,BC,A2,C9,6,8,C5,E
5,6,4,C5,7E,12,13,47,1A,A0,13,47,1
A,B0,13,77,C1,23,10,EF,E1,1,0,8,9,
C1,10,E3,C9,2A,7E                               <96F7>
1040 DATA A2,6,8,11,1C,A2,C5,E5,6,
4,1A,77,-3,13,124,23,10,F8,E1,1,0,
8,9,C1,10,EC,C9,FE,3,C0,DD,5E,0,1,
20,0,97,57,CD,80,A2,1,20,4E,9,E5,9
7,47,57,DD,5E,2                                 <9423>
1050 DATA DD,4E,4,CB,21,CB,21,C5,1
,50,0,CD,80,A2,C1,9,1,0,C0,9,D1,CD
,2B,A3,C9,6,8,C5,E5,6,4,1A,77,23,1
3,10,FA,E1,1,0,8,9,C1,10,EE,C9,FE,
1,C0,DD,56,1                                    <9130>
1060 DATA DD,5E,0,21,50,C0,1,E0,1,
C5,D5,E5,1A,4F,97,47,11,20,0,CD,80
,A2,1,20,4E,9,EB,E1,E5,CD,2B,A3,E1
,D1,13,-4,23,7,C1,B,78,B1,20,DC,C9
,-157,0                                         <8C37>
```

Fortsetzung von Seite 136

und höchstens vierzig Monster verteilt, mindestens ein Punkt besetzt und die Startposition definiert wurden.

Hinweise zum Abtippen:

Diskettenbenutzer tippen Listing 1 ab und speichern es unter dem Namen „Greifer" ab. Anschließend Listing 2 abtippen, unter einem beliebigen Namen abspeichern und erst dann laufen lassen. Es erzeugt das Binärfile Daten1.dat. Genauso beim Listing 3 vorgehen; von diesem Listing wird jetzt die Binärdatei "Greifer.bin" erzeugt. Jetzt ist das Programm lauffähig, sofern Sie keine Tippfehler gemacht haben.
Kassettenbenutzer arbeiten am besten mit zwei Kassetten: einer Programm- und einer Datenkassette.
Auf die Datenkassette werden die Binärfiles abgespeichert.
Das Programm fordert nach dem Starten auf, die Kassette mit den Binärfiles einzulegen und eine Taste zu drücken.
Viel Spaß beim Tippen und beim Spielen! Wem die Tipperei zuviel Arbeit ist, der kann bei unserem Verlag die Softbox bestellen. □

RAT UND HILFE FÜR BASTLER

Ich möchte zum Beispiel aus Papier eine Anzahl rechteckiger Formate verschiedener Höhe und Breite ausschneiden. Dazu wäre es günstig, ein Programm zu besitzen, das die günstigste Anordnung und das benötigte Format (A3 oder A4) errechnet. Wer kann mir weiterhelfen oder hat einen Literaturtip?

Wer Rat weiß, wende sich an:
Helmut Zinter,
Viehland 5,
2862 Worpswede

ABFRAGE DES SPEICHERPLATZES

Ich besitze einen CPC464. Ich weiß, daß ich, um freie Speicherplätze abzufragen, PRINT HIMEM eingeben muß. Der CPC antwortet dann mit 43904. Wenn ich ein Programm eingeladen habe und gebe PRINT HIMEM ein, antwortet der CPC wieder mit 43904. Welchen Befehl muß ich verwenden, um die tatsächlich freien Speicherplätze zu erfahren, nachdem ich ein Programm eingeladen habe?
Rüdiger Kwasny,
4100 Duisburg

Der Befehl PRINT HIMEM zeigt Ihnen die höchste verfügbare Speicherstelle Ihres CPC an. Das kann nützlich sein, um zu erfahren, ob ein Programm diese herabgesetzt hat, etwa durch einen SYMBOL-AFTER-Befehl oder die Installation eines Maschinenprogramms.
Wenn Sie erfahren wollen, wieviel Byte effektiv noch frei sind, so müssen Sie die FRE-Funktion verwenden. Sie hat zwei unterschiedliche Parameter, mit denen Sie unterschiedliche Auskünfte erhalten. Die Anweisung PRINT FRE(0) gibt Ihnen die Menge des verfügbaren freien Speicherplatzes in Byte aus. Wenn Sie diese Anweisung eingeben, nachdem ein Programm abgelaufen ist, sind in diese Angabe bereits alle Speicherstellen mit einbezogen, die von Variablen belegt werden. Wollen Sie hingegen wissen, wieviel Platz Ihr Programm ohne Variablenwerte beansprucht, so verwenden Sie PRINT FRE(" "). Bei diesem Befehl kann es sein, daß Sie auf die Antwort ein wenig warten müssen, da der CPC vor der Berechnung alle gespeicherten Variablen löscht.

BASIC2 AUF DEM CPC?

Was muß ich machen, damit ich auf meinem Schneider CPC464 mit BASIC2 programmieren kann? Und benötige ich dann eine Maus?
Beate Stix,
A-3830 Waidhofen

BASIC2 ist eine Programmiersprache, die Schneider zu den Geräten der PC-Klasse, PC1512 und PC1640 ausliefert. Diese Sprache ist speziell für diese Geräte bestimmt und auf dem CPC nicht lauffähig. Ihr Schneider CPC464 besitzt jedoch ein eingebautes BASIC, das sofort nach dem Einschalten des Gerätes zu Ihrer Verfügung steht.
Sie können sich am leichtesten in diese Programmiersprache einarbeiten, wenn Sie Schritt für Schritt die Beispiele im Handbuch durcharbeiten und sich anschließend mit der Funktion der einzelnen BASIC-Befehle auseinandersetzen. Auch in Zeitschriften veröffentlichte Programme sowie „Tips und Tricks" können Ihnen, wenn Sie einmal etwas Programmiererfahrung besitzen, weiterhelfen.
Der Anschluß einer Maus an den CPC464 ist zwar möglich, aber nicht vorgesehen. Für die reine Arbeit mit BASIC benötigen Sie sie nicht.

SOUNDDEMO

Hab acht!

Für die Freunde der Musikprogrammierung bringen wir diesmal zwei kleine Demos. Und zwar die deutsche und die englische Nationalhymne mit den dazugehörigen Flaggen.

Geschrieben hat diese Programme unser Autor Matthias Dühn. Sie laufen auf allen CPC-Typen. □

```
10 '*****************************  <2397>
20 '* DEMO#1 : Englische Hymne *  <2359>
30 '*           VON             *  <23B1>
40 '*      MATTHIAS DUEHN       *  <2317>
50 '*           FUER            *  <23B6>
60 '*    SCHNEIDER CPC-WELT     *  <23D8>
70 '*     CPC 464/664/6128   JE*  <2337>
80 '*****************************  <2322>
90 MODE 1:INK 0,1:BORDER 0:INK 1,6
:INK 2,26                         <1A3D>
100 FOR i=1 TO 10:PLOT i+20,1:DRAW
 639,399-20-i,2:NEXT              <2FED>
110 FOR i=1 TO 10:PLOT 1,20+i::DRA
W 639-20-i,399,2:NEXT             <30D6>
120 FOR i=1 TO 20:PLOT 1,i::DRAW 6
39-i,399,1:NEXT                   <2A16>
130 FOR i=1 TO 20:PLOT i,1:DRAW 63
9,399-i,1:NEXT                    <2964>
140 FOR i=-28 TO 30:PLOT i,399:DRA
W 639,i,1:NEXT                    <2960>
150 FOR i=1 TO 10:PLOT 34+i,399:DR
AW 639,i+20,2:NEXT                <2D55>
160 FOR i=1 TO 10:PLOT 1,399-20-i:
DRAW 639-30-i,1:NEXT              <2F3F>
170 FOR i=175 TO 185:PLOT 1,i:DRAW
 639,i,2:NEXT                    <261E>
180 FOR i=185 TO 245:PLOT 1,i:DRAW
 639,i,1:NEXT                    <26C2>
190 FOR i=245 TO 255:PLOT 1,i:DRAW
 639,i,2:NEXT                    <26EE>
200 FOR i=280 TO 290:PLOT i,1:DRAW
 i,639,2:NEXT                    <28AD>
210 FOR i=290 TO 350:PLOT i,1:DRAW
 i,639,1:NEXT                    <28EA>
220 FOR i=350 TO 360:PLOT i,1:DRAW
 i,639,2:NEXT                    <284C>
230 FOR i=185 TO 245:PLOT 1,i:DRAW
 639,i,1:NEXT                    <2627>
240 PLOT 1,1:DRAW 639,1,2:DRAW 639
,399,2:DRAW 1,399,2:DRAW 1,1,2    <2DFB>
260 REM *                         <080F>
270 REM * And now let's play the m
usic                              <25C6>
280 REM *                         <0837>
290 FOR i=1 TO 5                  <0E8E>
300 RESTORE 410                   <090B>
310 READ note,laenge              <1737>
320 WHILE note<>-1                <10F8>
330 WHILE laenge<>-1              <1272>
340 SOUND 1,note,laenge*1.45      <208D>
350 SOUND 2,note+1.7,laenge*1.45  <279A>
360 SOUND 4,note-1,laenge*1.45    <229C>
370 READ note,laenge              <17AF>
380 WEND:WEND                     <08B2>
390 FOR b=1 TO 1000:NEXT b        <161A>
400 NEXT                          <06E4>
410 DATA 358,45,2,2,358,45,319,45,
379,60,358,26,319,45,284,45,2,2,28
4,45,268,45,284,60,319,26,358,45  <6213>
420 DATA 319,45,358,45,379,45,358,
90,239,45,2,2,239,45,2,2,239,45,2,
2,239,60,268,26,284,45,268,45,2,2,
268,45,2,2,268,45,2,2,268,60      <809E>
430 DATA 284,25,319,45,284,45,268,
25,284,25,319,25,358,25,284,60,268
,26,239,45,213,26,239,15,268,15,28
4,45,319,45,358,90,2,60,-1,-1     <8174>
10 '*****************************  <2397>
20 '*    DEMO #2: DEUTSCHE      *  <2362>
30 '*  NATIONALHYMNE U. FLAGGE  *  <23CC>
40 '*           VON             *  <2389>
50 '*      MATTHIAS DUEHN       *  <23CA>
60 '*           FUER            *  <23CA>
70 '*    SCHNEIDER CPC-WELT     *  <2350>
80 '*     CPC 464/664/6128   JE*  <23F0>
90 '*****************************  <2336>
105 BORDER 26                     <082A>
110 MODE 1                        <0784>
120 INK 1,24:INK 2,0:INK 3,6      <14E3>
130 WINDOW 1,40,1,8:PAPER 2:PEN 1:
CLS                               <16B2>
140 WINDOW 1,40,9,17:PAPER 3:PEN 1
:CLS                              <17D9>
150 WINDOW 1,40,18,25:PAPER 1:PEN
2:CLS                             <18C5>
170 PRINT CHR$(22);CHR$(1)        <1223>
180 LOCATE 1,8:PRINT"Hymne + Flagg
e  ";CHR$(164);" 1987 by Matthias
Duehn";                           <3FF1>
190 REM *                         <0883>
200 REM * And now let's play the m
usic                              <2539>
210 REM *                         <08AB>
215 FOR i=1 TO 5                  <0EF6>
220 RESTORE 1000                  <09DF>
230 READ note,laenge              <1797>
240 WHILE note<>-1                <1057>
250 WHILE laenge<>-1              <12D0>
260 SOUND 1,note-1.7,laenge*1.35  <27C9>
261 SOUND 2,note,laenge*1.35      <209A>
262 SOUND 4,note+1.2,laenge*1.35  <2733>
270 READ note,laenge              <17E8>
280 WEND:WEND                     <08E9>
```

```
285 FOR b=1 TO 1000:NEXT b          <1647>
290 NEXT                            <0608>
1000 DATA 379,60,340,25,305,45,340
,45,286,45,305,45,340,24,415,25,37
9,51                                <45AF>
1010 DATA 230,45,260,45,286,45,305
,45,340,45,305,26,379,26,260,100    <3F72>
1020 DATA 379,60,340,25,305,45,340
,45,286,45,305,45,340,24,415,25,37
9,51                                <45D7>
1030 DATA 230,45,260,45,286,45,305
,45,340,45,305,26,379,26,260,100    <3F9B>
1040 DATA 340,45,305,45,340,26,415
,26,506,45,278,45,305,45,340,26,41
5,26,506,45                         <4C1D>
1050 DATA 260,45,286,45,305,65,2,1
,305,26,278,45,2,1,278,26,258,26,2
,1,258,100                          <4B33>
1060 DATA 185,75,201,27,2,1,201,27
,230,27,260,45,230,65,260,27,2,1,2
60,27,286,27,305,45                 <54AE>
1070 DATA 340,65,305,15,286,15,260
,27,230,27,286,27,340,27,379,45,30
5,27,340,27,379,100                 <54F1>
1080 DATA 185,80,201,27,2,1,201,27
,230,27,260,45,230,65,260,27,2,1,2
60,27,286,27,305,45                 <5465>
1090 DATA 340,60,305,15,286,15,260
,27,230,27,286,27,340,27,379,45,30
5,27,340,27,379,90,2,100,-1,-1      <5F24>
```

PROBLEME BEI DER DRUCKERANPASSUNG

Seit einiger Zeit besitze ich den CPC464 mit Diskettenstation. Ich verwende einige Programme von Data Becker, wie etwa Textomat oder Datamat. Aber ohne Drucker ist dies alles uninteressant. Deshalb habe ich mir einen Seikosha SP 180 AI gekauft. Obwohl ich mich, wie ich zumindest glaube, genau an die Anweisungen halte, erhalte ich keinen brauchbaren Ausdruck. Gibt es ein Programm, um den Drucker anzupassen, oder mache ich etwas falsch?

