IKS

INFORMATYKA KOMPUTERY SYSTEMY

CENA 120 ZŁ

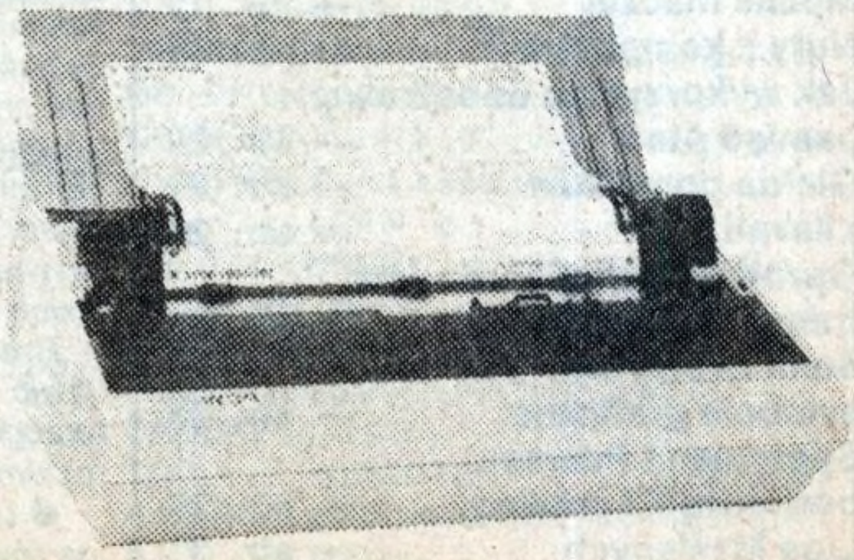
DODATEK DO „ŻOŁNIERZA WOLNOŚCI Nr 6/1988 ISSN 0860 — 2794

Komputer dla muzyka
— czyli komponować może każdy
„Dodatkową" pamięć Atari
odkrywamy na stronie 20

AMSTRAD 9512
RAM — 512 KB
System operacyjny CP/M+

AMSTRAD 1640
RAM — 640 KB
Mikroprocesor 8086 (8 Mhz)

EGA, Hercules,
MDA lub CGA



Zacznijmy nietypowo, bo od cytatu z rządowego „PROGRAMU ROZWOJU NAUKI I TECHNIKI NA LATA 1986—90" będącego załącznikiem do obecnie realizowanego Narodowego Planu Społeczno-Gospodarczego. Na str. 9 tego dokumentu sformułowano następujące zadanie związane bezpośrednio z informatyką:

„Wykorzystując wielkie tradycje polskiej szkoły matematycznej, zostanie w latach 1986—90 stworzony i rozwinięty przemysł oprogramowania, który może i powinien stać się naszą specjalnością eksportową. Za zadanie szczególnej wagi przyjmuje się także opracowanie i wdrożenie do produkcji seryjnej urządzeń komputerowych do wspomagania polskiego naukowca i inżyniera projektanta".

Czytając treść tego zadania można się niewątpliwie zgodzić, że oba wymienione tam cele są ambitne. W sytuacji, gdy na polskim rynku mini- i mikrokomputerowym dominuje oprogramowanie kradzione, z niekompetentną i często na kolanie sporządzoną dokumentacją, a oprogramowanie dużych i średnich komputerów cierpi na uwiąd starczy [1], rzetelna próba osiągnięcia tych celów spotkałaby się z rzeczywistym poparciem wielu ambitnych informatyków. Nic w tym dziwnego, przecież wielu z nas zdaje sobie coraz wyraźniej sprawę z jałowości pobieżnego rozpoznawania i „oswajania" mniej lub bardziej przypadkowo pozyskiwanego [2] z najdziwniejszych źródeł oprogramowania. Każdy może podać przykłady absolwentów najlepszych uczelni, którzy w poszukiwaniu pieniędzy lutują pakiety, w zdobycznych programach wymieniają komunikaty w języku angielskim na polskie, zajmują się sprowadzaniem, składaniem i sprzedażą dalekowschodniego sprzętu informatycznego. W efekcie tracą wiele ze swoich zdolności twórczych, często bezpowrotnie. Oczywiście takie działanie z ich punktu widzenia jest racjonalne i słuszne. Nie oznacza to jednak, że jest obiektywnie korzystne. Stąd wyżej sformułowane zamierzenie rządowe mogło i powinno być odczytane jako chęć stworzenia przez państwo szansy na nowe i nowoczesne wyposażenie miejsca pracy projektantów i programistów, na pracę planową, zorganizowaną i koncepcyjnie trudną, na przyciągającą najlepszych, możliwość uczestniczenia w międzynarodowym podziale informatycznego rynku pracy. Zgadzając się z tym trzeba uczciwie przyznać, że nie jest to zadanie łatwe. Po pierwsze dlatego, że opóźnienie w produkcji elementów i podzespołów elektronicznych oraz rodzimego sprzętu informatycznego tzn. sprzętu wytwarzanego z krajowych elementów, jest bardzo znaczne i wynosi około 10 lat.

W sytuacji, gdy cały wysoko i średnio rozwinięty świat uporał się z seryjną produkcją komputerów 32-bitowych nasz przemysł państwowy nadal nie jest w stanie wytworzyć elementów do produkcji całkowicie polskiego mikrokomputera 16-bitowego. Nie znaczy to, że domagam się pełnej samowystarczalności, odcięcia się od wyników światowych i namawiam do hołdowania zasadzie „każdy sobie..." Wręcz przeciwnie, jestem za aktywnym udziałem w międzynarodowym rynku produkcji i wymiany. Jednakże uważam, że z ogólnocywilizacyjnych względów kraj tej wielkości co Polska, powinien być w stanie samodzielnie produkować komputery określonych klas. Mikrokomputery 16-bitowe z pewnością do nich należą. Drugą ważną barierą jest to, że realizacja wspomnianego zadania przypadła na lata kryzysowe, których nieodłącznym atrybutem jest ciągły i dotkliwy brak wolnych środków finansowych (złotówkowych i dewizowych). Niezależnie od tych faktów można chyba sądzić, że autorom tego zapisu nie chodziło o spektakularne rozwiązanie. Miejmy nadzieję, że słusznie uznali, korzystając z doświadczeń innych krajów, że rozwój informatyki może być jednym ze sposobów przełamywania tendencji kryzysowych, środki i metody informatyki mogą nie tylko pośrednio wpływać na ogólny postęp gospodarczy, dzięki przyspieszeniu i zracjonalizowaniu procesu podejmowania decyzji, zmniejszaniu energochłonności procesów technologicznych, podniesieniu poziomu niezawodności i nowoczesności tysięcy urządzeń. Mogą być również przedmiotem bezpośredniego eksportu. Dotyczy to przede wszystkim, z uwagi na zły stan bazy elementowej, oprogramowania i usług konsultacyjno-serwisowych. Aby się tak stało trzeba odejść od tradycyjnego pojmowania przemysłu informatycznego i objąć nim również wytwarzanie oprogramowania sprzętu informatycznego. W tym miejscu zgadzam się całkowicie z tezą, że właśnie wytwarzanie oprogramowania jest w polskich warunkach szansą dla informatyki.

Rzeczywiste uznanie tego faktu musi przejawić się w potraktowaniu wytwarzania oprogramowania w kategoriach prawdziwego przemysłu. W szczególności należałoby:

- *uznać wytwarzanie oprogramowania za równorzędne produkcji sprzętu informatycznego,*
- *stworzyć normy chroniące prawa autorskie twórców oprogramowania i regulujące obrót oprogramowaniem komputerów,*
- *utworzyć ośrodki produkcyjne (np. na zasadzie spółek z kapitałem zagranicznym) produkcyjno-wdrożeniowe, dysponujące nowoczesną (być może unikalną w kraju) techniką komputerową, umożliwiającą wykonywanie oprogramowania na eksport,*
- *utworzyć agendy handlowe wyspecjalizowane w promocji i handlu oprogramowaniem.*

Pobieżna choćby analiza stanu informatyki krajowej prowadzi do wniosku, że mimo już prawie dwuletniej realizacji zacytowanego na wstępie zadania przesłanki powstania polskiego przemysłu wytwarzania oprogramowania nie powstały. Nadal jesteśmy bliżej **chałupnictwa** *niż przemysłu. W tej chwili nie ma samodzielnego wyniku oprogramowania komputerów. W dalszym ciągu jest ono dodatkiem do produkowanego sprzętu informatycznego. Nie istnieje ani jedna poważna firma nastawiona na produkcję oprogramowania, nie powstał żaden naprawdę duży produkt oprogramowania podstawowego. [3]. Tymczasem na świecie istnieją i znakomicie prosperują tysiące dużych i małych firm, trudniących się wyłącznie produkcją oprogramowania.*

Nie produkujemy obecnie, poza nielicznymi wyjątkami, konkurencyjnego sprzętu. Na szczęście niektóre uczelnie zapewniają dopływ bardzo dobrych programistów i projektantów. Trzeba to wykorzystać jak najprędzej. W tym miejscu chciałbym wreszcie wyjaśnić powody, dla których zdecydowałem się napisać ten odbiegający od ściśle informatycznych rozważań, felieton właśnie do „IKSa". Przede wszystkim chcę upowszechnić informację o tak sformułowanych zamierzeniach rządowych. Niezależnie od osiąganych postępów jest to dla informatyków fakt ważny. Po drugie uważam, że właśnie tysiące specjalistów i hobbystów informatyki stworzą wspólnie masę krytyczną, która doprowadzi do powstania nowoczesnego prawa chroniącego interesy wytwórców programów. Wreszcie po trzecie i najważniejsze: warto zawsze pamiętać, że informatyka jest jedną z najdynamiczniej rozwijających się dziedzin nauki i gospodarki. Dlatego bawiąc się lub profesjonalnie wykorzystując swoje mikro trzeba równolegle rozwijać głębszą wiedzę informatyczną, nadążać za nowymi technologiami organizacji działania i wykorzystania komputerów. Jednym słowem mając ATARI, czy COMMODORE, a w pracy wykorzystując IBM PC dobrze jest wiedzieć, jak działa i w jakich konfiguracjach jest wykorzystywany np. VAX. Wszystko po to, by z chwilą faktycznej budowy przemysłu wytwarzania oprogramowania być gotowym do podjęcia zadań wysokopłatnych i jednocześnie twórczych.

Bolesław SZAFRAŃSKI

---

[1] *Potwierdzeniem niech będzie to, że ELWRO wraz z komputerem RtA 34 oferuje jedyny SZBD MADES, którego pierwowzór skonstruowano na początku lat siedemdziesiątych.*

[2] *Można wg uznania wstawić: ukradzionego, załatwionego, przyniesionego,*

[3] *Jedynym chwalebnym wyjątkiem jest chyba SZBD RODAN, sprzedany przez ZOWAR do wielu krajów.*

W NUMERZE:

Gdy stwierdzisz, że gry komputerowe są do siebie podobne i coraz mniej atrakcyjne, że ZX Spectrum jako książka telefoniczna okaże się mniej użyteczny od tradycyjnej książki, a programowanie wyda się jednak nudne i mało ciekawe — wraca pytanie sprzed lat (jak ten czas leci!): jak można ciekawie wykorzystać personalny komputer w domu?

SIECI

Wydaje się, że odpowiedzią na to pytanie jest podłączenie komputera do sieci telefonicznej.

W warunkach europejskich jest to przedsięwzięcie stosunkowo tanie i nie wymagające specjalnego przygotowania, np. wyposażenie naszego wysłużonego „Spectrum" w port RS 232 oraz modem kosztuje niecałe 100 funtów angielskich. W efekcie stajemy się posiadaczami terminala domowego, który może połączyć nasz komputer z każdym zainstalowanym choćby w Australii. Zasadność takiego połączenia wynika z usług, jakie są dostępne. Na przykład w Wielkiej Brytanii, komputer przez sieć telefoniczną pozwala na szybki dostęp do aktualnych danych o rozrywkach (kina, teatry), pracy, szkole, hobby, sporcie oraz o wielu innych bardzo ciekawych rzeczach.

Aby komputer stał się „oknem na świat" niezbędne jest posiadanie następujących elementów:

1 komputer personalny,
2 szeregowy port RS 232,
3 komunikacyjne oprogramowanie,
4 modem,
5 znajomość numerów telefonicznych, pod które podłączone są komputery udostępniające poszukiwane przez nas informacje.

Jak już wspomniałem, każdy komputer może być podłączony do sieci telefonicznej (nawet Spectrum lub Commodore), a korzystanie z tego typu sieci nie jest drogie — cena nieco mniejsza od rozmowy telefonicznej plus ewentualna opłata, gdy korzystamy z komercyjnych sieci komputerowych (dodatkowy abonament).

Po połączeniu z innym komputerem odnosimy wrażenie, że „komputer rozmówca" jest częścią naszego komputera. Fakt, że druty łączące nasze dwa komputery mają długość np.: 20 000 km, jest bez znaczenia dla procesu przesyłania informacji. Przebiega on podobnie jak wówczas, gdy komputery stoją obok siebie w tym samym budynku. Nie jest ważny także typ naszego komputera, działa on jako terminal współpracujący z dowolnym innym komputerem (tutaj mezaliansów nie ma). W efekcie pozwala to nam korzystać z mocy obliczeniowej innych maszyn, „przenoszenia" tej mocy do naszego komputera.

