IKS
INFORMATYKA KOMPUTERY SYSTEMY

CENA 120 ZŁ

DODATEK DO „ŻOŁNIERZA WOLNOŚCI" NR 10/1988 ISSN 0860 — 2794

Informatyka w szkole - 16

SF 27

4

prenumerata tylko do 10 listopada

W NUMERZE:

Komputerowa rewolucja

— Wkrótce uzyskamy możliwość rozmowy z komputerami, później zaś zbudujemy człekopodobne komputery z inteligencją nadczłowieka — oświadczył Clive Sinclair. To, co się dzieje w tej dziedzinie, potwierdza tę opinię. Niemal każdy dzień przynosi dowody na błyskawiczny rozwój komputeryzacji.

I tak amerykańska firma „Cray" wprowadziła na rynek urządzenie oznaczone symbolem Y-MP. Ten superkomputer kosztujący 20 mln dolarów może przeprowadzić 2 do 4 mld operacji arytmetycznych w ciągu sekundy! To cudo techniki skonstruował Steven Chen, który nie pracuje już w „Cray'u", a tworzy obecnie w International Business Machines.

Z kolei ów koncern elektroniczny ogłosił niedawno, że jego konstruktorzy dokonali nowego przełomu, budując chip — podstawowy element pamięci komputerowej — zdolny do przyjęcia bitu informacji w ciągu 20 miliardowych części sekundy, czyli trzykrotnie szybszy od chipów stosowanych dotychczas. IBM wprowadza też w swych wyrobach nową architekturę budowy komputerów pozwalającą znacznie zwiększyć zasób informacji gromadzonych w pamięci maszyny oraz ułatwiający dostęp do nich. Taka konstrukcja daje możliwości około 8 tys. razy większe niż dotychczasowe systemy rozszerzonej pamięci, stosowane w komputerach IBM. Umożliwi to użytkownikom dużych komputerów dostęp do prawie 16 bilionów jednostek informacyjnych. Tego rodzaju konstrukcje, o czym są przekonani specjaliści, odegrają decydującą rolę w informatyce lat dziewięćdziesiątych.

Tymczasem także w Japonii trwają intensywne badania nad komputerem nowej generacji. Maszyna ta będzie mogła rozmawiać i myśleć podobnie jak człowiek, czyli podejmować samodzielne logiczne decyzje na podstawie posiadanych danych, a nie jak obecnie pracować tylko zgodnie z wprowadzonymi programami.

Opracowany został już w tym kraju typ komputera optycznego, który jest w stanie uczyć się, jak odróżniać obrazy. Jego budowa oparta jest na elektronicznych obwodach imitujących strukturę neuronów ludzkiego mózgu. W rezultacie ten „neuro-komputer" może sam siebie zaprogramować ucząc się rozpoznawania znaków. Jest to na razie konstrukcja w początkowym stadium rozwoju. Komputer w trakcie uczenia się odtwarza znaki czy kształty w sposób niedokładny, co wynika z niedokładnej pamięci komputera. Jednakże po przeanalizowaniu różnic między znakami „na wejściu" i „na wyjściu" koryguje on swą pamięć i odwzorowuje właściwe znaki, nawet jeśli są one trochę zniekształcone. Takie kojarzenie oznacza, że w przyszłości komputery optyczne będą czytały znaki pisane ręcznie.

W niezwykle szybko rozwijającej się komputeryzacji nie zamierzają próżnować specjaliści RWPG. Rozwiązali oni główne problemy techniczne i technologiczne budowy komputera, który potrafi przeprowadzić do miliarda operacji na sekundę. Obecnie w dalszych badaniach najważniejszą sprawą jest zapewnienie niezawodności środków techniki obliczeniowej i jakości ich obsługi technicznej.

(RAJ)

POMOC — AMSTRAD 6128

Jeżeli chcesz usprawnić proces uruchamiania własnych programów i ustrzec się przed nieprzewidzianymi skutkami jego działania, skorzystaj z POMOCY.

Program umożliwia, w dowolnej chwili, skopiowanie 64 k bajtów pamięci do drugiego bloku tej samej wielkości. Po tej operacji można wprowadzić do testowanego programu dowolne zmiany. Jeżeli uznasz, że są niepoprawne, to możesz odzyskać poprzednią postać programu.

POMOC działa poprawnie tylko na Amstradzie 6128 lub 464/664 z rozszerzoną pamięcią pod warunkiem, że drugi blok 64 k bajtów, nie podłączony aktualnie do szyny adresowej, nie będzie przez program wykorzystywany.

Wprowadź do komputera zamieszczony program. Jeśli nie będzie w nim błędów, to kod maszynowy zostanie automatycznie nagrany na dyskietce.

W celu skopiowania pamięci należy (w dowolnej chwili) wgrać POMOC. BIN i wykonać CALL&BFOO, natomiast aby ją odzyskać — CTRL—SHIFT, CALL&BFOO, 1.

Kod maszynowy został umieszczony od adresu &BFOO, ponieważ obszar &BFOO—&BFFF jest stosem i podczas zerowania pamięci przez CTRL—SHIFT—ESC nie jest niszczony.

```
10 'POMOC - AMSTRAD 6128 !
20 'ZACHOWANIE ZAWARTOSCI PAMIECI
30 '
40 adr=&BF00: ln=200
50 FOR i=0 TO 5
60 sum=0: READ a$,d
70 FOR j=1 TO LEN(a$) STEP 2
80 bajt=VAL("&"+MID$(a$,j,2))
90 POKE adr,bajt:sum=sum+bajt
100 adr=adr+1
110 NEXT
120 IF sum<>d THEN PRINT "Zla dana w linii ";ln:END
130 ln=ln+10
140 NEXT
150 SAVE "!POMOC.BIN",b,&BF00,&5E
160 PRINT "Przeslanie pamieci: CALL &BF00"
170 PRINT:PRINT "Odzyskanie pamieci: CALL &BF00,1":PRINT
200 DATA A72006214ABFCD1CBF2154BFCD1CBF3E,&6B9
210 DATA C001007FED79CD11BCC31CBD0605C5E5,&791
220 DATA CD2ABFE12323C110F5C97EE60FC6C001,&866
230 DATA 007FED797EE6F057237EE60FC63F477E,&7F0
240 DATA E6F0670E00596BEDB0C947C146804401,&788
250 DATA C04145C1C54140C1C741864004410000,&621
```

Rafał FOLTYNIEWICZ

Konkurs rozstrzygnięty

Do ogłoszonego na łamach „Żołnierza Wolności" i „IKS-a" konkursu na scenariusz programu komputerowego, którego tematem była historia i tradycje ludowego Wojska Polskiego zgłoszono kilkadziesiąt prac. Zadanie, jak się okazało było trudne, wymagało bowiem nie tylko inwencji twórczej, ale także umiejętności nadania scenariuszowi takiej formy, która pozwoliłaby na jego podstawie napisać program komputerowy.

Jury nie miało łatwego zadania. Spośród ocenianych prac za najlepszy uznano scenariusz zatytułowany „Ucieczka Orła". Autorem tej pracy i zarazem zdobywcą pierwszego miejsca jest płk dr inż. Andrzej Grzesiak.

Drugie miejsce zajął zespół programistów z Wyższej Oficerskiej Szkoły Wojsk Pancernych w Poznaniu w składzie: Danuta Kwasiżur, kpt. Mieczysław Skonieczny i Robert Zugehort, za zbiór scenariuszy i pakiet programów przeznaczony na mikrokomputer Amstrad CPC 6128.

Redaktor naczelny „Żołnierza Wolności" płk Zdzisław Janoś wręcza nagrodę (mikrokomputer TIMEX) zwycięzcy.

INFOGRYF '88

W zasadzie zrezygnowałem z planowanego wyjazdu na Infogryf'88 — koszt uczestnictwa w tej konferencji okazał się bowiem zbyt duży. Szczęśliwie dyrektor okazał się jednak hojny i opłata w wysokości około 55 tys. zł została dokonana.

Tradycyjnie już otwarcia Infogryfu dokonał prof. dr hab. **Tadeusz Wierzbicki**. Stwierdził, iż jest to dziewiąte spotkanie informatyków na tego rodzaju imprezie. Liczba uczestników wynosiła około 300 osób (stosunkowo mało — cena!) w tym goście z Bułgarii, Francji, NRD, RFN i ZSRR.

Na konferencję nadesłano 115 referatów, które zostały opublikowane w czterech tomach. Każda sekcja to oddzielny tom. Są tam referaty „perełki" (mało) i referaty „nie perełki" (dużo), w sumie zamiast zadrukowania 239+297+269+215=1020 stron można by „całkiem spokojnie" zużyć 100 może 105 kartek (np.: streszczając część materiałów). Gwoli ścisłości należy wymienić wyróżnione sekcje/tytuły tomów:

— sekcja pierwsza: Informatyka w zarządzaniu przedsiębiorstwem (nowe generacje systemów), Systemy Informatyczne zarządzania (struktury — funkcjonowanie — rozwój),

— sekcja druga: Informatyka w zarządzaniu przedsiębiorstwem (nowe generacje systemów), Systemy wspomagania decyzji (istota, zasady, możliwości zastosowań),

— sekcja trzecia: Informatyka w zarządzaniu przedsiębiorstwem (nowe generacje systemów), Metody tworzenia i rozwoju systemów (stan i perspektywy rozwoju — praktyczne zastosowanie),

— sekcja czwarta: Mikroskala'88 (wersje i kontrowersje zastosowań mikrokomputerów), Informatyka w zarządzaniu przedsiębiorstwem (nowe generacje systemów).

Prawda, że z tych tytułów „emanuje" precyzja informatyczna?

Referaty uzyskano z tak zwanego wolnego naboru, to znaczy wynagradzając PT Autorów satysfakcją, iż ich materiały zostały przepisane (nie bez błędów) i powielone w 320 plus 3 egzemplarzach (wszystkie!). Jest także tom piąty (stron: 451, czyli łącznie 1471 kartek razy 323), a raczej pierwszy, zawierający (na dużo lepszym papierze) referaty programowe konferencji w następujących językach: angielskim, niemieckim, polskim i rosyjskim (dla informatyków poliglotów?).

Zważywszy jednak na fakt, iż większość, to MY Polacy — tematy poruszane na konferencji, według słów profesora, dotyczyły będą naszych problemów — znad Wisły. Potwierdził te nadzieje (życzenia?), w sposób niezmiernie barwny w swym wystąpieniu, a konkretnie w „publicystycznym wykładzie inauguracyjnym" doc. dr hab. Wojciech Olejniczak — sekretarz naukowy konferencji. Wystąpienie owo było zresztą „awaryjne", planowany bowiem wykład nie mógł się odbyć bo prelegent nie dojechał.

Pan docent „obśmiał" chyba wszystkie przywary naszej informatyki, może nawet trochę więcej, zresztą wstęp wystąpienia wiele wyjaśnia. Oto cytat: „Referat ma charakter publicystyczny; w tym sensie rozmyślnie dokonano w nim przejaskrawień i aby było weselej, dla prezentacji obrazu użyto kilku krzywych zwierciadeł. Innymi słowy w referacie nie zawsze mówi się prawdę (choć nie są to kłamstwa), nie zawsze mówi się wszystko, lecz zawsze mówi się dokładnie to, co się chce powiedzieć", koniec cytatu.

Prelegent mówił jeszcze o pociągach: o pędzącym superekspresie i pociągu osobowym, który stoi przed semaforem..., korzystał także jeszcze z wielu innych metafor. Po tej, w sumie trochę przydługiej inauguracji, szefowie kolejnych sekcji zarządzili, by autorzy referatów (115) zgłosili się do nich i „się zobaczy", kto będzie występował, a kto nie. Pozostawiam tę konkurencję bez komentarza.

Równolegle odbywały się także imprezy towarzyszące, znacząco zwróciły one na siebie uwagę, pomijam oczywiście spotkanie towarzyskie (a 1500 zł), chodzi mi o firmy wystawiające sprzęt, a szczególnie kompetencje fachowców tam pracujących. Nadzwyczaj interesujące było dla mnie spotkanie z dr. Andrzejem Niemcem, reprezentującym ELWRO. Rozmowa z nim (na tematy zastosowań mikro do obliczeń statystycznych) była nieporównywalnie wartościowsza od szeregu referatów. Jaką mamy bowiem korzyść z wysłuchania np.: referatu, który dotyczył kompleksowego komputeryzowania stoczni na bazie sprzętu i oprogramowania rodem z VANGa? Z założeniem, iż stocznia ma dolary — dostatecznie dużo dolarów.

Spotkanie towarzyskie minęło bez ekscesów — jednak informatyka zdominowana jest przez mężczyzn, a szkoda...

O świcie następnego dnia rozpoczęła się Mikroskala równolegle z jeszcze inną sekcją. Zważywszy na nasze („IKS-a") zainteresowania — oczywiście wybrałem Mikroskalę. Na wstępie dzielenia się wrażeniami z tego uczestnictwa pragnę zwrócić uwagę na głośną sprawę sądową, o której opowiedział jeden z organizatorów. Oskarżonym była nasza rodzima mikroinformatyka, po wysłuchaniu obu stron — prokuratora i obrońców, sąd orzekł odroczenie sprawy — oskarżona jest po prostu nieletnia (to wiele tłumaczy).

Jako pierwszy wystąpił pan doktor z Politechniki Warszawskiej, mówiąc o trendach w rozwoju systemów operacyjnych dla mikrokomputerów. Był to w zasadzie wykład o systemach operacyjnych od „początku świata" plus totalna negacja wszelkich krajowych osiągnięć. Prelegent mówił o tysiącach osobolat, które spędzono na budowie systemów operacyjnych dla dużych maszyn typu ODRA 1305, o CPM-ie, DOS-ach (i ich popularności — 10 milionów instalacji), o UNIX-ie i oczywiście o OS2, który obecnie jest w trakcie bu-

dowy. Od trzech lat pracuje nad nim 35 osób. Wydaje się, że PS2 — jest nowym głównym kierunkiem „rynkowym" IBM-a, który między innymi przez „nałożone" patenty pragnie odciąć się od kłopotliwej konkurencji Dalekiego Wschodu. Niemniej za DOS-em opowiada się 50 proc. użytkowników, 25 proc. za OS2, pozostali za mutacjami UNIX-a. Wykładowca wspomniał o problemie komunikacji użytkownika z komputerem (ikony, ekrany dotykowe). W tym miejscu nie mogę sobie podarować uwagi co do znacznej zbieżności przedstawionych treści o zawartości ostatnich BAYT-ów. Pan doktor mówiąc o problemach krajowych nie powiedział dokładnie nic.

Pan Kajkowski jako pierwszy zabrał głos w dyskusji i zadeklarował kompetentną pomoc w sprzężeniu DOS-a z UNIX-em...

Kolejnym problemem był **WIRUS** — czarny strach padł na większość uczestników konferencji, w tym na piszącego te słowa. „Zasady" są podobne do AIDS-a, przekładając je na język użytkowników mikro należy:

— korzystać z zablokowanych (na zapis) dyskietek,
— skopiować (natychmiast!) wszystkie programy i zbiory,
— nowe programy powinny przechodzić kwarantannę na wyróżnionym w firmie komputerze, sprzęt ten trzeba odciąć od sieci — wirus biega także po drutach!

Oto kilka fragmentów głosów w tej kwestii: „czy jest szczepionka", „widziałem w akcji wirusa z Austrii", „północna Polska jest zakażona", „nie ma skutecznej ochrony", „znalazłem wirusa w COM", „porządne firmy zachodnie nie dają wirusa", itp.

Cóż na pewno należy się wystrzegać przypadkowych kontaktów naszego komputera z obcymi... dyskietkami (asceza?). Choroba ta przenosi się natychmiast na inne programy, BIOS-a i wylęga się. Trwa to pewien czas „bezobjawowo", a potem tragedia — twardy trzeba sformatować, bez wcześniejszej możliwości skopiowania czegokolwiek.

Po tych informacjach dostrzegłem za oknem piękne słońce — to był wspaniały spacer brzegiem morza.

Po obiedzie była giełda systemów i kolejne wystąpienia, między innymi, jak wykorzystać mikro do optymalizacji wykrojów (blachy). Temat „dyżurny", bo już w 1972 roku na komputerze ODRA 1013 w ZETO Bydgoszcz został ten problem rozwiązany. Na marginesie — ZETO Bydgoszcz ma doskonale pracującą sieć na bazie R32 i trzech JSG zainstalowanych we Włocławku (do 8 monitorów) w Grudziądzu (do 8 monitorów) i w Bydgoszczy tyle samo. Informacje „biegają" między miastami z szybkością 4800 bodów! Efekty tego systemu mogą ocenić niektórzy klienci ZUS.

Infogryf nabierał rozmachu. Wieczorem o 20.00 (to już maraton) rozpoczęła się giełda firm komputerowych. Czego tam nie było? Komputery z Singapuru, Tajwanu, wielodostęp, galanteria (komputerowa), Junior, 801AT („made in ELWRO" — „kości" z Tajwanu, obudowa nasza, 512 KB, 6 MHz, Hercules, DOS kupiony formalnie i twardy 20 MB — 60 ms!, cena 3,8 miliona zł, oczekiwanie 6 miesięcy), OA Link (bardzo ciekawa sieć — do 6 terminali będących samodzielnymi komputerami, które w postaci kart zainstalowane są w gniazdach płyty głównej centralnego AT), Winchestery 380 MB, plotery, streamery, pamięci taśmowe, 32-bitowe PCty z zegarem 25 MHz.

Była także dyskusja, czasami zabawna, między panami dyrektorami firm komputerowych, a uczestnikami konferencji. Powstał między innymi problem, czy system finansowo-księgowy instaluje się dwa dni czy rok...? Zdania były podzielone, pan prowadzący („po amerykańsku" z ręką w kieszeni) ledwo zapanował nad emocjami dyskutantów.

Dowiedzieliśmy się także, że miesięcznik „Mikroklan„ będący blisko bankructwa wykupiło ELWRO. Redaktorem naczelnym został dyrektor fabryki. Działem programowania kieruje dr A. Niemiec, który zadeklarował tematykę miesięcznika: „powyżej XT", promocja polskich programów, sieci i wielodostęp oraz szeroko pojęta edukacja komputerowa.

Następnego dnia, także o świcie, zważywszy na to, iż Mikroskala dobiegła końca, udałem się do sali IKARa, by posłuchać nowości w zakresie systemów wspomagania decyzji (DSS). „Wysokie loty" (IKAR) prelegentów przypominały mi tragiczną historię Ikara, im wyżej leciał tym słabsze były jego skrzydła... Dużo mówiono o DSS-ach sensu stricto i largo, osiągnięciach Zachodu i kraju za oceanem, różnicach między bazami danych i bazami wiedzy, systemach ekspertowych (eksperckich? doradczych?). Dopiero podczas wystąpienia gościa bułgarskiego powiało nieco praktyką (programy: gromadzące wiedzę ekspertów, tworzące bazy danych i przetwarzające te dane i wiedzę). Ale już za chwilę, po pewnym akcencie futurologicznym, pojawił się temat: „Zastosowanie teorii zbiorów rozmytych do symulacyjnego wspomagania decyzji" — natychmiast się „zmyłem". Słońce pięknie świeciło, mimo to zacząłem spisywać te oto słowa, bo termin składania mija za kilkanaście godzin.

Kończąc te na gorąco spisane wrażenia z INFOGRYFU'88, wydaje mi się konieczne podkreślenie faktu, iż większość luminarzy nauki związanych z informatyką nie docenia (wręcz nie dostrzega) fachowości tych na „dole" — autentycznych specjalistów (praktyków) wdrażania komputerów we wszystkich dziedzinach naszego życia codziennego. Trzeba obecnie tworzyć język, nowy język, porozumiewania się wielkiej nauki z praktyką komputerową nad WISŁĄ.

Kołobrzeg 20.10.88

UCZESTNIK

NAUKA ● TECHNIKA

Trzy placówki badawcze w USA poinformowały ostatnio o opracowaniu bardzo zaawansowanych komputerowych systemów rozpoznawania ludzkiej mowy. Mogą one rozróżnić do tysiąca słów wypowiadanych stosunkowo szybko, bez długich pauz między nimi. Również firma IBM jest bliska zakończenia prac nad tego rodzaju systemem o nazwie Tangora. Współpracowałby on z komputerami personalnymi.

Najdoskonalszy spośród trzech wspomnianych systemów opracowany został w Carnegie Mellon University, w Pensylwanii przez Kai-Fu-Lee. System o nazwie Sphinx wykorzystuje w swym działaniu tzw. statystyczny model rozpoznawania i generowania mowy. Dźwięk w tym systemie przedstawiany jest w postaci cyfrowej. Najważniejsze jest jednak to, że jest to system rozpoznający słowa niezależnie od tego, kto je wypowiada. Dotychczasowe natomiast wymagały zaprogramowania na określonego mówcę.

Dokładność rozpoznawania słów wynosi 96 proc. (gdy zdania są proste). Odsetek błędów rośnie do 10,1 proc., gdy zdania są bardziej skomplikowane. Nieco gorszy pod tym względem jest drugi system opracowany w laboratoriach MIT, gdyż odsetek błędów wynosi 13,6 proc.

Trzeci system rozpoznający ok. 1 tys. słów powstał w prywatnym laboratorium Bolt, Deranek and Newman w Cambridge, Massachusetts. Ten system, którego autorem jest John Makhoul określa się jako „adaptujący się do mówcy". Przez 30 minut przygotowuje się on do rozpoznawania określonego głosu.

Mimo tych niewątpliwych osiągnięć nadal systemy rozpoznawania mowy znajdują się w początkowej fazie rozwoju. Specjaliści m.in. z Carnegie Mellon University są zdania, że na urządzenie, które będzie w stanie rozpoznawać spontanicznie wypowiadane zdania, a więc nie będzie stwarzało żadnych ograniczeń, mówiącemu poczekać trzeba będzie co najmniej 10, a może nawet 30 lat.

* * *

Japońskie firmy komputerowe: NEC, Hitachi i Fujitsu opracowały własne konstrukcje superkomputerów, nie ustępujące produkcji światowego potentata w tej dziedzinie — amerykańskiej firmie Cray Research Corp.

Obecnie funkcjonuje na świecie ok. 260 superkomputerów. Najszybsze z nich potrafią wykonać ponad miliard działań w ciągu sekundy, zaś każdy z nich może zastąpić przeszło sto tysięcy komputerów osobistych.

Z ostatnich nowości: Cray Research wprowadził niedawno nowy superkomputer, który umożliwia trójwymiarowe modelowanie. Maszyna nazwana Y-MP — kosztująca 20 mln dolarów — wykorzystuje jednocześnie osiem mikroprocesorów.

* * *

Roboty wielkości ... komara, które mogłyby np. przedostawać się naczyniami krwionośnymi do serca, aby usunąć zagrażające życiu ludzkiemu schorzenia tego organu, powstają w laboratoriach amerykańskich w wyniku wytrawiania zaprojektowanych wzorów na płytkach krzemowych, a więc dzięki technologiom stosowanym w mikroelektronice. Tą samą techniką wytwarzane są mikroskopijne mechanizmy, trybiki, turbinki i silniczki. W laboratoriach amerykańskiego koncernu ATT otrzymano już kółka zębate, których ząbki mają zaledwie 15 mikronów szerokości!

Animacja w trybie znakowym

Tomasz MROWIEC, Ludwik PIELA

Jedną z mocniejszych stron 8-bitowych komputerów Atari była zawsze animacja, czyli ruchome rysunki. Podstawą większości animacji jest grafika znakowa, prosty i efektywny sposób tworzenia własnych kształtów poprzez składanie odpowiednio dobranych znaków, którymi mogą być litery, cyfry i znaki interpunkcyjne.

Atari dostarcza także pewnych „specjalnych" znaków, które mogą być używane do kreślenia rysunków. Gdy uda się je pomysłowo połączyć, można otrzymać interesujące kształty (patrz **program nr 13**).

Rysunki w trybie znakowym możemy tworzyć za pomocą instrukcji PRINT i POSITION, w taki sam sposób jak tekst. Grafika znakowa jest bardzo efektywna pod względem zajętości pamięci. Na przykład ekran w trybie o wysokiej rozdzielczości (CGRAPHICS 8) wymaga około 8000 komórek pamięci do przechowania wszystkich liczb reprezentacji bitowej. Używając znaków w zwykłym trybie tekstowym możemy zmieścić ten sam ekran w 960 bajtach pamięci (40 kolumn razy 24 wiersze).

Może wydawać się, że jesteśmy ograniczeni przez dysponowanie tylko dostępnymi w Atari znakami. Możliwe jest jednak tworzenie własnych znaków, dostosowanych do naszych aktualnych potrzeb. Zwykle nie musimy przerabiać całego zestawu znaków. Często kilka zmian wystarczy do utworzenia takiego obrazu, jaki chcemy.

Sposób tworzenia i wyświetlania znaków na ekranie opisaliśmy w drugim odcinku naszego cyklu zamieszczonym w „IKS-ie" nr 3 z 1987 roku. Dlatego postaramy się uniknąć powtarzania ograniczając się tylko do przypomnienia pewnych istotnych informacji.

Dobrze znanym sposobem umieszczania znaków na ekranie są instrukcje PRINT oraz POSITION. Poniżej przedstawiamy dwa mniej znane sposoby. Pierwszym z nich jest umieszczanie (za pomocą POKE) kodów znaków bezpośrednio w obszarze pamięci ekranu. W tym celu najpierw musimy znaleźć adres początku pamięci ekranu. Robimy to następująco:

```
10 PAMEKR = PEEK (88) + 256*PEEK (89)
```

Znając adres początku pamięci ekranu umieścimy w nim literę. W tym celu wprowadźmy następującą linię i RUN dla programu:

```
20 POKE PAMEKR, 33
```

W lewym górnym rogu ekranu pojawi się litera **A.** Kod wewnętrzny umieszczony w odpowiednim miejscu pamięci spowoduje wyświetlenie znaku na ekranie.

Musimy pamiętać, że podczas ustawiania do pamięci używamy kodów wewnętrznych, a nie ATASCII. Obejrzymy teraz cały zestaw znaków komputera. W tym celu wprowadźmy następujące linie i RUN dla programu.

```
 5 GRAPHICS 0
20 POSITION 2, 15:PRINT „Wstawianie" (POKE)"
30 FOR X=0 TO 255
40 POKE PAMEKR+X,X
50 NEXT X
```

Na ekranie otrzymamy wszystkie 128 znaków, a następnie negatywy każdego z nich. W taki sposób możemy umieścić znaki bezpośrednio gdzieś w pamięci.

Innym sposobem umieszczania znaków na ekranie jest użycie instrukcji PLOT w trybie tekstowym.

```
10 GRAPHICS 1
20 COLOR (33+160)
30 PLOT 5,7
```

Powyższe instrukcje umieszczą niebieską literę A w miejscu o współrzędnych 5, 7. Nic nie stoi na przeszkodzie aby następną instrukcją było:

```
40 DRAWTO 15,7
```

Teraz jesteśmy już przygotowani do animacji pewnych znaków. Wystarczy użyć jedną z powyższych technik do zmiany znaków w tym samym miejscu ekranu. Mały program, przedstawiony poniżej, będzie animował litery A B C.

Wprowadźmy NEW, a następnie poniższe linie:

```
10 GRAPHICS 0
20 POSITION 10,10
30 PRINT „A"
40 POSITION 10,10
50 PRINT „B"
60 POSITION 10,10
70 PRINT „C"
80 GOTO 20
```

Odbywa się to dosyć szybko. Szybkością możemy sterować za pomocą pętli opóźniającej. Wykona to prosta pętla FOR—NEXT Dodajmy zatem poniższe linie:

```
 5 OPO=100
35 GOSUB 100
55 GOSUB 100
55 GOSUB 100
100 FOR I-1 TO OPO: NEXT I:RETURN
```

Sterowanie szybkością uzyskamy poprzez zmianę wartości zmiennej OPO w linii 5.

Powyższy efekt, z wykorzystaniem liter ABC, nie jest zbyt interesujący. Aby uzyskać naprawdę ciekawe efekty, konieczne jest dysponowanie znakami o odpowiednich kształtach, które musimy utworzyć sami. Wiadomo że cały zestaw znaków zajmuje 1024 bajty pamięci. Za pomocą instrukcji POKE można zmienić kształt dowolnego znaku. Pewne kłopoty sprawia fakt umieszczenia ich w pamięci ROM, której zawartości nie można zmienić.

Zestaw znaków jest zbiorem wzorów, które definiuje kształt każdego znaku. Standardowy zestaw znaków Atari przechowywany jest w pamięci ROM. W celu utworzenia własnych znaków musimy umieścić nowy ich zestaw gdzieś w pamięci, a następnie poinformować komputer, aby używał nowego zestawu w miejsce starego. Podstawową sprawą jest określenie gdzie umieścimy nowy zestaw znaków.

Jeden ze sposobów rozwiązania powyższego problemu sugerowany jest w Dodatku 12 książki „Mapping the Atari". Specjalny program kopiuje system operacyjny z ROM do RAM i wyłącza ROM. Zatem komputer „widzi" ukryty obszar RAM. Po wykonaniu tego możemy, za pomocą POKE, umieścić własny zestaw znaków w obszarze zajmowanym początkowo przez zestaw międzynarodowy. Zaletą tej techniki jest to, że nowe znaki nie zmniejszają wielkości pamięci dostępnej dla naszego programu. W podobny sposób możemy zmienić kształt dowolnych znaków w standardowym zestawie.

Inny sposób polega na utworzeniu własnych znaków w wybranym obszarze RAM i poinformowanie o tym systemu operacyjnego. Jest to czynność pracochłonna i wykonywana bardzo rzadko, bowiem wymaga zdefiniowania całego zestawu znaków. Łatwiejsze i mniej pracochłonne jest modyfikowanie istniejącego zestawu, po uprzednim przepisaniu go z pamięci ROM do RAM.

W obu wymienionych ostatnio sposobach istotne jest wybranie bezpiecznego miejsca w pamięci dla przechowania zestawu znaków. Zawsze musimy chronić dane poprzez umieszczenie ich w takim miejscu, gdzie nie będą zniszczone. A system operacyjny, podobnie jak BASIC, stale przemieszcza swój kod. Na szczęście istnieje łatwy sposób tworzenia bezpiecznego miejsca.

Granice używanej pamięci sterowane są przez RAMTOP w komórce 106 i MENTOP w komórkach 741 i 742. Komórki te przechowują wartości reprezentujące 256-bajtowe stronice pamięci. Poprzez zmianę ich wartości możemy „przekonać" komputer, że pewien nieużywany obszar pamięci jest już zajęty.