Da uns der betreffende Drucker nicht zur Verfügung steht, leiten wir diese Frage an unsere Leser weiter. Wer Rat weiß, wende sich bitte an:
Ady Laschette
63, Rue P. Gansen
L-4570 Niederkorn

PROBLEME MIT DEM CHECKSUMMER

Ich bin ein regelmäßiger Leser Ihrer Zeitung und möchte auf diesem Wege Ihren Leserservice in Anspruch nehmen. Den Eingabechecksummer im Januarheft halte ich für eine gute Sache. Aber ist es möglich, daß er im zweiten Listing des Adressenmanagers, der auch im Januarheft abgedruckt ist, nicht läuft?
K.-H. Schwandrau,
5632 Wermelskirchen

Der Checksummer funktioniert in der abgedruckten Version ohne Probleme. Allerdings haben auch wir ihn mit der Januar-Ausgabe zum ersten Mal verwendet, und so ist uns beim Ausdruck des oben erwähnten Listings ein kleiner Fehler unterlaufen. Die im Heft abgedruckten Prüfsummen stimmen nicht mit denen überein, die Sie durch den Checksummer erhalten. Bei allen anderen Programmen können Sie den Checksummer jedoch anwenden.

PROBLEME MIT DEN LISTINGS

Vor einigen Tagen habe ich mir einen CPC464 von Schneider gekauft. Ich besorgte mir das SCHNEIDER AKTIV SPECIAL Nr. 3/87. Beim Eintippen der Listings stellte ich fest, daß diese am Schluß nicht richtig abliefen. Da ich noch Anfänger bin, verstehe ich die Programme und deren Ablauf nicht so gut wie ein Fortgeschrittener und kann daher keine Korrektur vornehmen. Ein weiteres Problem möchte ich ansprechen. Es handelt sich um das Zeichen "^", bei dem ich nicht weiß, wie es bei der Tastatur eingegeben wird.
Christian Albiez,
6600 Saarbrücken

Um einen Tippfehler so gut wie unmöglich zu machen, veröffentlichen wir jetzt regelmäßig einen sogenannten „Checksummer". Dieses Programm gibt nach dem Eintippen jder Zeile eine Prüfsumme auf dem Bildschirm aus. Stimmt diese mit der im Heft abgedruckten Zahl überein, so wurde die Zeile richtig eingegeben. Wenn Ihnen das Abtippen zu mühselig ist, so können Sie auch unseren Softbox-Service in Anspruch nehmen. Informationen darüber finden Sie auf den Bestellcopons in jedem Heft.

Das Zeichen "^" entspricht auf der Tastatur Ihres Computers dem nach oben gerichteten Pfeil. Sie finden ihn in der obersten Tastenreihe als dritte Taste von rechts unter dem englischen Pfundzeichen. Leider ist uns kein Drucker bekannt, der dieses Zeichen so ausgibt, wie Sie es auf der Tastatur und auf dem Bildschirm sehen. □

IMPRESSUM

SCHNEIDER CPC-WELT EXTRA erscheint zweimonatlich in der CA-Verlags GmbH (i.G.)

VERANTWORTLICH FÜR DEN INHALT: Gert Seidel

GESCHÄFTSFÜHRER (und verantwortlich für Anzeigen): Werner E. Seibt

ANSCHRIFT FÜR ALLE VERANTWORTLICHEN:
Postfach 1161,
8044 Unterschleißheim
Tel.: 089/1298011
Telex: 5214428 cav-d

Es gilt Preisliste Nr. 8 vom 1.1.1988
Media-Unterlagen bitte anfordern.

©1988 by CA-Verlags GmbH (i.G.), Heßstraße 90, 8000 München 40

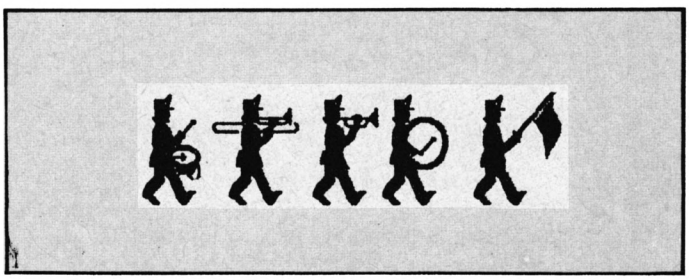
Für unaufgefordert eingesandte Manuskripte und Listings keine Haftung.

Bei Einsendung von Texten, Fotos und Programmträgern erteilt der Autor dem Verlag die Genehmigung für den Abdruck und die Aufnahme in den Softbox-Service zu den Honorarsätzen des Verlages. Das Copyright und das Recht der wirtschaftlichen Verwertung gehen auf den Verlag über.
Alle in dieser Zeitschrift veröffentlichten Beiträge sind urheberrechtlich geschützt. Jedwede Verwendung ist untersagt. Namentlich gezeichnete Beiträge unserer Mitarbeiter stellen nicht unbedingt die Meinung der Redation dar.

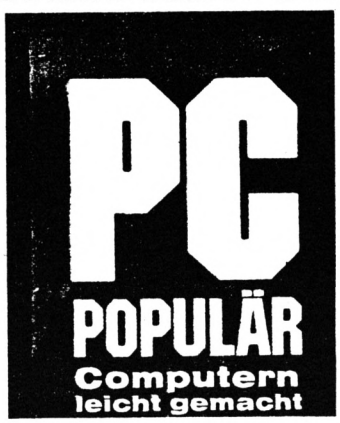
HARDCOPY IN ANDERE PROGRAMME EINBINDEN

Meine Frage betrifft SCHNEIDER AKTIV SPECIAL Nr. 1/88, Seite 35, zum Listing „Diagramm". Ich weiß nicht, wie ich die Hardcopy-Anpassung vornehmen soll. Für diese Auskunft wäre ich Ihnen dankbar.
Frank Hohn,
2800 Bremen 1

Ähnliche Probleme hatten bereits viele unserer Leser, so daß hierzu ein paar grundsätzliche Bemerkungen angebracht scheinen. Im selben Sonderheft wurde das Programm „Superhardcopy" veröffentlicht. Dieses Programm installiert auf dem CPC mehrere RSX-Befehle. RSX steht für „resident system extension" also „residente Systemerweiterung". Das ist im Prinzip nichts anderes als ein Programm in Maschinensprache, das einmal in den Speicher geladen wird und Ihnen ab dann neue Befehle zur Verfügung stellt, die jeweils mit einem senkrechten Strich eingeleitet werden.



Macht Computern so einfach wie Autofahren



Aus der Praxis
für die Praxis
Ohne Fachchinesisch
Von Computer-
Experten für
Computer-Einsteiger

An ausgewählten Kiosken
und im Bahnhofs-Buchhandel

Bei der „Superhardcopy" lautet der Befehl zum Ausdruck einer Bildschirmkopie /HARDCOPY. Um im Programm „Diagramm" eine Hardcopy ausdrucken zu können, müssen Sie also wie folgt vorgehen:
1. Tippen Sie das Programm „Superhardcopy" ab. Beachten Sie dabei die Anpassungen an die Schneider-Modelle 664 und 6128! Verwenden Sie nur die Programmzeilen, die für Ihren Rechner geeignet sind!
2. Speichern Sie die „Superhardcopy" ab – wenn Sie mit Kassetten arbeiten, am besten vor „Diagramm".
3. Ändern Sie den RSX-Befehl COPY in den oben angegebenen Zeilen von „Diagramm" in /HARDCOPY. Speichern Sie auch das so geänderte Programm.
4. Lassen Sie das Programm „Superhardcopy" laufen, bevor Sie „Diagramm" starten. Die RSX-Befehle sind jetzt in Ihrem Rechner installiert und damit jedem Programm bekannt, das anschließend geladen und gestartet wird. Diese Befehle werden nur durch ein Reset (CTRL-SHIFT-ESC) oder das Abschalten des Rechners wieder gelöscht.