Podstawowym elementem warunkującym realizację tego typu połączenia jest modem — małe urządzenie, które pośredniczy w przesyłaniu informacji między komputerem, a siecią telefoniczną. Modem zamienia wyjściowy sygnał komputera na sygnał akustyczny — postać sygnału, która może być przesyłana drutami telefonicznymi. Należy zatem włączyć modem do gniazdka telefonicznego, wykręcić odpowiedni numer, po usłyszeniu charakterystycznego sygnału adresata (modem przy drugim komputerze) należy nacisnąć odpowiedni klawisz w modemie — połączenie zostało wykonane — komputery mogą sobie „pogadać". Obecnie czynności te, dzięki odpowiedniemu oprogramowaniu, wykonuje sam komputer. „Samodzielność" ta wynika także z coraz doskonalszych konstrukcji modemów. Ma to miejsce na przykład podczas stosowania modemu HAYESa, który uzyskał w Polsce prawo do instalowania w komputerach korzystających z krajowej sieci telefonicznej.

Jak już wspomniałem, komunikacja komputerów połączonych siecią telefoniczną odbywa się za pośrednictwem wysyłanych drutami różnej wysokości tonów. W efekcie pisanie komend na naszej klawiaturze daje efekt w komputerze, z którym „rozmawiamy". Odpowiedzi komputera, do którego dzwoniliśmy, ukazują się na ekranie naszego mikro, tak jak gdyby generowane były przez nasz komputer.

W tym miejscu może pojawić się wątpliwość — po co modem, komputer, skoro można zadzwonić i po prostu pogadać z osobą, której chcemy przekazać określoną informację. Odpowiedź jest prosta — mówiąc „przesyłamy" 29—30 liter w czasie jednej sekundy (zakładając, że mamy dobrą dykcję), komputery „gadają" z szybkością 1200 liter na sekundę. Dokonując pewnych rachunków, łatwo dojść do wniosku, że proponowany sposób przesyłania informacji jest znacznie tańszy. Szczególnie wyraźnie jest to widoczne przy porównywaniu kosztów „przesyłania znaków" przez ocean. Wykorzystuje się tutaj tak zwane paczki informacji. Komputer gromadzi określoną liczbę znaków i potem szybko je przesyła na przykład za pośrednictwem sputnika. Usługi takie oferuje Bell System Radio Link, który przesyła informacje z szybkością 1,9 biliona bitów na sekundę. Przesłanie naszej encyklopedii trwałoby około jednej setnej sekundy!

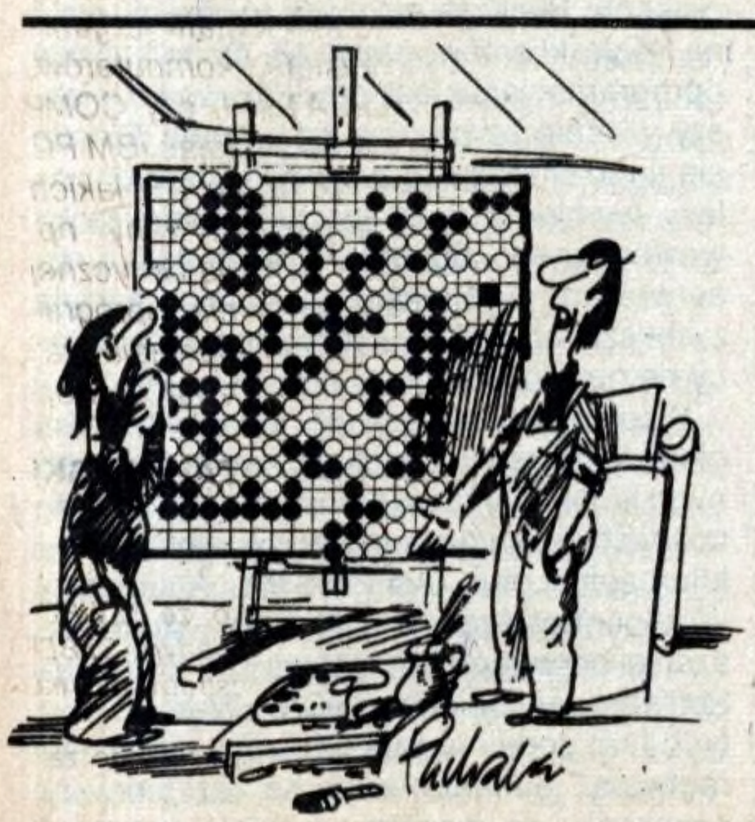
Wykorzystanie możliwości dołączania komputerów do sieci telefonicznej nie wymaga oczywiście umiejętności programowania. Dostępne są bowiem gotowe, tak zwane programy komunikacyjne.

Obecnie na świecie dostępne są tysiące baz danych. Przykładami są: The Source w USA i Micronet w Wielkiej Brytanii. Bazy te zawierają setki darmowych programów, gier, informacji, wiadomości i wiele innych użytecznych danych, np. słowniki, encyklopedie.

Brytyjski system biblioteczny BLAISE udostępnia ewidencję wszystkich książek, periodyków, broszur itp. Dane te mogą być wybierane poprzez tytuł, temat, autora, wydawcę lub podając szereg innych parametrów.

Korzystanie z komputera w domu staje się wielką przygodą, zabawą o „światowym wymiarze". Sądzę, że nie należy popadać w kompleksy, iż technika ta jest bardzo odległa od naszej domowej rzeczywistości. Została bowiem uruchomiona także u nas w kraju sieć komputerowa (poprzez łącza telefoniczne — nie są takie złe!). Obecnie „gromadzi" ona użytkowników (jednym z nich jest piszący te słowa), a jej promotorem jest między innymi nasza konkurencja KOMPUTER, w którym FIDO — owa sieć — wielokrotnie był tematem publikacji.

PRAKTYK



— A o sztuce komputerowej chyba słyszałeś?

Tekst CPC

Ten program wyświetla teksty w specjalnej, wyróżnionej formie. Program, a właściwie podprogram rozpoczynający się od linii 70 należy uruchamiać z następującymi parametrami:

a$ — tekst do wyświetlenia
x, y — współrzędne początku wyświetlanego tekstu
s — numer pióra (PEN) tła tekstu
t — numer pióra znaków tekstu

```
10 MODE 1:INK 0,10:INK 3,0
20 LOCATE 12,8:PRINT "Tekst 1"
30 s=3:t=1:x=8:y=12
40 a$="Tekst 2"
50 GOSUB 70
60 t=0:a$="Tekst 3":
x=12:y=18:GOSUB 70:PEN 1:
CALL &BB18:STOP
70 REM **WYROZNIONY TEKST**
80 PRINT CHR$(22)CHR$(1)
CHR$(23)CHR$(3):x=(x-1)*16:
gy=398-(y-1)*16:PLOT
-2,-2,s:PEN t
90 TAG:MOVE gx-2,gy:PRINT
a$:MOVE gx+2,gy:PRINT a$:
MOVE gx,gy+2:PRINT a$:
MOVE gx,gy-2:PRINT a$;:
TAGOFF
100 LOCATE x,y:PRINT
a$CHR$(22)CHR$(0)CHR$(23)CHR
$(0):RETURN
```

CHIŃSKA KRZEMOWA DOLINA

Chińskie mikrokomputery nie odbiegają od światowego standardu. Wkrótce mogą się stać konkurentami IBM'a i APPLE'a — pisze J. Maier w ostatnim numerze „BYTE'a" z ubiegłego roku.

Chińska Republika Ludowa już od 1984 roku ma swój własny superkomputer GALAXY (100 milionów operacji na sekundę), a od 1980 roku macierzowy procesor 100—MIPS 757. Chińczycy wytwarzają także własne mikrokomputery, których produkcję szacuje się na około 70 000 sztuk, w tym 25 000 jest zgodne z IBM PC XT. Wytwarza je fabryka na przedmieściach Beijng — CHINA COMPUTER DEVELOPMENT CORP (CCDC). CCDC produkuje także wielodostępny system BCMS 6800 bazujący za Motoroli 68000 oraz UNIX'ie. Maszyna ta jest ewidentnym dowodem dużej popularności UNIX'a w ChRL. Chińska Akademia Nauk rozpoczęła publikowanie tłumaczonych materiałów UNIX'a już w 1984 roku. W tym samym czasie zainstalowano w uniwersytetach i innych chińskich instytucjach ponad 2000 dużych zachodnich komputerów pracujących pod UNIX'em.

CHIŃSKA DOLINA KRZEMOWA

Podczas gdy liczba zainstalowanych w ChRL komputerów nie przekracza miliona (roczna produkcja USA), wzrost chińskiego potencjału informatycznego jest wciąż wykładniczy. Należy pamiętać, że jeszcze kilka lat temu potencjał ten był niewielki.

Minęło zaledwie 10 lat od czasu, gdy pierwszy mikrokomputer został zbudowany w kalifornijskiej Dolinie Krzemowej. Obecnie, za Oceanem, Szanghai i Beijng budują własne Doliny Krzemowe. Przetłumaczono dokładnie nawet ten zwrot na język chiński, brzmi on GUIGOU. Chiński potencjał domowych komputerów ciągle rośnie. Należy się spodziewać, że zaowocuje to pojawieniem się około 1990 roku na rynkach Trzeciego Świata komputerów o światowym standardzie „Made in China".

Szereg ostatnich zdarzeń sygnalizuje ten rozwój. Chińscy przywódcy zdecydowali się nie na import, lecz na własną produkcję. Chińczycy nie są zainteresowani w importowaniu mikroprocesorów, tak jak przed kilku laty, ale kupowaniem technologii. Potwierdzają to ostatnio podpisane umowy z takimi firmami jak: IBM, Hewlett-Packard, Sperry, Wang, Gould, Burroughs, ComputerLand, Solar i inne. Obecnie strategia chińskiego rozwoju informatyki to: *„pierwsza maszyna jest importowana, druga maszyna jest budowana w Chinach, trzecia maszyna jest eksportowana"*.

Opracowanie pt.: „Technology Transfer to China" spowodowało dyskusję nad zaniechaniem ograniczeń w eksporcie nowoczesnych technologii do Chin przez COCOM, Japończyków i większość członków NATO, co może sprawić, że Chiny będą ważnym partnerem państw zachodnich.

CHIŃSKIE POTRZEBY WOBEC WŁASNYCH MOŻLIWOŚCI

W czerwcu 1987 roku w Beijng odbyła się druga międzynarodowa konferencja „Komputery i ich zastosowania", której sponsorami były: amerykańska IEEE Computer Society i Chińska Federacja Komputerowa (CFK), a honory gospodarza konferencji pełniła Chińska Akademia Nauk. W konferencji uczestniczyli reprezentanci 17 krajów z Europy, Azji, USA, Kanady i Australii. Kraje te wydelegowały 300 naukowców informatyków. Zaprezentowali oni 160 referatów, a spośród 100 chińskich naukowców ponad 60 wygłosiło referaty na różne tematy — od lokalnych sieci do mikrokomputerów bazujących na wielojęzykowych systemach przetwarzania słów. Pakiet takiego systemu pracującego na PC XT pod chińską wersją DOSa (z czterema chińskimi dialektami) przedstawił profesor Xinjiang.

Chiny posiadają 170 000 profesjonalnych komputerów i należy oczekiwać przed rokiem 2000 wzrostu tej liczby do miliona. Jednocześnie produkowany będzie milion mikrokomputerów przed rokiem 1990, wobec 300 000 w roku 1987 (połowa importowana). Osiągnięcie tego celu wymaga 500-procentowego wzrostu produkcji w ciągu 3 lat.

Chińska Federacja Komputerowa jest doskonale zorganizowana. Założona w 1985 roku liczy obecnie 25 000 członków ze wszystkich prowincji Chin. CFK jest czynnym organizatorem krajowych i międzynarodowych konferencji, publikuje osiem profesjonalnych periodyków (700 000 egzemplarzy). Komitety Edukacji i Popularyzacji tej federacji skutecznie upowszechniają wiedzę informatyczną wśród miliarda Chińczyków. Komitety te przygotowują 50 różnych samouczków, kompletują książki popularyzujące wiedzę informatyczną oraz wydają specjalne serie dla nastolatków.

Te i wiele innych działań CFK efektywnie zrównuje chińską informatykę ze światowym poziomem produkcji i zastosowań komputerów. Dotyczy to także oprogramowania „Made in China", które w niedługim czasie pojawi się na światowym rynku. Na przykład technologia budowy systemów typu ekspert jest obecnie dobrze znana w Chinach. Ilustruje to między innymi MYCIN — komputerowy system chińskiej tradycyjnej diagnostyki medycznej (jest on powszechny w Chinach od kilku lat).

Chiny ulegają metamorfozie w stylu Japonii — pod względem ekonomicznym i technologicznym. ChRL jest jedną z 10 potęg ekonomicznych świata, których bogactwo rośnie od 5 do 7 proc. rocznie, w 2000 roku oczekuje się, że dochód narodowy Chin wyniesie 1,5 tryliona dolarów.

Wielu ekonomistów uważa, że XXI wiek będzie stuleciem KRAJÓW PACYFIKU.

W.G.

Opracowane na podstawie: The State of Chinese Computing, John H. Maier, BYTE vol. 12 No. 14, Dec. 1987.