RAMTOP przechowuje najwyższy numer stronicy wolnej pamięci RAM dostępnej do użycia. System operacyjny nie chce używać pamięci powyżej tej stronicy. Poprzez zmniejszenie jej numeru możemy używać RAM między oryginalną granicą i tą nową bez zakłóceń.

Po przesunięciu RAMTOP w dół o 2 KB (8 razy 256) otrzymamy bezpieczny obszar pamięci. Czasami pierwsze 1024 bajty są wciąż niepewne, dlatego będziemy używać drugiego 1 KB dla naszego zestawu znaków.

Komputer wymaga, aby zestaw znaków w trybie graficznym 0 zaczynał się od adresu

podzielnego przez 1024 (dla GR.1 lub 2 podzielny przez 5120. Oznacza to dwie, albo cztery stronice naraz.

Jesteśmy już gotowi do rozpoczęcia animacji. Musimy zmodyfikować trzy znaki. Lecz najpierw musimy przesunąć standardowy zestaw znaków z ROM do RAM. Możemy to zrobić następująco:

```
100 NOWYZ=PEEK (106)
110 NOWYZ=NOWYZ-8:POKE 106,
    NOWYZ
120 GRAPHICS 0
130 CHSET=57344
140 FOR BAJT=0 TO 1023
150 POKE NOWYZ*256+BAJT, PEEK
    (CHSET+BAJT)
160 NEXT BAJT
```

Wykonanie powyższego programu zajmuje około 15 s. Lepiej będzie wykorzystać do tego procedurę w języku maszynowym. Nie musimy wiedzieć jak ona działa, wystarczy umieć ją użyć. To znaczy, jak włączyć ją do programu w BASIC-u oraz jak wywołać i z jakimi parametrami.

```
10 DIM MM $ (41)
20 FOR ZNAK=1 TO 41
30 READ KOD
40 MM $ (ZNAK, ZNAK)=CHR$ (KOD)
50 NEXT ZNAK
60 DATA 104, 104, 133, 207, 104, 133,
    206, 104, 133, 209, 104,
70 DATA 133, 208, 104, 170, 160, 255,
    138, 208, 2, 104, 168, 177
80 DATA 206, 145, 208, 136, 192,
    255, 208, 247, 230, 207, 230
90 DATA 206, 202, 224, 255, 208, 233,
    96
130 CHSET=57344
140 X=USR (ADR) MM$, (CHSET,
    NOWYZ*256, 1024)
```

Teraz zestaw znaków Atari jest już w chronionym obszarze RAM. Możemy to sprawdzić zmieniając wartość CHBAS:

```
POKE 756,NOWYZ
```

Następnie należy podjąć decyzję, którym znakom chcemy nadać nowe kształty. Nie mamy potrzeby używania którejkolwiek z liter, dlatego podmienimy znaki CONTROL-A do CONTROL-C. Normalnie są to znaki graficzne Atari, ich kody ATASCII mają odpowiednio wartości od 1 do 3.

Jak już wspomnieliśmy, zestaw znaków ma inną kolejność niż ATASCII. W podręczniku języka BASIC możemy sprawdzić, że znaki te mają kody wewnętrzne 65 do 67.

Poniżej prezentujemy trzy różne kształty robota, a także linie programu, które to realizują:

```
###      =  00111000  =   56
# ###    =  01011100  =   92
########  =  11111110  =  254
 #####   =  01111100  =  124
 #   #   =  01000100  =   68
 #  # #   =  01001010  =   74
 #        =  01000000  =   64
# #       =  10100000  =  160

  ###     =  00111000  =   56
 ## ##    =  01101100  =  108
########  =  11111110  =  254
 #####   =  01111100  =  124
 #   #   =  01000100  =   68
 #   #   =  01000100  =   68
 #   #   =  01000100  =   68
# # # #   =  10101010  =  170

  ###     =  00111000  =   56
 #####   =  01111100  =  124
########  =  11111110  =  254
 #####   =  01111100  =  124
 #   #   =  01000100  =   68
#  #  #   =  10010010  =  146
#  #  #   =  10010010  =  146
#  #  #   =  10010010  =  146
```

```
170 ZES=NOWYZ*256

180 FOR ZNAK=65 TO 67

190 FOR BAJT=0 TO 7

200 READ DANE

210 POKE ZES+ZNAK*8+BAJT, DANE

220 NEXT BAJT

230 NEXT ZNAK

1000 DATA 56, 92, 254, 124, 68, 74, 64, 160

1010 DATA 56, 108, 254, 124, 68, 68, 68, 170

1020 DATA 56, 116, 254, 124, 68, 164, 4, 10
```

Nasz zestaw znaków jest już gotowy do działania, poinformujemy komputer, gdzie on jest:

```
240 POKE 756, NOWYZ
```

Wprowadźmy teraz kod poruszający robota:

```
250 POKE 752,1
260 POSITION 10,4
270 PRINT CHR$(4)
280 RESTORE 1100
290 FOR LP=1 TO 3
300 READ ZNAK
310 POSITION 10,2
320 PRINT CHR$ (ZNAK)
330 FOR CZEK=1 TO 2
340 NEXT CZEK
350 NEXT LP
360 GOTO 270
1100 DATA 1,2,3
```

Zauważmy, że animacja robota polega na wyprowadzaniu różnych wersji, jednej po drugiej, w tym samym miejscu ekranu, z małym opóźnieniem między nimi, w celu umożliwienia obejrzenia każdego z obrazów. Podobna technika używana jest przez część programów do realizacji prostej animacji.

Jeśli chcemy wypróbować zestaw znaków w trybach GR.1 i 2, dokonujemy następujących zmian w programie:

```
120 GRAPHICS 1:REM LUB 2
240 POKE 756,NOWYZ+2
270 PRINT#6 CHR $ (4)
320 PRINT #6 CHR $ (ZNAK)
```

Serca możemy usunąć z ekranu poprzez przedefiniowanie znaku serca na znak spacji. Jeśli potrzebne nam są wielkie i małe litery, możemy to uzyskać poprzez przedefiniowanie zestawu znaków tak, aby znaki graficzne stały się wielkimi literami. W celu szybkiego dokonania modyfikacji możemy użyć procedury przechowywania w zmiennej MMs. Następująca instrukcja przesunie wielkie litery w miejsce znaków graficznych w pamięci.

```
X=USR (ADR (MM$), 57377, NOWYZ *
256+65,26)
```

Teraz naciśnięcie CTRL-A da nam wielkie A.

Jeśli chcemy szybko przełączyć kilka zestawów znaków możemy wykorzystać jedną z poniższych technik:

1. Możemy wprowadzić kilka kształtów znaków do RAM i przełączać wskaźnik początku zestawu na odpowiedni krój. Jest to nieefektywne, ponieważ musimy przechowywać w pamięci wiele zestawów znaków.

2. Możemy wykorzystać procedurę MMs. Przesunie ona nasz zestaw znaków ze zmiennej tekstowej do RAM, a wskaźnik początku zestawu pozostaje niezmieniony. Potrzebny jest tylko jeden obszar dla wszystkich danych.

Po tej części opisowej wprowadźmy do komputera **program nr 2** i połączmy go z programem nr 1. **Program nr 3** przygotuje nam linie 110 i 150, które również należy dołączyć.

Program wynikowy wykonuje animację poprzez wyprowadzanie zmodyfikowanych znaków w tym samym miejscu ekranu.

Otrzymana sylwetka jest bardzo mała i wykonuje ruchy w miejscu. Można ją powiększyć poprzez użycie do kreślenia kilku znaków. W tym celu wprowadźmy **program nr 4** i przechowajmy go w pamięci zewnętrznej. **Program nr 5**, po wprowadzeniu i uruchomieniu, przygotuje nam linie 2110—2120, które należy dołączyć (za pomocą ENTER) do programu nr 4.

Program ten dodaje pewne nowe linie do poprzedniego, w celu powiększenia sylwetki robota, i usuwa inne. Dlatego wprowadźmy wszystkie instrukcje REM, które usuwają stare linie. Połączmy go z programem poprzednim.

W programie nr 4 występują dwa interesujące chwyty. Pierwszym jest przechowanie kształtu znaków w zmiennej tekstowej. Pozwala to zaoszczędzić pamięć, ponieważ każda liczba (jak np.: 240) przechowywana wcześniej w instrukcji DATA, teraz jest zapamiętana w jednym bajcie. Drugi chwyt, w liniach 200 i 210, powoduje szybkie przesłanie nowych kształtów do nowego zestawu znaków.

Prostym sposobem sprawdzenia poprawności utworzonych znaków jest zatrzymanie komputera po dokonanu przez niego zmiany znaków. Teraz wystarczy nacisnąć odpowiednie klawisze, aby zobaczyć na ekranie nowe kształty. W naszym wypadku naciśnijmy następujące klawisze:

abc def ghi jkl mno pqr

Powinno to wyświetlić różne części nowo utworzonych kształtów. W ten sposób wygodnej jest obejrzeć nowe znaki niż podczas działania programu (są one nieruchome).

Jak widać z powyższego spisu, tworzenie własnych znaków nie jest zadaniem trudnym, lecz jakże pracochłonnym. Dlatego w celu ułatwienia pracy proponujemy wykorzystać prezentowany edytor znaków. Oprócz normalnych własności posiada on dwie dodatkowe: animacje wybranego ciągu znaków oraz łączenie czterech znaków w jedną większą jednostkę.

W celu uzyskania własnej kopii edytora należy wprowadzić **program nr 6** i przechować go w pamięci zewnętrznej. **Program nr 7** przygotowuje pewne linie dla programu nr 6, które ciężko byłoby wprowadzić z wydruku. Dlatego po wprowadzeniu programu nr 7 należy uruchomić go i otrzymane w pamięci zewnętrznej linie dołączyć do programu nr 6.

Po uruchomieniu programu wynikowego zobaczymy ekran podzielony na pięć podstawowych obszarów roboczych.

1. Siatka edycji — każdy z 64 kwadratów w siatce reprezentuje bit — każdy poziomy wiersz bitów reprezentuje bajt. Jest to podstawowy obszar dla edycji lub animacji znaków

2. Obraz znaku — podczas edycji widzimy, jak będzie wyglądał znak w trzech różnych trybach graficznych. Każdy znak jest w innym kolorze, lecz tylko dla wyróżnienia go.

3. Menu — wykaz podstawowych rozkazów edytora. W celu wybrania funkcji należy wprowadzić literę umieszczoną z lewej strony. Po wybraniu funkcji (nie należy naciskać klawisza (RETURN) tekst podświetlany jest na niebiesko.

4. Obraz zestawu znaków — reprezentacja aktualnych kształtów znaków.

5. Obszar we—wy — służy do wyświetlania komunikatów błędów i innych informacji.

Tryb kursora u góry menu, podświetlany jest na niebiesko, kiedy działamy w siatce edycji. W tym trybie dostępne jest sześć następujących funkcji:

1. Przesuwanie kursora. Do ustawienia kursora używamy klawiszy strzałek. Nie należy naciskać (CONTROL) lub (SHIFT).

2. Kreślenie. Używamy jej do kreślenia i kasowania wewnątrz siatki edycji. Jeśli kursor jest na pustym kwadracie, naciśnięcie (CONTROL) i klawisza strzałki wypełni go. Jeśli kursor jest na wypełnionym kwadracie, użycie (CONTROL) i klawisza strzałki oczyści go. Kursor przesuwany jest w kierunku naciśniętej strzałki.

3. Powrót. Naciśnięcie (H) przesuwa kursor do pozycji początkowej: lewego górnego rogu siatki edycji. Naciśnięcie (CONTROL) (H) przesuwa kursor do prawego dolnego rogu siatki.

4. Przesuwanie siatki. Naciśnięcie (SHIFT) i klawisza strzałki przesuwa znak w siatce edycji o jeden wiersz w kierunku strzałki.

5. Wartość bajtu. Aby zobaczyć wartość bajtu każdego wiersza, należy nacisnąć (N). Dla wyłączenia tych wartości naciskamy (CONTROL) (N).

6. Przełączanie kursora. Naciśnięcie klawisza (SPACEBAR) wyłącza kursor. Naciśnięcie klawisza strzałki — włącza.

Funkcje wprowaź (L) i przechowaj () służą do przemieszczania zestawów znaków z do pamięci zewnętrznej. Po wybraniu ich musimy podać nazwę zbioru lub C:.

W celu pobrania znaku do siatki edycji należy nacisnąć (G) i (RETURN). Klawisze strzałek ustawiają kursor na znaku, który chcemy poprawiać.

W celu przeniesienia znaku z siatki edycji i umieszczenia go w zestawie znaków należy nacisnąć klawisz (P). Jest to funkcja odwrotna do pobrania. Można ją wykorzystać do tworzenia wielokrotnych kopii.

Naciśnięcie (C) czyści siatkę edycji. Kasuje tylko wybrany znak. Nie zmienia innych znaków.

Funkcja (I) inicjuje działanie edytora. Odtwarza wszystkie znaki w ich oryginalnych kształtach.

Naciśnięcie (A) umieszcza kolejno znaki zarówno w siatce edycji, jak i obok niej. Przedtem musimy podać znaki, które będą animowane.

Składany znak wygląda podobnie jak duży znak, lecz jest aktualnie składany z kilku standardowych. Naciśnięcie (B) powoduje przejście do budowy złożonego znaku i wymaga wprowadzenia czterech znaków. Pojawią się one w obszarze wyświetlania w trybie GR. 2.

Naciśnięcie () kończy pracę programu. Upewnia się on dodatkowo, czy na pewno chcemy to zrobić. Jakkolwiek odpowiedź inna niż (T) powoduje powrót do programu.