Selbstverständlich hat die Superhardcopy noch weitere Möglichkeiten wie etwa die Einstellung verschiedener Graustufen. Die entsprechenden RSX-Befehle sind in dem Artikel zum Listing beschrieben.
Daß mit dieser Antwort, die unser Leser selbstverständlich vor der Veröffentlichung erhielt, noch nicht alle Probleme gelöst waren, zeigte sich in einem Schreiben, das einige Zeit später bei uns einging:

HARDCOPY: EIN HÄUFIGES PROBLEM

Ich möchte Sie noch einmal um Hilfe bitten. Damit der Brief nicht zu lang wird, folgende Stichworte:

1. Listing Diskette. Bestellt und erhalten.
2. Versucht „Diagramm" zu drucken. Geht nicht.
3. Ihre Antwort auf mein erstes Schreiben erhalten.
4. „Superhardcopy" auf eine neue Disk gespeichert und an 6128 angepaßt. (Im Heft fehlt bei 35340 zwischen 205 und 50 ein Komma.)
5. Diagramm "BLT" auf gleiche Disk gespeichert; Zeilen 940, 1380 und 1720 geändert in /HARDCOPY.
6. RUN"HARDCOPY". Anschließend Abbruch mit zweimal ESC.
7. RUN"BLT".
8. Daten eingegeben.
9. Statt Balken erscheinen nur Striche (Hardcopy liegt bei).
10. Weiter = W, zur Liniengrafik. Linien erscheinen nicht.
11. Weiter = W, zur Kreisgrafik. Funktioniert ausgezeichnet.

Was mache ich also falsch?
Frank Hohn

Daraufhin haben wir uns beide Programme noch einmal angesehen und eine ausführliche Anleitung zu speziell diesem Problem niedergeschrieben, ebenfalls nur in Stichworten:
1. LOAD"HARDANP.128"
2. SAVE"HC.BAS",A
3. LOAD"HARDCOPY.464"
4. MERGE"HC.BAS"
5. DELETE –29999
6. EDIT 30370; ändern in: 30370 END
7. Sichern für alle Fälle
8. RUN
Nun ist der Befehl /HARDCOPY installiert.
9. LOAD"BLT"
10. Die Zeilen 940, 1380 und 1720 editieren, also statt des Befehls /COPY eingeben: /HARDCOPY.
11. Eventuell sichern
12. RUN
Und schon ist alles in Ordnung. Vorausgesetzt dazu ist allerdings, daß alle Programme richtig abgetippt sind oder Sie unsere Softbox bezogen haben. □

CHECKER- der Eingabechecksummer für alle CPC's

Von Ihnen lange gefordert, ist er nun endlich da, der Checksummer für alle CPCs. Die nähere Besprechung dieses Eingabe-Prüfprogrammes erfolgt im Rahmen der Artikelreihe über die Programmierung in Maschinensprache. Hier nun nur die Bedienungsanleitung:

Tippen Sie das Programm sorgfältig ab, denn trotz der Prüfsummen am Ende jeder DATA-Zeile können Statementvertauschungen nicht erkannt werden. Nach dem Abtippen gleich abspeichern, damit bei einem eventuellen Eingabefehler nicht alles neu eingegeben werden muß!

Nach einem Programmlauf stehen zwei RSX-Befehle zur Verfügung: EIN und AUS. RSX-Befehle beginnen mit einem senkrechten Strich (Taste neben P mit Shift eingeben), und dann folgt der Befehlsname. Mit EIN wird der Checksummer aktiviert, mit AUS wieder abgeschaltet. Bei aktivem Checksummer erfolgt nach der Eingabe einer Basiczeile die Ausgabe einer vierstelligen Hexademizalzahl, die in eckige Klammern eingeschlossen ist. Diese Prüfsumme muß mit der im einzugebenden Listing enthaltenen Summe (am Ende jeder Basiczeile) übereinstimmen. Ist dies nicht der Fall, dann stimmt irgendetwas nicht! Geben Sie die Zeile neu ein (kann auch mit Edit Zeilennummer oder mittels der COPY-Cursor-Methode geschehen!).

Da unser Checker – im Gegensatz zu anderen Checksummern – auch die Zeilennummer mit überprüft, muß auch diese korrekt sein, damit die Prüfsumme stimmt.

Haben Sie ein Listing fehlerfrei abgetippt, dann speichern Sie dieses ab! Erst dann sollten Sie das abgetippte Programm starten, denn es könnte ja sein, daß das Programm dort, wo unser Checker seinen Platz hat, selbst aktiv wird oder eigene RSX-Befehle mit gleicher Aufrufsbefehlsfolge hat und es deshalb Fehler gibt!

Da die Firmware der drei CPC-Typen leider unterschiedlich ist, verhält sich unser "Eingabe-Prüfer" beim Befehl AUTO unterschiedlich. Dieser Befehl kann beim CPC 464 nicht benutzt wwerden, da dann keine Prüfsumme ausgegeben wird. Beim CPC 664 und 6128 ist der Einsatz dieses Befehles möglich.
Viel Glück mit dem Prüfprogramm! *LM*□

*

```
100 MEMORY &A2FF:'CHECKI.HEX
110 a=&A300:e=&A3F6:zb=1000:e=e+1
120 PRINT"Moment bitte...":FOR i=a TO e:
READ d$:IF LEFT$(d$,1)="&"THEN flag=1
130 IF(flag AND p<>VAL(d$))THEN PRINT"Fe
hler in Zeile "zb+1:END
140 IF(flag AND i=e)THEN 180
150 IF flag THEN i=i-1:zb=zb+1:p=0:d$=""
:flag=0:GOTO 170
160 POKE i,VAL("&"+d$):p=p+VAL("&"+d$)
170 IF i<e THEN NEXT i
180 typ=PEEK(&BD71):IF typ=&E8 THEN 260
190 IF typ=&55 THEN 220
200 IF typ=&14 THEN 240
210 PRINT"CPC nicht erkannt !":END
220 POKE &A331,&D4:POKE &A338,&69:POKE &
A339,&E8:POKE &A399,&D4:POKE &A39D,&AA:P
OKE &A39E,&E7:POKE &A3C0,&2
230 POKE &A3C1,&AC:POKE &A3CF,&5B:POKE &
A3E0,&5B:POKE &A3F2,&2:POKE &A3F3,&AC:PO
KE &A3EF,&8A:GOTO 260
240 POKE &A331,&CF:POKE &A338,&64:POKE &
A339,&E8:POKE &A399,&CF:POKE &A39D,&A5:P
OKE &A39E,&E7:POKE &A3C0,&2:POKE &A3C1,&
AC
250 POKE &A3CF,&5E:POKE &A3E0,&5E:POKE &
A3F2,&2:POKE &A3F3,&AC:POKE &A3EF,&8A
260 CALL &A300:END
1001 DATA 01,0A,A3,21,19,A3,CD,D1,&0329
1002 DATA BC,C9,12,A3,C3,C3,A3,C3,&0526
1003 DATA D4,A3,45,49,CE,41,55,D3,&043C
1004 DATA 00,00,00,00,00,CD,BF,A3,&022F
1005 DATA F5,C5,D5,E5,2A,EF,A3,28,&0558
1006 DATA 6A,E5,CD,98,A3,E1,30,63,&04CB
1007 DATA CD,04,EE,ED,53,EB,A3,CD,&055A
1008 DATA A3,E7,22,E5,A3,ED,43,E7,&054B
1009 DATA A3,09,22,E9,A3,21,00,00,&027B
1010 DATA 22,ED,A3,3A,E7,A3,32,EE,&0496
1011 DATA A3,D6,05,47,DD,21,40,00,&0303
1012 DATA B7,DD,7E,00,5F,3A,ED,A3,&043B
1013 DATA 83,B7,07,32,ED,A3,DD,23,&0403
1014 DATA 10,EE,2A,EB,A3,B7,85,B7,&04A9
1015 DATA 07,B7,84,32,ED,A3,2A,ED,&041B
1016 DATA A3,3E,20,CD,5A,BB,CD,5A,&040A
1017 DATA BB,3E,5B,CD,5A,BB,7C,CD,&047F
1018 DATA A2,A3,7D,CD,A2,A3,3E,5D,&046F
1019 DATA CD,5A,BB,E1,D1,C1,F1,C9,&060F
1020 DATA CD,04,EE,D0,CD,D2,E6,37,&054B
1021 DATA 9F,C9,5F,0F,0F,0F,0F,E6,&02E9
1022 DATA 0F,CD,B3,A3,7B,E6,0F,CD,&046F
1023 DATA B3,A3,C9,FE,0A,38,02,C6,&0427
1024 DATA 07,C6,30,CD,5A,BB,C9,CF,&0477
1025 DATA 98,AA,F7,3E,FF,32,00,AC,&0454
1026 DATA 01,03,00,21,F4,A3,11,3A,&0207
1027 DATA BD,ED,B0,C9,3E,00,32,00,&0393
1028 DATA AC,01,03,00,21,F1,A3,11,&0276
1029 DATA 3A,BD,ED,B0,C9,00,00,00,&035D
1030 DATA 00,00,00,00,00,00,00,A4,&00A4
1031 DATA AC,CF,98,AA,C3,1D,A3,&0440
```

CPC – Schneider Flugtraining CPC 464, 664, 6128. Umfassende Bewertung der Flüge. A) Hubschrauber-Simulator in Aktion. 9 Cockpit-Anzeigen. 3 Flugprog. zur Wahl 29 DM B) Space Shuttle-Landung Echtzeitsimulation, nach NASA-Unterlagen. 29 DM C) Boeing-727 Simulator. Programm ist zur Anfänger- + Instrumenten-Flugschulung geeignet. Mit Anleitg. 34 DM Ab 2 Progr. jedes minus 5 DM Disk plus 7 DM. Info gegen Rückporto. Fluging. Jahnke Am Berge 1, 3344 Flöthe 1 Tel. 05341/91618

VERKAUFE Schneider CPC 664 m. Grünmon., Drucker CMC-CPM 80 GS, Programme, Joyst., Leerdisk., Data Becker Buch, orig. Textverarb., Preis VB 1.200,– DM. Ingo Böhm, Ackermannstr. 5, 3509 Morschen, Tel. 05664/1227

Suche gebrauchte Disketten bis 4,– DM. Angebote an: Oliver Scheu, Stettiner Str. 2, 4558, Bersenbruch, Tel. 05439/3351

Täglich frische Freeware! Tausch u. Versand v. US- u. dt. Freeware f. IBM-PC u. Kompatible. Info-Disk. gg. 1,5 DM Rückporto bei: Martin Nerger, AKKUSOFT PC-CLUB, Ankerstr. 3, 5650 Solingen 11

VERKAUFE: Programme u. Spiele f. CPC-Computer auf Disk. u. Kass. Info: Mario Pahmeyer, Bergstr. 10, 1000 Berlin 41

VERKAUFE: 6128 grün, DMP 2000, FD1, Keyboard 464, Preis VHS. Tel. 07427/2519

Die PUBLIC DOMAIN-USER-GRUPPE bietet Public-Domain-Software f. alle Schneidercomputer an: CPC 464, 6128, Joyce u. PC. Info gg. zwei 80er Rückporto. Katalogdisk 12,– DM (PC 2,40 DM). PDUG, PF 1118, 6464 Linsengericht

---

Verkaufe Schneider PC 1512 m. Vortex-Filecard DM 2.300.–, DMP-2000 f. DM 400,–. Tel. 089/701956, Sa. u. So.