Piraci końca XX wieku

Świat już jakby nieco ochłonął po wyścigu o najnowocześniejsze i najsprawniejsze komputery. Coraz częściej za to obserwujemy rywalizację w dziedzinie oprogramowania. I jest to zupełnie naturalna kolej rzeczy — po udoskonaleniu maszyn przyszła pora na znalezienie sposobu optymalnego ich wykorzystania. Szybko jednak okazało się, że rywalizacja ta może posłużyć do celów związanych nie tylko z rozwojem komputeryzacji.

Powszechnym zjawiskiem stało się podrabianie wyrobów znanych i co ważniejsze — uznanych firm. Małe wytwórnie, przeważnie z Dalekiego Wschodu, produkują „oryginalne" zegarki, perfumy, aparaty fotograficzne i inne towary, bezkarnie na ogół wykorzystując szyld renomowanych marek. Czerpią z tego nielegalnego procederu ogromne zyski, narażając prawdziwego producenta nie tylko na straty finansowe, ale także utratę zaufania klientów, jako że falsyfikaty są zazwyczaj bardzo niskiej jakości. Ostatnio, opócz tradycyjnie już fałszowanych towarów, rozkwita handel nielegalnie nagranych kaset wideo i oprogramowania komputerowego. No cóż, piraci nie próżnują, zmieniają się tylko obiekty ich zainteresowań.

Dostrzegając ten problem, a także odczuwając realne już straty finansowe, największe amerykańskie firmy zajmujące się programami komputerowymi postanowiły połączyć siły w walce z piratami na obcych rynkach. Rynki te stanowią ważną pozycję na liście klientów, popyt na amerykańskie oprogramowanie jest bowiem często większy właśnie za granicą niż w kraju. Składa się na to wiele przyczyn, wśród których należy wymienić duże już nasycenie rynku wewnętrznego. Spadek wartości dolara obserwowany w ostatnim czasie, także sprzyja zwiększeniu zysków i rozwijaniu współpracy na nowych rynkach.

Współpraca ta pociąga jednak za sobą niebezpieczeństwo dokonywania nielegalnych kopii. Aby temu przeciwdziałać pięć liczących się na rynku amerykańskim firm — Microsoft, Lotus, Ashton-Tate, Autodesk i Wordperfect-Apple utworzyło stowarzyszenie chroniące ich interesy — Business Software Association (BSA). Większość z tych firm podejmowała już próby walki z piractwem, jednakże teraz po raz pierwszy przystąpiły do wspólnego działania mając nadzieję na osiągnięcie skuteczniejszych rezultatów.

Czy możliwe jest skonstruowanie komputera działającego na podobnej zasadzie, jak mózg człowieka? Na obecnym etapie rozwoju techniki jest to niemożliwe, ale wiele firm prowadzi prace badawczo-konstrukcyjne nad komputerem, którego działanie oparte byłoby na podobnej zasadzie, jak funkcjonowanie mózgów tak nieskomplikowanych zwierząt, jak ślimaki lub żaby.

Neuronowy komputer

Na czym polega różnica w działaniu tradycyjnego komputera i komputera nietypowego, tzw. sieci neuronowej? Komputery tradycyjne pracujące zgodnie z zasadą von Neumanna mają zwykle jeden procesor, jeżeli posiadają więcej niż jeden, to są one połączone liniowo za pomocą odpowiedniego złącza (bus'a). W tego typu pracujących równolegle komputerach nie może być połączonych więcej niż 4 procesory. Mózg zawiera zdecydowanie więcej połączeń równoległych. W przeciętnym mózgu znajduje się około 4 milionów komórek nerwowych — neuronów, z których każda pracuje jak oddzielny procesor lub układ logiczny. Są one połączone ze sobą za pomocą tysięcy łączy zwanych synapsami. Gdy komputery pracujące według zasady von Neumanna wykonują działania ściśle określone za pomocą instrukcji, to sieć neuronowa mózgu obrabia informację poprzez zestawienie odpowiednich połączeń między neuronami. Konstruktorzy sieci neuronowej starają się skopiować tę zasadę działania za pomocą elementów symulujących. Do tego celu nie jest potrzebne specjalne oprzyrządowanie. Większość procesów symulacyjnych przeprowadzana jest za pomocą konwencjonalnych procesorów i pamięci RAM na specjalnie wykonanych tablicach, które następnie umieszczane są w obudowach zwykłych komputerów, takich jak np. IBM PC.

Kluczem do sieci neuronowej jest zastosowane oprogramowanie symulacyjne, które wykorzystuje standardowe elementy sprzętu, jako ciąg dyskretnych procesorów lub neuronów, ze zmienną liczbą połączeń — synaps między nimi.

Oprogramowanie określa rozmiar sieci (symulacja neuronu może być wykonana za pomocą kilku bitów pamięci) oraz ilość połączeń pomiędzy neuronami. Rozmiar sieci, a w szczególności ilość połączeń pomiędzy neuronami, decyduje o mocy i możliwościach całego systemu. Jest to zasadnicza różnica między siecią neuronową a tradycyjnym komputerem.

Zasady działania sieci neuronowej podobne są do zasad funkcjonowania mózgu. Sieć uruchamia największą liczbę neuronów i zestawia struktury różnych jednoczesnych połączeń między nimi. Każdej informacji odpowiada określona struktura połączeń między neuronami. Informacja jest przetwarzana jako ciągi takich struktur.

Jest to bardziej elastyczny sposób analizowania niektórych typów informacji, w szczególności, gdy informacja jest zamazana lub nieczytelna, jak w wypadkach systemów sztucznej inteligencji.

Gdy sieć ma dużą liczbę połączeń między neuronami, komputer może samoczynnie „nauczyć się" danych początkowych przez kilkakrotne powtarzanie ciągów struktur możliwych połączeń między neuronami. Dane nie muszą być ściśle precyzowane, zanim zostaną „wyuczone", jak jest to wymagane w tradycyjnych komputerach. W ten sposób mogą być tworzone łączenia różnych typów informacji w sposób o wiele bardziej elastyczny bez konieczności ograniczenia ich do kilku kluczowych cech.

W obliczeniach komputery sieci neuronowej są wolniejsze od tradycyjnych. Można także wykorzystać w ich pracy oprogramowanie standardowe. Jest jednak mało prawdopodobne, aby komputery te zastąpiły już teraz tradycyjne konstrukcje. Dzięki możliwości szybkiego przetwarzania danych nieczytelnych, zamazanych lub niejednoznacznych, posiadaniu pamięci skojarzeniowej i umiejętności uczenia się, a nie korzystania ze zbiorów danych, komputery te znajdą zastosowanie w systemach sztucznej inteligencji do rozpoznawania mowy i języków oraz analizowania obrazów dynamicznych np. zarejestrowanych na taśmach wideo. Wykorzystanie do takich celów komputerów tradycyjnych jest przeważnie bardzo trudne lub niemożliwe ze względu na ich małe możliwości „myślenia" abstrakcyjnego.

W najbliższych latach przewiduje się gwałtowny rozwój komputerów sieci neuronowej, dzięki którym znacznie powiększą się możliwości systemów sztucznej inteligencji. Najbardziej zaawansowane prace w tym kierunku prowadzone są w Stanach Zjednoczonych. Jak do tej pory wyniki tych prac objęte są ścisłą tajemnicą i niewiele firm zdecydowało się na zaprezentowanie swoich osiągnięć. Według nieoficjalnych informacji amerykańskie firmy TRW i IBM zbudowały procesory dla tych komputerów mające możliwość zestawienia ponad 500 000 połączeń. Firma Synaptics z San Diego w Kalifornii konstruuje układ elektroniczny do sieci neuronowej symulującej system wzrokowy żaby dla przetworzenia danych wizyjnych z oka umieszczonego na końcu ramienia robota. Firma Neuralware z Pensylwanii wypuściła zestaw oprogramowań różnych konfiguracji sieci neuronowej, możliwych do stosowania na IBM PC XT z pamięcią co najmniej 512 KB RAM pozwalających użytkownikowi dowolnie eksperymentować z dziewięcioma rodzajami sieci neuronowej. W firmie HNC skonstruowano przystawkę sieci neuronowej ze specjalną pamięcią 4 Mbajtową, procesorem centralnym MC 68020 i wspomagającym MC 68881, mogącą współpracować z komputerem IBM PC lub kompatybilnym. Jej koszt — 10 000 $.

Firma zbudowała także urządzenie — Nestor Writer, które na podstawie sieci neuronowej weryfikuje prawidłowość podpisów na czekach. Koszt tego urządzenia — 1535 $.

Komputery sieci neuronowej dopiero zaczynają swoją karierę. Ale nawet te największe obecnie istniejące zawierają zaledwie drobny fragment połączeń możliwych w mózgu organicznym. Do skonstruowania sztucznego mózgu w postaci kości elektronicznej jest jeszcze daleka droga. Nadal zbyt mało wiemy o mózgu, a kolejne próby zbudowania coraz doskonalszej sieci neuronowej stwarzają więcej problemów, niż rozwiązują.

MBO

Pierwszym udanym osiągnięciem BSA było wykrycie działalności fałszerzy w Hongkongu. Akcja doprowadziła do aresztowania 10 osób zajmujących się tym procederem oraz konfiskaty ponad 2 milionów dolarów uzyskanych ze sprzedaży nielegalnie skopiowanych programów. Sukces ten zwrócił uwagę innych firm także myślących o wspólnym zwalczaniu piractwa.

Mimo że stowarzyszenie działa już od roku, dopiero niedawno zostało formalnie zarejestrowane. Podczas pierwszego walnego zebrania, które odbyło się w San Francisco, ustanowiono statut oraz wytyczono główne kierunki działań. Zapadły także decyzje w sprawie liczebności członków — uznano, że BSA będzie bardziej aktywne i efektywne w działaniu, jeśli pozostanie organizacją niedużą. Dlatego postanowiono przyjąć tylko trzech nowych kandydatów, spośród takich firm jak Aldus, Adobe, Micropo lub Microrim. Ograniczenie liczby członków ma także na celu skupienie uwagi na przeciwdziałaniu piractwu na wielką skalę, bez angażowania się w drobniejsze akcje, które mogą prowadzić na własną rękę inne, mniejsze firmy.

Działalność BSA polega na zwalczaniu nielegalnego handlu oprogramowaniem. Podejmowane są więc konkretne działania. Chodzi tu przede wszystkim o zebranie dostatecznej liczby dowodów obciążających „podejrzaną" firmę zagraniczną i dostarczenie tych dowodów organom miejscowej policji bądź przedstawicielom władz celnych. Kompletowaniem dowodów zajmują się wynajęci przez stowarzyszenie prywatni detektywi. Oczywiście tego typu akcje kosztują. Ale BSA nie szczędzi środków, ponieważ straty powstałe w wyniku działalności piratów wielokrotnie przewyższają nakłady przeznaczone na ich zwalczanie.

Dlaczego jednak wielkie firmy amerykańskie same musiały zająć się ściganiem fałszerzy? W wielu krajach brak uregulowań prawnych problemów związanych z informatyką, w tym kwestii praw autorskich i ochrony prawnej ich dzieła — programu komputerowego. W Hongkongu, Kanadzie i krajach Commonwealthu zainteresowane firmy musiały same podjąć działania w celu wytropienia piratów, w przeciwnym bowiem wypadku nie mogłyby dochodzić swych praw i wyrównać strat. Mimo deklaracji poparcia ze strony rządów krajów o „podejrzanych" rynkach, tak jak w przypadku Hongkongu, zbieranie dowodów przez wynajętych detektywów trwało całe miesiące, zanim władze zdecydowały się także przystąpić do akcji.

Determinację firm stowarzyszonych w BSA najlepiej uzasadnia wysokość strat. W czasie śledztwa w Hongkongu wyszło na jaw, że piraci sprzedawali niektóre programy nawet po 2 dolary, podczas gdy ich oficjalna cena wynosi 500 dolarów! Warto więc ponosić nawet wysokie koszty i angażować się w zakrojone na szeroką skalę akcje, zwłaszcza że pirackie firmy są na swoim terenie niemal zupełnie bezkarne.

Małgorzata NOWOTNY

W szponach Atari (16)

GRAFIKA — wycinanie i zakrywanie

Tomasz MROWIEC, Ludwik PIELA

Dowolnie skomplikowany rysunek możemy utworzyć za pomocą dwóch podstawowych instrukcji: **PLOT X,Y** i **DRAWTO X,Y**. Liczba wyświetlanych punktów i linii zależy od tego, ile razy użyjemy tych instrukcji. Kształt uzyskanego rysunku zależy od wartości X i Y, których będziemy używać podczas kreślenia. Najczęściej stosowane są dwa podstawowe sposoby otrzymania wartości współrzędnych. Pierwszy z nich polega na wyliczaniu wartości współrzędnych na podstawie różnorodnych wzorów matematycznych. Używany jest głównie wtedy, gdy możemy dobrać odpowiedni wzór dla uzyskania żądanego kształtu lub kreślimy wykresy funkcji zadanych w postaci wzoru. Drugi sposób polega na wykorzystaniu do kreślenia wartości współrzędnych zadanych w postaci jawnej. Wymaga to zwykle wcześniejszego wprowadzenia ich do odpowiedniej tablicy lub tablic. Po bliższym przyjrzeniu okaże się, że w momencie kreślenia oba sposoby są identyczne. Punkt lub odcinek prostej tworzymy na podstawie wartości współrzędnych istniejących w postaci jawnej. Różnice polegają na sposobach przygotowania tych wartości. W pierwszym zadane są w sposób niejawny, w drugim — jawny. Rezultat jest podobny — w obszarze pamięci komputera, zwanym pamięcią ekranu, zostają ustawione odpowiednie bity. Na podstawie zawartości tego obszaru pamięci ANTIC wyświetla utworzony rysunek na ekranie monitora lub telewizora (patrz „IKS" nr 7—8 z 1987 roku).