Animacja w trybach graficznych wymaga nieco innego podejścia i być może będzie przedstawiona w ramach cyklu.

```
FK 1 REM ****************
FJ 2 REM *              *
XA 3 REM * Program nr 1 *
FL 4 REM *              *
FO 5 REM ****************
NL 6 REM
ON 1040 GRAPHICS 0:POKE 752,1
IX 1050 SETCOLOR 2,0,0:SETCOLOR 1
   ,0,10
VY 1060 FOR R=15 TO 22
LQ 1070 POSITION 2,R:? "_";
JI 1080 NEXT R
GI 1090 POSITION 2,14:? "H     "
WO 1100 POSITION 3,13:? "PPPPPP"
HH 1110 POSITION 1,15:? "H"
KV 1120 FOR R=6 TO 13
MG 1130 POSITION 8,R:? "_"
IY 1140 NEXT R
AL 1150 POSITION 1,9:? "RWRRWR"
RU 1160 POSITION 1,10:? "      "
RH 1170 POSITION 1,11:? "ZAMEK"
QF 1180 POSITION 13,0:? "HJ"
DH 1190 FOR R=1 TO 6
LW 1200 POSITION 13-R,R:? "H"
NF 1210 POSITION 14+R,R:? "J"
IU 1220 NEXT R
JQ 1230 FOR R=6 TO 11
KZ 1240 POSITION 19,R:? "_"
JD 1250 NEXT R
MV 1260 POSITION 20,11
BS 1270 ? "_____ ___ ___ _"
OU 1280 POSITION 24,10
LM 1290 ? "___ ___ ___"
SZ 1300 FOR R=10 TO 22
MX 1310 POSITION 33,R:? "_";
IW 1320 NEXT R
FK 1370 POSITION 13,5:? "FMMMG"
RZ 1380 POSITION 13,6:? "V   B"
JT 1390 POSITION 13,7:? "GNNNF"
EE 1400 POSITION 0,23
MD 1410 FOR I=1 TO 39:? "_";:NEXT
   I
OM 1420 RESTORE 1480
EB 1430 FOR GW=1 TO 10
ET 1440 READ C,R:POSITION C,R:? "
   "
SU 1450 NEXT GW
VG 1480 DATA 2,1,3,3,19,0,23,0,29
   ,0,18,2,24,5,26,1,31,2,38,2
```

```
FK 1 REM ****************
FJ 2 REM *              *
XU 3 REM * Program nr 2 *
FL 4 REM *              *
FO 5 REM ****************
NL 6 REM
QM 100 REM
QS 130 REM
IK 140 DIM ZNACZKI$(24)
MV 160 TM=PEEK(106)
MD 170 CB=TM-8:CA=CB*256:NCA=CA+2
   4
PI 180 POKE 106,CB-8
SI 190 X=USR(ADR(MM$),224*256,CA,
   1023)
NY 200 X=USR(ADR(MM$),ADR(ZNACZKI
   $),NCA,24)
PK 2000 POKE 756,CB
QN 2010 C=5:R=22
AS 2020 POSITION C,R
MF 2030 ? #6;"#";:GOSUB 2090
AY 2040 POSITION C,R
MX 2050 ? #6;"$";:GOSUB 2090
BE 2060 POSITION C,R
NP 2070 ? #6;"%";:GOSUB 2090
OY 2080 GOTO 2020
GJ 2090 FOR I=1 TO 50
ES 2100 NEXT I
AF 2110 RETURN
```

```
FK 1 REM ****************
FJ 2 REM *              *
YO 3 REM * Program nr 3 *
FL 4 REM *              *
FO 5 REM ****************
NL 6 REM
PJ 35 REM UTWORZENIE LINII 110-15
   0
KA 45 REM ZMIEN LINIE 70 JESLI PR
   ACUJESZ ZE STACJA DYSKOW
MG 50 DIM FN$(20),TEMP$(20),AR$(9
   3)
HO 60 DPL=PEEK(10592):POKE 10592,
   255
AJ 70 FN$="C:"
YS 80 GRAPHICS 0:? "     ANTIC'S
   GENERIC BASIC LOADER"
CD 90 ? ,"BY CHARLES JACKSON"
PW 100 POKE 10592,DPL:TRAP 170
JO 110 ? :? :? "Tworzenie ";FN$:?
    "...prosze czekac."
LQ 120 RESTORE :READ LN:LM=LN:DIM
   A$(LN):C=1
BK 130 AR$="":READ AR$
XW 140 FOR X=1 TO LEN(AR$) STEP 3
   :POKE 752,255
DT 150 LM=LM-1:POSITION 10,10:? "
   (Odliczanie...T-";INT(LM/10);"
   )    "
UY 160 A$(C,C)=CHR$(VAL(AR$(X,X+2
   ))):C=C+1:NEXT X:GOTO 130
OT 170 IF PEEK(195)=5 THEN ? :? :
   ? CHR$(125);"ZA DUZO LINII DAT
   A !":? "UTWORZENIE ZBIORU NIEM
   OZLIWE!":END
WU 180 IF C<LN+1 THEN ? :? CHR$(1
   25);"ZA MALO LINII DATA!":? "U
   TWORZENIE ZBIORU NIEMOZLIWE!":
   END
AL 200 OPEN #1,8,0,FN$
PP 210 POKE 766,1:? #1;A$;:POKE 7
   66,0
TY 220 CLOSE #1:GRAPHICS 0:? "KON
   IEC"
FK 1000 DATA 104
IT 1010 DATA 049049048032068073
   073207077036040052049041058073
   077036061034104104133207104133
   206104133209
ZF 1020 DATA 104133208104170160255
   138208002104168177206145208136
   192255208247230207230209202224
   255208233096
SZ 1030 DATA 034155049053048032090
   078065067090075073036061034056
   092254124068074064160056108254
   124068068068
MG 1040 DATA 170056116254124068161
   040040103415
```

```
FK 1 REM ****************
FJ 2 REM *              *
ZI 3 REM * Program nr 4 *
FL 4 REM *              *
FO 5 REM ****************
NL 6 REM
NA 99 GOTO 110
BH 100 FOR I=1 TO 50:NEXT I:RETUR
   N
QS 130 REM
IH 140 DIM ZNACZKI$(75):DLY=100
QW 150 REM
QY 160 REM
ET 170 CB=PEEK(740)-4:POKE 106,CB
   -12:CA=CB*256:GRAPHICS 0
RC 180 REM
TU 200 CA=CA+(96*8):FOR SEC=0 TO
   1:GOSUB 2110+10*SEC:X=USR(ADR(
   MM$),ADR(ZNACZKI$),CA,75)
```

```
GF 210 CA=CA+76:NEXT SEC
OV 1040 POKE 752,1
HX 2000 REM
XB 2010 X=5:Y=20
AJ 2020 POKE 756,CB+2
SJ 2030 POSITION X,Y:? #6;"ab":PO
   SITION X,Y+1:? #6;"cd":POSITIO
   N X,Y+2:? #6;"ef":GOSUB DLY
QU 2040 POSITION X,Y:? #6;"gh":PO
   SITION X,Y+1:? #6;"ij":POSITIO
   N X,Y+2:? #6;"kl":SCROLL=SCROL
   L+2:GOSUB DLY
PZ 2050 POSITION X,Y:? #6;"mn":PO
   SITION X,Y+1:? #6;"op":POSITIO
   N X,Y+2:? #6;"qr":SCROLL=SCROL
   L+4:GOSUB DLY
PF 2060 GOTO 2030
IS 2070 REM
IV 2080 REM
IY 2090 REM
HZ 2100 REM
```

```
FK 1 REM ****************
FJ 2 REM *               *
AC 3 REM * Program nr 5 *
FL 4 REM *               *
FO 5 REM ****************
NL 6 REM
GI 35 REM UTWORZENIE LINII 2110-2
   120
KA 45 REM ZMIEN LINIE 70 JESLI PR
   ACUJESZ ZE STACJA DYSKOW
MG 50 DIM FN$(20),TEMP$(20),AR$(9
   3)
HO 60 DPL=PEEK(10592):POKE 10592,
   255
AJ 70 FN$="C:"
YS 80 GRAPHICS 0:? "     ANTIC'S
   GENERIC BASIC LOADER"
CD 90 ? ,"BY CHARLES JACKSON"
PW 100 POKE 10592,DPL:TRAP 170
JO 110 ? :? :? "Tworzenie ";FN$:?
    "...prosze czekac."
LQ 120 RESTORE :READ LN:LM=LN:DIM
    A$(LN):C=1
BK 130 AR$="":READ AR$
XW 140 FOR X=1 TO LEN(AR$) STEP 3
   :POKE 752,255
DT 150 LM=LM-1:POSITION 10,10:? "
   (Odliczanie...T-";INT(LM/10);"
   )  "
UY 160 A$(C,C)=CHR$(VAL(AR$(X,X+2
   ))):C=C+1:NEXT X:GOTO 130
WE 170 IF PEEK(195)=5 THEN ? :? :
   ? CHR$(125);"ZA DUZO LINII DAT
   A !":? "UTWORZENIE ZBIORUNIEMO
   ZLIWE!":END
WU 180 IF C<LN+1 THEN ? :? CHR$(1
   25);"ZA MALO LINII DATA!":? "U
   TWORZENIE ZBIORU NIEMOZLIWE!":
   END
AL 200 OPEN #1,8,0,FN$
PP 210 POKE 765,1:? #1;A$;:POKE 7
   66,0
TY 220 CLOSE #1:GRAPHICS 0:? "KON
   IEC"
GH 1000 DATA 214
IX 1010 DATA 0500490490480320900
   7806506709007507303604004904405
   5054041061034000000000000000
   0000000000003
EX 1020 DATA 0050110110110050030
   0240232212244132232240040090
   1019019017016031016200200136
   20200200008248001
GA 1030 DATA 0030070140280560600
   6012819219222409611206006000701
   1023023023011007008224208168232
   2008208224016
BH 1040 DATA 0090170170170340580
   8206908408508207803215050049
   0500480320900780650670900750730
   36040049044055
LG 1050 DATA 0540410610340160160
   3100320820020023223200824819200
   3003015062120112056024192192192
   192192192240
```

```
ZS 1060 DATA 2400070110230230230
   1007008224208168232008208224
   6009009017016016016031003200
   2002324812000 8
EK 1070 DATA 2481920030070070070
   1501501501522422419212812819219
   2000034058082069084085082078
   32155
```

```
FK 1 REM ****************
FJ 2 REM *               *
AW 3 REM * Program nr 6 *
VJ 4 REM *    cz. 1      *
FM 5 REM *               *
FP 6 REM ****************
NM 7 REM
IC 2110 REM
SC 2120 GOTO 2640
ZY 2130 REM WCZYTANIE CIAGU ZNAKO
   W
ZB 2140 X=1
DX 2150 POSITION 1,22:? D$;M$;:M$
   =""
IA 2160 GET #1,KEY:? CHR$(KEY);
JS 2170 IF KEY=155 THEN 2210
NY 2180 IF KEY<>126 THEN 2200
IT 2190 IF X>1 THEN X=X-1:GOTO 21
   60
VJ 2200 M$(X)=CHR$(KEY):X=X+1:GOT
   O 2160
GU 2210 POKE 702,64:POSITION 1,22
   :? D$;
AK 2220 RETURN
GY 2230 REM POBRANIE NAZWY ZBIORU
    ZNAKOW
DV 2240 IF M$="" THEN M$="Q"
QE 2250 IF M$="Q" THEN POP :GOTO
   3170
YQ 2260 F$="D:":IF M$="C:" THEN F
   $=M$:GOTO 2350
DZ 2265 IF LEN(M$)<3 THEN 2300
ZC 2270 IF M$(2,2)=":" THEN 2290
GA 2280 IF M$(3,3)<>":" THEN 2300
RB 2290 F$=M$:GOTO 2310
TW 2300 F$(3)=M$
MH 2310 FOR I=1 TO LEN(F$)
HP 2320 IF F$(I,I)="." THEN 2350
FF 2330 NEXT I
DD 2340 F$(LEN(F$)+1)=".FNT"
PQ 2350 TRAP 3820:POKE 752,1:? F$
   ;
AY 2360 RETURN
DY 2370 REM MALUJ MATRYCE
GI 2380 POKE 752,1:POSITION 1,3:?
OQ 2390 X=USR(ADR(PX$),ADR(X$),MA
   )
ZG 2400 FOR I=0 TO 7
DN 2410 POSITION 10,I+5:? "   ";C
   HR$(30);CHR$(30);CHR$(30);
GH 2420 IF N=1 THEN ? PEEK(I+CHA)
FH 2430 NEXT I
AU 2440 RETURN
IU 2450 REM
LT 2460 IF CH<32 THEN CHR=CH+64
IY 2470 IF CH>31 THEN CHR=CH-32
SA 2480 IF CH>95 THEN CHR=CH
TE 2490 CHA=CA+CHR*8
QP 2500 X=USR(ADR(MB$),CHA,ADR(X$
   ))
MR 2510 POKE SA+22+4*40,CHR
FU 2520 X=USR(ADR(MM$),CHA,PA+542
   ,7)
EM 2530 IF BLD=0 THEN 2610
DA 2540 X=USR(ADR(MM$),BA(1),PA+6
   90,7)
OV 2550 X=USR(ADR(MM$),BA(3),PA+6
   98,7)
GU 2560 X=USR(ADR(MM$),BA(2),PA+8
   18,7)
HM 2570 X=USR(ADR(MM$),BA(4),PA+8
   26,7)
BI 2580 RETURN
ZS 2590 X=USR(ADR(FM$),PA+640,0,2
   55)
UG 2600 BLD=0:POKE 53249,138
NO 2610 X=USR(ADR(MM$),CHA,PA+693
   ,7)
AS 2620 RETURN
```

```
FK 1 REM ****************
FJ 2 REM *               *
AW 3 REM * Program nr 6 *
VZ 4 REM *    cz. 2      *
FM 5 REM *               *
FP 6 REM ****************
NM 7 REM

ZN 2630 REM DEKLARACJE
WF 2640 DIM B$(4),BA(4),C$(26),D$
   (38)
PZ 2650 DIM F$(20),M$(40),P(26),X
   $(64)
UM 2660 X$(1)=CHR$(127):X$(64)=X$
   :X$(2)=X$
AV 2670 D$(1)=" ":D$(19)=D$(1):D$
   (2)=D$
XO 2680 FOR K=20 TO 38:D$(K,K)=CH
   R$(30):NEXT K
YK 2710 TM=PEEK(106)
MZ 2720 PB=TM-4:PA=PB*256
OZ 2730 CB=PB-4:CA=CB*256
YZ 2740 POKE 106,CB-4
GA 2750 GRAPHICS 0:REM POKE 559,0
FG 2760 POSITION 1,3:?
JX 2770 DL=PEEK(560)+PEEK(561)*25
   6
HZ 2780 SA=PEEK(DL+4)+PEEK(DL+5)*
   256
AN 2790 MA=SA+1+5*40
KD 2800 FOR I=1 TO LEN(C$)
FU 2810 IF I<18 THEN B=PA+920
QL 2820 P(I)=B+4:B=B+4
FP 2830 NEXT I
ZI 2840 GOSUB 5000
FT 2850 OPEN #1,4,0,"K:"
XT 2860 REM GRAFIKA P/M
ZY 2870 POKE 54279,PB:POKE 53277,
   3
OQ 2880 POKE 623,1
AZ 2890 POKE 53256,0:POKE 53257,1
GH 2900 POKE 53258,1:POKE 53259,3
SR 2910 POKE 704,84:POKE 705,38
RP 2920 POKE 706,38:POKE 707,132
SN 2930 X=USR(ADR(FM$),PA+512,0,5
   12)
KF 2940 POKE 53248,150:POKE 53249
   ,138
WL 2950 POKE 53250,146:POKE 53251
   ,170
PV 2960 GOTO 3020
BC 2970 REM INICJOWANIE ZNAKOW
FY 2980 POSITION 1,22:? D$;
MX 2990 ? "INICJACJA (T/N)? ";:GE
   T #1,KEY
EJ 3000 POSITION 1,22:? D$;
UQ 3010 IF KEY<>84 THEN 3160
SI 3020 X=USR(ADR(MM$),224*256,CA
   ,1024)
NQ 3030 FOR I=0 TO 7
RV 3040 POKE CA+I+126*8,255
ZX 3050 POKE CA+I+127*8,84
XR 3060 IF I=0 THEN POKE CA+127*8
   ,0
WW 3070 IF I=7 THEN POKE CA+7+127
   *8,0
FP 3080 NEXT I
FR 3090 C1=15:R1=2:CH=ASC("@")
AB 3100 POKE 756,CB:POKE 82,1
YU 3110 POKE 709,10:POKE 710,0
AJ 3120 POKE 559,46
CQ 3130 REM MALUJ ZNAK
RK 3140 GOSUB 2460:GOSUB 2380:POK
   E 752,0
IA 3150 REM WEJSCIE Z KLAWIATURY
SL 3160 C=1:R=1:CX=1:RX=1
DA 3170 POSITION C,R+4:? CHR$(31)
   ;CHR$(30);
EL 3180 GET #1,KEY:POKE 752,0
KP 3190 REM KASUJ BLAD
IE 3200 IF  NOT E THEN 3250
OS 3210 POKE 752,1
ZA 3220 POSITION 1,22:? D$:E=0
CQ 3230 POSITION C,R+4:? CHR$(31)
   ;CHR$(30);
OM 3240 POKE 752,0
HP 3250 FOR B=1 TO LEN(C$)
FE 3260 IF KEY=ASC(C$(B,B)) THEN
   3300
CR 3270 NEXT B
SB 3280 POSITION 1,22:? CHR$(253)
   ;" BLAD "
```

```
OL 3290 E=1:GOTO 3170
OY 3300 X=USR(ADR(FM$),P(1),0,40)
VQ 3310 X=USR(ADR(FM$),P(B),255,4
   )
YR 3320 ON B GOTO 4580,4590,4600,
   4610
QD 3330 ON B-4 GOTO 4630,4640,465
   0,4660
PR 3340 ON B-8 GOTO 4550,4550,455
   0,4550
CY 3350 ON B-12 GOTO 3430,3440,34
   60,3480
TL 3360 IF B=17 THEN 3500
BL 3370 IF  NOT BLD OR B=26 THEN
   3390
NI 3380 IF B<20 OR B>21 THEN GOSU
   B 2590
HR 3390 ON B-17 GOTO 4240,3560,35
   20,3860
UZ 3400 ON B-21 GOTO 2980,3690,38
   60,3750
DF 3410 IF B=26 THEN 4890


FK 1 REM ****************
FJ 2 REM *                  *
AW 3 REM * Program nr 6 *
WP 4 REM *     cz. 3      *
FM 5 REM *                  *
FP 6 REM ****************
NM 7 REM
HF 3420 REM KURSOR POZ. POCZ.
NK 3430 C=1:R=1:CX=C:RX=R:GOTO 31
   70
SX 3440 C=8:R=8:CX=C:RX=R:GOTO 31
   70
IV 3450 REM
XK 3460 N=1:GOSUB 2400:GOTO 3170
JB 3470 REM
XI 3480 N=0:GOSUB 2400:GOTO 3170
HP 3490 REM WYLACZ KURSOR
PB 3500 POKE 752,1:GOTO 3170
IL 3510 REM
GS 3520 X=USR(ADR(FM$),ADR(X$),12
   7,64)
HA 3530 X=USR(ADR(MS$),ADR(X$),CH
   A)
QU 3540 GOTO 3140
EX 3550 REM BLD
DN 3560 M$="4 ZNAKI? ":GOSUB 2140
BS 3570 B$=M$:IF M$="Q" THEN 3170
DH 3580 IF LEN(B$)=4 THEN 3600
IO 3590 ? CHR$(253):GOTO 3560
OV 3600 X=USR(ADR(FM$),PA+690,0,1
   6)
JM 3610 POKE 53249,130:BLD=1
RF 3620 X=USR(ADR(FM$),PA+818,0,1
   6)
VN 3630 FOR I=4 TO 1 STEP -1
PM 3640 CH=ASC(B$(I,I)):GOSUB 246
   0
FP 3650 BA(I)=CHA
FV 3660 NEXT I
RF 3670 GOTO 3140
DR 3680 REM LADUJ ZNAKI
UZ 3690 M$="LADUJ? "
LS 3700 GOSUB 2140:GOSUB 2240
QX 3710 OPEN #3,4,0,F$:POKE 752,1
AI 3720 X=USR(ADR(CC$),3,7,CA,100
   8)
WC 3730 GOTO 3790
CW 3740 REM ZAPISZ ZNAKI
JG 3750 M$="ZAPISZ? "
MK 3760 GOSUB 2140:GOSUB 2240
TT 3770 OPEN #3,8,0,F$:POKE 752,1
EE 3780 X=USR(ADR(CC$),3,11,CA,10
   24)
PA 3790 CLOSE #3
CN 3800 POSITION 1,22:? D$
MQ 3810 POKE 752,1:GOTO 3140


OD 3820 CLOSE #3:POSITION 1,22:?
   D$;
OW 3830 ? CHR$(253);" ZLA NAZWA Z
   BIORU"
FX 3840 E=1:POKE 752,0:GOTO 3170
YA 3850 REM WEZ/WSTAW ZNAK
MR 3860 IF  NOT BLD THEN 3940
PK 3870 M$="WYBIERZ ":M$(9)=B$
PA 3880 M$(13)="? ":GOSUB 2140
ZD 3890 FOR I=1 TO 4
IJ 3900 IF M$=B$(I,I) THEN 3930
FM 3910 NEXT I
LU 3920 ? CHR$(253):GOTO 3870
ZQ 3930 CH=ASC(M$):GOTO 3140
DA 3940 POSITION C1+20,R1+15:? CH
   R$(31);CHR$(30);
DD 3950 POSITION C1+20,R1+15:? CH
   R$(31);CHR$(30);
IA 3960 GET #1,K
TM 3970 CHA=CA+(C1-1+(R1-1)*18)*8
AA 3980 IF K=81 THEN 3170
FR 3990 IF K=42 THEN C1=C1+1
GH 4000 IF K=43 THEN C1=C1-1
EZ 4010 IF K=45 THEN R1=R1-1
CG 4020 IF K=61 THEN R1=R1+1
FR 4030 IF R1=0 THEN R1=7
EK 4040 IF R1=8 THEN R1=1
IF 4050 IF C1=0 THEN C1=18
KO 4060 IF C1=19 THEN C1=1
HC 4070 IF K<>155 AND K<>82 THEN
   3860
DN 4080 CHR=(CHA-CA)/8
JR 4090 IF CHR<64 THEN CH=CHR+32
QQ 4100 IF CHR>63 THEN CH=CHR-64
ML 4110 IF CHR>95 THEN CH=CHR
ZV 4120 IF K<>82 THEN 4170
LI 4130 RA=(224*256)+(C1-1+(R1-1)
   *18)*8
BZ 4140 X=USR(ADR(MM$),RA,CHA,7)
QM 4150 GOSUB 2500:GOSUB 2380
TF 4160 POKE 752,0:GOTO 3940
OX 4170 IF B<>21 THEN 4200
RH 4180 X=USR(ADR(MB$),CHA,ADR(X$
   ))
RC 4190 GOTO 3140
GM 4200 X=USR(ADR(MS$),ADR(X$),CH
   A)
QG 4210 GOTO 3140



FK 1 REM ****************
FJ 2 REM *                  *
AW 3 REM * Program nr 6 *
XF 4 REM *     cz. 4      *
FM 5 REM *                  *
FP 6 REM ****************
NM 7 REM
ZL 4230 REM ANIMACJA
NS 4240 D=4:N=0:M$="ZNAKI? ":GOSU
   B 2140
HN 4250 IF LEN(M$)<2 THEN 3160
IR 4260 POKE 752,1:? "WOLNO SZYBK
   O KONIEC"
MK 4270 X=USR(ADR(MB$),CA,ADR(X$)
   )
DV 4280 GOSUB 2380
MX 4290 K=PEEK(764):POKE 764,255
TJ 4300 IF K=255 THEN 4420
ZY 4310 IF K<>46 THEN 4350
MF 4320 D=D+4:F=1:IF D>32 THEN D=
   32
QU 4330 POSITION 1,22:? "WOLNO SZ
   YBKO"
RM 4340 GOTO 4420
CS 4350 IF K<>62 THEN 4390
LW 4360 D=D-4:F=1:IF D<1 THEN D=0
UI 4370 POSITION 1,22:? "WOLNO SZ
   YBKO"
RY 4380 GOTO 4420
CS 4390 IF K<>5 THEN 4420
ES 4400 POSITION 1,22:? D$;
QK 4410 GOTO 3140
SF 4420 FOR I=1 TO LEN(M$)
UT 4430 CH=ASC(M$(I,I)):GOSUB 246
   0
OF 4440 X=USR(ADR(PX$),ADR(X$),MA
   )
MT 4450 FOR WAIT=1 TO D:NEXT WAIT
FS 4460 NEXT I
CL 4470 IF F=0 THEN 4530
JF 4480 POSITION 1,22:F=0
PS 4490 IF D=0 THEN ? "WOLNO
     "
TZ 4500 IF D=32 THEN ? "      SZY
   BKO"
FH 4510 IF D=0 OR D=32 THEN 4530
WP 4520 ? "WOLNO SZYBKO"
UC 4530 GOTO 4290
NM 4540 REM PRZESUNIECIE
JN 4550 POKE 752,1:X=USR(ADR(SF$)
   ,B,CHA)
NN 4560 GOSUB 2500:GOSUB 2380:GOT
   O 3170
FW 4570 REM RUCH KURSORA
JQ 4580 R=R-1:GOTO 4680
JB 4590 R=R+1:GOTO 4680
AS 4600 C=C-1:GOTO 4680
AD 4610 C=C+1:GOTO 4680



IR 4620 REM
RA 4630 R=R-1:W=1:GOTO 4680
QL 4640 R=R+1:W=1:GOTO 4680
JE 4650 C=C-1:W=1:GOTO 4680
ZT 4660 C=C+1:W=1
WE 4670 REM GRANICE RUCHU KURSORA
WY 4680 IF R>8 THEN R=1
YS 4690 IF R<1 THEN R=8
GN 4700 IF C>8 THEN C=1
IH 4710 IF C<1 THEN C=8
WQ 4720 IF  NOT W THEN 4830
JC 4730 REM PISZ  BIT
YL 4740 P=CX+(RX-1)*8:M$=X$(P,P)
CB 4750 IF M$=CHR$(126) THEN M$=C
   HR$(127):GOTO 4770
YV 4760 IF M$=CHR$(127) THEN M$=C
   HR$(126)
JI 4770 X$(P,P)=M$:? CHR$(27);M$;
   CHR$(30);
JL 4780 REM WYSWIETL OBRAZ
QB 4790 POKE 752,1
GY 4800 X=USR(ADR(MS$),ADR(X$),CH
   A)
NA 4810 GOSUB 2510:POKE 752,0
IV 4820 REM
UQ 4830 IF  NOT N THEN 4860
JA 4840 POSITION 10,RX+4
TQ 4850 ? "   ";CHR$(30);CHR$(30)
   ;CHR$(30);PEEK(RX-1+CHA)
XP 4860 CX=C:RX=R:W=0
SX 4870 GOTO 3170
XE 4880 REM KONIEC
TD 4890 IF BLD THEN GOSUB 2590:GO
   TO 3160
FC 4900 POSITION 1,22:? D$;
NE 4910 ? "KONIEC (T/N)? ";:GET #
   1,KEY
WS 4920 IF KEY=84 THEN 4940
GL 4930 POSITION 1,22:? D$:GOTO 3
   160
OZ 4940 POKE 106,TM:GRAPHICS 0
BG 4950 POKE 53248,0:POKE 53249,0
PS 4960 POKE 53250,0:POKE 53251,0
MZ 4970 POKE 82,2
VD 4980 X=USR(ADR("hlJ@"))
GZ 5000 FOR I=0 TO 959:READ A:POK
   E SA+I,A:NEXT I
AG 5010 RETURN


FK 1 REM ****************
FJ 2 REM *                  *
AW 3 REM * Program nr 6 *
XV 4 REM *     cz. 5      *
FM 5 REM *                  *
FP 6 REM ****************
NM 7 REM
TI 19990 REM EKRAN
IH 20000 DATA 81,82,82,82,82,82,8
   2,82,82,82,82,82,82,82,82,82
XE 20010 DATA 82,82,82,82,87,82,8
   2,82,82,82,82,82,87,82,82,82
RO 20020 DATA 82,82,82,82,82,82,8
   2,69,124,128,165,164,185,180,1
   75,178
BU 20030 DATA 128,128,128,128,128
   ,186,174,161,171,175,183,128,1
   24,128,175,162
XN 20040 DATA 178,161,186,128,124
   ,128,128,128,173,165,174,181,1
   28,128,128,124
JA 20050 DATA 90,82,82,82,82,82,8
   2,82,82,82,82,82,82,82,82,82
TR 20060 DATA 82,82,82,82,83,82,8
   2,82,87,82,82,82,83,82,82,82
GW 20070 DATA 82,82,82,82,82,82,8
   2,68,0,0,0,0,0,0,0,0
BA 20080 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,124,0,0,0
QF 20090 DATA 124,0,0,0,124,0,0,4
   3,53,50,51,47,50,0,0,124
```

```
MD 20100 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
LJ 20110 DATA 0,0,0,0,124,0,0,0,1
   24,0,0,0,124,33,0,33
VI 20120 DATA 46,41,45,33,35,42,3
   3,124,0,0,0,0,0,0,0,0
AJ 20130 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,124,0,0,0
IQ 20140 DATA 124,0,0,0,124,34,0,
   34,53,36,47,55,33,0,0,124
MX 20150 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
RM 20160 DATA 0,0,0,0,124,0,0,0,1
   24,0,0,0,124,35,0,43
LW 20170 DATA 33,51,53,42,0,0,0,1
   24,0,0,0,0,0,0,0,0
PE 20180 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,65,82,82,82
VX 20190 DATA 88,82,82,82,68,39,0
   ,55,37,58,0,0,0,0,0,124
MG 20200 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
ZJ 20210 DATA 0,0,0,0,124,0,0,0,0
   ,0,0,0,124,41,0,41
YO 20220 DATA 46,41,35,42,53,42,0
   ,124,0,0,0,0,0,0,0,0
AM 20230 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,124,0,0,0
VR 20240 DATA 0,0,0,0,124,44,0,44
   ,33,36,53,42,0,0,0,124

NA 20250 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
UX 20260 DATA 0,0,0,0,124,0,0,0,0
   ,0,0,0,124,48,0,55
OM 20270 DATA 51,52,33,55,0,0,0,1
   24,0,0,0,0,0,0,0,0
BG 20280 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,124,0,0,0
PU 20290 DATA 0,0,0,0,124,51,0,58
   ,33,48,41,51,58,0,0,124
MJ 20300 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
QK 20310 DATA 0,0,0,0,124,0,0,0,0
   ,0,0,0,124,49,0,43
KE 20320 DATA 47,46,41,37,35,0,0,
   124,0,0,0,0,0,0,0,0
OQ 20330 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,65,82,82,82
YN 20340 DATA 82,82,82,82,88,82,8
   2,82,82,82,82,82,82,82,82,68
ND 20350 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
RU 20360 DATA 0,0,0,0,124,128,128
   ,186,165,179,180,161,183,128,1
   28,186
CW 20370 DATA 174,161,171,175,183
   ,128,128,124,0,0,0,0,0,0,0,0
PK 20380 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,65,82,82,82
TN 20390 DATA 82,82,82,82,82,82,8
   2,82,82,82,82,82,82,82,82,68
MM 20400 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
LW 20410 DATA 0,0,0,0,124,0,1,2,3
   ,4,5,6,7,8,9,10
OE 20420 DATA 11,12,13,14,15,16,1
   7,124,0,0,0,0,0,0,0,0
DP 20430 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,124,18,19,20
XK 20440 DATA 21,22,23,24,25,26,2
   7,28,29,30,31,32,33,34,35,124
NG 20450 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
TY 20460 DATA 0,0,0,0,124,36,37,3
   8,39,40,41,42,43,44,45,46
EP 20470 DATA 47,48,49,50,51,52,5
   3,124,0,0,0,0,0,0,0,0
UF 20480 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,124,54,55,56
WJ 20490 DATA 57,58,59,60,61,62,6
   3,64,65,66,67,68,69,70,71,124
MP 20500 DATA 0,0,0,0,0,0,0,0,0,0
   ,0,0,0,0,0,0
XF 20510 DATA 0,0,0,0,124,72,73,7
   4,75,76,77,78,79,80,81,82
MK 20520 DATA 83,84,85,86,87,88,8
   9,124,81,82,82,82,82,82,82,82
RO 20530 DATA 82,82,82,82,82,82,8
   2,82,82,82,82,82,68,90,91,92
EX 20540 DATA 93,94,95,96,97,98,9
   9,100,101,102,103,104,105,106,
   107,124
QZ 20550 DATA 124,0,0,0,0,0,0,0,0
   0,0,0,0,0,0,0
YE 20560 DATA 0,0,0,0,124,108,109
   ,110,111,112,113,114,115,116,1
   17,118
QQ 20570 DATA 119,120,121,122,123
   ,124,125,124,90,82,82,82,82,82
   ,82,82
VX 20580 DATA 82,82,82,82,82,82,8
   2,82,82,82,82,82,88,82,82,82
RN 20590 DATA 82,82,82,82,82,82,8
   2,82,82,82,82,82,82,82,82,67

FK 1 REM ****************
FJ 2 REM *              *
BQ 3 REM * Prosram nr 7 *
FM 5 REM *              *
FP 6 REM ****************
NM 7 REM
OR 35 REM UTWORZENIE LINII 1170-1
   590 I 2690-2700
KA 45 REM ZMIEN LINIE 70 JESLI PR
   ACUJESZ ZE STACJA DYSKOW
MG 50 DIM FN$(20),TEMP$(20),AR$(9
   3)
HO 60 DPL=PEEK(10592):POKE 10592,
   255
AJ 70 FN$="C:"
YS 80 GRAPHICS 0:? "     ANTIC'S
   GENERIC BASIC LOADER"
CD 90 ? ,"BY CHARLES JACKSON"
PW 100 POKE 10592,DPL:TRAP 170
JO 110 ? :? :? "Tworzenie ";FN$:?
    "...prosze czekac."
LQ 120 RESTORE :READ LN:LM=LN:DIM
   A$(LN):C=1
BK 130 AR$="":READ AR$
XW 140 FOR X=1 TO LEN(AR$) STEP 3
   :POKE 752,255
DT 150 LM=LM-1:POSITION 10,10:? "
   (Odliczanie...T-";INT(LM/10);"
   )    "
UY 160 A$(C,C)=CHR$(VAL(AR$(X,X+2
   ))):C=C+1:NEXT X:GOTO 130
ZB 170 IF PEEK(195)=5 THEN ? :? :
   ? CHR$(253);"ZA DUZO LINII DAT
   A !":? "UTWORZENIE ZBIORUNIEMO
   ZLIWE!":END
ZB 180 IF C<LN+1 THEN ? :? CHR$(2
   53);"ZA MALO LINII DATA!":? "U
   TWORZENIE ZBIORU NIEMOZLIWE!":
   END
AL 200 OPEN #1,8,0,FN$
PP 210 POKE 766,1:? #1;A$;:POKE 7
   66,0
TY 220 CLOSE #1:GRAPHICS 0:? "KON
   IEC"
IK 1000 DATA 1307
PG 1010 DATA 0490490550480320870B
   4036040049041061034104104170224
   4000240025169255141255006138160
   006162255202
HS 1020 DATA 208253136208248034155
   0490490560480320870840360400505
   0052041061034206255006208241170
   202208231104
NX 1030 DATA 170224000240018141255
   0061600061622550341550490490570
   4803208708403604005205404106103
   4202208253
BY 1040 DATA 136208248206255006208
   2410960341550490500490480320680
   7307703207007703604005305204105
   8082069077

OI 1050 DATA 032070073076077069077
   0460790660741550490500500480320
   7070077036040049041061034104104
   133205104133
BV 1060 DATA 204104104133206104170
   2240002400161602551652061452040
   3415504905005104803207007703604
   0050052041
OC 1070 DATA 061034136208251145204
   2302052022082401041681920002400
   1416520613619200024003415504905
   0052048032
MO 1080 DATA 070077036040052054041
   0610340051452041362082511452040
   9603415504905005404803206807307
   7032077077
DT 1090 DATA 036040052057041058082
   0690770320770790860770690770460
   7906607415504905005504803207707
   7036040049
TU 1100 DATA 041061034104104133207
   1041332061041332051041332041041
   7022400024001816025517720603415
   5049050056
NO 1110 DATA 048032077077036040050
   0520410610341452041361922552082
   4723020723020520220823810416817
   7206145204
CT 1120 DATA 136192034155049050057
   0480320770770360400520540410610
   3425520824709603415504905104904
   8032068073
AZ 1130 DATA 077032067067036040051
   0550410580820690770320670730790
   6708407604607906607415504905105
   0048032067
QX 1140 DATA 067036040049041061034
   1041041041701690000241050162022
   0825017010410415706600310415706
   9003104034
HJ 1150 DATA 155049051051048032067
   0670360400500520410610341570680
   0310415707300310415707200307608
   6228034155
FO 1160 DATA 049051053048032068073
   0770320770830360400560500410580
   8082069077032077079086083084082
   046079066074
UE 1170 DATA 155049051054048032077
   0830360400490410610341041041332
   0510413320410413320710413320616
   9000141000
XW 1180 DATA 007169000141001007034
   1550490510550480320770830360400
   5005205204106103416900014100200
   7169128141003
UN 1190 DATA 007162008172000007238
   0000071772042011260341550490510
   5604803207708303604005205404106
   1034208009

ER 1200 DATA 173002007013003007140
   0200707800300702420220822817200
   1007238034155049051057048032070
   83036040
```

```
SH 1210 DATA 0540560410610340010
   0717300200714520617300000720106
   4208198096034155049052049048032068073077032
IB 1220 DATA 0770660360400570490410580820690770320770790860660890840460790660741550490520500480320770360
KR 1230 DATA 0400490410610341041041332051041332041041332071041332061600001691271452062001920642080341550490
QZ 1240 DATA 0520510480320770660360400500520410610342491690001410000071690001410010071720010071772041410020
AX 1250 DATA 0071691281410341550490520520480320770660360400520541061034003007162008172001007173002007056
QS 1260 DATA 2370030070480101410020071720000070341550490520530480320770660360400540560410610342041452061691261452060
CC 1270 DATA 0780030072380000072022082252380010071730000072010642080341550490520540480320770660360400570480
CZ 1280 DATA 0410610342000960341550490520560480320680730770320830700360400490490560410580820690770320830720
AB 1290 DATA 0730700840890460790660741550490520570480320830700360400490410610341041041041701041332051332070
XK 1300 DATA 1041332041332062240092402402401024000922403415504905304804803208307003604005005204106103401100
CE 1310 DATA 2400732240122400500961600071772041332082302061361772041452061360341550490530490480320830700360
VD 1320 DATA 0400520540410610342082491772041452061652081982061452060960160000177204133208230204177034155049
HE 1330 DATA 0530500480320830700360400540560410610342041452062001920072082471652081452060961600001772041332
ZN 1390 DATA 2022400321652040241050081332041440061652051050001332051650341550490530570480320800880360400520
DG 1400 DATA 0540410610342060241050401332061442181652071050001332071442100960341550500540570480320770360610
IZ 1410 DATA 0341041620161690071570660031041570690031041570680031041570730031041570720030760862280341550500
WN 1420 DATA 0550480480320670360610340450610430420280290300310951240920940720080780140320650660670710730760
AE 1430 DATA 0800830810341550
```

Tomasz MROWIEC
Ludwik PIELA

GRA W TYSIĄCA

Janusz JANIEC

Jest to gra waszych ojców. Może brać w niej udział dwóch graczy tzn. ty i twój komputer. Jak w klasycznej grze, tak i tu wykorzystano talię składającą się z 24 kart (od asa do „9"). Obowiązuje następująca hierarchia wartości: as — 11 punktów, „10" — 10, król — 4, dama — 3, walet — 2 oraz „9" — bez punktów.

Każdy z graczy posiada po 5 kart, które rozdaje generator liczb losowych. W związku z tym, że jest to tysiąc dobierany nie ma początkowej licytacji. Rozpoczynając grę wychodzisz pierwszy, natomiast komputer jest pierwszy w następnym rozdaniu. Na ekranie masz zobrazowany aktualny stan kart „na ręku".

Nie obawiaj się, że komputer w momencie podejmowania decyzji będzie podglądał twoje karty, z pewnością tego nie uczyni. Jeśli wychodzący ma parę, czyli damę i króla w tym samym kolorze, powinien to zameldować, a następnie wyjść w damę. Za meldunek gracz otrzymuje: 100 punktów za kier, 80 dla kar, 60 dla trefli oraz 40 dla pików. Gdy skończy się talia, komputer tasuje karty, podaje aktualny wynik, a następnie proponuje dalszą grę. Wygrywa ten, który zdobędzie jako pierwszy 1000 punktów.

Przyporządkowanie klawiszy poszczególnym kolorom oraz honorom kart przedstawia poniższa tabela.

KOLOR	KLAWISZ
KIER	K
TREFL	T
KARO	C
PIK	P
HONOR	**KLAWISZ**
AS	A
10	1
KROL	K
DAMA	D
WALET	W
* 9 *	9
MAM MELD.	M

Program można podzielić na dwa zasadnicze bloki. Pierwszy dotyczy wykonywania poleceń wydawanych przez ciebie i drugi tworzący z komputera gracza. Poszczególne podprogramy są oddzielone komentarzem. Powinno to ułatwić zrozumienie algorytmu programu oraz dociekliwszym umożliwić dokonanie przeróbek.

```
1000 REM ************************
1002 REM *                      *
1004 REM *    GRA W TYSIACA     *
1006 REM *      * Atari *       *
1007 REM *                      *
1008 REM ************************
1050 REM DEKLARACJA ZMIENNYCH
1052 DIM CZ$(15),D$(1),DD$(1),WW$(4),OD$(5)
1055 DIM TA(4,6),HO$(5),KO$(4),WA(6),W$(1),T$(1)
1060 WY=0:SUC=0:SUG=0
1065 RESTORE 1088
1066 FOR B=1 TO 6:READ D:WA(B)=D:NEXT B
1067 WW$="KCTP"
1068 CZ$="                    "
1070 KO$(1,1)=CHR$(0):KO$(2,2)=CHR$(96):KO$(3,3)=CHR$(16):KO$(4,4)=CHR$(123)
1081 DATA *AS* ,A
1082 DATA *10* ,1
1083 DATA KROL ,K
1084 DATA DAMA ,D
1085 DATA WALET,W
1086 DATA * 9 *,9
1088 DATA 11,10,4,3,2,0
1100 REM ZEROWANIE CZESCI DANYCH
1105 GRAPHICS 0:RE=0:KT=0:PO=1
1150 REM TWORZENIE TABLICY
1155 FOR A=1 TO 4:FOR B=1 TO 6
1160 TA(A,B)=0:NEXT B:NEXT A
1200 REM ILE KART W TALII
1204 ? CHR$(125)
1205 LI=0
1206 FOR A=1 TO 4:FOR B=1 TO 6
1210 IF TA(A,B)=0 THEN LI=LI+1
1215 NEXT B:NEXT A
1220 IF LI=0 THEN 1354
1250 REM LOSOWANIE KARTY
1255 Y=INT(RND(0)*4)+1:X=INT(RND(0)*6)+1
1300 REM SPRAWDZANIE CZY BYLA
1301 REM JUZ LOSOWANA KARTA
1305 IF TA(Y,X)<>0 THEN 1255
1307 RE=RE+1:BE=1:GOSUB 2005
1310 IF WY=0 THEN TA(Y,X)=1:GOTO 1352
1315 TA(Y,X)=2
1350 REM POBRANIE DANYCH HONORU
1351 REM KARTY I WYSWIETLENIE NA
1352 REM EKRANIE.

1354 RZ=0:? CHR$(125)
1355 FOR A=1 TO 4:FOR B=1 TO 6
1360 IF TA(A,B)=2 THEN 1385
1365 RESTORE 1080+B
1370 READ HO$
1372 IF LI=0 AND TA(A,B)=1 THEN 2105
1375 IF RE=10 AND TA(A,B)=1 THEN RZ=RZ+1:POSITION 5,4+RZ:? HO$;" ";KO$(A,A)
1385 NEXT B:NEXT A
1388 IF LI>0 AND RE<10 AND WY=1 THEN WY=0:GOTO 1205
1390 IF LI>0 AND RE<10 AND WY=0 THEN WY=1:GOTO 1205
1400 REM CZY KONIEC ROZDANIA
1405 IF RE=0 AND LI=0 THEN 1500
```

```
1410 REM KTO WYCHODZI
1415 IF KT=1 THEN 1805
1420 POSITION 20,5:? "WYCHODZ !
     "
1425 POSITION 20,6:? "----------
"
1430 POSITION 27,8:? " ":POSITIO
N 20,8:? "Kolor ";:INPUT W$:IF W
$<>"M" THEN 1460
1435 FOR A=1 TO 4
1437 IF TA(A,3)=1 AND TA(A,4)=1
THEN 1442
1439 NEXT A
1441 POSITION 11,16:? "Nie oszuk
uj !":FOR CZ=1 TO 300:NEXT CZ:PO
SITION 11,16:? CZ$:GOTO 1430
1442 POSITION 11,16:? "Masz meld
unek":POSITION 11,17:? "nie klam
iesz !"
1444 SUG=SUG+120-20*A:GOTO 1430
1460 POSITION 20,9:? "Honor ";:I
NPUT D$
1465 REM ODNALEZIENIE KOLORU W
1466 REM TABLICY
1470 FOR A=1 TO 4:IF WW$(A,A)=W$
 THEN 1600
1480 NEXT A:GOTO 1625
1500 REM SUMOWANIE WYNIKOW TABLI
CY
1505 FOR A=1 TO 4:FOR B=1 TO 6
1510 IF TA(A,B)=3 THEN SUG=SUG+W
A(B):GOTO 1520
1515 SUC=SUC+WA(B)
1520 NEXT B:NEXT A:REM *********
1530 ? CHR$(125):GRAPHICS 1:POSI
TION 4,6:? #6;"SUMA PUNKTOW"
1535 POSITION 4,7:? #6;"--------
----"
1540 POSITION 4,9:? #6;"** TY **
 ";SUG
1545 POSITION 4,11:? #6;"KOMPUTE
R ";SUC:SETCOLOR 0,8,14
1550 IF (SUC<1000) OR (SUG<1000)
 THEN ? :? "       Czy grasz dale
j [T/N]";:INPUT T$
1552 IF T$="T" THEN 1105
1555 ? :? "          * KONIEC *"
1560 ? CHR$(125):END :REM ******
1600 FOR B=1 TO 6
1605 RESTORE 1080+B
1610 READ HO$,DD$
1615 IF DD$=D$ AND TA(A,B)=1 THE
N 1645
1620 NEXT B
1625 FOR CZY=8 TO 11:POSITION 20
,CZY:? CZ$:NEXT CZY
1630 POSITION 11,16:? "Nie oszuk
uj !":FOR CZ=1 TO 300:NEXT CZ:PO
SITION 11,16:? CZ$
1635 GOTO 1430
1640 REM KOMPUTER SPRAWDZA CZY
1641 REM MA TEN KOLOR I HONOR
1645 Y=A:X=B
1648 REM * CZY MA WYZSZY HONOR *
1650 FOR B=1 TO 6
1655 IF TA(Y,B)=2 AND X>B THEN T
A(Y,B)=4:TA(Y,X)=4:KT=1:GOTO 174
5
1660 NEXT B
1663 REM DAJE NAJNIZSZY HONOR
1665 FOR B=6 TO 1 STEP -1
1670 IF TA(Y,B)=2 AND X<B THEN T
A(Y,B)=3:TA(Y,X)=3:KT=0:GOTO 174
5
1675 NEXT B
1678 REM ZRZUTKA GDY BRAK DANEGO
1679 REM KOLORU
1680 B=6
1685 FOR A=4 TO 1 STEP -1
1690 IF TA(A,B)=2 THEN TA(A,B)=3
:TA(Y,X)=3:KT=0:GOTO 1745
1700 NEXT A
1705 B=B-1
1710 GOTO 1685
1745 RE=RE-2
1750 POSITION 5,12:? "KOMPUTER D
AJE:"
1760 RESTORE 1080+B
1770 READ HO$
1785 POSITION 20,12:? HO$;" ";KO
$(A,A):FOR CZ=1 TO 300:NEXT CZ:G
OSUB 2000
1790 GOTO 1205
1800 REM WYCHODZI KOMPUTER
1805 FOR CZY=5 TO 11:POSITION 20
,CZY:? CZ$:NEXT CZY
1810 POSITION 20,5:? "WYCHODZE !
"
1815 POSITION 20,6:? "----------
"
1820 REM CZY NIE NA MELDUNKU
1825 FOR A=1 TO 4
1830 IF TA(A,3)=2 AND TA(A,4)=2
THEN POSITION 11,16:? "MAM MELDU
NEK !":SUC=SUC+120-20*A:X=4:GOTO
 1843
1835 NEXT A
1837 GOTO 1860
1840 REM WYCHODZI W DAME
1843 HO$="":HO$="DAMA"
1845 POSITION 20,9:? HO$;" ";KO$
(A,A)
1850 GOTO 1912
1855 REM WYCHODZI W NAJSTARSZA K
ARTE
1860 FOR A=1 TO 4:FOR B=1 TO 6
1865 IF TA(A,B)=2 THEN 1875
1870 NEXT B:NEXT A
1875 X=B
1880 FOR B=1 TO 6
1885 RESTORE 1080+B
1890 READ HO$
1895 IF X=B THEN 1905
1900 NEXT B
1905 POSITION 20,8:? HO$;" ";KO$
(A,A):GOSUB 2000
1910 REM GRACZ DOKLADA
1912 Y=A
1915 POSITION 20,10:? "KOLOR ";:
INPUT W$
1920 POSITION 20,11:? "HONOR ";:
INPUT D$
1925 REM ODNALEZIENIE KOLORU W T
ABLICY
1930 FOR A=1 TO 4:IF WW$(A,A)=W$
 THEN 1940
1935 NEXT A:GOTO 1915
1940 FOR B=1 TO 6
1945 RESTORE 1080+B
1950 READ OD$,DD$
1955 IF DD$=D$ AND TA(A,B)=1 THE
N 1980
1960 NEXT B
1965 FOR CZY=10 TO 11:POSITION 2
0,CZY:? CZ$:NEXT CZY
1970 POSITION 20,8:? "Nie oszuku
j !":FOR CZ=1 TO 200:NEXT CZ
1975 POSITION 20,8:? CZ$:GOTO 19
15
1980 REM POROWNANIE KART I ZAKOD
OWANIE WYNIKU
1983 RE=RE-2
1985 IF A=Y AND B<X THEN TA(A,B)
=3:TA(Y,X)=3:KT=0:GOTO 1205
1990 TA(A,B)=4:TA(Y,X)=4:KT=1:GO
TO 1205
2000 REM ROZDAWANIE KART
2005 FOR A2=1 TO BE
2010 FOR B2=2 TO 14
2015 SOUND 0,0,0,B2

2020 NEXT B2
2025 SOUND 0,0,0,0
2035 IF PO=0 THEN 2050
2040 NEXT A2
2045 PO=0
2050 RETURN :REM *********
2100 REM ******************
2105 RZ=RZ+1
2110 POSITION 5,4+RZ:? HO$;" ";K
O$(A,A)
2115 GOTO 1385
```