GEBE GÜNSTIG AB, klar Originale m. Handbuch usw.: Textomat, Datamat, Hanse, TexPack, Factura, Kassenbuch, Lagerverw., Videothek, Schneider Basic: der kompl. Sprachumfang, div. DB-Bücher f. 464 bzw. 6128, Tel. 07427/2519

LICHTGRIFFEL mit Programm für CPC 464 oder CPC 664 + CPC 6128 nur 49,– DM. Versand gegen Scheck/Nachnahme. Info gratis! Fa. Schißlbauer, Postfach 1171S, 8458 Sulzbach, Tel. 09661/6592 bis 21h

SUCHE Anwend.- u. Spielprog. auf 3" Disk. CPC 6128. Angebote an: P. Pfeiffer, Bismarckstr. 59, 6050 Offenbach

---

Wer kann mich darüber informieren, wie man von einem CPC 6128 m. Disk.-Laufw. Spiele auf einen CPC 464 m. Datasette überspielt? Die ersten Einsender erhalten ein Überraschungsgeschenk. Karl-Heinz Rullmann, Ravolzhauserstr. 8, 6451 Neuberg 2

Mache Sicherheitskopien: Orig.-Disk/Leer-Disk. 5,– DM inkl. Porto an: Oliver Scheu, Stettiner Str. 2, 4558 Bersenbrück, Tel. 05439/3351. Garantiert schnelle Bearbeitung innerhalb von 3 Tagen.

SUCHE f. CPC 6128 AFU Soft- u. Hardware. Möchte m. User Kontakt aufnehmen, der XRS-Schnittstelle f. 6128 hat od. umgebaut von 464 auf 6128 hat od. ein 3.5"-Laufwerk besitzt. Krüger, Schorlemer Str. 80E, 4440 Rheine, Tel. 05971/56719

*** SUCHE TAUSCHPARTNER *** Tausche Programme aller Art auf 3" u. 5.25" Disketten f. alle CPCs. Schickt Eure Listen an: Dirk Frankenhauser, Handjakobweg 7, 7562 Gernsbach

---

Verkaufe Schneider Drucker NLQ 401 m. Traktoraufs., 1 Jahr alt, VB 400,– DM. H. Stromer, Ditzenbacher Str. 8, 705 WN 5

ORIGINALE: Hi jack 20,– DM; RSE-Mathepack 25,– DM; Colossus 4.0 20,– DM; Tomahawk 18,– DM; alles zus. 60,– DM. Vortex SP128 3 Bänke f. 140,– DM. Bei Reiner Klohn, Silcherstr. 2, 7916 Nersingen, Tel. 07308/3229 ab 18 Uhr

---

Wir helfen bei Fußball-Turnieren m. Know How, Soft- u. Hardware. Fußball-Fanclubs werden bevorzugt. Gratis-Info v. DFFV, Eisackstr. 9, 1000 Berlin 62

SUCHE preiswerten Drucker DMP 2000/3000. Angebot schriftl. an: Klaus Gerke, Auf dem Damm 27, 4100 Duisburg 12

DRUCKER-FARBBAND TOPNEU! Mit Druckerfarbe u. Interninfo, einfach, schnell, dauerhaft, statt vieler neuer Farbbänder. Tel. 07427/2519

SUCHE Tauschpartner für Spiele f. CPC 6128. Garantierte Antwort!! H.-J. Schwebler, Kolleg, 7822 St. Blasien

UMSATZSTEUERVORANMELDUNG. Schneider CPC 464, 664, 6128, Kassen- und Girokontobuchführung mit UST-Voranmeldung. Einfache Anwendung für Haushalt, Vereine und Einzelfirmen für DM 70,– + Porto u. Vers. Siegfr. Bastian, Zusenhofener 15, 7602 Oberkirch 2, Tel. 07805-39 47

TAUSCHE Software auf Disk. 5 1/4 Zoll Vortex. Verk. 30 Computerhefte PM. Angebote an: Pomareda Luis, Kongreßstr. 23, App. 38, 5100 Aachen

---

Erstelle Sicherheitskopie auf 3-Zoll-Disk. f. CPC 6128. Schickt Orig.- u. Leer-Disk., Rückumschlag u. 5,– DM an: Mario Schwies, Beethovenstr. 8, 6970 Lauda-Königshofen

---

OKIMATE 20 Farbdrucker m. Centr.-Parallel-Schnittst. zu verkaufen. Preis VS. Außerdem div. Zeitschr. u. Spiele f. CPC auf Disk. Tel. 07942/3145

HILFE! Ich bin 14 Jahre alt u. im Besitz eines CPC 464 m. Kass.-Rec. Bin Anfänger. Wer kann mir auf Kass. Spiele aufnehmen. Sehr dringend. Meldet Euch. Die ersten Einsender erhalten ein Geschenk. Martin Delzlut, Zum Goldacker 6, 6450 Hanau 8, Tel. 06181/690864

1000 Berlin

SEI NICHT DUMM... KOMM RUM... CPC DIE GANZE WELT DES SCHNEIDER CPC

Herbert Köcher GbR

A + C Vertrieb, 1 Berlin 44, Emser Str. 18

Tel. 625 36 05, Mailbox 6252098

1000 Berlin

COMPUTERE

Wolfgang Hiller

Schneider COMPUTER DIVISION

Hardware
Software
Zubehör
Beratung
Literatur

Tempelhofer Damm 120
1000 Berlin 42
Am U-Bhf. Tempelhof
Tel.: 7 52 20 91

6090 Rüsselsheim

Computer u. Bürotechnik

Dipl.-Ing. Neuderth

Frankfurter Str. 23/Ecke Friedenspl.

6090 Rüsselsheim
Tel. 06142-68455

Gratisliste für alle CPC's anfordern bei: Friedrich Neuper, Postfach 72, 8473 Pfreimd

---

CPC 6128: Suche SOFTWARE-TAUSCHPARTNER, evtl. zur Gründung eines User-Club-Raum 235 Norderstedt. Nur Disk. Listings gg. Rückporto gratis! Jürgen Fürhoff, Gorch-Fock-Str. 22, 2359 Henstedt-Rhen, Tel. 04193/7262 ab 18 Uhr

---

CPC 664 (Floppy, Grünmon. GT 65, Auto-Bitcorder f. autom. Kass.-Betrieb) inkl. umfangreiche dt. Handbü., reichhaltige Software auf 40 Disk. (Mal-, Text-, Spielprog. u.v.m.) f. 720,– DM. Tel. 02421/71391 nach 18 Uhr

---

ADRESSENVERWALTUNG
mit Etikettenausdruck, Karteiverwaltung mit Einzeldruck od. Gesamtausdruck aller Adressen. (z.B. Rundschreiben) Geeignet für Haushalt, Vereine und Firmen.
Preis DM 40,– + Porto u. Vers.
Siegfr. Bastian, Zusenhofener 15
7602 Oberkirch 2
Tel. 07805 / 39 47

** VERKAUFE/TAUSCHE CPC-Software. SUCHE gute Anwend.-Prog. D. Maron, Bahnstr. 26, 2085 Quickborn

---

VERKAUFE CPC 6128 grün, DMP 2000 m. Zubehör, Lit., CPC-Magazinen, 11 Disk. m. Software (auch einzeln) gg. Gebot. Tel. 06633/1788 abends

---

VERKAUFE UND TAUSCHE Spiele! Habe Traktor, Arkanoid, Elite u.v.a. Listen an: Oliver Scheu, Stettiner Str. 2 4558 Bersenbrück, Tel. 05439/3351

SUCHE digitalen Infrarot-Joystick f. CPC 464, G. Neitzel, Am Westhang 74, 5860 Iserlohn

---

****SUPER-LOHN-EINKOMMENSTEUER****
Jahresausgleich 87 neu m. Datenspeicher, Kundenverw., Formularadr., Analyse; ab 70,–; jährl. Aktu. 10,–; Info gg. RP; PC-Demo-Disk 10,–.
**** MIET-, WOHNGELDBERECHNUNG ****
Mit allen Kreisen d. BRD! Jedes Prg. ab 70,–.
H-I-Software H. Ilchen, Niederfelderstr. 44, 8072 Manching, 08459/1669

BIETE f. PCs: billige Software u. Zubehör (Turbo Pascal, Turbo Basic, Framework II, dBase III etc.) Info kostenlos v. M. Lehmann, Mittelweg 11, 6000 Frankfurt

---

SUCHE!!! Software f. CPC 6128 (C/D) u. Drucker (Typenrad od. Matrix). Angebote an: R. Otte, Kahlerstr. 190, 4830 Gütersloh 1

VERKAUFE: selbstgeschriebene Programme, Info bei: Mario Pahhmeyer, Bergstr. 10, 1000 Berlin 41, Tel. 030/7928560

Verkaufe PC INTERNATIONAL-Hefte 10/85, 12/85, Jahrgänge 1986/87 komplett. Alle Hefte zus. DM 85,– inkl. Porto. Für Einzelhefte nachfragen: Gerd Rotfuchs, An der Weid 28, 6799 Etschberg, Tel. 06381/3438 nach 18 Uhr

---

100 Orig.-CPC-Spieletapes 300,– DM od. einzeln ab 5,– DM. Liste gg. Freiumschlag. Reinhard Ludwig, Schäringer-Str. 1, 8000 München 19

Löse meine Softwaresammlung auf! Kass. v. 5,– bis 15,– DM. Bücher ab DM 10,–. Auch noch Disk. vorhanden f. DM 18,–. Nur Orig.-Software. Kein Tausch! Liste gg. Rückp. od. anrufen! 06638/1503 v. 15–20 Uhr

---

TAUSCHE UND VERKAUFE Spiele auf Disk. (über 250 Stück). SUCHE gute Anwend.-Prog. User-Club-Gründung im Kreis Remseck angestrebt. Arne Krüger, Karl-Rohm-Weg 11, 7148 Remseck

---

Suche Tauschpartner f. 5 1/4-Zoll-Disk (Cumana). Interesse an Anwendung u. Spielen! Suche Disk. zu Zeitschr. (Listings), suche Drucker, suche Farbmon. (6128). T. Wimmer, Röthenstr. 15, 8643 Küps

---

MICROSOFT-BASIC (Orig. M.+T.) enthält Macro-Assembler; Basic unter CP/M 2.2 u. 3+ kein Problem mehr. Als Interpreter u. Compiler verwendbar. Für 464/664 m. Speichererw., 6128 od. Joyce DM 110,–. Mit umfangr. dt. Handbuch. Tel. 02421/71391 nach 18 Uhr

---

SUCHE Software jeder Art f. CPC 664 (nur Disk.) Angebote m. Preisen an: Gerhard Meßmann, Nobelstr. 4, 8418 Teublitz 2. PS: Besonders an Textverarb. interessiert!