Każdy z przedstawionych sposobów ma swoje zalety i wady. W pierwszym z nich cały ciężar wyznaczenia wartości współrzędnych punktów przerzucamy na komputer, który na podstawie dosyć lakonicznego zapisu podanego wzoru wyznacza setki, a nawet tysiące wartości, a następnie wykreśla je na ekranie. Program realizujący to zadanie wymaga zwykle mało pamięci.

Drugi sposób wymaga wcześniejszego przygotowania wartości współrzędnych kreślonych punktów lub linii. Najprostszym sposobem ich uzyskania jest wykreślenie rysunku na papierze milimetrowym, a następnie określenie współrzędnych charakterystycznych punktów i wprowadzenie ich do programu. Sposób jest dosyć żmudny, umożliwia jednak uzyskanie dowolnych kształtów. Dla programów w języku BASIC dane o rysunku umieszczane są zwykle w liniach DATA. Zaletą takiego sposobu przygotowania danych jest zachowanie wartości, na których podstawie wykreślono rysunek. Umożliwia to nie tylko szybkie odtworzenie rysunku, lecz także wykonanie pewnych manipulacji całym rysunkiem lub jego częścią.

Ważny jest sposób przechowywania danych o rysunku. Dla prostych rysunków można przechowywać współrzędne punktów w tablicach jednowymiarowych w takiej kolejności, w jakiej będą później połączone. W wypadkach bardziej złożonych przez jeden punkt może przechodzić kilka linii lub mogą istnieć fragmenty rysunku nie połączone liniami z resztą. Dlatego dane o rysunku wygodnie jest przechowywać w tablicach dwuwymiarowych. Pierwszy indeks elementu tablicy oznacza numer fragmentu rysunku, a drugi wskazuje numer punktu w tym fragmencie. W ten sposób para wartości (X (2,1), Y(2,1)) określa współrzędne pierwszego punktu drugiego fragmentu rysunku.

Wprawdzie rysunek zapamiętany jest również, bez względu na sposób tworzenia, w pamięci ekranu, jednak manipulowanie nim jest dosyć trudne. Przyczyną tego jest sposób odwzorowania w pamięci.

Przekształcenia mogą być wykonywane na całym rysunku lub na określonych jego fragmentach (szerzej napiszemy o tym w jednym z tegorocznych zeszytów programów komputerowych). Fragmenty te mogą być wydzielone

```
FK 1 REM ****************
FJ 2 REM *               *
XA 3 REM * Program nr 1 *
FL 4 REM *               *
FO 5 REM ****************
NL 6 REM
BK 20 GRAPHICS 0:POKE 752,1
LB 30 DIM X(15),Y(15),M$(1)
GK 40 XM=319:YM=159
BE 60 REM ** ODCZYT DANYCH WYKRES
   U **
ID 70 FOR I=1 TO 13
HQ 80 READ X,Y
WX 90 X(I)=X:Y(I)=Y
FR 100 NEXT I
TK 120 GOSUB 160
TX 130 GOSUB 260
QV 140 GOTO 770
XA 160 REM ** WYKRESLANIE **
AG 180 GRAPHICS 8:COLOR 1
XV 190 PLOT X(1),Y(1)
SW 200 FOR I=2 TO 13
ZQ 210 DRAWTO 2*X(I),Y(I)
FW 220 NEXT I
ZG 240 RETURN
LS 260 REM ** DODANIE OKNA **
KE 280 ? CHR$(125);"SRODEK OKNA (
   X,Y)";
KE 290 TRAP 280:INPUT XO,YO
EO 330 ? CHR$(125):? "WYSOKOSC I
   SZEROKOSC OKNA";
CM 340 TRAP 330:INPUT OW,OS
JC 350 TRAP 40000
NB 360 L=XO-OS/2
NF 370 P=XO+OS/2
OF 380 G=YO-OW/2
NA 390 D=YO+OW/2
YJ 400 IF L>P OR L<0 OR P>XM OR G
   >D OR G<0 OR D>YM THEN 330
GH 450 PLOT L,G
DB 460 DRAWTO P,G
BN 470 DRAWTO P,D
WW 480 DRAWTO L,D:DRAWTO L,G
TV 500 ? CHR$(125);"K--KONIEC","Z
   --ZMIANA OKNA"
EP 510 INPUT M$
HJ 520 IF M$="K" OR M$="k" THEN 7
   70
PQ 590 REM ** KASOWANIE OKNA **
XD 600 COLOR 0
GB 610 PLOT L,G
CV 620 DRAWTO P,G
BH 630 DRAWTO P,D
WQ 640 DRAWTO L,D:DRAWTO L,G
XY 650 COLOR 1
SI 680 ? CHR$(125);"CZY ODTWORZYC
   RYSUNEK (T/N)"
FG 690 INPUT M$
EK 720 IF M$="N" OR M$="n" THEN 2
   60
VC 730 GOSUB 190
OS 740 GOTO 260
ZN 750 RETURN
OK 770 END
JV 1000 DATA 0,159,10,120,20,60,3
   0,50,40,100,50,120,60,100,70,3
   0,80,40,90,100,100,120,110,110
   ,120,159
```

w specjalny sposób, mogą być usunięte z ekranu lub zachowane i przekształcone, a jednocześnie reszta rysunku będzie usunięta.

Często wymagane jest wyróżnienie pewnych elementów rysunku w celu skupienia na nich uwagi. Elementy te mogą być narysowane innym kolorem lub wykreślone liniami o zwiększonej jaskrawości. Taki sam efekt można uzyskać poprzez wykreślenie w wydzielonym obszarze linii podwójnej grubości: zamiast linii wykreśla się dwie linie w odległości jednego kroku rastra. Innym sposobem zwracania uwagi na wybrany obszar jest otoczenie go ramką, okrągłą lub prostokątną.

W celu nałożenia na rysunek okręgu wystarczy określić współrzędne jego środka (XC, YC) i promień R. Do programu należy dodać podprogram kreślenia okręgu, do którego zwracamy się za każdym razem, kiedy potrzebujemy wydzielić żądany obszar. Można napisać program, który pozwoli eksperymentować z położeniem i rozmiarami okręgu dla uzyskania najlepszego efektu.

Podczas używania, do wydzielania obszaru ekranu, prostokątnej ramki można określać wszystkie cztery lub tylko dwa przeciwległe rogi prostokąta. Można także napisać program, który prosi o podanie współrzędnych środka prostokąta i jego rozmiarów. Przykładem może być program nr 1, który wykreśla prostokąt o środku w punkcie (XB, YB) szerokości W i wysokości H.

Oprócz wyróżniania określonych obszarów ekranu, niekiedy trzeba usunąć części utworzonych rysunków. W takich wypadkach można wykorzystać metody wydzielania obszarów dla identyfikacji fragmentów rysunku, które należy zakryć (usunąć) lub wyciąć. W pierwszym wypadku usuwamy rysunek ze wskazanego obszaru ekranu, w drugim — zachowujemy rysunek tylko w tym obszarze, a usuwamy resztę.

Sposoby wydzielania obszaru ekranu, za pomocą okręgu lub prostokąta, można wykorzystać do określenia ścieranego obszaru. Potrzeba usuwania części rysunku może wynikać z wielu powodów. Na przykład chcemy uprościć złożony rysunek lub uwolnić miejsce na wyświetlenie dodatkowej informacji. Operacje te wygodnie jest wykonać podczas dobierania formatu wyświetlania rysunków w celu uwolnienia miejsca na ekranie podczas powiększenia lub innych przekształceń elementów rysunku.

Wydzielone fragmenty rysunku można oczywiście usunąć, zmieniając program tak, aby rysunek tworzony był bez nich. Prowadzi to do istotnych zmian w programie graficznym. Jeśli później chcemy odtworzyć usunięte fragmenty, program znowu trzeba będzie przerabiać. Jeśli trzeba często usuwać fragmenty rysunku, można specjalnie opracować uniwersalny program do kasowania całej informacji wewnątrz zadanego obszaru.

Wykorzystanie prostokąta do identyfikacji oczyszczonego obszaru umożliwia łatwe wycieranie znaków, punktów i części linii, znajdujących się wewnątrz prostokąta. Trzeba tylko zapełnić ten obszar liniami w kolorze tła.

Można czyścić także wewnątrz granic okręgu. Jednak w tym wypadku program będzie działać wolniej, ponieważ należy obliczać współrzędne punktów okręgu, które następnie wykorzystuje się do wskazania końców odcinków linii zapełniających wewnętrzny obszar okręgu.

W pewnych wypadkach należy przechować część rysunku i wykasować resztę. W tym celu wybrany obszar ekranu zamyka się w prostokątną ramkę lub okno. Kasowanie elementów rysunku na zewnątrz okna nazywamy wycinaniem.

Program, powodujący wycinanie dowolnego zobrazowania, powinien znajdować i przechowywać wszystkie napisy, oddzielne punkty i części linii mieszczące się w obszarze okna. Analizuje on współrzędne końców odcinków i położenie napisów w celu określenia przechowywanych elementów rysunku.

Przede wszystkim rozważmy działanie programu wycinania dla punktów i odcinków linii prostych. Będzie on wprowadzał współrzędne lewego górnego rogu okna *(XW, YW)* szerokość *WW* i wysokość okna *HW*. Współrzędne elementów rysunku przechowujemy w tablicach *X* i *Y*.

Oddzielny punkt (nie będący częścią linii) przechowywany jest przez program wycinania, jeśli znajduje się wewnątrz okna, tj. jeśli jego współrzędna *X* leży w przedziale od *XW* do *XW + WW* a współrzędną *Y* — w przedziale od *YW* do *YW+ HW*. Odcinek prostej jest przechowywany, jeśli oba jego końce znajdują się wewnątrz obszaru okna. Linia usuwana jest w całości. Jeśli wszystkie jej punkty leżą na zewnątrz okna. Jeśli część linii znajduje się na zewnątrz okna, jest odcinana. Przechowywana jest tylko ta część linii, która znajduje się wewnątrz.

Program nr 2 rozpoczyna działanie od analizy współrzędnych względem lewej granicy okna. Punkty, których współrzędne poziome przewyższają wartość *X* lewej granicy okna, zapamiętywane są w tablicach *X1* i *Y1*. Punkty przecięcia linii z lewą granicą ekranu także zapamiętywane są w tablicach *X1* i *Y1*. Następnie współrzędne z tablic *X1* i *Y1* porównywane są z położeniem górnej granicy okna.

Punkty przecięcia i wszystkie punkty izolowane, których współrzędna pionowa przewyższa wartość *Y* górnej granicy okna, zapamiętywane są w tablicach *X2* i *Y2*. Pozostałe punkty odnoszone są do prawej granicy okna. Tablice *X1* i *Y1* wykorzystywane są powtórnie do przechowywania współrzędnych punktów przecięcia i punktów, których współrzędna pozioma nie przewyższa współrzędnej *X* prawej granicy. W końcu, współrzędne punktów z tablic *X1* i *Y1* porównywane są z położeniem dolnej granicy okna. Tablice *X2* i *Y2* wykorzystuje się do przechowywania końcowej informacji o punktach i odcinkach linii, które powinny być zobrazowane po oczyszczeniu ekranu.