* SUMA KONTROLNA / ETYKIETA *

```
TE 1000 FW 1002 FK 1004 LV 1006
GQ 1007 UK 1008 IY 1050 UY 1052
GI 1055 JW 1060 SI 1065 EG 1066
WO 1067 JW 1068 RS 1070 HN 1081
YI 1082 AM 1083 EI 1084 EP 1085
BF 1086 CV 1088 EH 1100 ZZ 1105
YI 1150 DG 1155 JR 1160 UL 1200
GP 1204 UE 1205 CX 1206 HY 1210
NZ 1215 HE 1220 TK 1250 QQ 1255
VN 1300 YN 1301 PI 1305 YP 1307
BU 1310 CH 1315 QF 1350 KQ 1351
TI 1352 MP 1354 DK 1355 LV 1360
NI 1365 VZ 1370 OG 1372 YT 1375
OW 1385 TN 1388 SU 1390 QZ 1400
CN 1405 JH 1410 MZ 1415 CK 1420
VN 1425 XJ 1430 VT 1435 QI 1437
DG 1439 ZE 1441 HR 1442 OO 1444
CF 1460 KB 1465 VW 1466 IG 1470
PX 1480 JH 1500 CZ 1505 OV 1510
UN 1515 ZH 1520 TA 1530 PV 1535
WV 1540 PF 1545 LM 1550 LG 1552
DI 1555 ZA 1560 WM 1600 MW 1605
TV 1610 SA 1615 CI 1620 XD 1625
HR 1630 RM 1635 RH 1640 IZ 1641
LN 1645 JG 1648 XB 1650 MP 1655
CU 1660 JH 1663 UU 1665 EQ 1670
DR 1675 YZ 1678 RQ 1679 WI 1680
TS 1685 XL 1690 BT 1700 IJ 1705
WR 1710 XT 1745 SH 1750 MW 1760
WH 1770 OY 1785 RT 1790 BE 1800
VL 1805 TW 1810 VS 1815 GB 1820
VY 1825 TD 1830 CY 1835 VH 1837
TA 1840 YR 1843 TB 1845 TM 1850
WT 1855 DD 1860 QU 1865 OJ 1870
FX 1875 XO 1880 NY 1885 WP 1890
UB 1895 CI 1900 RD 1905 BB 1910
ET 1912 LX 1915 IS 1920 VD 1925
TC 1930 RX 1935 XE 1940 NO 1945
SK 1950 VG 1955 DA 1960 LI 1965
NR 1970 RX 1975 IG 1980 YB 1983
IT 1985 RD 1990 YX 2000 NJ 2005
PK 2010 EU 2015 YZ 2020 HX 2025
GR 2035 YU 2040 XB 2045 WM 2050
TJ 2100 JH 2105 KO 2110 VQ 2115
```

Rady i porady programisty (cz. 2)

Kontynuując rozpoczęty cykl o programach, które powinny znaleźć się w Twojej biblioteczce programów-narzędzi, przedstawiam program przeznaczony do porządkowania (sortowania) elementów zawartych w tablicy.

```
10 ' ***** PORRZADKOWANIE CIAGOW *****
20 INPUT"DLUGOSC CIAGU";n
30 DIM x(n)
40 FOR i=1 TO n
50 INPUT x(i)
60 NEXT i
70 PRINT"WARTOSCI WCZYTANEGO CIAGU"
80 GOSUB 140
90 GOSUB 230
100 PRINT"CIAG UPORZADKOWANY"
110 GOSUB 140
120 STOP
```

Poniższy program jest segmentem sterującym pozwalającym na wprowadzenie liczby n — długości ciągu i wartości do tablicy x, wywołanie podprogramu drukującego.

```
130 '***** PODPROGRAM DRUK *****
140 PRINT
150 i=0
160 WHILE i<>n
170 i=i+1
180 PRINT x(i);
190 WEND
200 PRINT
210 RETURN
```

Podprogram drukujący wywoływany jest dwukrotnie, pierwsze wywołanie drukuje elementy nieuporządkowanej tablicy, drugie zaś wartości tablicy po uporządkowaniu. W segmencie sterującym umieszczone jest wywołanie podprogramu sortującego.

```
220 '****** PODPROGRAM SORTOWANIA ******
230 j=n
240 j=INT(j/2)
250 IF j=0 THEN 380
260 s=n-j
270 FOR r=1 TO s
280 i=r
290 v=i+j
300 IF x(i)<=x(v) THEN 360
310 h=x(i)
320 x(i)=x(v)
330 x(v)=h
340 i=i-j
350 IF i>0 THEN 290
360 NEXT r
370 GOTO 240
380 RETURN
```

Podprogram sortowania umożliwia ustawienie elementów tablicy w porządku rosnącym, jeśli badany warunek w linii 300 ma postać x(i) <= x (v), w porządku malejącym warunek ma postać x(i) > x(v).

Jeżeli w linii 30 zmienimy definicję tablicy ze zmiennej numerycznej na łańcuchową, tj. linia 30 przyjmie postać DIM x$ (n) i w miejsce x wpiszemy x$, będziemy mogli porządkować łańcuchy znaków — np. ułożyć alfabetycznie dowolną listę nazwisk.

Jak ważne jest posiadanie tego podprogramu w twojej bibliotece przekonasz się o tym czytelniku jeszcze nieraz.

Zdzisław ŁUKASZEWICZ

```
/*****************************/
/*                           */
/*        S K A R B          */
/*          ATARI            */
/*                           */
/*****************************/
main()${
char kierunek,z,num[6],pierw[6];
char wydr[17];
int c,a,b,x,y,p,q;
c=iml();  /* inicjal. biblioteki */
do${
 x=0; y=0;
 a=rnd(19)-10;
 b=rnd(19)-10;
 printf("\f");
 do${
  printf("\nKierunek N,S,O,W ?");
  kierunek=getchar();
  getchar();
       if (kierunek=='N') y--;
 else if (kierunek=='S') y++;
 else if (kierunek=='W') x--;
 else if (kierunek=='O') x++;
 else ${ printf("\nUWAGA !");
         for(c=1;c<=30;c++)
         sound(0,100,10,9);
       sound(0,0,0,0);
    printf("\nTylko :N,S,O lub W");
       $}
  q=b-y; p=a-x; z=p*p+q*q;
 c=ifp(z,num);
 c=sqrt(num,pierw);
 c=fasc(pierw,wydr);
  printf("\nDo skarbu %s",wydr);
  for(c=1;c<=30;c++)
  sound(0,100,10,9);
 sound(0,0,0,0);
$}
while(z>0);
printf("\nSkarb : X=%d Y=%d",a,b);
$} while(czydalej());
$}
```

```
PY 590 REM ************************
YS 592 REM *                      *
DE 594 REM *     S K A R B        *
TT 595 REM *     * ATARI *        *
ZH 597 REM *                      *
QW 598 REM ************************
YW 600 DIM K$(2):X=0:Y=0
UJ 602 PRINT CHR$(125)
TK 604 A=INT(RND(0)*19)-10
TV 606 B=INT(RND(0)*19)-10
KZ 608 ? "KIERUNEK N,S,O,W ";:INPUT
    K$
CS 610 IF K$="N" THEN Y=Y+1:GOTO 63
    0
HA 612 IF K$="S" THEN Y=Y-1:GOTO 63
    0
HM 614 IF K$="W" THEN X=X-1:GOTO 63
    0
CG 616 IF K$="O" THEN X=X+1:GOTO 63
    0
TS 618 ? :? "UWAGA !":GOSUB 650
RF 620 GOTO 608
NI 630 P=A-X:Q=B-Y:Z=SQR(P*P+Q*Q)
DO 634 IF Z=0 THEN 640
NC 636 ? "DO SKARBU ";INT(Z*100)/10
    0:GOSUB 650:GOTO 608
BN 640 ? :? "SKARB : X=";A;" Y=";B
UR 642 ? :? "SZUKAMY JESZCZE RAZ ?"
XQ 644 ? :? "              ";CHR$(212);
YC 646 ? "/";CHR$(206);:INPUT K$
GC 648 IF K$=CHR$(84) THEN 602
PE 649 END
CP 650 FOR C=1 TO 15:SOUND 0,100,10
    ,9:NEXT C:SOUND 0,0,0,0
ZS 652 RETURN
```

SCHOWEK

„Schowek" — to procedura pomocnicza, która może się przydać przy pisaniu złożonych programów w standardowym BASIC-u ATARI. Umożliwi ona dostęp do tych obszarów pamięci RAM komputerów ATARI XL i XE, które normalnie są niewidoczne z poziomu BASIC-a, gdyż zasłania je ROM systemu operacyjnego lub sam BASIC. Prawie całą tę przestrzeń można wykorzystać do czasowego przechowywania danych. Miejsca do wykorzystania jest sporo, 22 kB „Schowek" organizuje tę pamięć w umowne sektory po 128 bajtów, numerowane od Ø do 175, dostępne — niezależnie od siebie — do zapisu bądź odczytu. Wymiany danych dokonuje się za pomocą funkcji USR z parametrami.

USR (*adres, numer, bufor, zapis*), gdzie:
adres — adres programu maszynowego,
numer — numer sektora,
bufor — adres bufora danych o pojemności 128 bajtów,
zapis — znacznik kierunku operacji: Ø to odczyt z pamięci, 1 to zapis.

Program maszynowy jest przemieszczalny, można więc umieścić go w zmiennej tekstowej w BASIC-u. Numer sektora winien się zawierać w granicach Ø do 175. Podanie zbyt dużego numeru spowoduje niewykonanie przez program maszynowy jakiejkolwiek operacji, funkcja USR przybiera wtedy wartość równą adresowi wywołania. Gdy numer będzie o wiele za duży (powyżej 255), to może nastąpić zapis lub odczyt z sektora o innym niż podany numerze. Można się o tym dowiedzieć z wartości funkcji USR, gdyż jeśli wykonała ona jakieś operacje na pamięci, to przybiera wartość równą numerowi odpowiedniego sektora. Jako adres bufora danych najwygodniej podać adres zmiennej tekstowej.

„Schowek" warto wykorzystywać tylko wtedy, gdy rzeczywiście brakuje miejsca, np. gdy chcemy mieć w pamięci jednocześnie trzy wykresy w grafice 8, a mieści się tylko jeden, albo gdy w programie typu disasemblera zakładamy jednocześnie duży bufor programu i duży zbiór etykiet itp. „Schowka" możemy także użyć do przekazywania danych między różnymi programami uruchamianymi kolejno, bez wyłączania komputera. W każdym z takich wypadków trzeba poświęcić trochę pracy na szczegółowe rozwiązanie współpracy programu w BASIC-u ze „Schowkiem", stosownie do potrzeb.

Dołączony program w BASIC-u zawiera procedurę maszynową w liniach DATA, oraz ilustruje samą zasadę wykorzystania „Schowka". Drugi program, w składni dostosowanej do Atari Macroassembler'a.

Robert MIZIŃSKI

```
0 REM Program SCHOWEK
1 REM Autor: Robert Mizinski
2 REM
10 REM DEMONSTRACJA DZIALANIA SCHOWKA
20 DIM SCHOWEK$(96)
30 FOR I=1 TO 96:READ A:SCHOWEK$(I)=CHR$(A):NEXT I
40 DATA 104,104,104,201,176,144,5,104,104,104,104,96,162,0,133
41 DATA 212,134,213,74,144,2,162,128,134,216,24,105,160,201,208
42 DATA 144,2,105,7,133,217,104,133,215,104,133,214,104,104,208
43 DATA 16,166,216,165,214,134,214,133,216,166,217,165,215,134,215
44 DATA 133,217,165,20,197,20,240,252,120,173,1,211,73,3,141
45 DATA 1,211,160,127,177,214,145,216,136,16,249,173,1,211,73
46 DATA 3,141,1,211,88,96
50 ? "Demonstracja:":? " tekstowa (1) lub graficzna (2)"
60 ? :? "Wybierz cyfre",:INPUT I
70 IF I=2 THEN 300
80 REM DEMONSTRACJA TEKSTOWA
90 DIM BUFOR$(128)
100 FOR I=1 TO 128:BUFOR$(I,I)="■":NEXT I
105 ? " ZAPIS CZY ODCZYT?":? "(1 - ZAPIS, 0 - ODCZYT)"
110 INPUT ZAPIS:IF  NOT ZAPIS THEN 130
120 ? "WPROWADZ ZNAKI DO BUFORA":INPUT BUFOR$
130 ? "KTORY NUMER SEKTORA ";:INPUT SEKTOR
140 X=USR(ADR(SCHOWEK$),SEKTOR,ADR(BUFOR$),ZAPIS)
150 IF X=ADR(SCHOWEK$) THEN ? "ZLY NUMER":GOTO 100
160 IF ZAPIS THEN ? "ZAPISANO";:GO TO 180
170 ? "ODCZYTANO";
180 ? " SEKTOR ";X
```

```
190 IF  NOT ZAPIS THEN ? "OTO BUFOR:":? :? BUFOR$
200 GOTO 100
290 REM DEMONSTRACJA GRAFICZNA
300 GRAPHICS 8:COLOR 1
310 FOR I=0 TO 159 STEP 2:PLOT 0,I:DRAWTO 319,I:NEXT I
320 ? "Zapis do schowka ekranu 1"
330 ZAPIS=1:SEKTOR=1:GOSUB 500
400 GRAPHICS 8:COLOR 1
410 FOR I=1 TO 319 STEP 2:PLOT I,0:DRAWTO I,159:NEXT I
420 ? "Zapis do schowka ekranu 2"
430 ZAPIS=1:SEKTOR=51:GOSUB 500
440 ? "Odczyt ze schowka ekranu 1"
450 ZAPIS=0:SEKTOR=1:GOSUB 500
460 ? "Odczyt ze schowka ekranu 2"
470 ZAPIS=0:SEKTOR=51:GOSUB 500
480 GOTO 440
490 REM ZAPIS/ODCZYT EKRANU ZE SCHOWKA
500 EKRAN=PEEK(88)+256*PEEK(89)
510 FOR I=0 TO 49
520 K=USR(ADR(SCHOWEK$),SEKTOR,EKRAN+I*128,ZAPIS)
530 SEKTOR=SEKTOR+1
540 NEXT I
550 RETURN
```

```
;Procedura "Schowek"
;Autor: Robert Mizinski
;
;Warunki wejscia:
;Na stosie lezy od gory: ile parametrow,
;numer sektora HI, LO,
;adres bufora danych HI, LO,
;znacznik zapis-odczyt HI, LO.
;
RTCLOK  EQU     $14         ;definicje
PORTB   EQU     $D301       ;etykiet
FR0     EQU     $D4         ;systemowych
ZRODLO  EQU     FR0+2       ;rezerwacja
CEL     EQU     FR0+4       ;strony 0
        ORG     $600
        PLA                 ;ile parametrow
        PLA                 ;HI numeru sektora
        PLA                 ;LO numeru
        CMP     #[$58*2]    ;sprawdz wartosc
        BCC     DOBRZE      ;numeru
ZLE     PLA! PLA! PLA! PLA! RTS
DOBRZE  LDX     #0          ;zwrot numeru
        STA     FR0         ;sektora
        STX     FR0+1       ;do BASIC'a
        LSR     A           ;oblicz LO
        BCC     JESTLO      ;rzeczywistego
        LDX     #$80        ;adresu
JESTLO  STX     CEL         ;schowka
        CLC                 ;oblicz HI adresu
        ADC     #$A0
        CMP     #$D0        ;omin obszar I/O
        BCC     PODD0
        ADC     #7
PODD0   STA     CEL+1
        PLA
        STA     ZRODLO+1
        PLA
        STA     ZRODLO
        PLA                 ;HI znacznika
        PLA                 ;LO znacznika
        BNE     PRZESL      ;idz jesli zapis
        LDX     CEL         ;tu odczyt
        LDA     ZRODLO      ;zamiana
        STX     ZRODLO      ;celu ze zrodlem
        STA     CEL
        LDX     CEL+1
        LDA     ZRODLO+1
        STX     ZRODLO+1
        STA     CEL+1
PRZESL  LDA     RTCLOK      ;czekanie na brak
CZEKAJ  CMP     RTCLOK      ;przerwan
        BEQ     CZEKAJ
        SEI
        LDA     PORTB       ;zamiana ROM(=)RAM
        EOR     #%00000011
        STA     PORTB
        LDY     #$7F        ;licznik
PETLA   LDA     (ZRODLO),Y
        STA     (CEL),Y     ;przeslanie danych
        DEY
        BPL     PETLA
        LDA     PORTB       ;zamiana RAM(=)ROM
        EOR     #%00000011
        STA     PORTB
        CLI
DODOSA  RTS                 ;powrot do BASIC'a
        END     DODOSA      ;lub DOS'a
```

INFORMATYKA

Temat nr 2: REPREZENTACJA INFORMACJI W KOMPUTERZE

1. System binarny, a cyfrowy zapis czarno-białego obrazu.
2. Liczby w systemie binarnym i dziesiętnym.
3. Interpretacja terminów: bit i bajt.
4. Przykłady operacji artymetycznych i logicznych na liczbach zapisanych w systemie binarnym.
5. Interpretacja „kodu znaku" i jego graficznego zapisu. Kod ASCII.
6. Atrybuty pikseli.

Temat nr 3: BUDOWA MIKROKOMPUTERA cz. 1

1. Pamięć RAM. Podstawowe parametry (pojemność, czas cyklu) i przeznaczenie.
2. Pamięć ROM — przeznaczenie.
3. Mikroprocesor, przeznaczenie, parametry.
4. Różnice między procesorami (zegar, liczba bitów).

Propozycja materiałów pomocniczych:

1. Komputery osobiste, D. Madej i inni, Warszawa WKiŁ 1987.
2. Mikrokomputery, A. J. Dirksen, red., Warszawa 1985, WKił.
3. Sami tworzymy symbole graficzne, M. Skonieczny, R. Zugehor, „IKS" nr 6/1988, s. 26.
4. Mikroprocesory 32-bitowe, A. J. Piotrowski, „Komputer" nr 4/87, s. 38

SPRZĘT INFORMATYCZNY cz. 1

(Materiały do podstaw informatyki cz. 2)

Usiłując zrozumieć znaczenie podstawowych parametrów sprzętu informatycznego nie można, niestety, uniknąć tematu: „system dwójkowy (binarny)".

Podczas ostatnich wakacji miałem okazję rozmawiać z Anglikiem. Rozmowa zeszła na temat mieszkań (tu i tam) i ich wielkości. Po krótkiej chwili okazało się, że porównanie wielkości mieszkań stanowi nie lada problem. Przyzwyczajeni jesteśmy bowiem do różnych miar. Dla niego 10 m^2 było tak abstrakcyjnym pojęciem, jak dla mnie 50 stóp kwadratowych. Na szczęście w rozmowie tej pomogła nam 10-letnia córka mego rozmówcy. Zna bowiem (ze szkoły) relacje obu miar.

Wydaje się, że przykład ten ilustruje kłopoty zrozumienia zwrotu „pojemność pamięci komputera" lub uświadomienie różnic między procesorami 8- a 16-bitowymi, osobie która nie zna pojęcia bit lub bajt.

W poprzednim nrze „IKS-a" omówiona została relacja między rzeczywistym obrazem, a jego cyfrową reprezentacją (zero-jedynkową) w pamięci komputera. W ten sposób zostało wskazane, iż zera i jedynki nie są abstrakcyjnymi pojęciami — odzwierciedlają bowiem czarne i białe punkty obrazu komputerowego lub inaczej mozaikę tworzącą wyświetlany na monitorze obraz. Zatem pamięć maszyny przechowuje informacje w postaci zer i jedynek.

Tezy do zajęć z Podstaw Informatyki

(Materiały do Podstaw Informatyki cz. 2)

Oczywiście w pamięci komputera zapisywane są nie tylko obrazy — przede wszystkim liczby, liczby i jeszcze raz liczby. W jaki sposób? Skoro dysponujemy tylko zerami i jedynkami. Możliwe jest to dzięki systemowi dwójkowemu, który różni się od stosowanego przez nas na co dzień tym, że wykorzystywane są w nim właśnie tylko dwa znaki **1** i **0** — stąd nazwa systemu — „dwójkowy", „binarny".

Zatem w komputerze każdą liczbę zapisujemy nie kombinacją dziesięciu cyfr (od 0 do 9), a tylko dwóch — zero i jeden. Zasada binarnego zapisu liczb jest prosta i wygląda następująco:

dziesiętny	System binarny (komputerowy)
0	00000000
1	00000001
2	00000010
3	00000011
4	00000100
5	00000101

itp.

Należy zwrócić uwagę, że każdej cyfrze dziesiętnej odpowiada aż osiem znaków w zapisie maszynowym — ogromne marnotrawstwo! Osiem znaków, by zapisać jeden! Rzeczywiście tak jest. Pamięć komputera zbudowana jest z „szufladek", niezmiennej wielkości „kontenerków", każdy z nich to osiem elementów pamiętających zero lub jedynkę. Zapisując zatem w komputerze liczbę 123 wykorzystujemy aż 24 elementy pamiętające — BITY lub inaczej trzy kontenerki — BAJTY. Czyli zapis owej liczby wygląda następująco:

00000001 (pierwszy bajt)
00000010 (drugi bajt)
00000011 (trzeci bajt)

Oczywiście istnieją inne, bardziej ekonomiczne, sposoby zapisywania liczb, jednak ze względu na ogólny charakter niniejszych rozważań proponuję poszukać ich opisu w dowolnej książce mającej w tytule „komputer" lub „mikrokomputer".

Wydawać by się mogło, że binarny zapis liczb jest nadzwyczaj sztuczny. Jak bowiem dodawać takie wartości, mnożyć, dzielić? Okazuje się, że nie jest to wcale trudne. Na przykład przy dodawaniu należy tylko pamiętać, że: 0+0=0, 1+0=1, 0+1=1 i 1+1=... właśnie, przecież nie ma znaku 2, czyli wynik należy zapisać w postaci „10".
1+2 w komputerze wykonuje się:

```
 0000001
+0000010
---------
 0000011  — to jest właśnie 3.
```

3+3 —
```
 00000011
+00000011
---------
 00000110   czyli 6.
```

System binarny pozwala w sposób bardzo prosty wykonywać wiele ciekawych i użytecznych operacji. Na przykład dzielenie lub mnożenie przez 2, 4, 8, itp., polega na „przesuwaniu" określonej liczby w lewo lub w prawo.

$4_{10} = 00000100_2$ dzieląc przez dwa, przesuwamy w prawo liczbę dwójkową o jedno miejsce — $00000010_2 = 2_{10}$. Podobnie mnożenie przez dwa polega na przesunięciu, ale w lewo, zatem w wyniku otrzymujemy: $000001000_2 = 8_{10}$ (oznaczenia, wskaźniki wyróżniają system w jakim zapisana jest liczba).

Wydaje się, że możemy teraz uznać, iż rozumiemy sposób zapisu czarno-białych obrazów oraz liczb w komputerze. Wiadomo jednak, że korzystając z maszyny, w zasadzie nigdy nie widzimy bitów. Na ekranie monitora komputera oglądamy normalne znaki, co prawda brakuje zazwyczaj wśród nich polskich liter (ń, ś itp.), ale dwójka to „2", a litery są podobne do tworzących te słowa. Jak to jest możliwe?

W tym miejscu bardzo ważne jest poprawne zrozumienie pojęcia „kodu znaku" i odpowiadającego mu „obrazka".

Jeśli idzie o „obrazek znaku", wiemy już, że w pamięci komputera tworzy go mozaika zer i jedynek, a na ekranie odpowiednio ułożone białe i czarne „punkty" (piksele). Powszechnie, a konkretnie w standardzie ASCII („aski" obrazków cyfr, małych i dużych liter, znaków interpunkcyjnych jest aż 96 (128—32 znaki sterujące). Każdy z tych obrazków ma swój kod lub inaczej „liczbową nazwę". Na przykład cyfry od 0 do 9 mają kody od 48 do 57, małym literom od „a" do „z" odpowiadają kody od 97 do 122, dwukropek to 58, itp.

Zatem w jednej części komputera zapisane mamy obrazki znaków. W drugiej natomiast kody znaków tworzące wprowadzane przez nas do komputera słowa. Na przykład w maszynie słowo „IKS" zapisane jest w trzech bajtach (umownych kontenerkach, każdy po 8 bitów!). W pierwszym bajcie mamy kod litery I — 73, w drugim — 75 (K) i w trzecim 83 (S). W postaci binarnej wygląda to następująco:

1001001 (I)
1001011 (K)
1010011 (S)

Przykład ten jeszcze raz zwraca uwagę na to, iż w pamięci komputera zapisywane są tylko liczby! Liczby te — to owe kody lub inaczej „nazwy" znaków, które maszyna potrafi wyświetlić. Natomiast obrazy znaków, a właściwie ich szablony, czcionki, komputer ma

zapisane w specjalnej „trwałej" części pamięci.

Nasuwa się w tym momencie pytanie — a gdyby zmienić tylko szablony? Właśnie tak się robi! Zmieniając pamięć „trwałą" ze starymi wzorcami na nową, np.: gdzie zamiast danych o postaci obrazka znaku „&" (kod — 38) odzwierciedlona będzie, odpowiednio ułożonymi jedynkami, postać naszej polskiej litery „ł". Zatem, gdy komputer „zechce" wyświetlić znak o kodzie 38, „pobierze" ze swej pamięci, odpowiadający temu kodowi obrazek, szablon litery „ł". Czyli na ekranie pojawi się nie znak „&", a litera „ł". W podobny sposób można zmienić postać wszystkich szablonów, tak by komputer „posługiwał się" cyrylicą lub znakami chińskimi.

Kończąc tematykę relacji: obraz na ekranie — jego cyfrowy zapis w pamięci komputera, celowym wydaje się przypomnienie, iż często na ekranie monitora ukazują się migające, a nawet kolorowe obrazki. Uzyskuje się to dzięki odpowiedniej elektronice oraz „mniej oszczędnemu" wykorzystaniu pamięci komputera. Upraszczając problem — każdy punkt obrazu, który dotychczas zapisywany był jako zero lub jedynka (w jednym bajcie mieściło się aż 8 punktów obrazu) zapisywany jest w jednym **bajcie.** Oznacza to, iż każdemu punktowi obrazu można przyporządkować liczbę mieszczącą się w jednym bajcie — od 0 do 256, innymi słowy 257 kolorów — od bieli przez szarości, żółty, niebieski, czerwony i inne, do czerni. Dzieje się to niestety kosztem wykorzystanej do tego celu pamięci. Przy takim sposobie zapisu obrazu potrzebujemy jej osiem razy więcej (nic za darmo!). Dlaczego 256? W każdym bajcie można zapisać 256 różnych kombinacji zer i jedynek.

Wydaje się, że można już uznać, że przebrnęliśmy jeden z najmniej ciekawych etapów naszego wstępu do informatyki. Było to jednak niezbędne, choćby po to, by móc odpowiedzieć na pytanie: co możemy zapisać w pamięci wielkości 48 000 bajtów (pamięć ZX SPECTRUM)? Przyjmując, że kartka maszynopisu ma 30 wierszy po 60 znaków, to w pamięci tej wielkości można zarejestrować... ponad 26 stron. Mało!