Verkaufe orig. Wordstar 464/664 (M&T) u. Budget Manager 464 (Data Becker), zus. f. 90,– DM. Tel. 02461/7266

VERKAUFE CPC 6128, div. Spiele, Wordstar 3.0, ca. 14 Disk., 1 Basic-Buch, ca. 40 Computerzeitschr., nur zus. f. FP 1.600,– DM. Tel. 07394/2169 ab 18 Uhr

---

ACHTUNG Kabel zum Anschl. f. CPC an Farb-TV m. Scartbuchse 35,– DM. Kabel zum Anschl. f. CPC an Farb-TV m. 6-pol. Videobuchse 45,– DM. Stereokabel CPC an Stereoanlage 12,– DM. 464 Monitorverl. 20,– DM. Nur gg. V-Scheck od Nachn. Tel. 026032849

SUCHE ZWEITFLOPPY FD-1, voll funktionsf. u. anschlußfertig f. CPC 464. Zahle bis 200,– DM. Tel. 069/7306925 ab 16 Uhr

---

FREEWAREGRUPPE IM AUFBAU! Habt Ihr selbstgeschriebene Programme f. CPC od. PC? Wenn diese gefallen, bekommt Ihr mehrere andere Programme nach Wahl dafür! Schreibt an: M. Merger, AKKUSOFT PC-Club, Ankerstr. 3, 5650 Solingen 11

++WANTED!++ Schüler sucht Drucker u. Zweitfloppy (Format unwichtig) f. CPC. Angebote an Heiko Weiland, Hauptstr. 16, 6791 Bosenbach

---

WACCI, Jeff Walker's europaweiter Armstrad-Schneider-Club sucht noch Mitglieder zum Mitmachen – auch Österreich u. Schweiz – Hard, Soft, Public Domain, Freesoft, Newsletter usw.: WACCI, Enzianstr. 10, 7464 Schömberg

---

Suche ERf.-Austausch im Raum FfM! Mit älterem CPC-User. Suche/biete außerdem Anwend.-Softw. (COBOL-Compiler, dBase u. Turbo Pascal) auf 3"-Disk. Suche auch Drucker, 2. Laufw. Antworte garantiert! H. Reich, Idsteiner Str. 59, 6 FfM 1, 069/7306329

7054 Korb

RICHARD HEINRICH & SOHN

Computer + Software
Winnender Str. 25, 7054 Korb
Tel. 07151/325 13

7700 Singen

Schneider COMPUTER DIVISION elektronik service

Ihr Fachhändler
Marianne Tröndle
Hauptstr. 11, 7700 Singen
(Hohentwiel), Tel. 07731/64 43 33

7700 Singen

Ihr kompetenter, autorisierter
SCHNEIDER-Fachhändler
Schellhammer
Hard- u. Software, Peripherie, Literatur
Fachkundige Beratung · kompletter Service
Hauptstr. 11, 7700 Singen
Tel. 07731/82020

VERDIENEN SIE GELD MIT IHREM COMPUTER!

WIR ZAHLEN BIS ZU 1000 DM!

Haben Sie einen CPC 464 oder 664? Einen 6128? Können Sie programmieren? In Basic oder Maschinensprache? Dann bietet SCHNEIDER CPC-WELT Ihnen die Möglichkeit, mit diesem Hobby Geld zu verdienen!

Wie? Ganz einfach. Sie senden uns die Programme, die Sie für einen Abdruck als geeignet halten, zusammen mit einer Kurzbeschreibung, aus der auch die verwendete Hardware — eventuelle Erweiterungen — benutzte Peripherie — hervorgehen muß, ein.

Benötigt werden: eine Datenkassette oder Diskette! Wenn die Redaktion sich überzeugt hat, daß dieses Programm läuft und sich zum Abdruck eignet, zahlen wir Ihnen pro abgedrucktem Programm je nach Umfang bis zu DM 300,-! Für das „Listing des Monats" sogar DM 1.000,-. Sie erhalten Ihre Kassette/Diskette selbstverständlich zurück, wenn Sie einen ausreichend frankierten Rückumschlag mit Ihrer Adresse beifügen.

Bei der Einsendung müssen Sie mit Ihrer Unterschrift garantieren, daß Sie der alleine Inhaber der Urheber-Rechte sind! Benutzen Sie bitte anhängendes Formular! (Wir weisen darauf hin, daß auch die Reaktion englische Fachzeitschriften liest und „umgestaltete" Programme ziemlich schnell erkennt).

Um Ihnen die Arbeit zu erleichtern, finden Sie hier ein Formular. Sie können es ausschneiden oder fotokopieren.

Name des Einsenders: _____
Straße/Hausnr./Tel.: _____
PLZ/Ort: _____

Hiermit biete ich Ihnen zum Abdruck folgende(s) Programm(e) an:

_____
_____
_____
_____

Benötigte Geräte: _____
_____

Beigefügt ☐ Listings ☐ Kassette ☐ Diskette

Ich versichere, der alleinige Urheber des Programms zu sein!
Hiermit ermächtige ich die Redaktion, dieses Programm abzudrucken und wirtschaftlich zu verwerten. Sollte es in den Kassetten-Service aufgenommen werden, erhalte ich auch dafür eine entsprechende Vergütung. Das Copyright geht an den Verlag über.

_____
Rechtsverbindliche Unterschrift

_____
Unterschrift des Erziehungsberechtigten bei Minderjährigen

SCHNEIDER CPC-WELT
PROGRAMM-REDAKTION
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

EINE HARDWARE-UHR FÜR DIE CPCs, TEIL 2

Beim CPC tickt es

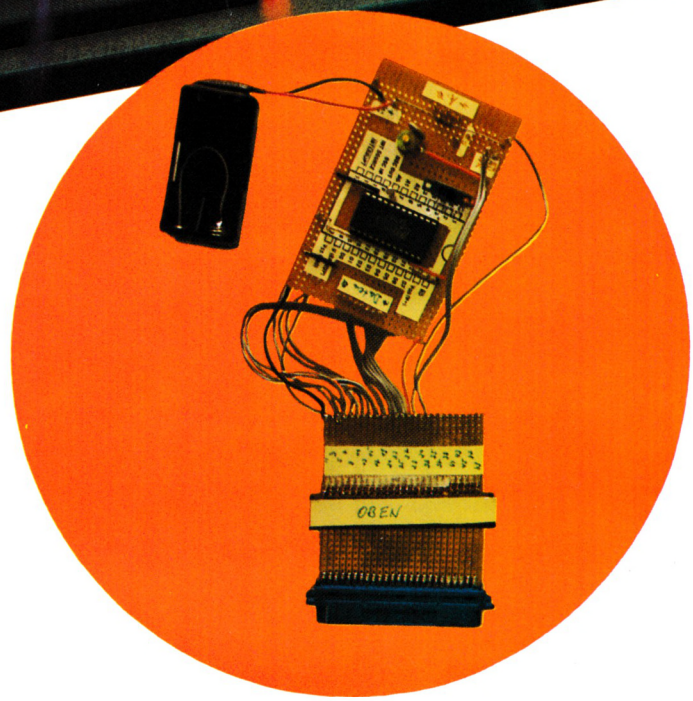
Im zweiten Teil unseres Artikels über die CPC-Hardware-Uhr lesen Sie, wie die Uhr gepuffert wird. Auch Hinweise auf die Register fehlen nicht. Außerdem erfahren Sie, wie die Genauigkeit justiert wird.





Der Autor hat bereits gute Erfahrungen mit dem 7170 von Intersil in Verbindung mit anderen Computern gemacht. Aus diesem Grunde fiel die Wahl auch diesmal wieder auf diese Echtzeit-Uhr. Wie manche anderen Uhrenbausteine hat er den Vorteil, relativ wenig externe Bauteile zu benötigen. Sprechen wir diesen Baustein nun detaillierter durch. Die Spannungsversorgung bräuchten wir nicht weiter zu erwähnen, wenn wir diesen Baustein so betreiben wollten, daß er nach Abschaltung der CPC-Versorgungsspannung sein Wissen verliert. Aber dann wäre eine

Das Platinenlayout veröffentlichen wir im nächsten Heft.



CPC ABGESCHALTET – UHR LÄUFT WEITER

Hardware-Uhr relativ sinnlos, denn der Vorteil ist, daß der CPC immer dann, wenn er die genaue Zeit und/oder das Datum benötigt, diese auch abholen kann.

Der ICM7170 geht, wenn er mit einer sogenannten Backup-Betriebsspannung von etwa 2,6 bis 3,2 Volt gepuffert wird, in einen Zustand über, der nur noch minimale Aktivitäten aufrecht erhält. Die Uhr läuft intern zwar noch, aber es können keine Daten mehr von außen angefordert werden. Der Stromverbrauch ist dabei so gering, daß die Uhr mo-

natelang aus einem Akku gepuffert werden kann. In den Datenblättern ist die Backup-Versorgungsspannung zwar mit 2,6 bis 3,2 Volt angegeben. Erfahrungen mit vielen Uhrbausteinen für den C64 und unsere praktischen Versuche mit dem „Labormuster" der Schaltung zeigten aber, daß auch 2,4 Volt ausreichten. Deshalb realisierten wir die Backup-Versorgung durch zwei 1,2-Volt-Akkus. Laut Herstellerangeben ist der Backup-Strom dabei typisch etwa 2 uA. Es wäre deshalb sogar möglich, eine Pufferung von einigen Minuten über einen gar nicht allzu großen Kondensator durchzuführen.

DIE REGISTER-AUSWAHL

Obwohl durch fünf Adreß-Eingänge (A0 bis A4) insgesamt zwei hoch fünf, also 32 Register angesprochen werden könnten, hat der ICM7170 nur 18. Die Funktionen und die Bedeutung können Sie *Bild 6* und dem Programmlisting entnehmen. *Bild 8* zeigt die Pinbelegung des Uhrenbausteines. Falls Sie Probleme mit den Registeradressen haben sollten: Register 0 liegt auf der Port-Adresse &F8E0, Register 1 bei &F8E1, das ergibt sich durch die Decodierung.