Rozszerzymy teraz program nr 2, aby zapewnić wykonanie operacji wycinania przy obecności napisów. Można przetwarzać napisy tak jak linie, to jest przechowywać tę część napisu, która znajduje się wewnątrz okna. Jednak dla uproszczenia programu będziemy przechowywać tylko te opisy, które całkowicie

```
FK 1 REM ****************
FJ 2 REM *               *
XU 3 REM * Program nr 2 *
FL 4 REM *               *
FO 5 REM ****************
NL 6 REM
AC 160 GRAPHICS 8:COLOR 1
OU 170 DIM X(8,20),Y(8,20),X1(8,2
   0),Y1(8,20),X2(8,20),Y2(8,20),
   NE(8),M$(1),ME(20)
SJ 180 XM=319
TK 190 YM=159
BH 195 REM ********************
WQ 200 REM ODCZYT WSPOLRZEDNYCH
AQ 205 REM ********************
XJ 210 READ N
FN 220 FOR P=1 TO N
RB 230 READ NE:NE(P)=NE
UZ 240 FOR E=1 TO NE
SP 250 READ X,Y:X(P,E)=X:Y(P,E)=Y
EQ 260 NEXT E
IY 270 NEXT P
EZ 280 GRAPHICS 8:COLOR 1:POKE 75
   2,1
RN 290 GOSUB 340:REM KRESLENIE RY
   SUNKU
MM 300 GOSUB 450:REM OKRESLENIE O
   KNA
VP 305 IF M$="Z" THEN 330
YC 310 GOSUB 850:REM WYCINANIE OK
   NA
ZH 320 GOSUB 2480:REM KRESLENIE W
   YCIETYCH PUNKTOW
CY 330 IF PEEK(764)=255 THEN 330
XO 335 POKE 764,255:GOTO 280
BG 338 REM ********************
UR 340 REM PODPROGRAM KRESLENIA
AZ 345 REM ********************
RH 350 ? #6;CHR$(125);
FW 360 FOR P=1 TO N
CP 370 IF NE(P)>1 THEN 400
BD 380 PLOT X(P,1),Y(P,1)
OL 390 GOTO 430
IG 400 FOR E=1 TO NE(P)-1
EQ 410 PLOT X(P,E),Y(P,E):DRAWTO
   X(P,E+1),Y(P,E+1)
EK 420 NEXT E
IS 430 NEXT P
ZI 440 RETURN
ZA 445 REM ****************
HN 450 REM OKRESLENIE OKNA
ZC 455 REM ****************
LP 460 ? CHR$(125);
PL 465 ? "PARAMETRY OKNA:"
IX 470 ? "Lewy gorny rog (x,y):";
XY 480 TRAP 460:INPUT XW,YW
ZE 520 ? "Wysokosc i szerokosc:";

KU 530 TRAP 490:INPUT HW,WW
LM 540 ? CHR$(125);
JE 550 TRAP 40000
RV 560 L=XW
AF 570 R=XW+WW
TU 580 T=YW
RU 590 B=YW+HW
WS 600 IF L>=R OR L<0 OR R>XM OR
   T>=B OR T<0 OR B>YM THEN 460
ZT 660 PLOT L,T:DRAWTO L,B
BL 670 DRAWTO R,B
LF 680 DRAWTO R,T
IN 690 DRAWTO L,T
SD 700 ? "K - kasowanie ramki W -
    wycinanie              Z - zakr
   ywanie";
ER 710 INPUT M$
XA 720 IF M$="Z" OR M$="z" THEN G
   OSUB 5000:RETURN
TM 730 IF M$="W" OR M$="w" THEN R
   ETURN
ZB 735 REM ****************
JG 740 REM KASOWANIE RAMKI
ZD 745 REM ****************
LQ 750 ? CHR$(125);
XQ 760 COLOR 0
ZY 780 PLOT L,T:DRAWTO L,B
BQ 790 DRAWTO R,B
KR 800 DRAWTO R,T
HZ 810 DRAWTO L,T
XU 820 COLOR 1
PA 830 GOTO 450
VL 840 REM ********************
WE 850 REM WYCINANIE WZGLEDEM
DH 860 REM LEWEGO BRZEGU
WE 865 REM ********************
DZ 870 REM W X1 I Y1
XD 880 P1=1
GH 890 FOR P=1 TO N
UD 900 E1=0
IN 910 FOR E=1 TO NE(P)-1
JE 930 IF X(P,E)<L THEN 1020
QB 940 REM PIERWSZY PUNKT JEST W
   SRODKU
DF 950 E1=E1+1
RP 960 X1(P1,E1)=X(P,E)
SL 970 Y1(P1,E1)=Y(P,E)
YI 980 REM CO Z DRUGIM PUNKTEM?
DF 990 IF X(P,E+1)>=L THEN 1070
BS 1000 GOSUB 1190
QE 1010 GOTO 1070
OO 1020 REM PIERWSZY PUNKT JEST P
   OZA
SE 1030 REM CO Z DRUGIM PUNKTEM?
UQ 1050 IF X(P,E+1)>=L THEN GOSUB
    1190
DS 1070 NEXT E
IU 1080 REM
KL 1090 IF X(P,NE(P))<L THEN 1130

AE 1100 E1=E1+1
MX 1110 X1(P1,E1)=X(P,NE(P))
NW 1120 Y1(P1,E1)=Y(P,NE(P))
RL 1130 IF E1=0 THEN 1160
SU 1140 ME(P1)=E1
HC 1150 P1=P1+1
II 1160 NEXT P
PH 1170 MN=P1-1
RM 1180 GOTO 1260
IZ 1190 REM
AG 1200 E1=E1+1
ET 1210 M=(Y(P,E+1)-Y(P,E))/(X(P,
   E+1)-X(P,E))
RG 1220 Y1(P1,E1)=M*(L-X(P,E))+Y(
   P,E)
KN 1230 X1(P1,E1)=L
AP 1240 RETURN
EV 1250 REM ********************
OI 1255 REM WYCINANIE WZGLEDEM
UW 1260 REM GORNEGO BRZEGU
AO 1265 REM ********************
PA 1270 P1=1
TN 1280 FOR P=1 TO MN
MJ 1290 E1=0
FN 1300 FOR E=1 TO ME(P)-1
OX 1310 IF Y1(P,E)>=T THEN 1330
AW 1320 IF Y1(P,E)<T THEN 1420
HT 1330 REM PIERWSZY PUNKT LINII
AY 1335 REM JEST W SRODKU
AU 1340 E1=E1+1
BU 1350 Y2(P1,E1)=Y1(P,E)
BB 1360 X2(P1,E1)=X1(P,E)
LJ 1370 REM CO ZDRUGIM PUNKTEM?
WW 1380 IF Y1(P,E+1)<T THEN GOSUB
    1580
JD 1390 REM
RS 1400 GOTO 1460
IK 1420 REM
FS 1440 IF Y1(P,E+1)>=T THEN GOSU
   B 1580
IT 1450 REM
DX 1460 NEXT E
EX 1470 REM BADANIE PUNKTU KONCOW
   EGO
HS 1480 IF Y1(P,ME(P))<T THEN 152
   0
BL 1490 E1=E1+1
NE 1500 Y2(P1,E1)=Y1(P,ME(P))
ML 1510 X2(P1,E1)=X1(P,ME(P))
TX 1520 IF E1=0 THEN 1550
SZ 1530 ME(P1)=E1
HH 1540 P1=P1+1
IN 1550 NEXT P
PM 1560 MN=P1-1
UN 1570 GOTO 1680
JE 1580 REM
BN 1590 E1=E1+1
PJ 1600 IF X1(P,E+1)<>X1(P,E) THE
   N 1630
AS 1610 X2(P1,E1)=X1(P,E)


SN 1620 GOTO 1650
SS 1630 M=(Y1(P,E+1)-Y1(P,E))/(X1
   (P,E+1)-X1(P,E))
ON 1640 X2(P1,E1)=(T-Y1(P,E))/M+X
   1(P,E)
QM 1650 Y2(P1,E1)=T
BD 1660 RETURN
```

```
FJ 1670 REM ********************
OF 1680 REM WYCINANIE WZGLEDEM
XO 1685 REM PRAWEGO BRZEGU
GO 1687 REM ********************
PO 1690 P1=1
SZ 1700 FOR P=1 TO MN
LV 1710 E1=0
GB 1720 FOR E=1 TO ME(P)-1
SY 1730 IF X2(P,E)<=R THEN 1750
HF 1740 IF X2(P,E)>R THEN 1840
IZ 1750 REM
BI 1760 E1=E1+1
BW 1770 X1(P1,E1)=X2(P,E)
CV 1780 Y1(P1,E1)=Y2(P,E)
JL 1790 REM
GS 1800 IF X2(P,E+1)>R THEN GOSUB
    2000
IP 1810 REM
VC 1820 GOTO 1880
IY 1840 REM
NI 1860 IF X2(P,E+1)<=R THEN GOSU
   B 2000
JH 1870 REM
EL 1880 NEXT E
JN 1890 REM
NX 1900 IF X2(P,ME(P))>R THEN 194
   0
AX 1910 E1=E1+1
NG 1920 X1(P1,E1)=X2(P,ME(P))
OF 1930 Y1(P1,E1)=Y2(P,ME(P))
ZD 1940 IF E1=0 THEN 1970
TN 1950 ME(P1)=E1
HV 1960 P1=P1+1
JB 1970 NEXT P
QA 1980 MN=P1-1
SF 1990 GOTO 2070
HX 2000 REM
AG 2010 E1=E1+1
WA 2020 M=(Y2(P,E+1)-Y2(P,E))/(X2
   (P,E+1)-X2(P,E))
FV 2030 Y1(P1,E1)=M*(R-X2(P,E))+Y
   2(P,E)
OF 2040 X1(P1,E1)=R
AP 2050 RETURN
ZR 2060 REM ********************
OI 2065 REM WYCINANIE WZGLEDEM
QY 2070 REM DOLNEGO BRZEGU
AO 2075 REM ********************
PA 2080 P1=1
TN 2090 FOR P=1 TO MN
LH 2100 E1=0
FN 2110 FOR E=1 TO ME(P)-1
BF 2120 IF Y1(P,E)<=B THEN 2140

SI 2130 IF Y1(P,E)>B THEN 2250
IL 2140 REM
AU 2150 E1=E1+1
BU 2160 Y2(P1,E1)=Y1(P,E)
BB 2170 X2(P1,E1)=X1(P,E)
IX 2180 REM
LC 2190 IF Y1(P,E+1)>B THEN GOSUB
    2390
IB 2200 REM
RS 2210 GOTO 2270
RZ 2230 REM PIERWSZY PUNKT POZA
QM 2250 IF Y1(P,E+1)<=B THEN GOSU
   B 2390
IT 2260 REM
DX 2270 NEXT E
IZ 2280 REM
US 2290 IF Y1(P,ME(P))>B THEN 233
   0
AJ 2300 E1=E1+1
NE 2310 Y2(P1,E1)=Y1(P,ME(P))
ML 2320 X2(P1,E1)=X1(P,ME(P))
TX 2330 IF E1=0 THEN 2360
SZ 2340 ME(P1)=E1
HH 2350 P1=P1+1
IN 2360 NEXT P
PM 2370 MN=P1-1
BE 2380 RETURN
JE 2390 REM
AL 2400 E1=E1+1
PJ 2410 IF X1(P,E+1)<>X1(P,E) THE
   N 2440
AS 2420 X2(P1,E1)=X1(P,E)
SN 2430 GOTO 2460
SS 2440 M=(Y1(P,E+1)-Y1(P,E))/(X1
   (P,E+1)-X1(P,E))
CT 2450 X2(P1,E1)=(B-Y1(P,E))/M+X
   1(P,E)
FK 2460 Y2(P1,E1)=B
BD 2470 RETURN
AW 2475 REM ********************
FJ 2480 REM KRESLENIE WYCIETYCH
TO 2485 REM PUNKTOW
BH 2487 REM ********************
FH 2490 ? #6;CHR$(125);
FZ 2500 PLOT L,T:DRAWTO L,B
KJ 2510 DRAWTO R,B
UW 2520 DRAWTO R,T
RZ 2530 DRAWTO L,T
TI 2540 FOR P=1 TO MN
GH 2550 FOR E=1 TO ME(P)-1
KS 2560 PLOT X2(P,E),Y2(P,E):DRAW
   TO X2(P,E+1),Y2(P,E+1)
ED 2570 NEXT E
IX 2580 NEXT P
BL 2590 RETURN
ZL 2600 REM ********************
MV 2610 DATA 8
EK 2630 DATA 12,195,87,210,90,232
   ,65,232,60,213,50,200,35,165,3
   7
HU 2640 DATA 145,50,85,55,70,30,6
   0,30,60,57
EH 2650 DATA 3,69,66,85,80,152,85
VZ 2670 DATA 6,160,70,138,128,145
   ,135,170,135,178,128,200,75
FO 2680 DATA 6,203,40,198,5,195,0
   ,170,0,165,5,167,36
RY 2700 DATA 3,73,35,78,35,80,45
UB 2710 DATA 4,65,50,53,70,65,70,
   75,60
IM 2730 DATA 7,230,63,235,63,230,
   25,240,25,230,100,240,100,235,
   63
UH 2731 DATA 6,208,80,210,80,210,
   82,208,82,208,80,210,82
FO 2740 END
XN 5000 COLOR 0
JN 5010 FOR X=L TO R
VE 5020 PLOT X,T:DRAWTO X,B
MC 5025 NEXT X
GI 5030 ? CHR$(125)
AP 5040 RETURN
```

mieszczą się w granicach ramki. Analiza położenia napisów polega na sprawdzeniu, czy początek tekstu znajduje się na prawo od lewej granicy okna, a koniec na lewo od prawej granicy. Oprócz tego napis powinien znajdować się w wierszu położonym między górną i dolną granicą ramki.

Zmienna *T$* wykorzystywana jest do przechowywania napisów — ciągów znaków. Tablice *XL* i *XR* używane są do przechowywania współrzędnych *X* początków i końców wszystkich ciągów. W tablicach **YT** i **YB** przechowywane są współrzędne górnej i dolnej granicy każdego ciągu znaków.

Na jeden symbol wydziela się 8×8 elementów rastra. Faktycznie zajmuje on obszar 7×7, to znaczy, że odstęp między literami w wierszu równy jest jednemu elementowi rastra. Taki sam jest odstęp między wierszami. Symbole zaczynają się w punktach o współrzędnych *X*: Ø, 8, 16, 24, ... i kończą się w punktach 7, 15, 23, 31, ... Górna granica symboli w wierszach ma współrzędne *Y* 0, 8, 16, ... a dolna — 7, 15, 23, ... W celu określenia początku obszaru wyświetlania pewnego znaku musimy odjąć 1 od numeru pozycji w wierszu i pomnożyć przez 8. W podobny sposób określa się koniec obszaru wyświetlania, górną i dolną jego granicę. Zresztą i tak będzie to za nas robił komputer. Odpowiednie obliczenia zawarte są w programie nr 3, który jest uzupełnieniem programu nr 2.