Sądzę, że usiłując omówić budowę mikrokomputera (bardzo ogólnie) uzasadnione jest rozpoczęcie od pamięci. Wiemy już bowiem do czego służy (przechowywanie informacji w postaci liczb binarnych), potrafimy także szacować jej wielkość (bajtami). Praktycznie stosuje się jednak nieco większe miary pojemności pamięci:

1 KB (kilobajty) tj. około 1000 bajtów (dokładnie (1024)
1MB (megabajty) — 1 000 000 bajtów
1 GB (gigabajt) — 1 000 000 000 bajtów.

Typowy IBM PC/XT wyposażony jest w pamięć 512 KB (tj. 512 000 bajtów).

Należy podkreślić, iż jest szereg różnych pamięci. Umownie dzielimy je na: zewnętrzne i wewnętrzne. Na początek zajmiemy się pamięcią wewnętrzną — o niej to właśnie dotychczas mówiliśmy. Tworzą je małe elementy elektroniczne, tak zwane układy scalone („kości") wielkości niewielkiej gumki do wycierania. W komputerach klasy IBM PC wykorzystuje się najczęściej kostki o pojemności 64 i 256 Kb (kilobity). Obecnie dostępne są także pojemniejsze kostki — nawet do jednego mega! Mówimy teraz o tak zwanych kostkach RAM. Tworzą one pamięć dostępną dla nas — użytkowników komputera. Możemy w niej zapisywać informacje i odczytywać. RAM zazwyczaj przechowuje aktualnie wykonywany program oraz przetwarzane dane (np. kolumny sumowanych liczb).

Poza pojemnością ważna jest szybkość pamięci, to znaczy czas wyszukania i odczytania (lub zapisania) np. jednego z 500 000 znaków (które wcześniej były zapisane w pamięci). Czas ten (czas cyklu) zazwyczaj wynosi mniej niż 400 ns (ns—nano sekund tj. 10^{-9}s.). Oznacza to, że podczas jednej sekundy można użyć tę pamięć (czytając z niej lub zapisując) aż dwa i pół miliona razy (obecnie dostępne są oczywiście także znacznie szybsze pamięci).

Niestety nie wszystkie cechy RAM są takie „sympatyczne". Wystarczy bowiem, że wyłączą prąd i wszystkie zapisane w niej pracowicie informacje bezpowrotnie giną! Przydała by się zatem taka pamięć, która nie gubi in-

...aż człowiek stworzył komputer

Szybki w latach osiemdziesiątych rozwój techniki komputerowej, powstanie mikrokomputerów mających zdolności obliczeniowe dużych komputerów z lat siedemdziesiątych, spowodowały ogromne zainteresowanie możliwościami coraz doskonalszej techniki obliczeniowej. A jakie były początki urządzeń liczących. Kogo i co możemy dzisiaj uważać za antenatów czy protoplastów współczesnych twórców komputerów i coraz doskonalszych maszyn liczących.

Za przodków komputera uznać należy wszystkie urządzenia służące do liczenia. Zanim jednak zaczęto myśleć o urządzeniach musiano nauczyć się liczyć. Zaczynano od liczenia na palcach. Liczby oznaczano węzełkami, muszelkami czy kamyczkami. Przodkiem komputera było z pewnością liczydło wynalezione 5000 lat temu, w Babilonie i rozpowszechnione następnie w Chinach, Japonii, Egipcie, Grecji i Rzymie. Około 3000 roku p.n.e. powstał pierwszy przyrząd do przeprowadzania obliczeń zwany abakusem. Początkowo były to deski posypane piaskiem, na których zaznaczano kreskami zliczane ilości, a następnie deski z wyżłobionymi rowkami, w których przesuwano przedmioty np. kamyczki. Około 200 roku p.n.e. w starożytnej Grecji opracowano „kalkulator analogowy" używany do obliczeń ruchu gwiazd i planet. W sto lat później **Heron z Aleksandrii** opracował licznik drogowy, zwany drogomierzem Herona, który dokonywał samoczynnego pomiaru długości drogi przebytej przez pojazd.

Pod koniec pierwszego wieku n.e. Arab **Alfazari** zastosował cyfry 0,1,2,3,4,5,6,7,8,9 używane do dnia dzisiejszego. Cyfry są pochodzenia indyjskiego, do Europy Zachodniej zostały przeniesione w X—XII wieku (według niektórych publikacji w 1226 roku). W 960 roku **Gerbert z Francji** udoskonalił **abakus,** co pozwoliło na dokonywanie obliczeń pozycyjnych. Około 1200 roku wprowadzono pierwsze liczydła skonstruowane na bazie stosowanego powszechnie abakusa. W 1464 roku niemiecki matematyk i astronom Regiomontanus wprowadził tablice sinusów dla ułatwienia mnożenia dwu liczb wielocyfrowych. W 1494 roku Włoch **L. Pacioli** opisał system księgowości podwójnej, uważany za początek prowadzenia ewidencji o zjawiskach gospodarczych w sposób kompletny i ciągły. W 1614 roku szkocki matematyk **John Napier** wprowadził pojęcie logarytmu. Opracowane tablice logarytmów uprościły czynności mnożenia, dzielenia, potęgowania i pierwiastkowania. W 1617 roku John Napier zastosował sztabki rachunkowe do obliczania wielokrotności liczb całkowitych (metodą dodawania logarytmów).

Sztabki rachunkowe znane są również jako „kostki Napiera". W 1620 roku **E. Gunter** wynalazł suwak logarytmiczny, mający dwie ruchome skale logarytmiczne do wykonywania mnożenia i dzielenia. Należy tu zaznaczyć, że w niektórych publikacjach Gunterowi przypisuje się wynalezienie skali logarytmicznej, natomiast suwaka logarytmicznego — **Oughtredowi** (w 1621 lub 1624). W 1623 roku profesor języków biblijnych i astronomii w Tybindze **Wilhelm Schickard** wynalazł pierwszą maszynę do dodawania i odejmowania (a więc pierwszy arytmometr), która wykonywała przeniesienie liczb dziesiętnych przez sześć miejsc.

Można było również, za pomocą tabliczki mnożenia, wykonywać operacje mnożenia i dzielenia. Niezależnie od Wilhelma Schickarda w 1645 roku francuski matematyk i filozof **Blaise Pascal** przedstawił skonstruowaną w 1642 roku maszynę do sumowania, opartą na zasadzie obrotowych kół zębatych. Liczby były przedstawione za pomocą wielkości kątowych. Arytmometr Pascala miał możliwość automatycznego przeniesienia dziesiętnego, wykonywania odejmowania przez odwrotny obrót tarczy liczącej oraz mnożenia przez wielokrotne dodawanie. Urządzenie to było maszyną jednookresową, tzn. ustawianie cyfr liczby i otrzymanie wyniku realizowane było w tym samym cyklu pracy. W 1666 roku skonstruował maszynę do sumo-

INFORMATYKA W SZKOLE

formacji także wówczas, gdy wyłączymy zasilanie. Rolę tę pełnią pamięci zewnętrzne (o czym będzie mowa innym razem) oraz tak zwana pamięć ROM. Trwałość zapisu w ROM uzyskana jest niestety kosztem możliwości zapisania w niej czegokolwiek. Oznacza to, że informacje zawarte w niej są niezmienne — to co wpisał do ROM producent można czytać tysiące razy, ale nigdy zmienić. Pamięć Tylko Do Czytania. Potrzebę takiej pamięci uzasadniają „szablony znaków" — raz zapisane w fabryce mogą niezmiennie być używane przez lata. Pozostałe cechy pamięci ROM są podobne do RAM.

Wracając do szybkości RAM, po co tak duża? Kto w pełni wykorzysta tę cechę? Jest jeden jej „współpracownik", w zasadzie „pracodawca" — **mikroprocesor** (procesor) — także układ scalony. Praktycznie to w nim wykonywane są wszystkie operacje związane z przetwarzaniem informacji. Układ ten między innymi dodaje, mnoży, dzieli, porównuje. Czynności te (i wiele innych) wykonywane są bardzo szybko, w takt tak zwanego **zegara**. W popularnym procesorze 8066 wymusza on (zegar) wykonanie do 10 milionów mikrooperacji (10 Mega) w czasie jednej sekundy. Jest to jednak już przeszłość, dzisiaj bowiem coraz częściej słychać o procesorach z zegarem powyżej 20 Mega. Co to praktycznie oznacza? Każdy takt zegara to wykonanie przez procesor jednej mikrooperacji typu: 1. Pobierz z RAM pierwszy argument, 2. Pobierz z RAM drugi argument, 3. Dodaj oba argumenty, 4. Wynik zapisz w RAM. To przykładowe dodawanie dwóch liczb składa się z czterech mikrooperacji. W wypadku, gdy zegar ma szybkość 4 Mega — czas wykonania przez procesor operacji dodawania (zgodnie z podanym przykładem) wyniesie 1/1000000 s. Owe mikrooperacje, to nic innego, jak czynności wykonywane przez nas, gdy korzystamy z kalkulatora — wprowadzanie argumentów i zapisywanie wyniku w zeszycie. Różnica polega głównie na szybkości. Aby wprowadzić do kalkulatora liczbę, jedna sekunda może się okazać zbyt krótkim czasem — komputer wykonuje tę czynność co najmniej milion razy szybciej. Możliwe jest to dzięki temu, iż mikroprocesor sam odczytuje argumenty — stąd potrzeba dużej szybkości RAM, aby procesor się nie „nudził" czekając na podanie kolejnych argumentów.

Argumenty, to oczywiście liczby — jak duże? Zależy to od procesora. Taki, który może operować na ośmiu bitach jednocześnie — na przykład dodawać liczby mieszczące się w jednym bajcie każda, nazywamy **procesorem ośmiobitowym.** Zatem procesor taki w jednej chwili może dodawać liczby nie większe od wartości równej 256 — taką bowiem maksymalną liczbę można zapisać w ośmiu bitach. Gdy chcemy sumować większe liczby, robi się to, „na raty". Na przykład najpierw komputer dodaje jedności, potem dziesiątki, itp. Wpływa to znacznie na szybkość działania komputera. Stąd postęp technologii budowy komputerów widoczny jest w budowie szesnastobitowych procesorów (np.: 80286) — mogą operować one, w każdym takcie, liczbami o wielkości do 32 000. Na przykład operację 31245:13567 komputer 16-bitowy z zegarem 10 Mega wykona w czasie: 1 s/10^7 x-liczba mikrooperacji niezbędnych do wykonania operacji dodawania.

Obecnie coraz częściej słychać o procesorach 32-, a nawet 64-bitowych!

Wszystko jest jasne, prawie wszystko — bo skąd on (procesor) wie, i to tak szybko, gdzie znajdują się argumenty (mogą przecież być ich setki tysięcy) i co z nimi zrobić? Dzielić? Dodawać? Co zrobić z wynikiem? Gdzie go zapisać? Przecież pamięć jest bardzo duża. Odpowiedzi na wszystkie pytania (także na inne) bardzo sprawnie i szybko (tak by się procesor nie nudził) udziela RAM. Odpowiedzi te, to instrukcje naszego programu komputerowego — jednego z tematów naszego następnego spotkania.

Włodzimierz GOGOŁEK

---

wania, którą następnie ulepszył **Ch. Mahon** wprowadzając mechanizm przeniesień.

W 1674 roku niemiecki matematyk **Gottfried Wilhelm Leibniz** opracował model arytmometru działającego na zasadzie walców schodkowych stosowanych także we współczesnych konstrukcjach. Konstrukcja modelu przewidywała wykonywanie 4 działań arytmetycznych tj. mnożenia, dzielenia, dodawania i odejmowania. Arytmometr był maszyną dwuokresową, tzn. w jednym cyklu ustawiało się cyfry liczby, natomiast w drugim — za pomocą ruchu korbką ustawione cyfry, liczby były wprowadzone do licznika.

W następnych latach G. W. Leibniz opisał operacje rachunkowe w systemie dwójkowym oraz opracował projekt maszyny liczącej w systemie dwójkowym, by w 1694 roku skonstruować pierwszy działający model arytmometru. Udoskonalony przez **Charlesa Thomasa de Calmara** arytmometr Leibniza był pierwszą, sprzedawaną maszyną liczącą.

W 1784 roku francuski matematyk **L. Carnot** założył pierwsze w świecie biuro obliczeniowe. Na początku XIX wieku **Joseph M. Jacquard** wynalazł automatyczny warsztat tkacki, w którym zastosował zasadę przedstawiania informacji przez rozmieszczenie otworów w karcie papierowej. Za pomocą tych otworów realizowane było sterowanie zmian koloru przędzy przy produkcji wielobarwnej tkaniny. Powszechnie uważa się, że zastosowana przez Jacquarda papierowa karta dziurkowana stała się pierwowzorem kart dziukowanych.

W dalszym wyliczaniu nazwisk wybitnych uczonych i wynalazców warto dłużej zatrzymać się przy nazwiskach **Charles'a Babbage'a** i **Hermana Holleritha**. Angielski uczony Charles Babbage opracował koncepcję, a następnie zbudował „machinę różnicową przeznaczoną do obliczania i drukowania tablic matematycznych" (np. tablic kwadratów liczb, dużych tablic wielomianów, itp.). W 1833 roku zaprojektował „maszynę analityczną" będącą ideą automatycznej maszyny liczącej ze sterowaniem programowym. Maszyna ta nie została jednak wykonana, gdyż potrzebne do jej budowy części mechaniczne, w tym przede wszystkim koła zębate, nie mogły być wykonane dostatecznie precyzyjnie.

Maszyna analityczna Ch. Babbage'a zawierała wszystkie elementy współczesnego komputera. Program oraz dane miały być wprowadzane do pamięci z kart dziurkowanych (według zasady Jacquarda). Mogły być przechowywane w pamięci, a także istniała możliwość modyfikacji obliczeń w zależności od użytkowych wyników.

W 1887 roku amerykański wynalazca Herman Hollerith opracował mechaniczny system rejestrowania danych przez wycinanie dziurek w długim pasku taśmy papierowej oraz zbudował maszynę sortującą do zliczania opracowanych danych. Dziurkowany pasek papieru przesuwany był ręcznie z szybkością 50—75 kart na sekundę nad pojemnikami z rtęcią. Rząd igieł teleskopowych badał kartę i w wypadku stwierdzenia dziurki zamykał obwód elektryczny, powodując w ten sposób zwiększenie stanu odpowiedniego licznika o 1. Wynalazek Holleritha zastosowano w pracach statystycznych, a przede wszystkim do opracowania wyników spisów ludności (1890 rok — USA, 1891 rok — Kanada, 1897 rok — Rosja). Było to pierwsze urządzenie, zastosowane na tak szeroką skalę do przetwarzania danych. Pozwoliło na skrócenie czasu opracowania wyników z 6 lat do 6 tygodni. Po raz pierwszy zastosowano masowo technikę kart dziurkowanych. W 1896 roku Herman Hollerith założył przedsiębiorstwo Tabulating Machine Company, które zajęło się produkcją i sprzedażą maszyn jego konstrukcji. Przedsiębiorstwo to po reorganizacji i z dwoma innymi przedsiębiorstwami przyjęło w 1911 roku nazwę Computing. Tabulating-Recording Company zmienioną w 1924 roku na International Business Machines Corporation (IBM). W tym też roku po raz pierwszy na świecie uruchomiono produkcję maszyn licząco-analitycznych. Firma IBM jest obecnie największym na świecie producentem komputerów wraz z oprogramowaniem i sprzętu komputerowego.

Warto zwrócić uwagę, że w Polsce po raz pierwszy do prac statystycznych zastosowano maszyny licząco-analityczne w roku 1921. Prace nad stworzeniem pierwszego komputera prowadziło wiele zespołów naukowo-badawczych. W 1936 roku **Allan M. Turing** przedstawił teoretyczny model uniwersalnej, abstrakcyjnej maszyny liczącej zwanej później „maszyną Turinga". W tym samym czasie **John von Neumann** zaproponował zastosowanie w maszynach liczących dwójkowego systemu liczenia w miejsce stosowanego dotychczas systemu dziesiętnego, a Ludwik Conffigual opisał sterowaną programem maszynę cyfrową mającą pracować w systemie dwójkowym. W 1939 roku Georg Stibitz skonstruował przekaźnikową, cyfrową maszynę liczącą nazwaną Complex Computer, a rok później pokazał współpracę prostej maszyny przekaźnikowej połączonej linią telefoniczną z końcówką. Wydarzenie to traktowane jest obecnie jako pierwsze zastosowanie zdalnego dostępu. W 1941 roku **Kondrat Zu-**

se opracował pierwszą całkowicie sterowaną programowo maszynę cyfrową nazwaną 23, pracującą na przekaźnikach, uznaną nawet przez niektórych uczonych za pierwszy, pracujący komputer.

W latach 1939—1944 zespół pod kierownictwem Amerykanina **Howarda Aikena** z uniwersytetu Harwarda we współpracy z firmą IBM prowadził prace nad skonstruowaniem maszyny elektromechanicznej (przekaźnikowej) opartej na idei maszyny liczącej Charlesa Babbage'a. Zbudowaną maszynę, którą nazwano Automatic Sequevice Controlled Colenlator, sterowano automatycznie za pomocą programu zapisanego na dwudziestoczterościeżkowej taśmie dziurkowanej. Maszyna ta zwana również Mark I uważana jest za pierwszy komputer. Maszyny pracujące na przekaźnikach elektromechanicznych zaliczane są obecnie do zerowej generacji komputerów.

W 1946 roku **J. M. Brainerd, J. P. Eckert, J. W. Mauchly** i **H. H. Goldstine** z Uniwersytetu Pensylwańskiego zakończyli budowę pierwszej na świecie elektronicznej maszyny cyfrowej do celów obliczeniowych nazwanej ENIAC (Elektronic Integrator and Computer), która zapoczątkowała erę współczesnych, elektronicznych komputerów. W pierwszych komputerach elektronicznych używano lamp elektronowych. Maszyny te zajmowały bardzo dużo miejsca, pobierały bardzo dużo prądu i były zawodne np. ENIAC zbudowany z około 1800 lamp elektronowych zajmował przestrzeń 150 m^2 i pobierał 150 KW energii. Mimo zawodności ENIAC pracował jak na owe czasy bardzo szybko. W czasie pracy wykonywał 10000 operacji w każdej sekundzie. Dla przykładu podamy, że czas dodawania dwóch liczb dziesiętnych, dziesięciopozycyjnych wynosił około 0,2 ms. ENIAC zbudowany na potrzeby amerykańskiej artylerii do wykonywania obliczeń balistycznych został praktycznie zastosowany w 1947 roku po zainstalowaniu na poligonie Aberdeen w Instytucie Balistyki Szefostwa Artylerii Armii USA. Na kolejne wersje lepszych maszyn nie trzeba było długo czekać. Już w rok po zainstalowaniu ENIAC-a John von Neuman przedstawił koncepcję komputera cyfrowego operującego na liczbach binarnych. Według koncepcji von Neumana program powinien być zapamiętywany w pamięci operacyjnej. Rozkazy i dane należało przedstawić w pamięci operacyjnej jednakowo, tzn. w postaci liczb binarnych, a rozkazy powinny być wykonywane automatycznie. Opierając się na tej koncepcji zbudowano na potrzeby armii amerykańskiej komputer EDVAC (Electronic Discrete Varizble Automatic Computer). Pierwszym komputerem z pamiętanych programem był skonstruowany w 1948 roku pod kierunkiem **M. W. Wilkesa** z uniwersytetu Cambridge komputer EDSAC (Electronic Delay Starage Automatic Calculator).

W 1949 roku zbudowano pierwsze komputery uniwersalne umożliwiające zastosowanie w problematyce przetwarzania danych. Były to skonstruowany w USA — UNIVAC, a w Wielkiej Brytanii — LEO.

Maszyny pracujące na lampach elektronowych zaliczane są obecnie do pierwszej generacji komputerów.

Na początku lat pięćdziesiątych pojawiły się pierwsze języki programowania jako interpretery. **Grace Hopper** z firmy Remington Rand opisała pierwszy kompilator. Uwzględnienie w konstrukcji komputerów możliwości adresowania symbolicznego oraz adresowania pośredniego umożliwiło rozwój języków programowania. W 1956 roku zostaje wprowadzony przez firmę IBM wysokopoziomowy język programowania FORTRAN. W roku 1960 wprowadzono języki ALGOL i COBOL.

Postęp w elektronice umożliwiał dalszy rozwój komputerów. W latach sześćdziesiątych lampy elektronowe zastąpiono tranzystorami. Sam tranzystor wynaleźli J. Bardeen, W. H. Brattai i W. Shockly w 1948 roku. Pierwszy tranzystor został wyprodukowany w laboratorium firmy Bell. Pierwszy tranzystorowy komputer zbudowano w 1956 roku w Massachusettes Institute of Technology, a w cztery lata później firma IBM uruchomiła seryjną produkcję komputerów tej klasy.

Na początku lat sześćdziesiątych uzyskano szybkość pracy komputera równą jednemu milionowi operacji na sekundę, a także skonstruowano pierwsze układy scalone (początkowo układ scalony zawierał w jednej obudowie kilkadziesiąt tranzystorów i innych elementów elektronicznych), które zaczęto stosować przy budowie nowych maszyn. Dzięki temu komputery stały się znacznie mniejsze, a jednocześnie zwiększyła się szybkość ich działania.

Maszyny pracujące na tranzystorach zalicza się obecnie do drugiej generacji, a na układach scalonych do trzeciej generacji komputerów.

W 1965 roku w USA rozpoczęto seryjną produkcję komputerów cyfrowych serii IBM 360 zaliczanych do trzeciej generacji oraz pojawiły się pierwsze komputery analogowe zbudowane na układach scalonych.

W roku 1967 również w USA zastosowano pierwsze abonenckie systemy informatyczne, pracujące w systemie podziału czasu, a także rozpoczęto produkcję komputerów hybrydowych na układach scalonych.

Z ciekawostek warto dodać, że w tym samym roku odbył się pierwszy mecz szachowy pomiędzy radzieckimi i amerykańskimi komputerami. Rozegrano cztery partie. Dwie partie wygrał program opracowany w ZSRR, dwie partie zakończyły się remisem.

W latach sześćdziesiątych udoskonalono urządzenia zewnętrzne wspomagające pracę użytkowników sprzętu komputerowego. W 1968 roku rozpoczęto produkcję czytników dokumentów przystosowanych do automatycznego odczytu, a także graficznych urządzeń zewnętrznych tj. grafoskopów i alfaskopów, miniaturyzacja elementów elektronicznych, produkcja układów scalonych o coraz większym upakowaniu podstawowych elementów elektronicznych doprowadziła do skonstruowania pierwszych minikalkulatorów (kalkulatorów) na układach scalonych LSI (large scale integradion — dużej skali integracji), a także pojawiły się pierwsze kalkulatory kieszonkowe. W 1969 roku wydawca nowojorski William Morrow and Co. dokonał pierwszej próby wykorzystania komputera do wykonania składu książki.

Na przełomie lat sześćdziesiątych i siedemdziesiątych dokonał się ogromny skok w dziedzinie mikroelektroniki. W 1969 roku projektanci z amerykańskiej firmy Datapoint zaprojektowali bardzo prosty procesor i zlecili firmom Texas Instruments i Intel wykonanie go w postaci jednego układu scalonego. Z postawionego zlecenia wywiązała się firma Intel, choć okazało się, że pierwszy mikroprocesor działał z szybkością około 10 razy mniejszą od szybkości oczekiwanej przez firmę Datapoint. Z tego powodu zerwano umowę, ale Intel wykonał jednak prototyp zaprojektowanego procesora. Zdecydowano się także na produkcję seryjną procesora. W ten sposób na rynku pojawił się pierwszy mikroprocesor Intel 4004, o długości słowa równej czterem bitom. W 1974 roku pojawiły się mikroprocesory na ulepszonych układach scalonych i większej długości słowa (8 bitów) i słowa adresowego (16 bitów). W tym samym roku odbyły się pierwsze komputerowe mistrzostwa świata w szachach. Rozegrano je w Sztokholmie na kongresie IFIP, (International Federation for Information Processing). Uczestniczyło 13 programów z 8 krajów. Tytuł mistrzowski zdobył program „Kaissa" opracowany w ZSRR. W 1976 roku opracowano pierwszy mikrokomputer zawierający się w jednym układzie scalonym (seria Intel MCS-48).

Szybki postęp w technologii produkcji układów scalonych (zaczęto wytwarzać układy scalone bardzo dużej skali integracji VLSI-very large scale integradion) oraz komputerów sprawił, że zaczęto mówić o rewolucji komputerowej. Komputery zbudowane z układów VLSI zaczęto zaliczać do czwartej generacji.

W początkowym okresie komputery dla pojedynczego użytkownika produkowano w Hewlett-Packard i IBM. Komputery te były zaprojektowane dla naukowców (kosztowały około 10 000 $), a można je było programować w Basicu i używać od razu po włączeniu do sieci.

Pierwszym mikrokomputerem był Mark 8 opisany jako konstrukcja dla amatorów w czasopiśmie Radio-Electronics w lipcu 1974 roku. W styczniu 1975 roku rozpoczęto produkcję komputera Altair, opartego na 8-bitowym procesorze Intel 8080.

W połowie lat siedemdziesiątych Stere Woźniak i Stere Computer — Apple dla masowego odbiorcy (cena początkowa około 900 $) Apple był zbudowany na bazie mikroprocesora typu 6502 firmy MOS Technology, współpracował z klawiaturą oraz wykorzystywał ekran telewizyjny. Wszystkie układy komputera były umieszczone na jednej płytce. Po udanej konstrukcji Apple wyprodukowano znacznie ulepszony komputer osobisty Apple II, który ciągle modyfikowany, produkowany jest do dziś. Kolejnym sukcesem firmy Apple cieszyła się konstrukcja z 1983 roku Macintosh. W ciągu 100 dni wyprodukowano około 70 000 sztuk tego komputera.

Z innych bardzo udanych modeli mikrokomputerów należy wymienić wyprodukowany w 1980 roku przez firmę Clive Sinclaira Z 80 — pierwszy komputer na każdą kieszeń, w cenie poniżej 200 $. W 1981 roku Sinclair wyprodukował ZX 81, a w rok później tak popularny w Polsce ZX Spectrum. Obecnie kierunek rozwoju komputerów osobistych wytyczają Atari serii ST, Commodore Amiga oraz IBM.

Największy na świecie producent sprzętu komputerowego, amerykańska firma IBM (około 70% produkcji światowej), przystąpił do produkcji komputerów osobistych w 1981 roku. Opracowano od razu komputer IBM PC z 16-bitowym procesorem, choć w tamtym okresie powszechnie stosowano mikroprocesory 8-bitowe. Konstrukcję IBM PC łatwo jest modyfikować poprzez wymianę lub dołączanie nowych modułów, co zwiększa możliwości obliczeniowe i graficzne komputera oraz pozwala na dołączenie nowych urządzeń zewnętrznych. Model IBM PC stał się standardem komputera osobistego przeznaczonego do zastosowań profesjonalnych.

oprac. Jacek SZANIAWSKI

PROGRAM 60

ZX SPECTRUM

Projektant — znaki

Szanowna Redakcjo !

W numerze 3 Waszego pisma zamieszczono program na "C-64" "Projektant-znaki", a ponieważ posiadam podobny program na ZX SPECTRUM - "UDG", więc postanowiłem przerobić go na język polski i wysłać jego wydruk do Was.

Program działa bezbłędnie, o czym może świadczyć fakt, że polskie litery, którymi się właśnie posługuje zostały utworzone na tym programie.

Program ten po przepisaniu na komputer należy zapisać na taśmę instrukcja:

SAVE "UDG" LINE 400.