Das Kommando-Register spielt eine besondere Rolle, denn durch die dort eingeschriebenen Werte bestimmen wir die Betriebsart. Der Baustein kann beispielsweise im Zwölf- oder 24-Stunden-Modus arbeiten und mit verschiedenen Quarzfrequenzen betrieben werden. *Bild 9* gibt hierüber Auskunft.
Wer weitergehende Informationen benötigt, weil er die Uhr in anderer Weise betreiben will, sollte sich die – mehrere Seiten umfassenden – Unterlagen zu diesem Baustein besorgen.

BAUTEILE-LISTE:

1 x 74LS30
1 x 74LS32
1 x ICM7170
1 x Quarz 32.768 kHz
1 x Widerstand 220 Ohm
1 x Widerstand 2,2 kOhm
1 x Widerstand 2,7 kOhm
1 x Kondensator 18 pF
1 x Trimm-Kondensator 4 – 30 pF
1 x Diode 1N4148
1 x Schalter (einpolig – um)
1 x 50poliger Platinenstecker
1 x Lochrasterplatine zweiseitige als Durchführungsplatine
1 x Uhrenplatine
1 x Akku 2,4 Volt (2 x 1,2 V)

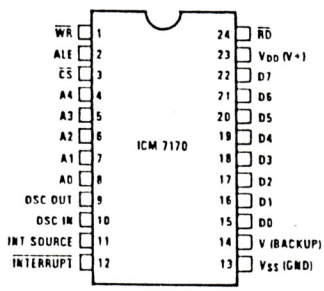


Bild 8: Die Pin-Belegung des ICM7170

Der Datenverkehr in beide Richtungen geschieht über die acht Datenleitungen des Datenbusses. Damit dem Baustein aber mitgeteilt werden kann, ob er Daten ausgeben oder annehmen soll, müssen ihm noch die Signale für Schreiben oder Lesen zugeführt werden. Diese stehen beim CPC ebenfalls am Erweiterungsanschluß, an den unsere Schaltung angeschlossen werden muß, an.
Wer einen Schaltplan des CPC besitzt, findet diese Anschlüsse unter den Bezeichnungen WR und RD. Der Strich über diesen Buchstaben weist darauf hin, daß die Signale Low-aktiv sind.
Wie Sie unserem *Schaltplan* entnehmen konnten, führt der Anschluß WR vom 7170 auf einen Schalter. Der Sinn dieses Schalters ist, nach dem Einschreiben der Zeitwerte, also dem „Stellen" der Uhr, keine weiteren, versehentlichen, Schreibvorgänge mehr zuzulassen. Zu diesem Zweck wird die vom Uhrenbaustein kommende Leitung auf Pluspotential gelegt.
Der Rest der Schaltung ist relativ einfach. Der Quarz bildet, in Verbindung mit den beiden an den Pins 9 und 10 liegenden Kondensatoren und der Innenschaltung des ICs, einen sehr frequenzstabilen CMOS-Pierce-Oszillator. Der an Pin 10 liegende Trimmkondensator dient zur Korrektur der Taktfrequenz.

ES GEHT AUCH OHNE GROSSEN MESSGERÄTEPARK

Da der meßtechnische Aufwand zur genauen Einstellung der Oszillatorfrequenz wegen der kapazitiven Belastung durch Meßmittel sehr hoch ist, empfehlen wir, auf folgende Weise vorzugehen:

- ☐ Bei der Erst-Inbetriebnahme den Trimmer CT in Mittelstellung bringen.
- ☐ Eine Stunde nach dem erstmaligen Stellvorgang mit einer anderen, aber möglichst genau gehenden, Digitaluhr verstellen. Richtung der Verstellung notieren.
- ☐ Uhr neu stellen. Wieder eine Stunde warten und, je nachdem, ob die Abweichung größer oder kleiner geworden ist, in die eine oder andere Richtung drehen.
- ☐ Die Vorgänge innerhalb größerer Zeitspannen wiederholen.

Nach ein paar Tagen der Beobachtung und der kleinen Justagen haben Sie eine äußerst genaue Echtzeit-Uhr für Ihren CPC, die so leicht nichts aus der Ruhe bringt. Da

```
100 '*****************************
*************************          <3B18>
110 '* Programm fuer die Echtzeit-
Uhr von Lothar Miedel *            <3BDA>
120 '*****************************
*************************          <3B40>
130 std=16:minu=34:sek=0:day=29:y$
="1987":m$="November":w$="Sonntag"
                                   <5776>
140 DIM a$(15),b$(7):bas=&F8E0:cre
g=bas+&11:bl$=STRING$(20," ")      <456A>
150 FOR q=1 TO 12:READ a$(q):NEXT
q                                  <214A>
160 FOR q=0 TO 6:READ b$(q):NEXT q
                                   <2080>
170 MODE 2:PRINT"Echtzeit-Uhr      <167C>
180 INPUT"Uhr einstellen (j/n)";v$
:v$=LOWER$(v$)                     <2FE9>
190 IF v$="n"THEN 420              <125C>
200 PRINT:PRINT"Bitte Schreib-Scha
lter nach links":CALL &BB18        <30AF>
210 GOTO 240                       <0919>
220 INPUT"Stunden ";std:INPUT"Minu
ten ";m:INPUT"Sekunden ";sek       <3CDB>
230 INPUT"Monat";m$                <121D>
240 FOR q=1 TO 12                  <0FD6>
250 IF m$=a$(q)THEN monat=q        <23FC>
```

```
260 NEXT q                                  (0A9A)
270 GOTO 290                                (0928)
280 INPUT"Datum ";d:INPUT"Jahr";f:
INPUT"Wochentag";w$                         (3253)
290 FOR q=1 TO 7:IF w$=b$(q)THEN w
day=q                                       (2C4F)
300 NEXT q                                  (0AEA)
310 year=VAL(RIGHT$(y$,2))                  (1BBF)
320 OUT(bas+17),12                          (14B6)
330 OUT(bas+1),std:REM Stunden-Ein
stellung                                    (2D7D)
340 OUT(bas+2),minu                         (18EF)
350 OUT(bas+3),sek                          (172F)
360 OUT(bas+4),monat                        (192A)
370 OUT(bas+5),day                          (1778)
380 OUT(bas+6),year                         (1833)
390 OUT(bas+7),wday                         (180B)
400 OUT creg,12                             (1034)
410 CLS:PRINT"Bitte Schreib-Schalt
er umstellen auf normal!":CALL &BB
18                                          (3B6A)
420 CLS:LOCATE 1,8:PRINT"Std.:              (1333)
430 LOCATE 1,10:PRINT"Min.:                 (12F0)
440 LOCATE 1,12:PRINT"Sek.:                 (12D1)
450 REM Zeit holen                          (1145)
460 xx=INP(bas)                             (15E0)
470 h=INP(bas+1):IF h>60 THEN 470           (23D4)
480 m=INP(bas+2):IF m>60 THEN 480           (2331)
490 sek=INP(bas+3):IF sek>60 THEN
490                                         (27A8)
500 r=INP(bas+4):IF r>12 THEN 500           (23AB)
510 mtag=INP(bas+5):IF mtag>31 THE
N 510 ELSE IF mtag=0 THEN 510               (3988)
520 y=INP(bas+6):IF y=255 THEN 520
                                            (23B7)
530 wtag=INP(bas+7):IF wtag>7 THEN
 530                                        (287A)
540 IF sek=alts THEN 460                    (1891)
550 LOCATE 10,12:PRINT bl$:LOCATE
1,10                                        (1828)
560 IF r>12 THEN r=12                       (15D1)
570 IF r<1 THEN r=1                         (133E)
580 IF ud<2 THEN ud=2                       (1517)
590 LOCATE 10,8:PRINT h                     (1008)
600 LOCATE 10,10:PRINT m                    (117B)
610 LOCATE 10,12:PRINT sek                  (1321)
620 LOCATE 1,5:PRINT"Datum: ";b$(w
tag);" - ";mtag". "a$(r);" ";y              (478B)
630 alts=sek                                (1314)
640 GOTO 470                                (09AB)
650 DATA Januar,Februar,Maerz,Apri
l,Mai,Juni,Juli,August                      (360A)
660 DATA September,Oktober,Novembe
r,Dezember                                  (2A01)
670 DATA Sonntag,Montag,Dienstag,M
ittwoch,Donnerstag,Freitag,Samstag
                                            (4204)
```

die Schaltung nicht in den Redaktionsräumen entwickelt wurde, sondern zu Hause beim Autor, wurde unser Labormuster mehrere Tage lang hin- und hertransportiert. Nach jedem Anstecken des Moduls ergab sich: Die Uhrzeit stimmte, und das, obwohl das Modul ein ziemlicher Drahtverhau war.

ZUM MECHANISCHEN AUFBAU

Da die Anschlüsse der CPCs unterschiedlich sind, hier eine Anleitung für den CPC464/664. 6128-Besitzer müssen entweder Zwischen-Adapter verwenden oder die Uhrenplatine anders aufbauen.
Am besten ist es, wenn Sie mit zwei Platinen arbeiten. Auf einer Platine wird die komplette Schaltung der Uhr untergebracht.
Im Bereich der Busanschlüsse wird ein Durchbruch geschaffen, durch den eine zweiseitige Lochraster-Platine gesteckt werden kann.

Bit5 Test	*0 = Normal-Mode*	*1 = Test-Mode*	
Bit4 Interrupt	*0 = Int. disabled*	*1 = Int. enabled*	
Bit3 Run	*0 = Stop*	*1 = Run*	
Bit2 Stunden-Modus	*0 = 12 Std.*	*1 = 24 Std.*	
Bit1 Frequenz	*00 = 32.768 kHz*	*01 = 1.048576 MHz*	
Bit0 Frequenz	*10 = 2.097152 MHz*	*11 = 4.194304 MHz*	

Bild 9: Bedeutung der Bit für das Kommando-Register

Die zweite Platine (es wird nur ein kurzes Stück von etwa vier bis fünf Zentimetern Länge benötigt) wird auf der einen Seite mit den Kontakten eines 50poligen Platinensteckers verlötet (pro Anschlußkontakt ein „Streifen", also keine Kurzschlüsse erzeugen). Danach wird die mit dem Stecker versehene Platine so durch die Uhrenplatine gesteckt, daß die weitere CPC-Pheripherie dort wieder angesteckt werden kann.
Damit Sie auch gleich ein Programm haben, mit dem Sie die Uhr stellen und auslesen können, finden Sie zu diesem Artikel ein kleines *Listing*, das als Beispiel für Ihr eigenes Programm dienen kann. Zum Stellen der Uhrzeit muß der Schalter S1 so stehen, daß die Verbindung CPC-WR und 7170-WR besteht, danach umschalten. In Zeile 210 des Listings haben wir die Zeit stehen. Passen Sie für den einmaligen Stellvorgang diese Zeile an oder entfernen Sie sie, dann erfolgt beim Programmlauf die Abfrage der einzelnen Daten.