Program nr 3 może być tak zmodyfikowany, aby przechowywać dowolną część tekstu mieszczącego się wewnątrz granic okna. Można też ulepszyć algorytm wycinania linii, w celu zmniejszenia liczby wykonywanych obliczeń. Będzie to ważne przy dużej liczbie linii.

Po wykonaniu wycinania można w dowolny sposób przekształcić zobrazowanie wewnątrz okna:

```
FK 1 REM ****************
FJ 2 REM *               *
YO 3 REM * Program nr 3 *
FL 4 REM *               *
FO 5 REM ****************
NL 6 REM
VI 171 DIM TX$(100),RO(10),CO(10)
   ,TL(10),TR(10),TT(10),TB(10),T
   $(10),LT(10),PT(10)
ZS 271 REM ODCZYTANIE TEKSTOW I W
   SPOLRZEDNYCH
AG 272 READ TN:F=1
HF 273 FOR K=1 TO TN
RR 274 READ T$,RO,CO:TX$(F)=T$:RO
   (K)=RO:CO(K)=CO:LT(K)=LEN(T$):
   PT(K)=F
CQ 275 TL(K)=(CO(K)-1)*8
QT 276 TR(K)=(TL(K)-1)+LEN(T$)*8
LQ 277 TT(K)=(RO(K)-1)*8
DF 278 TB(K)=TT(K)+7
VI 279 F=F+LEN(T$):NEXT K
MV 431 REM UMIESZCZENIE TEKSTOW
GW 432 FOR K=1 TO TN
NY 433 Y=RO(K):X=CO(K):T$=TX$(PT(
   K),PT(K)+LT(K)-1)
SV 434 GOSUB 10000
HJ 435 NEXT K
WP 2581 FOR K=1 TO TN
GI 2582 IF TL(K)<L OR TR(K)>R OR
   TT(K)<T OR TB(K)>B THEN 2589
VG 2583 Y=RO(K):X=CO(K):T$=TX$(PT
   (K),PT(K)+LT(K)-1)
BW 2584 GOSUB 10000
IE 2589 NEXT K
LT 2732 DATA 4
NM 2733 DATA KADLUB,12,10
BB 2734 DATA OGON,7,3
UM 2735 DATA SKRZYDLO,18,16
PV 2736 DATA SMIGLO,2,26
WV 9999 REM ZNAKI W GRAFICE 8
ZT 10000 START=PEEK(89)*256+PEEK(
   88)+40*Y*8+X
YC 10010 FOR E1=1 TO LEN(T$):E3=A
   SC(T$(E1))
EL 10020 E3=E3+64*(E3<32)-32*(E3>
   31 AND E3<96)
MK 10030 ZN=PEEK(756)*256+E3*8
EA 10040 FOR E2=7 TO 1 STEP -1
VW 10050 POKE START+E2*40,PEEK(ZN
   +E2):NEXT E2
SI 10060 X=X+1:IF X>=40 THEN STAR
   T=START+40*8:X=0
BT 10070 START=START+1:NEXT E1:RE
   TURN
```

— przenieść go x inne miejsce ekranu, powiększyć lub pomniejszyć lub też obrócić;

— zachować początkowy rysunek, nałożywszy na niego (np.: w kącie ekranu) wydzielony za pomocą okna fragment rysunku (być może powiększony).

Miejsce umieszczenia zawartości okna, po ewentualnych przekształceniach, nazywamy polem wprowadzania. Jego położenie, podobnie jak dla okna, można opisać podając współrzędne lewego, górnego rogu i rozmiary. Okno określa „co" chcemy zobaczyć, pole wyprowadzania wskazuje, „gdzie" chcemy zobaczyć to na ekranie. Rozmiary pola wyprowadzania mogą być większe lub mniejsze od rozmiarów okna lub takie same. Pole wyprowadzania może zajmować cały ekran lub być niewielką wstawką na ekranie. Okno i pole wyprowadzania można rozmieścić w różnych obszarach ekranu lub uczynić je przecinającymi się. Za pomocą pola wyprowadzania, którego rozmiary są znacznie większe od rozmiarów okna, można powiększyć zobrazowanie oddzielnych elementów i obejrzeć szczegóły, które są początkowo słabo rozróżniane.

Program przekształcający obszar okna w pole wyprowadzania powinien przekształcać współrzędne wszystkich punktów wewnątrz okna na współrzędne punktów w polu wyprowadzania.

Nowe współrzędne punktów w polu wyprowadzania związane są ze współrzędnymi

odpowiadających punktów w obszarze okna następującymi zależnościami:

$$XN = (X - XW) * (WY/WW) + XV$$

$$YN = (Y - YW) * (HV/HW) + YV$$

Parametry *(WV/WW)* i *(HV/HW)* są współczynnikami skalowania. Jeśli są one równe 1, rozmiary elementów zobrazowania podczas przekształcania zmieniają się. Wartości większe od 1 powodują zwiększenie rozmiarów, a wartości mniejsze od 1 — zmniejszenie. Jeśli stosunek *WV/HV* nie jest równy *WW/HW* obserwuje się naruszenie proporcji zobrazowania, co równoważne jest nierównomiernemu skalowaniu wzdłuż osi *X* i *Y*. Człony *XV* i *YV* w powyższych równaniach określają wielkość przemieszczenia.

Przekształcanie zobrazowania z obszaru okna w pole wyprowadzania wykonuje **program nr 4.** Aby zobaczyć jego działanie, należy dołączyć go do programu nr 2.

```
FK 1 REM ****************
FJ 2 REM *                *
ZI 3 REM * Program nr 4 *
FC 4 REM *                *
FO 5 REM ****************
NL 6 REM
VI 175 DIM XN(8,20),YN(8,20)
XL 335 POKE 764,255:GOTO 2750
IX 2740 REM
JA 2750 REM
EH 2810 GOSUB 2850
BY 2820 GOSUB 3070
BM 2830 GOSUB 3150
EG 2840 IF PEEK(764)=255 THEN 284
   0
EI 2845 POKE 764,255:GOTO 290
JC 2850 REM
TT 2860 ? CHR$(125);
SJ 2870 ? "Lewy gorny rog x,y ";
RQ 2880 TRAP 2860:INPUT XV,YV
UC 2890 ? CHR$(125);
FT 2920 ? "Podaj wysokosc i szero
   kosc ";
DE 2930 TRAP 2890:INPUT HV,WV
PA 2940 TRAP 40000
TS 2950 ? CHR$(125);
MX 2960 L=XV
DL 2970 R=XV+WV
PH 2980 T=YV
TY 2990 B=YV+HV
YE 3000 IF L>=R OR R>XM OR T>=B O
   R T<0 OR B>YM THEN 2860
AT 3060 RETURN
JH 3070 REM PRZELICZENIE OKNA
TL 3080 FOR P=1 TO MN
CZ 3090 FOR E=1 TO ME(P)
MH 3100 XN(P,E)=(X2(P,E)-XW)*(WV/
   WW)+XV
HK 3110 YN(P,E)=(Y2(P,E)-YW)*(HV/
   HW)+YV
DH 3120 NEXT E
IB 3130 NEXT P
AP 3140 RETURN
ZF 3150 REM WYKRESLENIE WYCIETYCH
    PUNKTOW
LJ 3160 ? #6;CHR$(125)
GN 3170 PLOT L,T:DRAWTO L,B
KX 3180 DRAWTO R,B
VK 3190 DRAWTO R,T
RL 3200 DRAWTO L,T
SU 3210 FOR P=1 TO MN
FT 3220 FOR E=1 TO ME(P)-1
AU 3230 PLOT XN(P,E),YN(P,E):DRAW
   TO XN(P,E+1),YN(P,E+1)
DP 3240 NEXT E
IJ 3250 NEXT P
AX 3260 RETURN
CZ 3270 REM ************************
   *****
FR 3280 END
```

```
FK 1 REM ****************
FJ 2 REM *                *
XU 3 REM * Program nr 2 *
TG 4 REM * uzupelnienie *
FM 5 REM *                *
FP 6 REM ****************
NM 7 REM
NN 8 REM
PV 10 REM W programie nr 2 z IKS
QJ 12 REM nr 2 z br. omylkowo nie
PJ 14 REM zamieszczono podpro-
OA 16 REM gramu, bedacego zasad-
BZ 18 REM nicza czescia przykla-
VF 19 REM du.
DT 20 REM Przepraszamy za zwia-
LA 22 REM zane z tym klopoty.
HE 24 REM Ponizszy program nale-
NK 26 REM zy wprowadzic do kom-
FN 28 REM putera i zapisac na
EK 30 REM tasmie magnetycznej.
JQ 32 REM Nastepnie nalezy dola-
IJ 34 REM czyc go do programu
IC 36 REM nr 2.
NK 70 GOSUB 31000
XV 31000 W=0:V=1:RESTORE 31050
RS 31010 READ X,Y:FOR I=X TO Y:RE
   AD Z:POKE I,Z:NEXT I
KY 31020 READ Z:IF Z<>-1 THEN ? "
   BLAD W DANYCH! SPRAWDZ LINIE D
   ATA !!!":END
MI 31030 W=W+1:IF W<V THEN 31010
DR 31040 RETURN
CX 31050 DATA 1536,1700
NV 31060 DATA 104,141,185,6,141,1
   86,6,162,0,104,157,173,6,157,1
   81,6,104,157,169,6
XO 31070 DATA 157,177,6,169,0,157
   ,165,6,232,236,186,6,208,231,1
   69,7,162,6,160,43
FV 31080 DATA 76,92,228,173,185,6
   ,240,88,169,3,141,15,210,169,0
   ,141,8,210,141,185
US 31090 DATA 6,170,189,165,6,208
   ,52,168,189,173,6,133,1,189,16
   9,6,133,0,177,0
JU 31100 DATA 72,138,10,168,104,1
   53,0,210,169,164,153,1,210,160
   ,1,177,0,157,165,6
ZM 31110 DATA 136,17,0,240,22,189
   ,169,6,24,105,2,157,169,6,144,
   3,254,173,6,169
GG 31120 DATA 127,238,185,6,222,1
   65,6,232,236,186,6,208,185,76,
   98,228,162,0,189,181
EA 31130 DATA 6,157,173,6,189,177
   ,6,157,169,6,232,236,186,6,208
   ,238,173,186,6,141
DT 31140 DATA 185,6,76,133,6,-1
```

Opracowując powyższy artykuł korzystaliśmy z książki wydanej w 1987 roku w Moskwie przez Wydawnictwo „Mir" pod tytułem: „Microcomputer graphics. Techniques and Applications" napisanej przez D. Hearn i M. P. Baker.

Tomasz MROWIEC
Ludwik PIELA

Skalowanie wykresu

Krzysztof POŹNIAK

Problem umieszczenia wykresu funkcji na ekranie dla wielu początkujących programistów jest zadaniem nie do rozwiązania. Jeżeli jeszcze rysunek ma być umieszczony tylko na wybranym obszarze ekranu i do tego w skali logarytmicznej... Lepiej o tym nie myśleć!

Proponuję zacząć systematycznie i od rzeczy najprostszych.

Na ekranie komputera możemy wykreślić fragment funkcji $Y = f(X)$, np. takiej:

rysunek 1

gdzie: X_{min} — początkowa wartość argumentu funkcji,
X_{max} — końcowa wartość argumentu funkcji,
Y_{min} — minimalna wartość funkcji dla przedziału $X \in \langle X_{min}, X_{max} \rangle$,
Y_{max} — maksymalna wartość funkcji dla przedziału $X \in \langle X_{min}, X_{max} \rangle$,
uwaga: wartości Y_{min} i Y_{max} mogą być podawane z odpowiednimi zapasami, odpowiednio w dół i w górę, ale wtedy funkcja nie obejmie całego obszaru, na którym będzie rysowana.

Prezentowany fragment wykresu chcemy umieścić na zadanym obszarze ekranu, np:

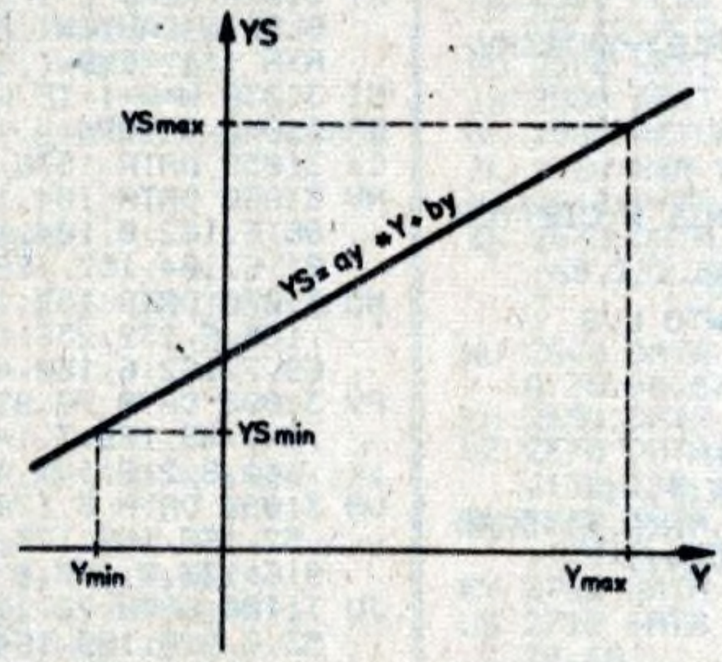
rysunek 2

gdzie: XS_{min} — początkowy punkt na osi poziomej XS ekranu,
XS_{max} — końcowy punkt na osi poziomej XS ekranu,
YS_{min} — najniższy punkt na osi pionowej YS ekranu,
YS_{max} — najwyższy punkt na osi pionowej YS ekranu.