Zbigniew Zych

```
  1 REM    polska wersja
**************ZBIGNIEW ZYCH**************
                 05.11.01.
  10 GO TO 8000
 100 PLOT 63,96: DRAW 129,0
 110 PLOT 128,160: DRAW 0,-129
INK 0
 120 IF INKEY$="" THEN GO TO 120
 130 IF INKEY$="1" THEN LET a=1
 LET a$="■": LET b$="■"
 135 IF INKEY$="4" THEN GO SUB 1
800
 140 IF INKEY$="0" THEN LET a=0
 LET a$="*": LET b$=" "
 145 IF INKEY$="3" THEN GO SUB 1
000
 147 IF INKEY$="2" THEN GO SUB 1
324
 150 LET x$=INKEY$
 155 PRINT AT y,x,b$
 160 LET x=x+(x$="8" AND x<23)-(
x$="5" AND x>0)
 170 LET y=y+(x$="6" AND y<17)-(
x$="7" AND y>0)
 180 PRINT AT y,x,a$
 185 IF a=1 AND POINT (x*8+5,175-y
*8)=1 OR a=0 AND POINT (x*8+5,175-y
*8)=0 THEN GO TO 290
 190 IF x<15 THEN LET b=2
 195 LET c=(b-1)*16+y-1
 200 IF POINT (x*8+1,175-y*8)=1
THEN LET a(c)=a(c)+2↑(7-(x-8*b))
 205 IF POINT (x*8+1,175-y*8)=0
THEN LET a(c)=a(c)-2↑(7-(x-8*b))
 210 IF a=0 AND POINT (x-8,177-y
)=0 THEN GO TO 250
 220 PLOT INK 0, OVER 0+(1 AND P
OINT (x-8,177-y)=1 AND a=0),x-8,
177-y
 230 PLOT INK 0, OVER 0+(1 AND P
OINT (232+x,177-y)=1 AND a=0),23
2+x,177-y
 255 PRINT AT y,3+(b-1)*23,"    "
 260 PRINT AT y,6+(b-1)*23-LEN S
TR$ a(c), INK 0;a(c)
 270 LET b=1
 290 BEEP .02,-10: BEEP .02,0: B
EEP .02,10
 300 GO TO 100
```

```
 400 POKE 23609,50: CLS: PRINT
AT 8,17;"■   ■";AT 9,13;"■■■■■";
AT 10,13;"■■■■■";AT 11,13;"■■■■■
■";AT 12,18;"■■";: PAUSE 100
: GO TO 8000
1000 REM FUNKCJE
1010 PRINT AT 19,0;"   P - POKE
   S - SAVE                "
1015 PRINT #1;AT 1,0;"
1020 PRINT AT 20,0;"   K - KONIE
C    C - CONTINUE            "
1025 PRINT AT 21,0;"
1030 IF INKEY$="p" THEN GO SUB 1
100
1040 IF INKEY$="s" THEN GO SUB 1
300
1050 IF INKEY$="k" THEN GO SUB 1
500
1060 IF INKEY$="c" THEN GO SUB 1
700: RETURN
1070 BEEP .005,0: GO TO 1010
1100 REM FUNKCJA POKE
 PONIZSZE ZNAKI ■■ GRAFICZNE !
1110 PRINT AT 19,0;"A  B  C  D
E  F  G  H  I  J  K"
1120 PRINT AT 20,0;"L  M  N  O
P  Q  R  S  T  U  "
1125 FOR n=1 TO 30: NEXT n: LET
m=0: POKE 23658,8: FOR f=1 TO 4
1130 PRINT #1;AT 1,1;f;". Litera
 (A-U) :": LET p$=INKEY$
1133 IF CODE p$=13 THEN FOR n=1
TO 30: NEXT n: GO TO 1165
1140 IF CODE p$<65 OR CODE p$>85
 THEN GO TO 1130
1160 FOR n=0 TO 7: LET m=m+1: PO
KE USR CHR$ (CODE p$+79)+n,a(m):
 NEXT n
1165 REM
 PONIZSZE ZNAKI ■■ GRAFICZNE !
1170 PRINT AT 19,0;"A  B  C  D
E  F  G  H  I  J  K"
1180 PRINT AT 20,0;"L  M  N  O
P  Q  R  S  T  U  "
1183 NEXT f
1185 POKE 23658,0: PRINT #1;AT 1
,1;"
1190 PRINT AT 21,0; OVER 1;"
Wcis"+CHR$ 8+"'nij dowolny klawi
sz !": PAUSE 0: RETURN
1300 REM SAVE
 PONIZSZE ZNAKI ■■ GRAFICZNE !
1310 PRINT AT 19,0;"A  B  C  D
E  F  G  H  I  J  K"
1320 PRINT AT 20,0;"L  M  N  O
P  Q  R  S  T  U  "
1321 FOR n=1 TO 30: NEXT n
1322 GO TO 1330
1324 PAUSE 5: PRINT #1;AT 1,9;"
COPY  ? (t/n)": IF INKEY$="" THE
N GO TO 1324
1325 IF INKEY$="t" THEN PRINT AT
 y,x;"■": COPY : PRINT AT y,x;a$
1326 GO SUB 1700: RETURN
1327 FOR n=1 TO 30: NEXT n
1330 PRINT #1;AT 1,0;"          $
AVE  ? (t/n)      ": IF INKEY$="" T
HEN GO TO 1330
1335 IF INKEY$="n" THEN RETURN
1336 IF INKEY$<>"" THEN GO TO 13
36
1340 INPUT "Nazwa  ";n$: IF LEN
n$>10 THEN GO TO 1340
1345 IF n$="" THEN LET n$="udg"
1350 SAVE n$CODE USR "a",168
```

```
1353 FOR n=1 TO 30: NEXT n
1355 PRINT #1;AT 1,8;" VERIFY ?
(t/n)": IF INKEY$="" THEN GO TO
1355
1356 IF INKEY$<>"t" THEN RETURN
1360 PRINT AT 19,0;"
                    ";AT 19,8; F
LASH 1; OVER 1;"Przemin"+CHR$ 8+
"  tas"+CHR$ 8+"  me"+CHR$ 8+", !
";AT 20,8;"     VERIFY !   "
1365 PRINT AT 18,0
1370 VERIFY p$CODE
1380 RETURN
1500 REM KONIEC
1510 FOR n=1 TO 30: NEXT n
1520 PRINT #1;AT 1,8;"KONIEC ?
(t/n)": IF INKEY$="" THEN GO TO
1520
1525 IF INKEY$="t" THEN GO TO 99
99
1530 RETURN
1700 REM
1710 PRINT AT 19,1;" 1-RYSUJ
          0-MAZ"+CHR$ 8; OVER 1
;" "
1720 PRINT AT 20,1;" 3-MENU
4-CLS       2-COPY "
1725 PRINT #1;AT 1,0;"
                    "
1730 RETURN
1800 REM CLS
1810 FOR n=1 TO 30: NEXT n
1820 PRINT #1;AT 1,0;"
   CLS  ? (t/n)          ": IF INKEY
$="" THEN GO TO 1820
1830 IF INKEY$="t" THEN GO TO 90
00
1840 PRINT #1;AT 1,0;"
                    ": RETURN
8000 REM INSTRUKCJA
8005 BORDER 7: PAPER 7: INK 0
8010 CLS : PRINT AT 0,5;"User De
fined Graphics";AT 1,4;"
                    "

8020 PRINT "Program ten umozliwi
a tworzenie 21 wlasnych symboli
 graficznych.": PLOT 123,159
8030 PRINT "Marker ""*"" przesu
wany jest kur sorami (5-8)."
8035 PRINT "Wybor litery,pod kt
ora zdefinio-wany zostanie dany
 symbol,jest dowolny.": PLOT 27,
111: PLOT 27,110: PLOT 155,111:
PLOT 155,110: PLOT 175,105: PLOT
 175,106
8040 PRINT "Mozliwy jest zapis
wszystkich  symboli na tasmie i
ub otrzymaniekopii na drukarce
(wymaga to  zmiany jednej z ins
trukcji !).": PLOT 19,79: PLOT 1
07,71: PLOT 107,70
8050 PRINT "Zestaw symboli na t
asmie mozna  zaladowac zleceniem
 LOAD""""CODE  i wykorzystac w i
nnym programie.": PLOT 163,39: P
LOT 163,38: PLOT 219,39: PLOT 18
,27: PLOT 20,28: PLOT 21,29: PLO
T 68,31: PLOT 68,30: PLOT 100,23
: PLOT 100,22
8060 PRINT #1;AT 1,3; OVER 1;"Ac
is"+CHR$ 8+" nij dowolny klawisz
 !"

8100 PAUSE 0
9000 REM
9000 BORDER 6: PAPER 7: INK 0: C
LS
9010 PRINT AT 0,5;"User Defined
Graphics"
9015 INK 0: PLOT 38,166: DRAW 17
2,0
9020 PLOT 63,160: DRAW 129,0: DR
AW 0,-129: DRAW -129,0: DRAW 0,1
29
9025 PLOT 63,96: DRAW 129,0: PLO
T 128,160: DRAW 0,-129
9030 FOR n=28 TO 161 STEP 133
9040 FOR m=64 TO 193 STEP 8
9050 PLOT m,n: DRAW 0,2: NEXT m:
 NEXT n
9060 FOR n=60 TO 195 STEP 134
9070 FOR m=32 TO 164 STEP 8
9080 PLOT n,m: DRAW 2,0: NEXT m:
 NEXT n
9090 PRINT AT 20,1;" 3-MENU
4-CLS       2-COPY"
9095 PRINT AT 19,1;" 1-RYSUJ
          0-MAZ"+CHR$ 8; OVER 1
;" "
9100 FOR n=5 TO 26 STEP 23: FOR
m=2 TO 17: PRINT AT m,n; INK 0;0
: NEXT m: NEXT n
9110 PRINT PAPER 7;AT 0,0;"  ";A
T 1,0;"  ";AT 0,30;"  ";AT 1,30
;"  "
9120 INK 0: PLOT 0,159: DRAW 16,
0: DRAW 0,16: DRAW 1,0: DRAW 0,-
17: DRAW -17,0: DRAW 0,-1: DRAW
18,0: DRAW 0,18
9130 PLOT 237,175: DRAW 0,-18: D
RAW 18,0: DRAW 0,1: DRAW -17,0:
DRAW 0,17: DRAW 1,0: DRAW 0,-16:
 DRAW 16,0: INK 0
9500 REM ZMIENNE
9510 LET y=2: LET x=8: DIM a(40)
9520 LET a=0: LET a$="*"
9530 LET b$=" ": LET b=1
9998 GO TO 180
9999 BORDER 0: CLS : PRINT AT 7,
0; FLASH 1;" Podanie instrukcji
CONTINUE ,  pomoduje wyzerowani
e komputera.": STOP : RANDOMIZE
USR 0
```

Autor(zy) M&G Soft.

UWAGA !!

U liniach :

1110,1120,1170,1180,1310 i 1320

Litery w instrukcjach PRINT są znakami graficznymi !

Pierwsze uruchomienie : RUN 400, następne : RUN .

UWAGI DOTYCZĄCE DZIAŁANIA PROGRAMU

PROGRAM 61

DWA GŁOSY NA SPECTRUM

Powszechnie znane ułomności popularnego ZX-Spectrum, to jego małe możliwości akustyczne. W komputerze tym zainstalowany jest jeden generator akustyczny, który ma tylko dwa zmienne parametry: wysokość tonu i czas trwania dźwięku.

Pozornie niemożliwe jest naśladowanie jakichś instrumentów, czy też granie melodii na dwa głosy. Zamieszczony niżej zestaw trzech programów niemożliwe czyni możliwym. Pierwszy z nich naśladuje dźwięk mandoliny; gra na dwa głosy starą rosyjską melodię „Samotna harmonia". Po wprowadzeniu do pamięci uruchamia się go dyrektywą **RUN.**

Następnie, nie wymazując poprzedniego programu, dopiszcie treść drugiego. Nie zrażajcie się, jeśli powtórzy się wiersz o jakimś numerze. Dyrektywa RUN spowoduje zagranie drugiej, równie pięknej melodii „Suliko".

Jeśli, nie wymazując dwu pierwszych programów, wprowadzicie trzeci, to będziecie mogli wybierać na zmianę: pierwszą lub drugą melodię. Aktualnie wybierana melodia wskazywana jest przez mrugający kursor. Zmiany wyboru dokonujemy klawiszem **SPACE,** a wybór potwierdzamy klawiszem **ENTER.** Stop programu następuje po naciśnięciu klawisza **BREAK.**

```
    1 REM Samotna harmonia
    5 DEF FN m()=1-2* INT (1/2)
    10 DATA "aCECazHDFEaHCafczHCDFEF
CHCDFEaEDDCHaHCDEFFFGFGFEaEDDCHazaa"
    20 DATA "eaCafeaHCaczafchezaHCDD
azaHCCeCHHazezaHCDDDFDFDCeCHHazeee"
    60 LET t=18: LET tm=23672
    70 LET h=-1: LET c=0: LET d=2:LET e=4:
LET f=5: LET g=7:LET z=8: LET a=9
    80 READ p$: READ d$
    150 FOR k=1 TO 3
    160 LET i=1
    170 FOR l=1 TO 6
    180 LET buf=i+7+ FN m(): FOR j=i
TO buf: GOSUB 1010: NEXT i: LET t=t+
(5- FN m()): GOSUB 1010: LET i=i+1:
LET t=18
    190 NEXT l
    200 NEXT k
    1000 CLS: STOP
    1010 LET w= VAL p$(i): LET u= VAL d$(i)
    1020 IF d$(i) <= "Z" THEN LET w=w+12
    1030 IF p$(i) <= "Z" THEN LET u=u+12
    1040 POKE tm,0
    1050 BEEP 1/60,w: BEEP 1/60,u
    1060 POKE tm, PEEK tm+2: IF PEEK tm < t
THEN GOTO 1050
    1070 RETURN
```

```
2 REM Suliko
    30 DATA "ggefgafg0HHHHHDHC0CCHCDCHaHagaaggfgg0"
    40 DATA "eecdefde0gggggHgg0aagaHagfgfeffeedeec0"
    50 DATA "1111211821111211821111111112251111121182"
    60 LET t=18: LET tm=23672
    70 LET h=-1: LET c=0: LET d=2: LET e=4: LET f=5:
LET g=7: LET z=8: LET a=9
    90 READ q$: READ b$: READ v$: GOTO 250
    250 FOR k=1 TO 3
    260 LET x=1: GOSUB 1200
    270 LET x=19:GOSUB 1200
    280 NEXT k
    1200 FOR i=x TO LEN q$
    1210 IF q$(i) ="0" THEN PAUSE t* VAL v$(i):GOTO 1290
    1220 LET w= VAL q$(i)
    1230 LET u= VAL b$(i)
    1240 IF q$(i) <= "Z" THEN LET w=w+12
    1250 IF b$(i) <= "Z" THEN LET u=u+12
    1260 POKE tm,0
    1270 BEEP 1/60,w: BEEP 1/60,u
    1280 POKE tm, PEEK tm+2: IF PEEK tm < VAL v$(i) THEN
GOTO 1270
    1290 NEXT i
    1300 RETURN

      90 READ q$: READ b$: READ v$
      95 DIM s$(2,16): LET s$(1)="Samotna harmonia":
 LET s$(2)="Suliko"
      100 PRINT AT 10,1;OVER 1;"Kto"; CHR$ 8;", ra";
 CHR$ 8;", melodie";CHR$ 8;", wybierasz?"
      105 PRINT AT 12,4;s$(1); AT 13,4;s$(2): LET l=0
      110 PRINT AT 12+l,1; FLASH 1;" "
      120 IF INKEY$ =" " THEN PRINT AT 12+l,1;" ":
LET l=l+1: LET l=FN m(): GOTO 110
      130 IF INKEY$ = CHR$ 13 THEN CLS : PRINT AT l+
10;s$(l+1): GOTO 150+l*100
      140 GOTO 120
      1000 CLS : GOTO 100
```

M P

Programowanie nie jest trudne, czyli MINI-GEM

Siadając do pracy przy komputerze, trudno sobie uzmysłowić, że jeszcze kilka lat temu te elektroniczne cacka nie były tak przyjaźnie „ustosunkowane" do człowieka jak dzisiaj. Starsi informatycy ze wstrętem wspominają czasy, gdy program wprowadzano do komputera za pomocą kart perforowanych, albo nawet taśmy dziurkowanej.

Jeszcze przed pięcioma laty tekst programu wprowadzało się do pamięci maszyny za pomocą prostego edytora, a trzeba było przed przystąpieniem do pracy dokładnie przestudiować instrukcję jego obsługi. Kto raz pracował na dzisiejszym komputerze oprogramowanym w systemie GEM, ten nigdy dobrowolnie nie będzie chciał wrócić do wcześniejszych metod.

Co to jest GEM? Jest to skrót od wyrazów angielskich, które można zinterpretować: „Zarządzanie środowiskiem komputerowym za pomocą symboli graficznych". Wykonanie jakiejś funkcji można sprowokować nie poprzez pracowite wystukiwanie dyrektywy na klawiaturze z możliwością popełnienia błędu, lecz dzięki wskazaniu na monitorze komputera odpowiedniego symbolu graficznego. Zamiast pisać „DELETE" (usuń) wystarczy wskazać na ekranie symbol kosza na śmieci. Wyboru symbolu dokonujemy strzałką, która wędruje po całym ekranie dzięki, tzw. myszy. Pomyłka jest prawie wykluczona.

Wbrew pozorom nie jest to takie trudne nawet wtedy, gdy nie mamy do dyspozycji „myszy". Nieco mniej komfortowym urządzeniem jest joystick. Poza tym można wykorzystać klawisze ze strzałkami, które znajdują się na klawiaturach wszystkich minikomputerów.

Żeby nie być gołosłownym przedstawiamy wam krótki program, dzięki któremu możemy przesuwać w jednym z czterech kierunków dowolny tekst. Program napisany jest w BASIC-u dla komputera ZX-Spectrum. Widać więc, że nawet w tak bardzo ganionym języku i jednocześnie na tak małym i prostym komputerze można zaimplementować opisaną wcześniej metodę sterowania programem.

Dla ułatwienia wspomnę, że zmiany jednej z sześciu opcji dokonujemy klawiszem **SPACE,** a wybór potwierdzamy klawiszem **ENTER.** Dwie „niekierunkowe opcje to:

C — czytaj tekst,

R — ruch całego tekstu.

Resztę powiedzą wam symbole. Stop programu nastąpi po wciśnięciu klawisza **BREAK.**

```
10 FOR n = 0 TO 79: READ a:
 POKE USR "a" + n,a: NEXT n
20 GOSUB 9000: LET t$ = u$:
 GOSUB 9000: LET i = 0: GOTO
 1000
30 LET t = i*4+1: PRINT AT 10,2
; t$(1 TO t);FLASH 1;t$(t+1 TO
 t+2); FLASH 0; t$(t+3 TO LEN
 t$)'TAB 2; u$(1 TO t); FLASH
 1; u$(t+1 TO t+2); FLASH 0;
```

Wydruk ekranu

Czy zawsze, gdy chcesz coś wydrukować, potrzebujesz edytora tekstu lub programu graficznego? Z pewnością nie. Do prostych prac przy formacie 40 znaków × 24 wiersze, może posłużyć Ci poniższy program. Trudno oczywiście mówić o tworzeniu złożonej grafiki, gdy pracujemy w trybie graficznym 0. Można jednak, wykorzystując zestaw znaków możliwy do uzyskania na drukarce 1029, tworzyć nieskomplikowaną grafikę, tabele, teksty o maksymalnie 40-znakowym wierszu.

Po uruchomieniu programu, wybieramy rodzaj drukowania. Chodzi o to, że w trybie tekstowym stosuje się odstęp między wierszami. W ten sposób na jeden cal wchodzi sześć wierszy tekstu. W trybie graficznym nie stosuje się odstępu między wierszami, dzięki czemu w jednym calu mieści się dziewięć wierszy. Gdy dokonasz wyboru, program przejdzie w stan wyczekiwania. Będzie gotów do tworzenia obrazu. Przesuwanie kursora po ekranie odbywa się tradycyjnie, np. naciśnięcie klawisza CONTROL i któregoś ze strzałką, przesuwa kursor w kierunku wskazywanym przez strzałkę; naciśnięcie klawisza RETURN powoduje przesunięcie kursora na początek następnego wiersza, itd.

Drukowanie inicjuje się przez naciśnięcie klawisza ESC. W związku z tym, że obraz przechowywany jest tylko w pamięci ekranu, po wydrukowaniu zostaje skasowany. Jest to z pewnością niedogodność, którą rekompensuje fakt, że obrazy czy teksty drukowane tą metodą, będzie można otrzymać znacznie szybciej niż przy zastosowaniu np. złożonych edytorów, których czas wprowadzania jest znaczny.

Fragment zaczynający się od etykiety 600, a kończący się na 670, to zasadnicza część drukująca. Może ona również służyć jako podprogram drukujący dowolne fragmenty wykonującego się programu np. tekst, tabele itp. Sens wykorzystania go do tego celu jest jednoznaczny. Przy dużej różnorodności zmiennych, stosowanie do każdej z nich instrukcji LPRINT staje się często niewygodne, gdyż z reguły chcemy również oglądać je na ekranie. Program niepotrzebnie rozrasta się, jak również staje mniej czytelny. Nie należy przy tym zapomnieć o zadeklarowaniu długości ciągu tekstowego a$.

```
PK 100 REM ************************
KL 105 REM *                      *
GR 110 REM *  TWORZENIE PROSTEJ   *
BH 115 REM *  GRAFIKI  I WYDRUK   *
LZ 120 REM *  NA DRUKARCE 1029.   *
KP 125 REM *                      *
PQ 130 REM ************************
JX 200 REM *                      *
DV 205 REM *  RODZAJ DRUKOWANIA   *
JZ 210 REM *                      *
GU 230 GRAPHICS 0:OPEN #2,8,0,"P:"
ZU 235 ? :? "RODZAJ DRUKOWANIA :"
PK 240 ? "--------------------"
CV 245 ? :? CHR$(177);" TEKSTOWE"
AM 250 ? :? CHR$(178);" GRAFICZNE"
UN 255 ? :INPUT RD
UV 260 IF RD=2 THEN PRINT #2;CHR$(
   27);"9";
KV 345 REM *                      *
UE 350 REM *  TWORZENIE GRAFIKI   *
KX 355 REM *                      *
MA 360 OPEN #1,4,0,"K:":POKE 82,0
TT 365 GRAPHICS 0:DIM A$(40)
TS 370 FOR C=1 TO 20:? CHR$(31);:N
   EXT C:FOR C=1 TO 11:? CHR$(29);
   :NEXT C
KS 375 GET #1,R
QB 380 IF R=27 THEN 600
SH 385 ? CHR$(R);
SB 390 GOTO 375
KA 500 REM *                      *
AG 505 REM * PROGRAM REALIZUJACY  *
IN 510 REM * DRUKOWANIE ZAPRO -   *
MK 515 REM * JEKTOWANEJ GRAFIKI.  *
KE 520 REM *                      *
QA 600 FOR A=0 TO 23
XU 610 A$=""
VB 620 FOR B=0 TO 39
LN 630 LOCATE B,A,X
GB 640 D=B+1:A$(D,D)=CHR$(X)
DO 650 NEXT B
YD 660 IF RD=2 THEN PRINT #2;A$:GO
   TO 670
HT 665 LPRINT A$
DI 670 NEXT A
KC 700 REM *                      *
MI 710 GRAPHICS 0:POKE 82,4
DF 720 ? :? "CZY CHCESZ KONTYNUOWA
   C T/N"
AV 730 ? :INPUT TN
OC 740 IF TN=84 THEN RUN
OG 750 END
```

Janusz JANIEC

```
 u$(t+3 TO LEN u$)
5Ø IF INKEY$ = "" THEN GOTO 5Ø
6Ø IF INKEY$ = CHR$ 13 THEN
 GOTO 3ØØ+i
7Ø IF INKEY$ = " " THEN LET i
 = i+1: IF i > 6 THEN LET i = Ø
8Ø GOTO 3Ø
3ØØ GOTO 1ØØØ
3Ø1 GOTO 1Ø1Ø
3Ø2 GOTO 1Ø5Ø
3Ø3 GOTO 125Ø
3Ø4 GOTO 12ØØ
3Ø5 GOTO 11ØØ
3Ø6 STOP
999 REM  Czytanie tekstu
1ØØØ CLS: INPUT "Napisz jakis
 tekst" 'LINE p$: GOTO 3Ø
1ØØ9 REM Przesuw w lewo
1Ø1Ø FOR k = Ø TO LEN p$-1
1Ø2Ø FOR L = 3Ø TO k STEP -1
1Ø3Ø PRINT AT 15,1;p$(k+1);" "
1Ø4Ø NEXT 1: NEXT k: GOTO 3Ø
1Ø49 REM Przesuw w prawo
1Ø5Ø FOR k = LEN p$-1 TO Ø
 STEP -1
1Ø6Ø FOR 1 = Ø TO 31-LEN p$+k
1Ø7Ø PRINT AT 15,1;" ";p$(k+1)
1Ø8Ø NEXT 1: NEXT k: GOTO 3Ø
1Ø98 REM Plynny ruch calego
1Ø99 REM tekstu w lewo
11ØØ LET g$ = " "
1110 FOR k = 1 TO 5
1120 LET g$ = g$+g$
1130 NEXT k
1140 LET o$ = g$+p$+g$+" "
1150 FOR k = 1 TO LEN o$-32
1160 PRINT AT 15,Ø;o$(k TO
 k+31): PAUSE 5
1170 NEXT k: GOTO 7Ø
1199 REM Przesuw do gory
12ØØ LET w = LEN p$: IF n > 21
 THEN LET w = 21
1210 FOR k = 1 TO w
1220 FOR 1 = 2Ø TO k-1 STEP -1
1230 PRINT AT 1,Ø;p$(k): PRINT
 AT 1+1,Ø;" "
1240 NEXT 1: NEXT k: GOTO 3Ø
1249 REM Przesuw do dolu
1250 LET w = LEN p$: IF w > 21
 THEN LET w = 21
1260 FOR k = w TO 1 STEP -1
1270 FOR 1 = 1 TO k
1280 PRINT AT 1,Ø;p$(k):PRINT
 AT 1-1,Ø;" "
1290 NEXT 1: NEXT k: GOTO 3Ø
7999 REM Grafika uzytkowa
8ØØØ DATA 192,96,48,24,12,6,3,1
8Ø1Ø DATA 1,3,6,12,24,48,96,192
8Ø2Ø DATA 24,24,24,24,24,24,24
,24
8Ø3Ø DATA Ø,Ø,Ø,255,255,Ø,Ø,Ø
8Ø4Ø DATA 31,31,24,24,24,24,31
,31
8Ø5Ø DATA 252,254,7,3,3,7,254
,252
8Ø6Ø DATA Ø,7,15,28,24,24,24,24
8Ø7Ø DATA 24,24,24,24,28,15,7,Ø
8Ø8Ø DATA Ø,254,255,3,Ø,Ø,Ø,Ø
8Ø9Ø DATA Ø,Ø,Ø,Ø,3,255,254,Ø
8799 REM Plansza
88ØØ DATA 15Ø,152,32
8811Ø DATA 32,145,147,32
8822Ø DATA 32,147,144,32
8833Ø DATA 32,146,146,32
8844Ø DATA 32,145,144,32
8855Ø DATA 32,148,149,32
8866Ø DATA 32,146,145,32
8877Ø DATA 151,153,32
8888Ø DATA 32,144,147,32
8899Ø DATA 32,147,145,32
89ØØØ DATA 32,144,145,32
8911Ø DATA 32,146,146,32
8922Ø DATA 32,146,144,32
8933Ø DATA 32,146,144,32
8999 REM Czytanie planszy
9ØØØ LET u$ = " "
9Ø1Ø FOR k = 1 TO 27
9Ø2Ø READ a: LET u$ = u$+CHR$ a
9Ø3Ø NEXT k: RETURN
```

Mieczysław PŁACHETA

Lilavati (9)

Spłowiałe rękopisy

Danuta KWASIŻUR Mieczysław SKONIECZNY

Z treścią dzisiejszego zadania rozwiązywanego z udziałem mikrokomputera zapoznaje użytkownika ekran informacyjny, który pokazuje się po zainicjowaniu pracy programu. Jego zawartość jest następująca:

```
 Odnaleziono bardzo cenne, stare reko-
pisy matematyczne, ktore jednak wskutek
niewlasciwego przechowywania splowialy
tak dalece, ze tylko zarysy niektorych
cyfr mozna bylo odroznic.

   Cyfry nieczytelne oznaczone sa
   gwiazdkami. Nalezy w ich miejsce
   wstawic pierwotnie tam wpisane
   cyfry.

KLAWISZE STERUJACE:

do sterowania kursorem : ↑,↓,←,→
klawisz 'z' - wstawianie cyfry
klawisz 'ENTER' - koniec rozwiazywania
                  zadania

   NACISNIJ DOWOLNY KLAWISZ
```

Do rozwiązania jest 8 zadań, których kolejność wykazuje coraz większą skalę trudności. Po naciśnięciu klawisza ENTER na ekranie wyświetla się informacja o poprawności rozwiązania. Jeżeli rozwiązanie było błędne to pada pytanie:

„CZY CHCESZ SPRÓBOWAĆ JESZCZE RAZ (T/N)?"

i dalsza akcja programu jest uzależniona od odpowiedzi użytkownika.