Nach dem Programmstart werden Sie gefragt, ob die Uhr gestellt werden soll. Beim ersten Mal antworten Sie mit ja, ansonsten mit nein.
Die Aufforderung „Schreibschalter nach links beziehungsweise nach rechts" kann bei Ihnen allerdings falsch sein, denn wir wissen nicht, welchen Schalter (Schiebeschalter oder Kippschalter) Sie bei Ihrem Uhren-Modul verwenden.
Einige BASIC-Zeilen (ab 470) enthalten IF-Abfragen. Sie dienen dazu, bei Werten, die nicht gültig sein können, nochmals auszulesen. Normalerweise aber werden diese Abfragen nicht benötigt.
Die Uhr können Sie bei ausgeschaltetem CPC jederzeit abziehen und anstecken, das Modul arbeitet weiter. Wenn mit Akkus gearbeitet wird, werden diese bei eingeschaltetem CPC über den Fünf-Volt-Anschluß geladen.
Wenn Sie Interesse an solchen Basteleien haben, teilen Sie es uns bitte mit. Je nach Leser-Reaktion werden wir noch andere Erweiterungen veröffentlichen. *LM* □

Bastard - echt kriminell

Die meisten Computerfreaks beschäftigen sich neben der Lektüre von Fachmagazinen und Büchern bisweilen auch mit etwas leichterer Kost in Form von Filmen, die sich mit dem Computer beschäftigen. Da es in solchen Produkten meist mit erheblichem Hang zu Klischees und übertriebenen Utopien zugeht, nahmen wir einmal – zur Information für Sie – die Produktion eines Filmes mit dem Schwerpunkt Computer unter die Lupe.

Filme wie Wargames der L.I.S.A. dürften den meisten unserer Leser bekannt sein. Hauptgegenstand dieser Spielfilme war – wie auch in anderen Werken – der Computer. Doch zumeist neigen solche Streifen zu übertriebenen Utopien

MEHR SCIENCE-FICTION ALS REALITÄT

und tendieren mehr zur Science-fiction als zur Realität.
Um so mehr wurde unser Interesse von einer Presseinformation aus den Bavaria Filmstudios geweckt. Quasi vor unserer Haustür fanden die Dreharbeiten zu einer sechsteiligen Kriminalserie für das deutsche Fernsehen statt, deren Hauptgegenstand das Thema Computerkriminalität ist. Wir wollten hinter die Kulissen sehen und feststellen, wie solche Filme überhaupt zustande kommen. In Gesprächen mit Schauspielern und Produzent sowie der Besichtigung von Kulisse und einem der Drehorte informierten wir uns vor Ort.
Die Serie wird voraussichtlich Ende des Jahres unter dem Titel Bastard im deutschen Fernsehen ausgestrahlt. Es ist die Geschichte eines Computerfreaks, der sich auf eine Baleareninsel zurückgezogen hat und dort Spiele programmiert. Auf seinen nächtlichen Reisen

VERBRECHER-ORGANISATION MIT DATENBANK

in internationalen Datennetzen verfängt er sich in den Maschen einer Datenbank, die offensichtlich von einer weltweit operierenden Verbrecherbande benutzt wird. Solch böse Bits bringen den Herrn wieder ins Alltagsleben zurück, und er widmet sich der Verbrechensbekämpfung. Um diesen Anfang rankt sich dann die gesamte Handlung.
Das Grundproblem eines solchen Filmes sieht Produzent Peter Bradatsch darin, daß zum einen die Technik zwar wahrheitsgetreu, zum anderen aber möglichst effektvoll in

UND PIEPEN MUSS ER AUCH

Szene gesetzt werden soll. So sollen Zuschauer, die mit dem Medium Computer vertraut sind, keine unmöglichen Techno-Märchen aufgetischt bekommen. Andererseits müssen aber von der Materie Unberührte von der wundersamen Computerwelt gefesselt werden. Auf der einen Seite also die nüchterne Realität eines ganz normalen Personal Computers, auf der anderen einen blinkendes und piependes Etwas, welches mit der eigentlichen Realität nichts mehr zu tun hat.
Um hierbei nicht allzu sehr auf das Niveau von Filmen wie Wargames zu sinken, bediente man sich bei der Produktion von Bastard eines Programmierers, welcher durch Bera-





Oben: Der Darsteller der Titelrolle, ein echter Bastard, wird von seinen Kollegen verwöhnt.

tung und Entwicklung geeigneter Programme möglichst sämtliche Belange unter einen Hut bringen sollte. Bradatsch meint, so das Problem in den Griff bekommen und eine gewisse Ausgewogenheit erreicht zu haben. Der Kompromiß besteht beispielsweise in einem realitätstreuen Terminalprogramm, welches aber dem besseren Effekt zu-

BERATUNG DURCH PROGRAMMIERER

liebe mit überdimensionaler Schriftgröße und piependem Bildschirmaufbau versehen wurde. Vom breiten Publikum wird ein Computer anscheinend nur dann als ein solcher erkannt, wenn er kräftig Laut gibt. Freaks mögen es dem deutschen Fernsehen verzeihen. Interessiert hat uns auch ein weiterer Aspekt: Wie sieht es mit Unterstützung

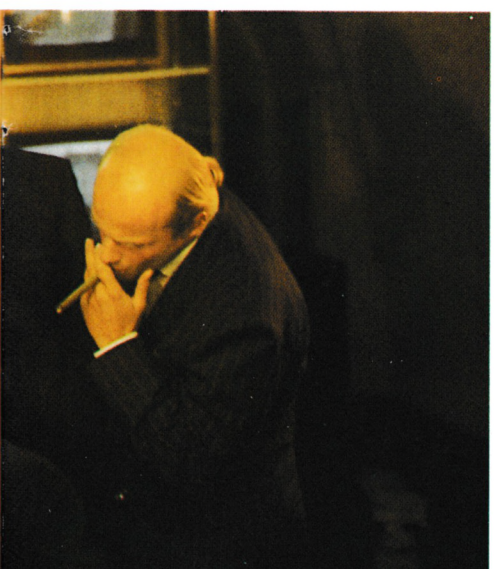
UNTERSTÜTZUNG VON HERSTELLERN

seitens der Computerhersteller aus? Wohl jeder assoziiert inzwischen das neumodische Wörtchen Product Placement, wenn beispielsweise Schimanski im Tatort sehr ausführlich ein Hustenbonbon zu Munde führt. Ähnlich verhält es sich auch im Bereich der Geräte-Ausstattung, wobei Sparmaßnahmen allerdings im Vordergrund stehen und die daraus resultierende Werbung in Kauf genommen wird. Schließlich wäre auch der Zuschauer, der letzten Endes die Requisiten mit seinen Gebühren bezahlt,

SPARMASSNAHMEN IM SINNE DES ZUSCHAUERS

unzufrieden, wenn bei den Produktionen ein sechsstelliger Betrag für technische Kulissen ausgegeben würde. Auch bei Bastard war es so: Man wandte sich an drei große Hersteller – welche, werden Sie im Herbst selbst sehen – von PCs, die





Oben: Gudrun Landgrebe und Peter Sattmann. Links: Alle Darsteller der Fernsehserie

prompt zusagten und die Produktion mit der notwendigen Hardware versorgten.
Darüber hinaus gab es aber auch noch Rat und Hilfe vor Ort. So sorgte beispielsweise die Bildwiederhol-Frequenz von sechzig Hertz eines PC für Probleme bei den Aufnahmen: Ein unschöner Streifen wanderte über den Schirm. Jeder, der schon einmal versucht hat, ein Bildschirmfoto zu machen, kennt die Balken, die entstehen, wenn die Belichtungszeit zu kurz ist. Und Filmen ist schließlich nichts anderes als Fotografie in Serie. Um dies zu beheben, wurden flugs einige Techniker der betreffenden Firma eingesetzt, um den Rechner mit einer anderen Grafikkarte auszurü-

TECHNISCHE PROBLEME MIT GRAFIKKARTE

sten. Gratis und mit Freuden, versteht sich, denn die Firma hatte schließlich kein Interesse an wandernden Bildern.
Die Sparsamkeit in der Requisite ermöglichte vielleicht die ausgezeichnete Besetzung des Filmes. Peter Sattmann spielt die Hauptrolle, jenen ausgestiegenen Programmierer und DFÜ-Freak. Für ihn ist der Computer nichts Neues. Er sei schon seit Jahren

ECHTE STARBESETZUNG

durch mehrere private Rechner damit vertraut, erklärt er. Aus der Sammlung hat er den Apple zum Lieblingsspielzeug erkoren.
Ein Bonbon für die Fotografen war natürlich Gudrun Landgrebe. Als Lisa Pallmann darf sie dem Kämpfer um Computer-Gerechtigkeit zur Seite stehen. Weitere Rollen wurden mit Heinz Reincke und Dieter Krebs besetzt.
Zusammen mit dem Regisseur Ulrich Stark geht also keine unbekannte Truppe ans Werk. Mit einer Ausnahme allerdings: dem Bastard. Das Hunde-Viecherl, das der Serie den Namen gab, nennt sich im Film Diddle und heißt in Wirklichkeit Hexie. Es ist eben alles nur Schein beim Film, doch auch hier wieder mit einer Ausnahme: der Bastard ist tatsächlich einer. *TS* □

XBC BASIC-COMPILER

Bedingt einsatzbereit

Unter dem Namen Extended BASIC-Compiler (XBC) ist für die Schneider-CPC-Rechner ein Programm erhältlich, das den Eindruck erweckt, es könne Ihre Programme erheblich beschleunigen. Ob XBC hält, was sein Name verspricht, wollten wir genauer wissen und haben das Programm für Sie unter die Lupe genommen.

Die Aufmachung des XBC-Compilers macht einen semiprofessionellen Eindruck. In einem stabilen DIN-A5-Ordner erhalten Sie die Diskette nebst Handbuch. Letzteres besteht aus einer Sammlung fotokopierter Blätter; aber die Kopien sind wenigstens gut, und die Beschreibungen sind für erfahrene Programmierer leicht zu verstehen. Für Anfänger ist XBC wohl nicht das Wahre, und zwar nicht nur aufgrund des Handbuchs.

STARTSCHWIERIGKEITEN

Zunächst möchten Sie wohl den Compiler starten. Und da wartet auch schon die erste Hürde. Die Diskette wird im Vendor-Format ausgeliefert, so daß Sie erst einmal sämtliche Dateien auf eine Systemdiskette kopieren müssen. Das ist zwar selbstverständlich, sollte aber dennoch in der Beschreibung erwähnt werden. Wer sich mit CP/M noch wenig befaßt hat, dürfte hier Schwierigkeiten haben.