Tak więc naszym zadaniem jest znaleźć takie przekształcenie, aby przejść z poziomej osi wykresu *X* na poziomą oś ekranu *XS* i z pionowej osi wykresu *Y* na pionową oś ekranu *YS*.

Zajmijmy się na początek osiami pionowymi. Graficzne przejście pomiędzy osią *Y*, a *YS* można przedstawić następująco:

rysunek 3



Jak widać funkcja przekształcająca jest linią prostą, czyli przekształcenie można zapisać w ten sposób: $YS = a_y * Y + b_y$. Parametry oblicza się następująco:

$$a_y = \frac{YS_{max} - YS_{min}}{Y_{max} - Y_{min}},$$

$$b_y = YS_{min} - a_y * Y_{min} = YS_{max} - a_y * Y_{max} = \frac{YS_{min} * Y_{max} - YS_{max} * Y_{min}}{Y_{max} - Y_{min}}$$

Sprawdźmy nasze przekształcenia na prostym przykładzie: niech wartość funkcji zmienia się w zakresie od —3 do 2, a chcemy aby na ekranie funkcja rysowała się w pionie między 100-nym, a 200-nym punktem na ekranie.

Rozwiązanie: dane: $Y_{min} = -3$
$Y_{max} = 2$
$YS_{min} = 100$
$YS_{max} = 200$

stąd

$$a_y = \frac{200 - 100}{2 - (-3)} = 20; \quad b_y = \frac{100 * 2 - 200 * (-3)}{2 - (-3)} = 160$$

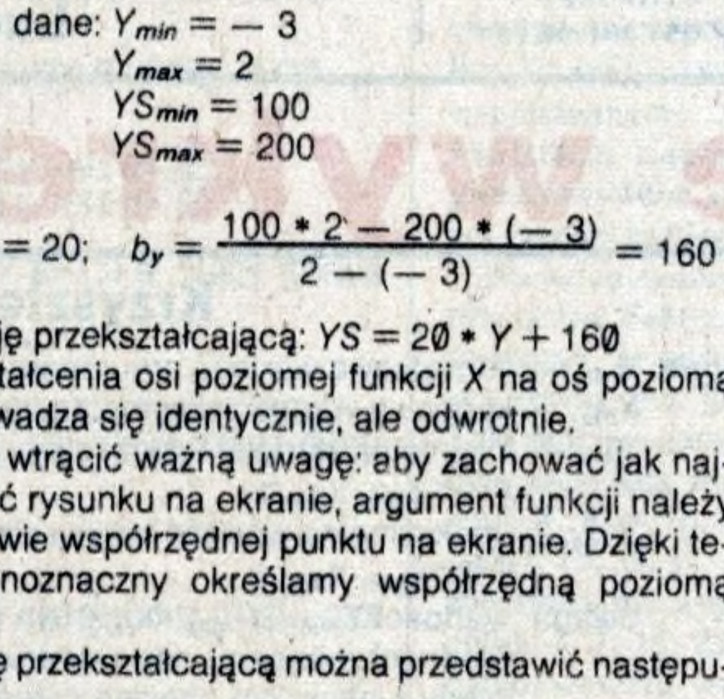
otrzymujemy funkcję przekształcającą: $YS = 20 * Y + 160$

Problem przekształcenia osi poziomej funkcji *X* na oś poziomą ekranu *XS* przeprowadza się identycznie, ale odwrotnie.

I tutaj chciałbym wtrącić ważną uwagę: aby zachować jak największą dokładność rysunku na ekranie, argument funkcji należy obliczać na podstawie współrzędnej punktu na ekranie. Dzięki temu w sposób jednoznaczny określamy współrzędną poziomą punktu na ekranie.

Graficznie funkcję przekształcającą można przedstawić następująco:

rysunek 4



Jest to również linia prosta, stąd $X = a_x * XS + b_x$, gdzie:

$$a_x = \frac{X_{max} - X_{min}}{XS_{max} - XS_{min}}$$

$$b_x = X_{min} - a_x * XS_{min} = X_{max} - a_x * XS_{max} = \frac{X_{min} * XS_{max} - X_{max} * XS_{min}}{XS_{max} - XS_{min}}$$

Zajmijmy się teraz innym problemem: skalą logarytmiczną.

Dla osi pionowej sprawa jest prosta — należy wartość funkcji zlogarytmować. Pozostałe przekształcenia są identyczne, jak dla poprzednio obliczanej funkcji przeliczającej oś *Y* na oś *YS*, z tą różnicą, że zamiast Y_{min} podstawiamy $m * log_a(Y_{min})$, a zamiast Y_{max} podstawiamy $m * log_a(Y_{max})$, gdzie:
m — pewien współczynnik liniowy
a — podstawa logarytmu
(np. dla skali w decybelach [*dB*] $m = 20$, $a = 10$)

Otrzymujemy następującą funkcję przekształcającą:

$$YS = a_w * m * log_a(Y) + b_w$$

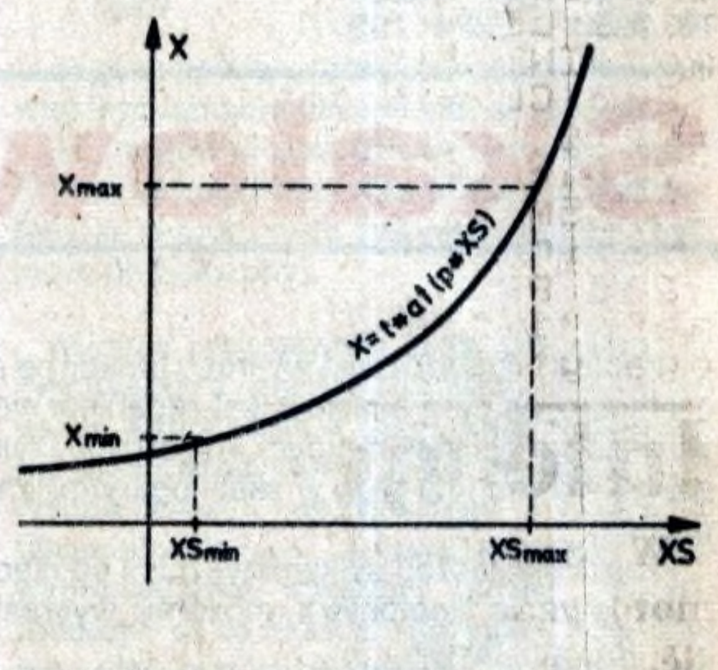
gdzie:

$$a_w = \frac{YS_{max} - YS_{min}}{m * log_a\left(\frac{Y_{max}}{Y_{min}}\right)}$$

$$b_w = YS_{min} - m * a_w * log_a(Y_{min}) = YS_{max} - m * a_w * log_a(Y_{max}) = \frac{YS_{min} * log_a(Y_{max}) - YS_{max} * log_a(Y_{min})}{log_a\left(\frac{Y_{max}}{Y_{min}}\right)}$$

Bardziej złożone przekształcenie zachodzi dla osi poziomych, dla której argument obliczamy na podstawie współrzędnej punktu na ekranie. Funkcją odwrotną do logarytmicznej jest funkcja wykładnicza, stąd graficzne przekształcenie osi poziomych wygląda następująco:

rysunek 5



Otrzymujemy następującą funkcję przekształcającą:
$X = t * a!(p * XS)$ gdzie:
t — znak potęgowania
a — podstawa funkcji wykładniczej

$$p = \frac{log_a\left(\frac{X_{max}}{X_{min}}\right)}{XS_{max} - XS_{min}}$$

$$t = a^{\left(\frac{XS_{max} * log_a(X_{min}) - XS_{min} * log_a(X_{max})}{(XS_{max} - XS_{min})}\right)}$$

Po tych przerażających obliczeniach pora przejść do części praktycznej. Program pozwala wyrysować na dowolnym obszarze ekranu obrysowanego ramką funkcję wprowadzoną przez użytkownika. Tam, gdzie to było możliwe, zachowałem nazwy zmiennych używanych w części teoretycznej artykułu, aby łatwo można było prześledzić proces obliczeniowy.

W skali logarytmicznej oś pionowa przedstawiana jest w [*dB*].
Jeśli funkcja wychodzi poza ramkę, to program kontynuuje obliczanie, ale nie stawia punktu.

```
  10 REM ************************
  11 REM
  12 REM
  13 REM     Krzysztof Poźniak
  14 REM
  15 REM    Skalowanie  wykresu
  16 REM
  17 REM                    ©1988
  18 REM
  19 REM ************************
  80 LET XSkon=255: LET YSkon=17
5
  90 REM wprowadzanie danych
 100 INPUT "podaj wzor funkcji "
;"y=f(x)="; LINE x$
 110 INPUT "podaj minimalna wart
osc argumentu Xmin=";Xmin;"podaj
 maksymalna wartosc argumentu Xm
ax=";Xmax
 111 IF Xmin>=Xmax THEN GO TO 11
0
 120 INPUT "podaj minimalna wart
osc na osi Y: Ymin=";Ymin;"podaj
 maksymalna wartosc na osi Y: Ym
ax=";Ymax
 121 IF Ymin>=Ymax THEN GO TO 12
0
 130 INPUT "podaj minimalna wspo
lrzedna ekranu na osi X: XSmin="
;XSmin;"podaj maksymalna wspolrz
edna ekranu na osi X: XSmax=";XS
max
 131 IF XSmin>=XSmax OR XSmin<0
OR XSmax>XSkon THEN GO TO 130
 132 LET XSmin=INT XSmin: LET XS
max=INT XSmax
 140 INPUT "podaj minimalna wspo
lrzedna ekranu na osi Y: YSmin="
;YSmin;"podaj maksymalna wspolrz
edna ekranu na osi Y: YSmax=";YS
max
 141 IF YSmin>=YSmax OR YSmin<0
OR YSmax>YSkon THEN GO TO 140
 142 LET YSmin=INT YSmin: LET YS
max=INT ySmax
 200 CLS : PRINT "funkcja y=f(x)
=";x$;"Xmin=";Xmin,"Xmax=";Xmax;
"Ymin=";Ymin,"Ymax=";Ymax;"XSmin
=";XSmin,"XSmax=";XSmax;"YSmin="
;YSmin,"YSmax=";YSmax
 205 PRINT ;"opcje:";"0-nowe da
ne";"1-skala liniowa";"2-skala l
ogarytmiczna"
 210 LET i$=INKEY$:
 211 IF i$="0" THEN GO TO 100
 212 IF i$="1" THEN GO TO 300
 213 IF i$="2" THEN GO TO 400
 215 GO TO 210
 300 REM skala liniowa
 310 LET ax=(Xmax-Xmin)/(XSmax-X
Smin): LET bx=(Xmin*XSmax-Xmax*X
Smin)/(XSmax-XSmin)
 315 LET ay=(YSmax-YSmin)/(Ymax-
Ymin): LET by=(YSmin*Ymax-YSmax*
Ymin)/(Ymax-Ymin)
 320 GO TO 500
 400 REM skala logarytmiczna-dec
ybelowa
 410 IF Xmin<=0 THEN PRINT "skal
a rzednych X ma wartosci ujemne"
 411 IF Ymin<=0 THEN PRINT "skal
a odcietych y ma wartosci ujemne
"
 412 IF Xmin<=0 OR Xmax<=0 THEN
BEEP .1,20: PAUSE 1: PAUSE 0: GO
 TO 200
 415 LET m=20
 420 LET aw=(YSmax-YSmin)/(m*LN
(Ymax/Ymin)/LN 10)
 421 LET bw=(YSmin*LN Ymax-YSmax
*LN Ymin)/LN (Ymax/Ymin)
 425 LET bw=(YSmin*LN Ymax-YSmax
*LN Ymin)/LN (Ymax/Ymin)
 430 LET p=LN (Xmax/Xmin)/(XSmax
-XSmin)
 431 LET t=EXP ((XSmax*LN Xmin-X
Smin*LN Xmax)/(XSmax-XSmin))
 500 REM ramka
 505 CLS : PLOT XSmin,YSmin: DRA
W XSmax-XSmin,0: DRAW 0,YSmax-YS
min: DRAW XSmin-XSmax,0: DRAW 0,
YSmin-YSmax
 520 REM petla rysowania funkcji
 530 FOR k=XSmin TO XSmax
 540 IF i$="1" THEN LET x=ax*k+b
x
 541 IF i$="2" THEN LET x=t*EXP
(k*p)
 550 IF i$="1" THEN LET YS=ay*VA
L x$+by
 551 IF i$="2" THEN LET YS=m*aw*
LN (VAL x$)/LN 10+bw
 560 IF YS>=YSmin AND YS<=YSmax
THEN PLOT k,YS
 570 NEXT k
 580 PRINT #1;"wcisnij dowolny k
lawisz": PAUSE 1: BEEP .1,20: PA
USE 0: GO TO 200
```

K. POŹNIAK

Inteligentny samochód (?)

W ponad stuletniej historii motoryzacji przyzwyczailiśmy się, że charakterystycznymi elementami każdego samochodu są cztery koła, kierownica i silnik. Tymczasem niepostrzeżenie do elementów tych dołączył komputer.