Wersja źródłowa segmentu głównego programu zajmującego 6 KB pamięci RAM jest następująca:

```
10 MODE 1:INK 0,24:INK 1,0:INK 2,6:INK 3,26:BORDER 0:PAPER 0:CLS
20 SYMBOL AFTER 126
30 SYMBOL 126,0,0,0,255,0,0,0,0
40 KEY DEF 15,1,126
50 PEN 1:LOCATE 1,3:PRINT "  Odnaleziono bardzo cenne, stare reko- pisy matematy
czne, ktore jednak wskutek niewlasciwego przechowywa
nia splowialy  tak dalece, ze tylko zarysy niektorych  cyfr mozna bylo odroznic.
"
60 WINDOW £1,3,36,10,15:PAPER £1,1:CLS £1:PEN £1,0
70 LOCATE £1,2,2:PRINT £1,"Cyfry nieczytelne oznaczone sa":PRINT £1," gwiazdkami
. Nalezy w ich miejsce":PRINT £1," wstawic pierwotni
e tam wpisane":PRINT £1," cyfry."
80 LOCATE 2,17:PRINT "KLAWISZE STERUJACE:"
90 PRINT:PRINT " do sterowania kursorem : ";CHR$(240);",";CHR$(241);",";CHR$(242
);",";CHR$(243)
100 PRINT " klawisz 'z' - wstawianie cyfry
110 PRINT " klawisz 'ENTER' - koniec rozwiazywania                    zadania"
120 LOCATE 5,24:PRINT "NACISNIJ DOWOLNY KLAWISZ"
130 CLEAR INPUT
140 IF INKEY$="" THEN GOTO 140
150 FOR k=1 TO 8
160 READ wym(k,1),wym(k,2),il(k):NEXT k
170 DATA 5,4,4,5,5,5,5,4,4,5,4,4,5,4,4,6,8,17,6,8,18,10,14,27
180 FOR k=1 TO 8
190 MODE 1:LOCATE 14,5:PRINT "ZADANIE";K
200 WINDOW £1,14,14+wym(k,1)+2,8,8+wym(k,2)+1:PAPER £1,1:CLS £1:PEN £1,0
210 WINDOW £2,1,40,1,3:PAPER £2,1:PEN £2,0:CLS £2
220 GOSUB 380
230 GOSUB 700
240 GOSUB 1020
250 IF BL=0 THEN GOTO 350
260 CLS £2:PRINT £2," TWOJE ROZWIAZANIE ZAWIERA BLEDY !!":INPUT £2," CHCESZ SPRO
BOWAC JESZCZE RAZ (T/N)";T$
270 IF T$="T" OR T$="t" THEN CLS £2:GOTO 230
280 IF T$="N" OR T$="n" THEN GOTO 300
290 GOTO 260
300 IF K=8 THEN GOTO 370
310 INPUT £2," NASTEPNE ZADANIE (T/N)";T$
320 IF T$="T" OR T$="t" THEN CLS £2:GOTO 360
330 IF T$="N" OR T$="n" THEN GOTO 370
340 GOTO 300
350 PRINT £2," TWOJE ROZWIAZANIE JEST POPRAWNE !!":GOTO 300
360 NEXT k
370 END
```

Do programu głównego jest dołączonych 5 podprogramów spełniających następujące funkcje:

1. Podprogram wczytywania danych do aktualnego zadania.

```
380 REM 'podprogram wczytywania danych do aktualnego zadania'
390 IF k>1 THEN ERASE tabl$:ERASE GWIA
400 DIM tabl$(wym(k,2),wym(k,1))
410 DIM GWIA(IL(K),3)
420 FOR i=1 TO wym(k,2)
430 FOR j=1 TO wym(k,1)
440 READ tabl$(i,j):LOCATE £1,1+j,1+i:PRINT £1,tabl$(i,j)
450 NEXT j
460 NEXT i
470 FOR i=1 TO il(k)
480 READ gwia(i,1),gwia(i,2),gwia(i,3)
490 NEXT i
500 RETURN
510 DATA " ","*","8","4","*","+","2","*","*","3"," ","~","~","~","~"," ","6","5"
,"2","9"
520 DATA 1,2,3,1,5,6,2,3,6,2,4,8
530 DATA " "," ","1","0","5"," ","*","*","1","7","+","5","8","*","*","~","~","~"
,"~","~","*","0","8","4","6"
540 DATA 2,2,4,2,3,9,3,4,2,3,5,4,5,1,1
550 DATA " ","5","*","1","7","+","*","4","*","8"," ","~","~","~","~"," ","6","8"
,"1","*"
560 DATA 1,3,3,2,2,1,2,4,9,4,5,5
570 DATA " ","8","*","*","7","-","*","3","5","*"," ","~","~","~","~"," ","6","1"
,"7","7"
580 DATA 1,3,5,1,4,2,2,2,2,2,5,0
590 DATA " ","3","*","5","7","-","*","9","8","*"," ","~","~","~","~"," "," ","4"
,"*","6"
600 DATA 1,3,4,2,2,2,2,5,1,4,4,7
610 DATA " "," "," ","*","*","*"," ",X," ","4","5","7"," ","~","~","~","~","~","
"," ","*","*","*","*"," ","1","7","0","5"," ","*","
*","*","*"," "," ","~","~","~","~","~","~","*","*","*","*","*","*"
620 DATA 1,4,3,1,5,4,1,6,1,4,3,2,4,4,3,4,5,8,4,6,7,6,1,1,6,2,3,6,3,6,6,4,4,8,1,1
,8,2,5,8,3,5,8,4,8,8,5,3,8,6,7
630 DATA " "," ","*","*","*","3"," ",X," ","*","*","*"," ","~","~","~","~","~","
"," ","*","7","0","*"," ","*","2","1","3"," ","*","
*","*","9"," "," ","~","~","~","~","~","~","*","*","*","*","*","*"
640 DATA 1,3,1,1,4,2,1,5,1,2,4,3,2,5,1,2,6,8,4,3,9,4,6,4,5,2,1,6,1,3,6,2,6,6,3,3
,8,1,3,8,2,8,8,3,5,8,4,7,8,5,3,8,6,4
650 DATA " "," ","*","*","8","*"," "," "," "," ","~","~","~","~","~","~"," "," "
," "," ","*","*","*","*","*","*",":","*","*","*","*"
,"*","*"," "," "," "," "," "," "," ","~","~","~"," "," "," "," "," "," "
660 DATA "3","0","2","*"," "," "," "," "," "," ","*","6","9","5"," "," "," "," "
," "," "
670 DATA "~","~","~","~"," "," "," "," "," "," "," ","*","*","1","1"," "," "," "
," "," "," ","3","*","8","*"," "," "," "," "," "
,"~","~","~","~"," "," "," "," "," "," "," ","2","*","1","*"," "," "," "," "," "
," ","*","*","*","*"," "," "," "," "
680 DATA " ","~","~","~","~"," "," "," "
690 DATA 1,3,1,1,4,7,1,6,6,3,1,6,3,2,8,3,3,7,3,4,6,3,5,1,3,6,0,3,8,3,3,9,8,3,10,
5,4,1,3,4,2,8,4,3,5,6,4,6,7,1,2,9,2,3,9,3,2,10,3,0,1
0,5,0,12,4,3,12,6,0,13,3,2,13,4,3,13,5,1,13,6,0
```

2. Podprogram sterowania kursorem na ekranie.

```
700 REM  'podprogram sterowania kursorem'
710 i=1:j=1
720 PAPER £1,3:PEN £1,2:LOCATE £1,j+1,i+1:PRINT £1,tabl$(i,j)
730 a$=INKEY$
740 IF a$="" THEN GOTO 730
750 IF INKEY(6)=0 THEN RETURN
760 IF a$="z" OR A$="Z" THEN GOTO 900
770 IF a$=CHR$(240) THEN GOTO 820
780 IF a$=CHR$(241) THEN GOTO 840
790 IF a$=CHR$(242) THEN GOTO 860
800 IF a$=CHR$(243) THEN GOTO 880
810 SOUND 1,458:GOTO 730
820 IF i=1 THEN GOTO 810
830 GOSUB 1200:i=i-1:GOTO 720
840 IF i=wym(k,2) THEN GOTO 810
850 GOSUB 1200:i=i+1:GOTO 720
860 IF j=1 THEN GOTO 810
870 GOSUB 1200:j=j-1:GOTO 720
880 IF j=wym(k,1) THEN GOTO 810
890 GOSUB 1200:j=j+1:GOTO 720
900 FOR V=1 TO IL(K)
910 IF J=GWIA(V,2) AND I=GWIA(V,1) THEN GOTO 940
920 NEXT V
930 GOSUB 1130:GOTO 730
940 LOCATE £2,2,2:PRINT £2,"WPISZ CYFRE"
950 CLEAR INPUT
960 A$=INKEY$
970 IF A$="" THEN GOTO 960
980 IF ASC(A$)>47 AND ASC(A$)<58 THEN GOTO 1000
990 GOSUB 1190:GOTO 940
1000 PAPER £1,0:PEN £1,1:LOCATE £1,J+1,I+1:PRINT £1,A$:TABL$(I,J)=A$
1010 CLS £2:GOTO 720
```

3. Podprogram badania poprawności rozwiązania.

```
1020 REM 'podprogram badania poprawnosci rozwiazania'
1030 bl=0
1040 FOR v=1 TO il(k)
1050 IF J=GWIA(v,2) AND I=GWIA(v,1) AND tabl$(i,j) <>"*" THEN PAPER £1,0:PEN £1,
1:LOCATE £1,j+1,i+1:PRINT £1,tabl$(i,j):GOTO 1080
1060 NEXT v
1070 PAPER £1,1:PEN £1,0:LOCATE £1,j+1,i+1:PRINT £1,tabl$(i,j)
1080 FOR V=1 TO IL(K)
1090 IF RIGHT$(STR$(GWIA(V,3)),1)=TABL$(GWIA(V,1),GWIA(V,2)) GOTO 1110
1100 BL=1
1110 NEXT V
1120 RETURN
```

4. Podprogram z komunikatami o błędach.

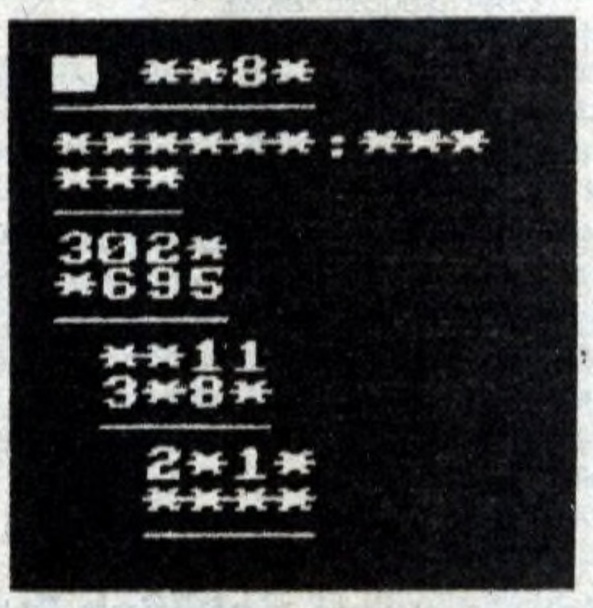
```
1130 REM  'podprogramy z komunikatami o bledach'
1140 CLS £2:LOCATE £2,2,2
1150 IF tabl$(i,j)<"1" OR tabl$(i,j)>"9" THEN GOTO 1170
1160 CLS £2:LOCATE £2,2,2:PRINT £2,"TEJ CYFRY NIE MOZNA ZMIENIAC":GOTO 1180
1170 CLS £2:LOCATE £2,2,2:PRINT £2,"TEJ POZYCJI NIE MOZNA ZMIENIAC"
1180 SOUND 1,458:FOR T=1 TO 1000:NEXT T:CLS £2:RETURN
1190 CLS £2:LOCATE £2,2,2:PRINT £2,"TO NIE JEST CYFRA":SOUND 1,458:FOR T=1 TO 10
00:NEXT T:CLS £2:RETURN
```

5. Podprogram badania pozycji cyfry.

```
1200 REM 'podprogram badania pozycji cyfry'
1210 FOR V=1 TO IL(K)
1220 IF J=GWIA(V,2) AND I=GWIA(V,1) AND tabl$(i,j) <>"*" THEN PAPER £1,0:PEN £1,
1:GOTO 1250
1230 NEXT V
1240 PAPER £1,1:PEN £1,0
1250 LOCATE £1,j+1,i+1:PRINT £1,tabl$(i,j):RETURN
```

Przykładowa postać jednego z zadań jest następująca:

ZADANIE 8



Życzymy Czytelnikom bezbłędnego rozwiązania wszystkich 8 zadań.

Danuta KWASIŻUR,
Mieczysław SKONIECZNY

POLSCRIPT — edytor tekstu z polskimi literami dla C-64

I. Wstęp

POLSCRIPT jest przeróbką programową znanego edytora tekstu EASY SCRIPT, który został opracowany w wersji pierwotnej przez Simona Tranmera i wypuszczony na rynek angielski przez firmę Precision Software Limited w roku 1982. Poniższy opis, bardzo skrótowy (oryginalny opis EASY SCRIPT zajmuje kilkadziesiąt stron), dotyczy zarówno wersji oryginalnej, jak i polskiej modyfikacji, ponieważ różnice między nimi są minimalne.

Autor zetknął się z przeróbkami programu dla drukarek MPS-801, MPS-802, MPS-803, z tym, że na co dzień używa wersji dla drukarki MPS-803, ponieważ pracuje z nią poprawnie posiadana drukarka (OKIDATA-120).

II. Uruchomienie programu

POLSCRIPT jet programem rezydującym i współpracuje równie dobrze z magnetofonem, jak i stacją dysków.

Zajmuje w pamięci komputera 73 bloki (Polscript 803). Po załadowaniu i uruchomieniu edytora komendą RUN ukazuje się menu główne, gdzie jesteśmy proszeni o podanie:

— szerokości tekstu (jest to szerokość tekstu na monitorze, najczęściej wynosi ona 40 znaków),

— rodzaju urządzenia (stacja dysków czy magnetofon),
— rodzaju drukarki (dla drukarek Commodore i zgodnych należy wybrać 0).

Następnie pytani jesteśmy o rodzaj liter (polskie, czy angielskie) i układ klawiatury (polski, czy angielski (QWERTY)).

Przechodzimy do trybu: „EDIT" — pisanie tekstu. Do dyspozycji mamy praktycznie cały ekran monitora. W górnej części ekranu (STATUS LINE) znajduje się informacja o trybie pracy, w jakim aktualnie znajdujemy się oraz o położeniu kursora (wiersz i kolumna). Zmianę kolorów ramki, tła i kursora uzyskuje się przez równoczesne naciśnięcie klawiszy CTRL i odpowiednio 3,2 i 1.

III. Pisanie tekstu

Każdy nowo tworzony dokument (pisany tekst) powinien zostać zatytułowany. W tym celu należy nacisnąć klawisz F3, napisać nb, podać w cudzysłowie nazwę dokumentu (pliku) i nacisnąć RETURN.

Klawisz F3 służy zawsze do zaznaczenia początku linii komend, w której podaje się między innymi, oddzielone od siebie dwukropkiem następujące komendy dotyczące formatu wydruku pisanego tekstu:

lmxx	— lewy margines (xx — nr kolumny),
rmxx	— prawy margines,
plxx	— długość strony,
tlxx	— długość tekstu,
spx(1,2,3)	— odstęp między liniami,
1nxx	— ilość pustych wierszy w tekście,
1pxx	— ilość wierszy na cal,
txx	— ilość liter na cal.

Każda komenda lub linia komend musi być zakończona klawiszem RETURN. Standardowe ustawienie powyższych parametrów wygląda następująco: lm=1, rm=80, pl=66, tl=60, lp=6, pt=10.

Klawisz RETURN służy do rozpoczynania nowego akapitu (paragrafu). Pojedyncze znaki usuwa się klawiszem DEL. Naciśnięcie CTRL i W powoduje przejście do pierwszej litery następnego słowa, a CTRL i „O" do ostatniej litery ostatniego słowa.

IV. Uzyskiwanie polskich liter

Polskie litery uzyskuje się następująco:
ą — średnik
ę — SHIFT + średnik
ś — SHIFT + przecinek
ń — SHIFT + kropka
ć — SHIFT + „/"
ó — „δ"
ź — SHIFT + „α"
ż — znak „+"
Ż — SHIFT + znak „+"
ł — dwukropek
Ł — SHIFT + dwukropek

Ulega również zmianie znaczenie niektórych klawiszy tj:
klawisz „gwiazdka" =„:"
SHIFT + „gwiazdka" = „;"
SHIFT + „—" =„+"
„strzałka pionowa" =„zł"
znak funta =„ϕ"
SHIFT + „6" =„?"

V. Znaczenie klawiszy funkcyjnych

Klawisz F1 służy do zmiany trybu pracy. Podstawowe kombinacje z klawiszem F1 to:
— F1+0+V — przeglądanie tekstu na monitorze w postaci, w jakiej będzie wydrukowany (do przesuwania tekstu służą klawisze kursora, klawisz Commodore, F5, F7, SPACJA.
— F1+0+P — drukowanie tekstu. Dodanie „C" powoduje wydruk ciągły. Przejście od „oglądania" strony na ekranie do wydruku następnej na drukarce uzyskuje się po naciśnięciu „P".
— F1+L — załadowanie pliku (dokumentu) do pamięci komputera (po podaniu nazwy i naciśnięciu RETURN),
— F1+F — zapisanie pliku (zamiast wpisywania nazwy pliku możemy ją skopiować, o ile plik został uprzednio zatytułowany za pomocą klawisza F2)
— F1+E+A — usunięcie całego kształtu tekstu,
— F1+E+R — usunięcie tekstu od pozycji kursora w dół,
— F1+E+P — usunięcie paragrafu,
— F1+E+S — usunięcia zdania,
— F1+D — usunięcie tekstu podświetlonego kursorem,
— F1+I — przejście do trybu pracy z rozsuwaniem tekstu (INSERT) lub odwrotnie,
— F1+G+E — przejście kursora na koniec tekstu,
— F1+G+xx — przejście kursora do linii xx,
— F1+SPACJA — wyświetlenie zawartości następnego ekranu,
— F1+SHIFT+SPACJA — wyświetlenie zawartości poprzedniego ekranu,
— F1+INST/DEL — usunięcie linii,
— F1+R+oświetlenie kursorem — definiowanie zakresu,
— F1+T+H lub V — tabulacja (pozioma i / lub pionowa),
F7 — przejście do następnej pozycji i tabulatora przy tabulacji poziomej,
F8 — jw., tylko przy tabulacji pionowej,
F1+C — kasowanie tabulacji,
F1+A — duplikacja tekstu (po zaznaczeniu zakresu (RANGE),
F1+S — wyszukiwanie tekstu i zastępowanie,
F5 — załączanie dużych liter,
F4 — przejście do trybu operacji dyskowych. W trybie tym istnieje możliwość między innymi;

a) wyświetlenia katalogu dyskietki na ekran lub jego wydrukowania (odpowiednio F4+„$" i F4+„+"+„$"+O+P,
b) załadowania dowolnego pliku z katalogu do pamięci komputera przez kombinację klawiszy: F4+„+"+„$"+L+F2,

VI. Inne możliwości programu

Omawiany program pozwala również między innymi na:
— formatowanie dyskietki,
— usuwanie, kopiowanie i zmianę nazwy plików,
— justyfikację i centrowanie tekstu,
— drukowanie w nagłówku i w stopce,
— automatyczną numerację stron,
— tworzenie długich dokumentów za pomocą wiązania plików (linked files) i obróbkę takich dokumentów (formatowanie, wydruk),
— tworzenie nowego pliku z zaznaczonych paragrafów,
— wydruk pogrubiony, podkreślony i odwrócony (reverse),
— resetowanie programu bez naruszenia tekstu znajdującego się w pamięci (RUN) STOP +RESTORE) lub z jego skasowaniem (F1+RUN) STOP).

Program umożliwia również definiowanie własnych znaków. Maksymalna wielkość tworzonego pliku wynosi 764 wiersze przy 40 kolumnach.

VII. Podsumowanie

Z przedstawionego powyżej, z konieczności skrótowego opisu edytora POLSCRIPT (EASY SCRIPT) wynika, że w zasadzie spełnia on wszystkie wymagania stawiane „poważnym", profesjonalnym programom tego typu. Niewątpliwą jego wielką zaletą jest „pisanie po polsku". Sam program nie jest może zbyt „przyjazny", bo komend koniecznych do zapamiętania jest dużo, ale z doświadczeń autora wynika, że jest to tylko kwestia ilości godzin, czy też stron napisanych za pomocą programu. Zaletą programu jest również możliwość współpracy z magnetofonem. Jedyną wadą programu odkrytą w ciągu kilku miesięcy jego użytkowania jest zniekształcanie liter po znakach „," i „—" przy stosowaniu automatycznej numeracji stron. Zjawisko to występowało zarówno podczas używania drukarki MPS-803, jak i OKIDATA -120. Nie potrafiąc sobie z tym problemem poradzić autor niniejszego artykułu zastosował „ręczne" numerowanie stron. A może ktoś z czytelników i użytkowników programu POLSCRIPT-803 wie, jak powyższą wadę usunąć?

Tadeusz CISEK

ZAGADKA

Martin CARR

Alma Cari nie była sobą, jej głos znany doskonale wszystkim mieszkańcom Dean Verry, drżał i łamał się, a twarz z nienaturalnym uśmiechem wyglądała niczym maska.

Frank Ctraks, który dopiero teraz włączył swój odbiornik zdążył usłyszeć tylko końcowy fragment komunikatu: — Dlaczego z Banku Myśli zginął mózg arcyzbrodniarza Citta Seana — trudno w tej chwili odpowiedzieć. Prezydent policji zapowiada, iż sprawą zajmie się osobiście.

Teraz dopiero Frank zrozumiał dlaczego został w trybie pilnym wyrwany ze swego urlopu. Jadąc do centrum starał sobie przypomnieć co wiedział o Citt Seanie. Tak... to musiał być niezwykły facet, jeżeli przez całe dziesięciolecia prowadził swoją prywatną wojnę z cywilizacją ziemską. Miał takie słowo na określenie sylwetki — anarchista niepodporządkowujący. Zwalczająca go policja kontynentalna była bezsilna. Jego dziełem było uprowadzenie najnowocześniejszego na owe czasy pojazdu kosmicznego i wymuszenie za zwrot od Rady Obywatelskiej wysokiego okupu — 10 weryklejsów czyli jego udziału w 10 ekspedycjach pozagalaktycznych. On też porwał prom kosmiczny na Eksporą Gardis i poleciał nim na wycieczkę w przestrzeń Oriona.

Kiedy wreszcie umarł, jego mózg trafił do gabinetów najwybitniejszych uczonych psychiatrów i neurologów. Ich ekspertyza była wyczerpująca, dopatrzyli się wielu odchyłek i nieprawidłowości w funkcjonowaniu. Nazwali go mózgiem arcyzbrodniarza i zalecili umieszczenie w Banku Myśli. Olbrzymi materiał badawczy, jaki zgromadzono, posłużył do opracowania nowej teorii, a zarazem nowej dziedziny nauki zwanej przez opracowujących mianem HMC. Rada Obywatelska wydała wtedy zezwolenie na wprowadzenie korekt u nowo narodzonych, jeżeli stwierdzono jeden z symptomów Citta. Dzięki temu przez całe wieki nie zdarzało się nic, co mogłoby źle wpłynąć na cywilizację ziemską, a życie toczyło się spokojnie.

Kiedy Frank trafił do centrum policji spostrzegł niezwykłe ożywienie. Na ekranach monitorów widać było, jak cymborgi i roboty funkcjonalnie uzbrojone w analizatory poszukiwawcze przeczesywały teren, dom po domu, rezydencja po rezydencji, biurowce i zakłady produkujące. Kiedy wyświetlił swój numer identyfikacyjny usłyszał: proszę iść niebieską drogą i natychmiast wyświetliły się niebieskie światełka. Ruszył. Szedł przez kilka kondygnacji, aż znalazł się przed gabinetem prezydenta policji. Tam na niego czekano, nie zdążył bowiem nawet nacisnąć guzika, kiedy drzwi się rozsunęły i wszedł do gabinetu. Na jego powitanie podniósł się ubrany w mundur policji miejskiej wysoki mężczyzna. Wskazał mu miękki fotel, sam podsunął wózek z napojami i zajął miejsce naprzeciw.

— To, co teraz powiem należy do najściślejszych tajemnic nie znanych ogółowi naszego miasta, przecieki mogą wywołać niepotrzebne napięcia. Dlatego proszę o zachowanie tego dla siebie. Jest pan najlepszym ekspertem w rozwiązywaniu niejasnych spraw. To że zaginął mózg arcyzbrodniarza to jedno, ale wiąże się z tym cała seria innych wydarzeń. Dziesięć dni po jego zaginięciu spłonął Bank Centralnych Danych, dwa dni po tym Centrala Rozdzielnictwa Dóbr. Przyczyn pożarów i sprawców nie ustalono. Poprosiliśmy wtedy o pomoc policję kontynentalną. Gdy awioloty z posiłkami lądowały w naszym mieście został obrabowany Bank Funduszów — zginęło 1000 regredów czyli środki na utrzymanie i zapewnienie wymogów życiowych dla 1000 naszych obywateli na jeden rok.

Nasza hipoteza jest taka: osoba lub grupa osób wykradła mózg i wbudowała go jakiemuś cymborgowi, po to, aby destabilizować sytuację w naszym mieście. Nie mamy tylko odpowiedzi — dlaczego? To, co się jeszcze może zdarzyć trudno przewidzieć. Prosząc pana o pomoc liczymy, że pan potrafi sprawę rozwiązać, ze swej strony udostępnimy wszystko czego będzie panu potrzeba.

Rozmowa była skończona i Frank postanowił wrócić do siebie, musiał głęboko się zastanowić, jak rozwiązać postawione przed nim zadanie.

Ale jeszcze tego dnia dowiedział się, że z oddalonego o dwadzieścia mil od miasta ośrodka terminku kosmicznego zginął ładunek trafiału, najnowszego paliwa do awiolotów B, używanych, jako niszczyciele kosmicznych meteorytów. Następnego dnia rano niezidentyfikowany pojazd z serii awiolotów B zniszczył prom kosmiczny z Lukseru przywożący ładunek platyny, dwa dni później ten sam los spotkał prom z Algera-1. Rada Ziemi podjęła wtedy decyzję wysłania eskadr ochronnych. Nie na wiele się to zdało, gdyż w niewyjaśnionych okolicznościach zaginął kolejny statek kosmiczny, tym razem z ładunkiem almogeru niezbędnego do produkcji walidu, materiału, z którego budowano pokrycia promów kosmicznych.

W czasie, kiedy wystąpiły te tajemnicze wydarzenia Frank pracował z największym napięciem. Za pomocą programów symulacyjnych sprawdzał własną hipotezę. Po kolejnej chyba dwudziestej serii wszystkie wydarzenia zaczęły układać się w jedną spójną całość.

Jak później, już po rozwiązaniu sprawy, mówił na konferencji prasowej nie był pewny, czy poszedł dobrą drogą. Najpierw bowiem udało mu się ustalić listę potomków Citta Seana. Później przebadał je jedną po drugiej i wtedy znalazł rozwiązanie. Okazało się, że tylko dwaj ludzie nie byli przebadani na HMC i to do tego właśnie Harry i Reed Seanowie. Motorem działania był Harry, którego Rada Lekarska z powodu obniżonego progu inteligencji skierowała na najniższy X poziom kierowania. Tam stykając się bezpośrednio z robotami i cymborgami rozpoczął wbudowywanie im układów posłuszeństwa wobec siebie. Mając wielkie aspiracje, a jednocześnie zdając sobie sprawę z tego, że nie będzie mógł być podniesiony na wyższy poziom, postanowił sam doprowadzić do własnego awansu. O co mu chodziło? Chciał zostać dyktatorem Ziemi. Kiedyś w młodości znalazł gdzieś kasetę magnetowidową z zapisem o dyktatorach. To mu się spodobało. A ponieważ jego umysł nie został skorelowany z potrzebami wobec cywilizacji, działał bez skrupułów. Miał swego pomocnika w bracie, który znajdując się w najwyższym pułapie kierowniczym poziomu A-1 dostarczał potrzebnych informacji. Jak wykazały badania był szantażowany i kilkakrotnie naruszono jego nietykalność osobistą. Roboty, które były w dyspozycji Harrego, kilkakrotnie niszczyły ulubione sprzęty Reeda i to wystarczyło.

Kradzież mózgu przodka miała dać legendę i zarazem natchnąć odwagą do działania.

Muszę zapewnić państwa, że obydwaj znajdują się w ośrodku zdrowia psychicznego i właśnie w tej chwili poddawani są zabiegowi dostosowania społecznego. Wyciągnęliśmy też daleko idące wnioski. Nigdy już tak nie będzie, aby ktoś nie był badany na HMC. Rada Obywatelska podjęła decyzję, aby certyfikat HMC był stale noszony na lewej ręce. To byłoby wszystko — zakończył.

GIEŁDA POMYSŁÓW

Szanowna Redakcjo!

Od pierwszych numerów „IKS-a" jestem jego stałym czytelnikiem. Posiadam mikrokomputer Atari 800XL, stację dysków LDW 2000 oraz drukarkę Atari 1025. Od kilku miesięcy układam programy w języku Basic, Pascal, Asembler. Jeden z nich chciałbym opublikować na łamach waszego pisma.

Program ten służy do robienia spektoramu pamięci komputera. Po wpisaniu go do pamięci komputera podajemy adresy, w zakresie których program ma robić spektoram. Program ten pracuje w trybie graficznym 8. Po zapisaniu całego ekranu następuje jego zamazanie i zarysowanie od początku. Program umożliwia wczytanie dowolnego programu z dysku do pamięci komputera i późniejsze wykonanie wykresu.

Myślę, że program ten przyniesie wiele satysfakcji w „badaniu" budowy programów. Ciekawe efekty można uzyskać w posiadaniu różnych programów np.:

Ciekawych efektów w zakresie spektografii komputerowej
— życzy autor.

Maciej STANUSCH

```
0 DIM NA$(14)
5 GOTO 125
10 SETCOLOR 2,0,0:? CHR$(125):POSITION 0,5
15 ? "PODAJ OD JAKIEGO ADRESU MAM TESTOWAC     PAMIEC ADR1=";:INPUT A
20 POSITION 0,10:? "PODAJ DO JAKIEGO ADRESU MAM TESTOWAC PAMIEC ADR2=";:INPUT B
25 GRAPHICS 8:SETCOLOR 2,0,0:COLOR 1:POKE 752,1
30 FOR Q=A TO B:X=X+1:IF X=320 THEN X=0:GRAPHICS 8:SETCOLOR 2,0,0:COLOR 1:POKE 7
52,1
35 PLOT X,150:DRAWTO X,150-PEEK(Q)*0.58823
40 IF PEEK(53279)=6 THEN RETURN
45 ? ,Q,PEEK(Q):NEXT Q
50 ? CHR$(125):? "GOTOWE...":? "NACISNIJ START"
55 IF PEEK(53279)<>6 THEN 55
60 RETURN
65 REM WCZYTANIE PROGRAMU Z DYSKU
70 GRAPHICS 0:SETCOLOR 2,0,0:POSITION 0,5:? "PODAJ NAZWE ZBIORU W POSTACI D:NAZW
A":POSITION 5,7:INPUT NA$
75 MEMLO=PEEK(14)+256*PEEK(15)
80 APPMHI=PEEK(741)+256*PEEK(742)
85 TRAP 105:Q=MEMLO
90 OPEN #1,4,128,NA$:POKE 559,0
95 GET #1,X:MEMLO=MEMLO+1:IF MEMLO=APPMHI THEN 105
100 POKE MEMLO,X:GOTO 95
105 GRAPHICS 0:SETCOLOR 2,0,0:CLOSE #1
110 ? "ZALADOWANO ";MEMLO-Q;" BAJTØW "
115 ? :? "OD ADRESU ";Q;" DO ADRESU ";MEMLO:?
120 GOSUB 15
125 REM MENU
130 X=0
135 GRAPHICS 0:SETCOLOR 2,0,0:COLOR 1:? :? :? :? :?
140 ? "1.TESTOWANIE PAMIECI":?
145 ? "2.WCZYTANIE PROGRAMU Z DYSKU":?
150 ? :? :? "PODAJ NR OPCJI I NACI$NIJ RETURN":INPUT SKOK
155 ON SKOK GOSUB 10,65
160 GOTO 125
```

Postscriptum do programu „Enigma"

W zamieszczonym programie „ENIGMA" (4/88 „IKS") nie ma błędu. Opis jego autor przedstawił może zbyt lakonicznie, ale wyraźnie napisał, że składa się z trzech części. Każdą z tych części należy wprowadzić na taśmę oddzielnie.

Pierwsza z nich, to fragment o etykietach 0, 1, 2, część druga rozpoczyna się od etykiety 3, a kończy na 11, zasadniczy fragment programu zaczyna się od etykiety 20. Każdą z części wprowadzamy kolejno na kasetę instrukcją CSAVE. Program pierwszej części ładujemy z taśmy przez wpisanie CLOAD. Gdy zostanie wprowadzony, program zatrzyma się — będzie oczekiwał na wpisanie instrukcji RUN. Gdy to nastąpi rozpocznie się wprowadzanie następnych części, które będą uruchamiały już automatycznie.