Mit welchem Befehl XBC gestartet wird, verschweigt das Handbuch dezent. Dafür beginnt es mit einer Liste der möglichen Fehlermeldungen. Das mag zwar interessant sein, aber nicht zu diesem Zeitpunkt. Nach einigem Suchen stoßen Sie auf die Information, der Compiler werde mit X gestartet. Ein Blick ins Directory der Diskette schafft Gewißheit: Auch das ist falsch. In einem Nachtrag zur CPC-Version ist schließlich zu erfahren, daß sich auf der Disk zwei Versionen befinden: XC für die 64KByte-Geräte und X6 für den 6128.
Die Befehle für den Compiler und den integrierten Editor sind quer durchs Handbuch verstreut. Die Dokumentation ist jedoch vollständig und enthält auch detaillierte Informationen wie zum Beispiel zur internen Darstellung von Variablen. Daß es auch die Möglichkeit gibt, ein Listing auszudrucken, finden Sie irgendwo gegen Ende des Buches.
Leider hat der Autor ein Stichwortverzeichnis für unnötigen Luxus gehalten; aufgrund des geringen Umfangs von nur fünfzig Seiten ist aber eine Suche im Inhaltsverzeichnis und anschließendes hektisches Blättern gerade noch akzeptabel. Das Ringbuchsystem erweist sich dabei als sehr nützlich. Sie können sich jederzeit die gerade benötigten Blätter auf dem Schreibtisch zurechtlegen.



Ein gutes Handbuch ergänzt die Software

EIN NEUES BASIC

Nachdem Sie den Compiler zum Laufen gebracht haben, wollen Sie wahrscheinlich schnell ein altes BASIC-Programm durchjagen, das Ihnen schon immer viel zu langsam war. Doch so einfach macht es Ihnen XBC nicht. Weder als BASIC- noch als ASCII-Version ließ sich eine bereits vorhandene Datei in den Editor laden. Aber selbst wenn das funktioniert hätte, wäre der Erfolg höchst zweifelhaft gewesen. XBC steht nicht umsonst für „Extended BASIC Compiler": Das Locomotive BASIC der Schneider CPCs ist für ihn das gleiche wie Chinesisch für einen Deutschen.
Die Sprache, die XBC versteht, heißt Extended BASIC und ist eine Eigenentwicklung des Autors. Extended BASIC arbeitet ohne Zeilennummern und verwendet ausschließlich Labels. Außer den bekannten Datentypen existieren noch Byte für Werte bis 255 und Cardinal für positive Ganzzahlen. Besonders störend macht sich bemerkbar, daß XBC eine Variable ohne besondere Kennzeichnung als Cardinal-Wert versteht, während die meisten BASIC-Dialekte den Typ Real, also eine Fließkommazahl, voraussetzen. Die dadurch nötige Umgewöhnung führt anfangs häufig zu Fehlern.

Der Befehl GOSUB wurde zwar implementiert, ist aber eigentlich unnötig, da Unterprogramme ähnlich wie in Pascal durch die Namen ihrer Labels aufgerufen werden können. Als Vorteil gegenüber anderen BASICs sind die zahlreichen Schleifenoperationen zu werten. REPEAT und WHILE gibt es ebenso wie FOR, das beim XBC auch mit DOWNTO funktioniert. Hier wurde eindeutig von Pascal abge-

kupfert, was sich aber keineswegs nachteilig auswirkt.
Einen besonderen Sinn sah der Autor wohl in dem Befehl LOOP, der eine Endlosschleife erzeugt. Ob dies wirklich nützlich ist, sei dahingestellt. Die Möglichkeit, das Programm über den WAIT-Befehl für eine bestimmte Zeitdauer zu unterbrechen, kann auf jeden Fall von Vorteil sein. Mit den herkömmlichen FOR-Schleifen können Sie die Zeitdauer nur grob abschätzen; WAIT erledigt das für Sie auf die Millisekunde genau.
Weitere Features sind etwa die Umwandlung zwischen zwei verschiedenen Datentypen oder die Bestimmung einer maximal zulässigen String-Länge in Array-Definitionen. Arithmetische Funktionen sind in der Regel mehrfach vorhanden, so daß Sie für jeden Datentyp eine eigene Funktion verwenden müssen. Wahrscheinlich versprach man sich dadurch einen Geschwindigkeitszuwachs, wie auch durch die Implementierung spezieller Funktionen für häufig benötigte Rechenvorgänge. So kann zum Beispiel x zum Quadrat mit einer eigenen Funktion SQR berechnet werden. Das kann wiederum zu Verwechslungen mit der SQR-Funktion in anderen BASIC-Versionen führen, wo diese Abkürzung bekanntlich für die Quadratwurzel einer Zahl steht.

MANGELNDE ANPASSUNG AN DEN CPC

Die Anpassung des XBC an den Schneider CPC ist leider nur mangelhaft erfolgt. Die vielfältigen Möglichkeiten für Sound und Textfenster bleiben ungenutzt, von den Grafikroutinen wurden nur einige in den Sprachschatz des Extended BASIC übernommen. Auch die Verwaltung von Dateien läuft völlig anders ab als gewohnt; dafür haben Sie die Möglichkeit, Dateien mit wahlfreiem Zugriff zu erzeugen.
Erfahrene Programmierer können die nicht implementierten Befehle durch die Einbindung von Assembler-Routinen simulieren. Um diese Option jedoch wirklich ausnutzen zu können, sollten Sie außer BASIC wenigstens die Grundzüge von Assembler beherrschen und den Aufbau der CPC-Firmware kennen.

DER EDITOR: ZEILENORIENTIERT, ABER KOMFORTABEL

Wenn Sie schon mit einem völlig neuen BASIC arbeiten müssen, so sollte der zugehörige Editor wenigstens bequem zu bedienen sein. Das Einlesen von einem beliebigen Textverarbeitungsprogramm ist zwar theoretisch möglich; in der Praxis stellen sich Ihnen jedoch diverse Hindernisse in den Weg, die darauf beruhen, daß XBC ein besonderes Format der Texte erwartet. In der Regel lohnt sich der Aufwand einer Dateiumwandlung nicht, so daß Sie auf den integrierten Editor zurückgreifen werden.
Dieser Editor arbeitet zeilenorientiert. Jede Zeile erhält folglich eine Nummer, die jedoch für den Compiler keine Bedeutung hat. Dieser arbeitet, wie bereits erwähnt, ausschließlich mit den von Ihnen vergebenen Labels.
Während des Editierens haben Sie Zugriff auf zahlreiche CP/M-Funktionen. So können Sie sich etwa zwischendurch ein Directory ansehen oder eine Datei umbenennen.
Für einen Zeileneditor besitzt das Programm so angenehme Möglichkeiten wie das Verschieben ganzer Programmblöcke oder das Suchen nach einem bestimmten Ausdruck. Das Umbenennen einer Variablen im ganzen Programm ist ohne großen Aufwand möglich – ein Vorteil des Editor-Compiler-Systems gegenüber einem Interpreter.

COMPILIERTE FILES SELBSTÄNDIG LAUFFÄHIG

Ein einmal erstelltes Programm kann im Speicher oder auf Diskette compiliert werden. Auf dem CPC 6128 haben Sie etwa 31,5 KByte für Text und Compilat zur Verfügung; die 64KByte-Geräte bieten Ihnen nur knapp zehn KByte an. Bei Compilierung auf Diskette entspricht dieser freie Platz gleichzeitig der maximalen Größe Ihres BASIC-Programms.
Ihre Programme werden, wenn Sie die entsprechende Option anwählen, als COM-Version auf Diskette abgelegt. Mit dem Compiler erwerben Sie zugleich das Recht, die von Ihnen mit Hilfe dieses Systems erstellten Programme gewerblich zu nutzen. Allerdings haben sämtliche dieser COM-Files einen kleinen Schönheitsfehler: Ein vor dem eigentlichen Programm erscheinendes Titelbild weist Ihre Kunden darauf hin, daß das Programm mit dem XBC-Compiler erstellt wurde. Das müßte nicht sein und wurde von anderen Firmen auch eleganter gelöst, etwa durch eine versteckte Copyright-Notiz in der COM-Datei.
Die Runtime-Bibliothek von XBC ist ziemlich umfangreich. Ein Textprogramm, das nur aus einer Kommentarzeile und dem Befehl END bestand, nahm auf der Diskette ganze 17 KByte in Anspruch. Wenn man die geringe TPA des Betriebssystems CP/M bedenkt, die auch auf dem 6128 nur 48 KByte beträgt, ist das nicht gerade wenig.

HOHE GESCHWINDIGKEITEN

An Schnelligkeit schlägt XBC das normale Locomotive BASIC jedoch um Längen. Ein Kreis wird von XBC in nur sechs Sekunden auf den Bildschirm gebracht, während der normale Interpreter dazu 11,5 Sekunden benötigt. Einfache Rechenoperationen in einem zweitausend Zahlen großen Array bewältigte XBC in zwei Minuten und zwei Sekunden. Locomotive BASIC war hier nur um zehn Sekunden langsamer, was aber hauptsächlich an der Bildschirmausgabe von immerhin viertausend Zahlen gelegen haben dürfte. Der Berechnungsteil lief in XBC wesentlich schneller ab.
Seine wahre Stärke beweist der Compiler bei den transzendenten Funktionen. Die Aufgabe, zwanzigtausend verschiedene Werte der Funktion sin(x)*cos(x) zu berechnen, wurde in sechs Minuten und 37 Sekunden erledigt. Das normale BASIC benötigte dazu zehn Minuten und 21 Sekunden. Bei arithmetischen Operationen war der Compiler stets um einen Faktor von etwa 1,6 schneller als der Locomotive-Interpreter; beim Setzen einzelner Grafikpunkte betrug dieser Faktor sogar 1,9. Bildschirmausgaben im normalen Textmodus liefen dagegen nicht merklich schneller ab.

XBC: COMPILER FÜR SPEZIELLE ANWENDUNGEN

Die mangelnde Sound- und Grafik-Unterstützung machen XBC ungeeignet für effektreiche Programme wie etwa Spiele. Die Übernahme bereits vorhandener Programme können Sie getrost vergessen, da XBC im Grunde genommen kein BASIC-Compiler ist, sondern ein Compiler-BASIC mit eigener Programmiersprache und Entwicklungsumgebung.
Wenn Sie jedoch mathematische oder technische Probleme auf Ihrem CPC lösen wollen, Ihnen hohe Rechengeschwindigkeiten wichtig sind, und Sie außerdem keine Scheu davor haben, BASIC quasi neu zu erlernen, so ist der XBC-Compiler ein nützliches Hilfsmittel. Der Preis in Höhe von 99 Mark hält sich ebenfalls in erschwinglichem Rahmen.

AE □



Zwei starke Helfer für jeden CPC-User!

Test-Jahrbuch 88

Software-Jahrbuch 88

Alles über und für den CPC

An ausgewählten Kiosken und im Bahnhofs-Buchhandel