Janusz JANIEC

INFORMATYCZNY SŁOWNIK ANGIELSKO-POLSKI

FRAME TABLE - tablica w systemach operacyjnych ze stronicowym podziałem pamięci wirtualnej, ustalająca zgodność między stronami wirtualnymi a fizycznymi,
FRAMEWORK - plan, szkielet, struktura,
FRAMING - ramkowanie, obramowanie, także: ustawianie obrazu, synchronizacja obrazu,
FRAMING BIT - bit ramki,
FREE - wolny, swobodny, także: bezpłatny,
FREE-FORM - dowolnego formatu, dowolnej struktury,
FREE FORM CODING - kodowanie niepozycyjne,
FREE FROM ... - nie zawierający ..., pozbawiony ..., wolny od ...,
FREE FROM INTERFERENCE - niezakłócony,
FREE-HAND DRAWING - tryb rysowania swobodnego, tryb rysowania odręcznego,
FREE OF CHARGE - bezpłatny,
FREE OF ERROR - bezbłędny,
FREE OF TURN - bez czekania w kolejce, poza kolejnością,
FREE-RUNNING MODE - tryb swobodnego dostępu,
FREE SPACE - wolna pamięć, wolna przestrzeń pamięci,
FREE VARIABLE - zmienna swobodna,
FREE VECTOR - wektor swobodny,
FREEZING - zamrożenie,
FREQUENCY - częstotliwość,
FREQUENCY DIVISION MULTIPLEXING - zwielokrotnienie częstotliwości,
FREQUENCY MODULATION - modulacja częstotliwościowa,
FREQUENCY MODULATION RECORDING - zapis z modulacją częstotliwościową, zapis FM,
FREQUENCY SHIFT KEYING - kluczowana modulacja częstotliwości,
FRICTION FEED - podawanie papieru przy pomocy wałka,
FRIENDLY ENVIRONMENT - otoczenie sprzyjające, otoczenie dopasowane, dopasowanie,
FRIENDLY SOFTWARE - oprogramowanie "przyjazne", oprogramowanie dopasowane, oprogramowanie sprzyjające,
FROM ABOVE - z góry, od góry,
FRONT-END - interfejs zewnętrzny, także: procesor komunikacyjny, preprocesor,
FRONT-END COMPONENT - podsystem dostępu, także: podsystem wstępnego opracowania danych,
FRONT-END COMPUTER - komputer czołowy,
FRONT-END DATA PROCESSING - przetwarzanie danych czołowe,
FRONT-END INTERFACE - interfejs zewnętrzny,
FRONT-END PROCESSOR - procesor czołowy, specjalizowany procesor wprowadzania i wyprowadzania (danych), także: procesor komunikacyjny,
FS - patrz: FILE SEPARATOR,
FSK - patrz: FREQUENCY SHIFT KEYING,
FULL ADDER - sumator pełny,
FULL DUPLEX - pełny dupleks (jednoczesna komunikacja w obu kierunkach)
FULL-DUPLEX CHANNEL - kanał dupleksowy,
FULL PATHNAME - pełna (kompletna) ścieżka nazwy,
FULL-SCREEN EDITOR - edytor pełnoekranowy,
FULL STOP - znak kropki,
FULL WORD - słowo maszynowe,
FULLWORD - patrz: FULL WORD,
FULLY CONNECTED NETWORK - sieć w pełni połączona,
FULLY INVERTED FILE - plik (zbiór) całkowicie zmieniony (np. przez indeksację według innego indeksu),
FULLY STARTED JOB - zadanie w pełni uruchomione,
FUNCTION - funkcja,
FUNCTIONAL - funkcyjny, także: funkcjonalny, czynnościowy,
FUNCTIONAL CHARACTER - znak sterujący, znak funkcyjny,
FUNCTIONAL DEPENDENCE - zależność funkcyjna,
FUNCTIONAL DESIGN - projekt (schemat) funkcjonalny,
FUNCTIONAL DIAGRAM - patrz: FUNCTIONAL DESIGN,
FUNCTIONALITY - możliwości funkcjonalne,
FUNCTIONAL LANGUAGE - język funkcjonalny, język funkcjonalnego programowania,
FUNCTIONAL QUALITY - jakość użytkowa,
FUNCTIONAL UNIT - zespół funkcjonalny,
FUNCTION BUTTON - klawisz funkcyjny,
FUNCTION CALL - wywołanie funkcji,
FUNCTION GENERATOR - generator funkcji,
FUNCTION KEY - patrz: FUNCTION BUTTON,
FUNCTION KEYBOARD - klawiatura funkcyjna,
FUNCTION MENU - wykaz (spis) funkcji, menu,
FUNCTION MULTIPLIER - układ mnożący dla funkcji,
FUNCTION TABLE - tablica funkcji,
FUNCTOR - funktor, element logiczny,
FUNDAMENTAL - podstawowy,
FUNDAMENTAL FORMULA - wzór podstawowy,
FUNDAMENTAL TYPE - typ podstawowy (danych),
FURNISH - zaopatrywać, wyposażać,
FURNISH INFORMATION - dostarczać informacji,
FUSE BASE - gniazdo bezpiecznikowe,
FUZZY - postrzępiony, rozmyty, także: niedokładny, nieścisły,
FUZZY LOGIC - logika nieścisła, logika niedokładna (w systemach ekspertowych),

G

GAIN - wzrost, także: korzyść, wzmocnienie,
GAIN EXPERIENCE - zdobyć doświadczenie,
GAME - gra, także: grać,
GAME THEORY - teoria gier,
GAME TREE - drzewo gry,
GAMING - teoria gier użytkowych,
GAMIST - specjalista teorii gier,
GAMM-MIX - mieszanka GAMM,
GANG-PUNCH - masowe wprowadzanie do kart dziurkowanych identycznej treści,
GANGPUNCHING - kopiowanie wieloseryjne (kart dziurkowanych),
GAP - przerwa międzyblokowa, przerwa blokowa (np. między dwoma blokami na taśmie lub dysku magnetycznym), także: brak impulsu lub sygnału,
GAP DIGIT - cyfra techniczna,
GAPLESS - bezszczelinowy,
GAPLESS TAPE - taśma magnetyczna z ciągłym zapisem danych,
GAPPED TAPE - taśma magnetyczna z blokami danych,
GARBAGE - dane zbędne, dane pomyłkowe,
GARBAGE COLLECT - czyścić pamięć (usuwać zbędne dane),
GARBAGE COLLECTION - usuwanie danych zbędnych (np. z ciągu znaków),
GARBAGE COLLECTOR - program czyszczący pamięć (usuwający zbędne dane),
GARBAGE IN, GARBAGE OUT - wprowadzisz błędne dane, to uzyskasz błędne wyniki,
GASP - język informacyjnego modelowania,
GAS PANEL - monitor plazmowy (jarzeniowy),
GAS-PLASMA DISPLAY - monitor plazmowy,
GATE - bramka, element logiczny,
GATE CIRCUIT - układ bramkowy,
GATED - bramkowany,
GATEWAY - śluza w sieciach komputerowych (sprzętowe i programowe środki zabezpieczające łączność w sieci),
GATEWAY SERVER - stacja łączności z zewnętrzną siecią komputerową,
GATHER WRITE - pisanie rozproszone, wypis rozproszony (z pamięci),
GB - patrz: GIGABYTE,
GE - sieć komputerowa GE,
GE - patrz: GREATER OR EQUAL,
GEM - patrz: GRAPHICS ENVIRONMENT MANAGER,
GENERAL - ogólny,
GENERAL FLOW - ogólna sieć działań,
GENERALIZATION - uogólnienie,
GENERALIZE - uogólniać,
GENERALIZED DATA BASE - baza danych ogólnego przeznaczenia,
GENERALIZED FORMULA - wzór ogólny,
GENERALIZED ROUTINE - program ogólny, program uniwersalny,
GENERAL LOCAL NAME - ogólna nazwa lokalna,
GENERALLY ACCEPTED - powszechnie przyjęty,
GENERAL PROBLEM SOLVING PROGRAM - ogólny program rozwiązywania problemów,
GENERAL-PURPOSE - ogólnego zastosowania, uniwersalny, wielocelowy,
GENERAL-PURPOSE COMPUTER - komputer uniwersalny, komputer wielozadaniowy,
GENERAL PURPOSE INTERFACE BUS - wspólna magistrala wejścia/wyjścia,
GENERAL-PURPOSE REGISTER - rejestr wielozadaniowy,
GENERAL REGISTER - rejestr ogólnego przeznaczenia,
GENERATE - wytwarzać, wywoływać, generować,

GENERATED ADDRESS - adres generowany,
GENERATE (TO) - generować,
GENERATING PROGRAM - generator programów, program generujący,
GENERATION - generacja (np. komputerów), także: generowanie, wytwarzanie,

GENERATION NUMBER - numer generacji,
GENERATIVE - wytwarzający,
GENERIC - rodzajowy, uniwersalny, ogólny,
GENERIC DESCRIPTION - opis uogólniony,
GENERIC FUNCTION - funkcja ogólna,
GENERIC NAME - nazwa ogólna,
GENERIC OPERATION - operacja ogólna,
GENERIC PACKAGE - pakiet uniwersalny, pakiet z możliwością strojenia,
GENERIC PROCEDURE - procedura ogólna,
GENERIC TYPE - typ uniwersalny, typ ogólny, typ parametryzowany (pojęcie programowania dotyczące klasy typów danych),
GENUINE - autentyczny, prawdziwy,
GEOMETRIC(AL) - geometryczny,
GEOMETRICAL PROGRESSION - postęp geometryczny,
GEOMETRIC MEAN - średnia geometryczna,
GEOMETRY - geometria,
GET - otrzymywać, uzyskiwać, dostać, stać się, wpływać, także: powstać,
GET ACQUAINTED WITH ... - zapoznać się z ...,
GET WORSE - pogarszać się, psuć się,
GIBSON-MIX - mieszanka Gibsona,
GIGABYTE - gigabajt (1048576 bajtów),
GIGACYCLE - gigacykl (1048576 cykli),
GIGO - patrz: GARBAGE IN, GARBAGE OUT,
GIVE INFORMATION - dostarczać informację,
GIVEN QUANTITY - wielkość dana,
GIVE (PROFESSIONAL) ADVICE - konsultować,
GIVE SERVICES - świadczyć usługi,
GKS - patrz: GRAPHICS KERNEL SYSTEM,
GLASS TELETYPE - "szklany dalekopis" (wykorzystanie monitora dla sukcesywnego wprowadzania lub wyprowadzania linii tekstu bez użycia operacji ekranowych i sterowania kursorem),
GLOBAL - globalny, także: ogólnoświatowy,
GLOBAL IDENTIFIER - identyfikator globalny, oznacznik globalny,
GLOBAL OPTIMIZATION - globalna optymalizacja (optymalizacja programu na poziomie instrukcji i procedur), także: znajdywanie minimum lub maksimum funkcji celu,
GLOBAL VARIABLE - zmienna globalna,
GLOSSARY - słownik (wyjaśniający, specjalistyczny),
GLOW - jarzyć się, żarzyć się, świetlić,
GLUABILITY - sklejalność,
GLUE - kleić, sklejać (np. fragmenty tekstu w trakcie edycji),
GLUING - klejenie, sklejanie,
GO - idź, także: stawać się, zapalać się (o lampie),
GOAL DRIVEN - wnioskowanie wsteczne (w systemach ekspertowych),
GOAL FUNCTION - funkcja celu (w zadaniach optymalizacji),

GOAL-INVOKED INTERPRETATION - interpretacja od celu (proceduralna interpretacja zasady "jesli A to B"),
GOTO - przejście, przechodzenie, przekazanie sterowania,
GOTO STATEMENT - instrukcja "GOTO", instrukcja przejścia,
GOVERNING - regulacja, sterowanie,
GOVERNOR - regulator,
GPS - patrz: GENERAL PROBLEM SOLVING,
GRACEFUL DEGRADATION - "amortyzacja uszkodzeń", także: płynne obniżenie efektywności,
GRADUAL - stopniowy,
GRADUALLY - stopniowo,
GRADUATE - kończyć (np. szkołę, uczelnię),
GRAINED - ziarnisty,
GRAMMAR - gramatyka,
GRAMMATICAL - gramatyczny,
GRAMMATICS - patrz: GRAMMAR,
GRANDFATHER - grupa danych starsza o dwa pokolenia od danych rozpatrywanych,
GRAND TOTAL - suma całkowita, suma ogólna, suma łączna,
GRANULARITY - stopień detalizacji, także: granulacja,
GRAPH - wykres, graf,
GRAPH COLORING - kolorowanie grafu,
GRAPHEME - grafon (elementarna jednostka tekstu np. litera),
GRAPH FOLLOWER - urządzenie wprowadzania konturowych obrazów graficznych,
GRAPHIC - znak drukarski (np. litera, cyfra),
GRAPHICAL - wykreślny, graficzny,
GRAPHICAL DATA PROCESSING - przetwarzanie danych graficznych,
GRAPHICAL DEVICE - patrz: GRAPH PLOTTER,
GRAPHICAL DISPLAY UNIT - grafoskop, ekranopis graficzny,
GRAPHICAL KERNEL SYSTEM - system jądra graficznego,
GRAPHICAL LANGUAGE - język graficzny,
GRAPHICAL METHOD - metoda wykreślna,
GRAPHICAL OUTPUT PRIMITIVE - niepodzielny element obrazu (np. kropka, przecinek, myślnik, itp.),

GRAPHICAL SYMBOLS - symbole graficzne,
GRAPHIC ART - grafika,
GRAPHIC CHARACTER - znak graficzny,
GRAPHIC DATA - dane graficzne,
GRAPHIC DISPLAY - zobrazowanie graficzne,
GRAPHIC FORMAT - postać graficzna,
GRAPHIC FORMULA - wzór strukturalny,
GRAPHIC INPUT DEVICE - graficzne urządzenie wejściowe,
GRAPHIC INTERFACE - interfejs graficzny, środki graficznego współdziałania (oddziaływania),
GRAPHIC LANGUAGE - język graficzny,
GRAPHIC PALLET - paleta graficzna (zgodność pomiędzy kodami kolorów, a kolorami zobrazowanymi na ekranie monitora),
GRAPHIC PRIMITIVE - element obrazu (np. punkt, odcinek, znak alfanumeryczny, itp.),
GRAPHIC PROCESSOR - procesor graficzny,
GRAPHICS - grafika,
GRAPHIC SCHEDULE - harmonogram,
GRAPHICS CURSOR - kursor graficzny,
GRAPHICS DIGITIZER - urządzenie (cyfrowe) wprowadzania obrazów,
GRAPHICS EDITOR - edytor graficzny, edytor zobrazowania,

GRAPHICS ENVIRONMENT MANAGER - zarządzanie środowiskiem grafiki,
GRAPHICS KERNEL SYSTEM - bazowy system graficzny,
GRAPHICS MODE - tryb graficzny,
GRAPHICS PAD - plansza graficzna, rysownica graficzna,
GRAPHICS PROGRAMMING LANGUAGE - język graficzny,
GRAPHICS TERMINAL - terminal graficzny,
GRAPHIC TABLETT - rysownica graficzna (urządzenie do wprowadzania konturów obrazów, składające się z tabliczki oraz specjalnego pióra świetlnego), koordynatograf,
GRAPHIC VIEWPORT - wziernik graficzny,
GRAPH PLOTTER - urządzenie rysujące, pisak cyfrowy, ploter,
GRAPH THEORY - teoria grafów,
GRAVE ACCENT - znak akcentu,
GRAY CODE - kod Gray'a,
GRAY LEVEL - poziom jaskrawości (dla obrazów czarno-białych),
GRAY SCALE - skala jaskrawości,
GRAY-SCALE IMAGE - zobrazowanie półtonowe (z różną skalą jaskrawości),
GRAY SCALE SIGNAL - sygnał skali kontrastów,
GREATER OR EQUAL - większy lub równy,
GREATER-THAN SIGN - znak większości ("większy od"),
GREATER THEN - większy od,
GREAT MAJORITY - znaczna większość,
GROSS ERROR - gruby błąd,
GROSS INDEX - indeks główny, indeks podstawowy,
GROUND - uziemienie,
GROUND COLOUR - kolor tła,
GROUNDWORK - podłoże, baza, podstawa, fundament,
GROUP - grupa, zespół,
GROUP DELAY - opóźnienie grupowe,
GROUPED RECORDS - zapisy zgrupowane,
GROUPING - grupowanie,
GROUPING OF DATA - grupowanie danych,
GROUP SYMBOL - symbol grupy,
GROWING SYSTEM - system rozszerzalny, system dostosowany do rozbudowy,
GT - patrz: GREATER THEN,
GUARANTEE - gwarancja, poręczenie, także: gwarantować, zapewnić,
GUARANTEE CERTIFICATE - karta gwarancyjna,
GUARANTEED ACCURACY ALGORITHM - algorytm o zagwarantowanej dokładności,
GUARD - zabezpieczenie, osłona, dozór, także: osłaniać, zabezpieczać,
GUARD BIT - bit zabezpieczenia,
GUARD DIGIT - pozycje chronione (wyniku),
GUARDED COMMANDS - komendy zabezpieczone,
GUARD PLATE - płyta ochronna,
GUESS - domysł, przypuszczenie, próba,
GUIDANCE - wskazówka, wytyczne,
GUIDE - poradnik, także: prowadzić,
GUIDE BOOK - poradnik (książka), przewodnik (książka), informator,
GUIDE HOLE - dziurka prowadząca, dziurka przesuwu,
GUIDE ROLLERS - rolki prowadzące (np. taśmę magnetyczną),
GUIDE TO OPERATION - wprowadzenie do pracy,
GULP - grupa bajtów przetwarzana jako jedna całość,

LIGA MYŚLĄCYCH

ZADANIE 1

Dwa ciała poruszają się po okręgu w tym samym kierunku i spotykają się co 56 minut. Gdyby ciała te poruszały się z tymi samymi prędkościami w przeciwnych kierunkach, to spotykałyby się co 8 minut. Przy ruchu w przeciwnych kierunkach odległość (mierząc po okręgu) między zbliżającymi się ciałami zmniejszyłaby się z 40 metrów do 26 metrów w ciągu 24 sekund. Ile metrów na minutę przebywa każde ciało oraz jaki jest obwód okręgu, po którym poruszają się oba ciała?

ZADANIE 2

Naczynie o pojemności 8 litrów napełniono powietrzem zawierającym 16% tlenu. Z naczynia tego odprowadzono pewną ilość powietrza i doprowadzono taką samą ilość azotu, po czym ponownie odprowadzono taką samą (jak za pierwszym razem) ilość mieszaniny i ponownie doprowadzono taką samą ilość azotu. W powstałej mieszaninie było 9% tlenu. Wyznaczyć po ile litrów gazu odprowadzono i doprowadzono za każdym razem z naczynia.

ZADANIE 3

Barka o ładunku 600 ton była wyładowana w ciągu 3 dni, przy czym pierwszego i trzeciego dnia wyładowano 2/3 całego ładunku. W drugim dniu wyładowano mniej, niż w pierwszym, a w trzecim mniej niż w drugim, przy czym różnica między procentowym zmniejszeniem wyładunku trzeciego dnia w stosunku do wyładunku drugiego dnia i procentowym zmniejszeniem wyładunku drugiego dnia w stosunku do wyładunku pierwszego dnia była równa 5. Ile ton ładunku wyładowano każdego dnia?

ZADANIE 4

Należy wyznaczyć kąt rozwarcia stożka, opisanego dookoła czterech równych kul ułożonych w ten sposób, że każda z nich jest styczna do trzech pozostałych.

ZADANIE 5

Z całej ilości towaru a % sprzedano z zyskiem p %, zaś b % sprzedano z zyskiem q %. Jaki był zysk sprzedaży pozostałej części towaru, jeżeli ogólny procent zysku wyniósł r %?

Rozwiązania zadań prosimy przesyłać do redakcji do końca grudnia br., z dopiskiem „Liga Myślących". Punktacja zależy od liczby prawidłowych rozwiązań. Wśród uczestników rozlosujemy książki, a na zwycięzców „Ligi" czekają dodatkowe nagrody.

PROGRAM 63 ATARI

Trałowanie

Znane przysłowie wojskowe mówi, że saper myli się tylko raz. Dotyczy ono również i załogi trałowców. Pewną różnicą charakteryzującą ich zadania jest to, że miny na morzu mogą zmieniać swoje położenie. Potrzeba dużo czasu i cierpliwości, by je unieszkodliwić. Do pomocy mamy urządzenia wykrywające. Po każdym ruchu meldują nam, w którym kierunku mamy poruszać się, aby dotrzeć do miny np. miny z lewej. Okrętem sterujemy za pomocą klawiszy: L — lewo, P — prawo, N — naprzód, W — wstecz.

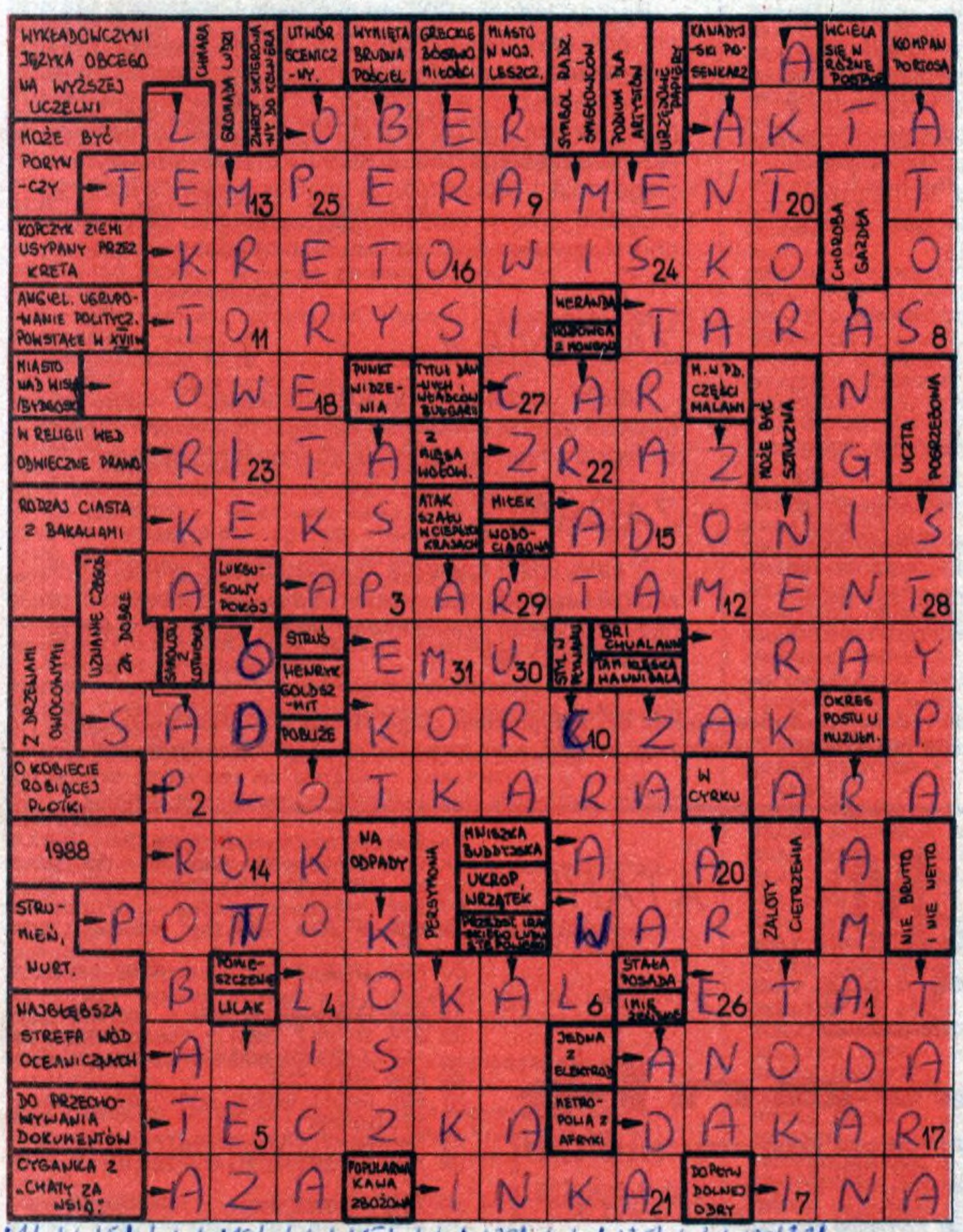
Jeśli na otrzymany meldunek zareagujemy prawidłowo, to zostaniemy powiadomieni, że miny są bliżej. Po kilku ruchach minę należy zdetonować. Gdy to ci się nie uda, uważaj się za jednego z tych pechowców, którzy pomylili się ten jeden raz.

```
PG 500 REM ************************
YA 502 REM *                      *
HY 504 REM *  T R A L O W A N I E *
GS 505 REM *     *Atari*          *
YP 507 REM *                      *
QH 509 REM ************************
MK 510 DIM T$(65),TS$(8),A$(1),B$(1
   )
VG 515 DATA Z LEWEJ ,Z PRAWEJ,Z PRZ
   ODU
QX 516 DATA Z TYLU  ,W LEWO  ,W PRA
   WO ,NAPRZOD ,WSTECZ  ,K
CI 520 RESTORE
MF 525 FOR K=1 TO 9:READ TS$
FK 530 T$(LEN(T$)+1)=TS$:NEXT K
BR 535 ? CHR$(125)
FB 537 ? "T R A L O W A N I E"
GA 539 EP=3:GOSUB 800
LL 541 M=5+INT(RND(0)*6):KA=0:N=M
XB 545 FOR K=1 TO N:A$="":B$="":OD=
   INT(RND(0)*3)+3
TA 548 P=0:R=INT(RND(0)*4)+1
DV 552 ? "M I N Y ! ";:RR=R*8:? T$(
   RR-7,RR)
EC 555 GOSUB 800:A=P:IF R>2 THEN 61
   0
TB 560 ? T$(33,40);"/";T$(41,48);"=
   ";
QR 565 ? "L/P";:INPUT A$
MK 570 IF (A$="L" AND R=1) OR (A$="
   P" AND R=2) THEN P=P+1:GOTO 650
PQ 580 GOTO 640
DV 610 ? T$(49,56);"/";T$(57,64);"=
   ";
TX 620 ? "N/W";:INPUT B$
CK 630 IF (B$="N" AND R=3) OR (B$="
   W" AND R=4) THEN P=P+1:GOTO 650
UJ 640 P=P-1
AW 650 IF P>A THEN ? " MINY BLIZEJ
   !"
OJ 660 IF P<A THEN ? " MINY DALEJ !
   "
FN 665 EP=2:GOSUB 800
RC 670 IF P>2 THEN ? "DETONUJESZ ?"
   :EP=3:GOSUB 800:GOTO 730
QQ 680 GOTO 552
NY 690 Q=Q+1:GOSUB 830:? Q;" MINA Z
   NISZCZONA"
XJ 700 NEXT K:WYNIK=INT((N*100)/M)
GR 705 ? "ZNISZCZONO ";WYNIK;" % MI
   N"
DV 710 ? "TRALUJEMY T/N";:INPUT A$
JS 715 IF A$="N" THEN ? CHR$(125):E
   ND
YK 720 Q=0:GOTO 535
LH 730 ? "      T/N ";:INPUT A$
CV 735 IF A$="T" AND P=OD THEN 690
ZN 740 IF P=OD THEN ? "WPADLES NA M
   INE !":GOSUB 830:END
AF 745 IF A$="T" AND P<OD THEN ? "P
   UNKT KARNY !":KA=KA+1
XG 746 IF KA=3 THEN N=N-1:KA=0
QL 750 GOTO 552
RI 800 FOR BE=1 TO EP:FOR G=10 TO 0
    STEP -1
WK 805 SOUND 0,100,10,G:NEXT G:NEXT
    BE
ZG 810 RETURN
GO 830 FOR S=20 TO 240 STEP 3:SOUND
    0,S,4,8:NEXT S:SOUND 0,240,0,14
DF 840 FOR S=0 TO 40:NEXT S
IJ 850 FOR S=15 TO 0 STEP -1:POKE 7
   06,62:SOUND 0,250,4,S:SOUND 2,24
   0,0,S:NEXT S
ZQ 860 RETURN
```

Janusz JANIEC

** W algorytmie programu wykorzystano pomysł J. Millera*

KRZYŻÓWKA NR 8



Litery z kratek ponumerowanych od 1 do 31 utworzą nazwy 5 komputerów, które wystarczy podać jako rozwiązanie zadania, przesyłając pod adresem redakcji na kartach pocztowych, w terminie do końca grudnia, naklejając kupon „IKS-a". Wśród autorów prawidłowych odpowiedzi rozlosujemy bony pieniężne i nagrody książkowe.

GRAFIKA KOMPUTEROWA

Jedną z najbardziej dynamicznie rozwijających się dziedzin zastosowania maszyn cyfrowych jest grafika komputerowa. Jeszcze do niedawna była ona domeną szybkich komputerów profesjonalnych. Obecnie dzięki zastosowaniu nowoczesnych technologii mogą się nią posługiwać posiadacze współczesnych komputerów osobistych.

Z tą myślą ciekawe i tanie rozwiązanie zaproponowała angielska firma Digithrust Ltd. Oferuje ona zestaw składający się z pakietu programów i pełnowymiarowej karty rozszerzającej do komputerów zgodnych z IBM PC AT, która pozwala połączyć komputer z magnetowidem lub kamerą wideo. Używając tego zestawu można zapamiętać obraz w postaci cyfrowej — na monitorze RGV z rozdzielczością 640 na 400 punktów. Specjalny algorytm pozwala uzyskać efekt płynnego przejścia między sąsiednimi kolorami i uzyskać ten sam obraz na monitorze i karcie EGA. Obraz można zapamiętać na nośniku magnetycznym w postaci akceptowanej przez programy graficzne PC Paint, GEM Paint i Ventura Publisher. Funkcja „format" określa, w której części ekranu ma pojawić się przetworzona grafika. Aby przyspieszyć operacje ustawienia kierunku, ostrości i przysłony kamery przewidziano możliwość przyjęcia przez komputer obrazu czarno-białego. Grafika komputerowa jest narzędziem o niekwestionowanej już dziś przydatności w pracy architektów, inżynierów czy lekarzy. Dlatego coraz więcej firm oferuje dodatkowe karty graficzne, które zainstalowane wewnątrz komputera znacznie rozszerzają jego możliwości i to pod względem rozdzielczości obrazu, jak i szybkości realizowania skomplikowanych operacji ekranowych.

Nowością w tej dziedzinie jest karta VTP amerykańskiej firmy Verticom. Jej przeznaczeniem jest monochromatyczny tryb 1280 na 960 punktów, chociaż może też pracować w trybie CGA. W karcie zastosowano najnowszy procesor graficzny INTEL 82786 — o nie wykorzystanych do końca możliwościach — taktowany z częstotliwością 20 MHz, 512 KB pamięci RAM i 8 KB ROM. VTP współpracuje z szeroko rozpowszechnionymi programami, na przykład Microsoft Windows, Auto CAD ADI, GEM, Ventura Publisher. Zainstalowanie takiej karty porównywalne jest ze wstawieniem drugiego komputera do jednej obudowy.

(RAJ)

„IKS" — dodatek „Żołnierza Wolności". Redaguje Wiesław Cetera (kierownik zespołu); Rada programowa: Krzysztof Chmarra, Romuald Głąb, Włodzimierz Gogołek, Janusz Janiec, Henryk Krasuski, Ireneusz Miernik, Ludwik Piela, Jacek Szaniawski. Adres redakcji: 00-950 Warszawa ul. Grzybowska 77, telefon centrali 20-12-61 w. 486. Telex 313664. Rękopisów nie zamówionych redakcja nie zwraca i zastrzega sobie prawo do skrótów. Nakładem: Wydawnictwa „Czasopisma Wojskowe", Warszawa ul. Grzybowska 77. Fotoskład i druk offsetowy — Wojskowe Zakłady Graficzne im. gen. dyw. A. Zawadzkiego. Nr zam. 1811. Nr ind. 361682. U-11