W czołowych firmach samochodowych świata przeprowadzane są zaawansowane prace nad pojazdami XXI wieku, w których komputer będzie odgrywał niezwykle istotną rolę. Jego podstawowym zadaniem ma być sterowanie pracą silnika oraz wszystkich podzespołów mechanicznych, a także ich kontrolowanie. W wypadku wykrycia nieprawidłowości w pracy pojazdu kierowca będzie o tym natychmiast informowany wraz ze wskazaniem sposobu naprawienia samochodu. Jeżeli kierowca nie będzie w stanie poradzić sobie z naprawą, komputer poinformuje go, gdzie znajduje się najbliższy serwis.

Komputer będzie także wspomagał kierowcę w prowadzeniu pojazdu. Każdy samochód będzie wyposażony w system automatycznego utrzymywania bezpiecznej odległości od innych użytkowników drogi. Radarowe czujniki będą śledzić otoczenie pojazdu i w razie wykrycia zagrożenia komputer automatycznie uruchomi hamulec lub przyspieszy w razie potrzeby. Proces hamowania ma być oczywiście kontrolowany, przez co uniknie się blokowania hamulców.

Znacznie rozbudowany będzie także system komputerowego radia kierowców. Kierowca wsiadając do samochodu wprowadzi do pamięci komputera informację o docelowym miejscu podróży. W miastach oraz wzdłuż wszystkich dróg publicznych rozmieszczone będą nadajniki przekazujące drogą radiową komunikaty o aktualnych warunkach drogowych i atmosferycznych. Komputer automatycznie śledząc wszystkie komunikaty na bieżąco naniesie poprawki dotyczące trasy przejazdu.

W każdym samochodzie znajdzie się także radiotelefon, za którego pomocą będzie można uzyskać połączenie z dowolnym abonentem, w dowolnym miejscu.

Elektroniczny numer rejestracyjny umożliwi w razie potrzeby znajdowanie skradzionych samochodów oraz identyfikację pojazdów, które naruszyły przepisy drogowe.

Komputery wyręczą także policję drogową w kontrolowaniu ruchu pojazdów. Wszelkie przewinienia drogowe za sprawą odpowiednio rozmieszczonych kamer będą automatycznie wychwytywane i każdy użytkownik drogi, który dopuścił się wykroczenia, będzie natychmiast identyfikowany i drogą pocztową otrzyma do zapłacenia stosowny przekaz.

MBO

Przesyłanie kodu maszynowego do instrukcji REM

Prezentowany program jest wygodnym narzędziem dla tych, którzy chcą przechowywać ciągi bajtów (m. in. kody maszynowe) w programie Basic'a. Program wprowadza dowolnie długi ciąg bajtów z każdej części pamięci komputera do linii komentarza **REM**, którą sam tworzy.

Procedurę wywołujemy przez **FN**f(*a, b, c*), gdzie:

a — adres początku bajtów w pamięci, 0≤a≤65535

b — ilość bajtów, 0≤b≤65535

c — numer linii **REM**, w której umieścimy bajty, 1≤c≤9999

Po wykonaniu procedury funkcja użytkownika (**FN**) przekazuje adres pierwszego elementu ciągu bajtów (można go użyć jako adresu startu kodu maszynowego).

Program jest zabezpieczony przed błędnymi danymi. W momencie wykrycia błędu system przerywa pracę i wypisuje odpowiedni komunikat:

Q Parrametr error — gdy nie ma zgodności ilości parametrów lub gdy parametry nie zawierają się w dopuszczalnych granicach.

B Integer out of range — gdy numer linii wynosi 0 lub przekracza 9999.

4 Out of memory — gdy zabraknie miejsca na utworzenie linii.

G No room for line — gdy próbujemy utworzyć linię o numerze linii już istniejącej w programie.

Dodatkowa funkcja: *jeżeli ustalimy długość ciągu bajtów równą zero, to procedura nie utworzy nowej linii tylko poda przez FN adres początku bajtów w tej linii.*

Ważna uwaga: *ponieważ tworzenie linii przesuwa obszary pamięci, to aby uniknąć błędów systemu, należy utworzyć linię* **REM** *za linią zawierającą definicję funkcji (***DEF FN***) i linią wywołującą funkcję (***FN***). W wypadku prezentowanego programu za linią 300.*

Program jest relokowalny i można go umieścić w dowolnym miejscu za **RAMTOP-em**, tak aby nie kolidował z innymi procedurami maszynowymi.

```
10 REM ************************
11 REM
12 REM     Krzysztof Poźniak
13 REM
14 REM    'przesylanie kodu'
15 REM    ' maszynowego do '
16 REM    ' instrukcji REM '
17 REM
18 REM                  ©1988
19 REM
20 REM ************************


  25 CLEAR 65299
  30 DEF FN f(a,b,c)=USR 65300
 100 REM wczytywanie kodu
 110 RESTORE 9000
 120 FOR k=0 TO 9
 130 LET s1=0: LET s2=0: LET z=1
 140 FOR l=1 TO 15
 150 READ a: POKE 65299+k*15+l,a
: LET s1=s1+a: LET s2=s2+z*a: LE
T z=-z
 160 NEXT l: READ sk1,sk2: IF sk
1<>s1 OR sk2<>s2 THEN BEEP .1,20
: PRINT "blad w linii ";9000+10*
k: STOP
 170 NEXT k
 200 CLS : BEEP .1,20: PRINT "pr
ogram zostal wczytany"
 210 INPUT "podaj adres poczatku
 kodu maszynowego: ";adres
 220 INPUT "podaj dlugosc ciagu
bajtow: ";dlugosc
 230 INPUT "podaj nr linii przez
naczonej na zapis bajtow: ";nrli
nii
 300 CLS : PRINT "adres poczatku
 ciagu bajtow :",FN f(adres,dlug
osc,nrlinii)
 310 LIST nrlinii



9000 DATA 221,42,11,92,6,3,221,2
29,225,35,35,175,17,4,0,1316,156
9010 DATA 182,35,182,35,35,35,18
2,25,16,246,254,0,32,39,221,1519
,689
9020 DATA 78,12,221,70,13,3,3,19
7,3,3,3,3,197,205,5,1016,30
9030 DATA 31,221,110,20,221,102,
21,125,180,40,7,229,1,240,216,17
64,-190
9040 DATA 9,48,15,46,10,195,85,0
,46,25,195,85,0,46,15,820,-70
9050 DATA 195,85,0,225,205,110,2
5,8,221,126,12,221,182,13,40,166
8,92
9060 DATA 50,8,40,235,43,193,205
,85,22,35,35,221,126,21,119,1438
,-158
9070 DATA 221,126,20,35,119,35,1
93,113,35,112,35,54,234,19,62,14
13,425
9080 DATA 13,18,35,235,221,110,4
,221,102,5,221,78,12,221,70,1566
,-210
9090 DATA 13,213,237,176,193,201
,1,5,0,9,193,193,229,24,245,1932
,290
```

```
00001  ;*********************************
00002  ;*                               *
00003  ;*     Krzysztof Poźniak         *
00004  ;*                               *
00005  ;*   program przesyla kod        *
00006  ;*  maszynowy do instrukcji       *
00007  ;*    jezyka Basic: REM          *
00008  ;*                               *
00009  ;*                    ©1988      *
00010  ;*********************************
00011  ;
00012  ; program jest relokowalny
00013  ; i calkowicie zabezpieczony
00014  ; przed blednymi danymi przesy-
00015  ; lanymi z poziomu Basic'a przy
00016  ; pomocy funkcji uzytkownika FN
00017  ;
00018  ; adresy procedur ROM-u
00019  linead equ  196eh
00020  makero equ  1655h
00021  testro equ  1f05h
00022  error  equ  55h
00023  ; adres zmiennej systemowej
00024  defadd equ  5c0bh
00025  ; wartosci stale programu
00026  nrmax  equ  9999
00027  illicz equ  3
00028  skok   equ  4
00029  ENTER  equ  13
00030  REM    equ  234
00031  komunB equ  10
00032  komunG equ  15
00033  komunQ equ  25
00034  ;
00035  ;        P R O G R A M
00036  ;
00037  start  ld   ix,(defadd)
00038  ;kontrola poprawnosci danych
00039         ld   b,illicz
00040         push ix
00041         pop  hl
00042         inc  hl
00043         inc  hl
00044         xor  a
00045         ld   de,skok
00046  petla  or   (hl)
00047         inc  hl
00048         or   (hl)
00049         inc  hl
00050         inc  hl
00051         inc  hl
00052         or   (hl)
00053         add  hl,de
00054         djnz petla
00055         cp   0
00056         jr   nz,blad_Q
00057         ld   c,(ix+12)
00058         ld   b,(ix+13)
00059         inc  bc
00060         inc  bc
00061         push bc
00062         inc  bc
00063         inc  bc
00064         inc  bc
00065         inc  bc
00066         push bc
00067         call testro
00068         ld   l,(ix+20)
00069         ld   h,(ix+21)
00070         ld   a,l

00071         or   h
00072         jr   z,blad_B
00073         push hl
00074         ld   bc,-nrmax-1
00075         add  hl,bc
00076         jr   nc,dalej
00077  ; obsluga bledow
00078  blad_B ld   l,komunB
00079         jp   error
00080  blad_Q ld   l,komunQ
00081         jp   error
00082  blad_G ld   l,komunG
00083         jp   error
00084  dalej  pop  hl
00085         call linead
00086         ex   af,af'
00087         ld   a,(ix+12)
00088         or   (ix+13)
00089         jr   z,adres
00090         ex   af,af'
00091         jr   z,blad_G
00092  ; dane poprawne
00093  ; czesc wykonawcza
00094         dec  hl
00095         pop  bc
00096         call makero
00097         inc  hl
00098         inc  hl
00099         ld   a,(ix+21)
00100         ld   (hl),a
00101         ld   a,(ix+20)
00102         inc  hl
00103         ld   (hl),a
00104         inc  hl
00105         pop  bc
00106         ld   (hl),c
00107         inc  hl
00108         ld   (hl),b
00109         inc  hl
00110         ld   (hl),REM
00111         inc  de
00112         ld   a,ENTER
00113         ld   (de),a
00114         inc  hl
00115         ex   de,hl
00116         ld   l,(ix+4)
00117         ld   h,(ix+5)
00118         ld   c,(ix+12)
00119         ld   b,(ix+13)
00120         push de
00121         ldir
00122  koniec pop  bc
00123         ret
00124  adres  ld   bc,5
00125         add  hl,bc
00126         pop  bc
00127         pop  bc
00128         push hl
00129         jr   koniec
00130         end
```

K. POŹNIAK

COMMODORE C-64 — edytor tekstu

T. CISEK

W cyklu artykułów pod tym tytułem autor pragnie podzielić się z czytelnikami „IKS-a" swoimi doświadczeniami w stosowaniu różnych edytorów tekstu dla komputera Commodore 64. Omówiony zostanie sposób ich działania (instrukcje), możliwości, wady i zalety.

Autor „wypróbował" wszystkie omówione edytory, dysponując następującym wyposażeniem: drukarką OKIDATA 120 (120 znaków/s w trybie draft i 20 w trybie NLQ), stacją dysków FSD-1 „EXCELERATOR" +", kompatybilną ze stacją 1541 i oczywiście mikrokomputerem C-64.

GEOWRITE

Edytor tekstu Geowrite stanowi część programu GEOS (Graphic Environment Operating System), który został opracowany w 1985 r. przez firmę Berkeley Softworks z Kalifornii. Poniższy opis dotyczy edytora zawartego w programie GEOS, wersja 1.2.

1. Uruchamianie edytora

Uruchamianie programu Geowrite odbywa się w sposób taki jak każdego innego zawartego w Geosie, to znaczy można dokonać tego przez:

a) jednokrotne naciśnięcie przycisku Fire joysticka po ustawieniu wskaźnika (strzałki) na ikonie Geowrite i otwarcie pliku przez wybranie „OPEN" z menu „FILE".

b) dwukrotne, szybkie naciśnięcie przycisku Fire na ikonie Geowrite. Po załadowaniu Geowrite należy dokonać wyboru jednej z trzech możliwości: CREATE — rozpoczęcie nowego dokumentu, OPEN — otwarcie dokumentu istniejącego, QUIT — zakończenie pracy edytora. Należy zaznaczyć, że wszystkie rozkazy i funkcje edytora wybierane są za pomocą strzałki kierowanej joystickiem i naciśnięciu przycisku *FIRE*.

Po wybraniu CREATE należy podać nazwę dokumentu. Po wybraniu OPEN program wyświetla nazwy plików (dokumentów) znajdujących się na dyskietce, z których jeden może zostać wprowadzony do pamięci komputera. Można również otworzyć istniejący dokument w taki sam sposób, w jaki został załadowany Geowrite (dwie możliwości). Przy takim postępowaniu Geowrite zostanie załadowany automatycznie.

2. Zawartość ekranu (okna) Geowrite

Po uruchomieniu edytora i wybraniu CREATE ekran Geowrite wygląda tak jak na rys. 1, przy założeniu, że nowo tworzony dokument został zatytułowany „Test".

Rozwinięcie poszczególnych funkcji zawartych w menu wygląda następująco:

GEOS	FILE	EDIT
GEOWRITE INFO	CLOSE	CUT
CALCULATOR	UPDATE	COPY
PREFERENCE MGR	PREVIEW	PASTE
NOTE PAD	RECOVER	
PHOTO MANAGER	RENAME	
TEXT MANAGER	PRINT	
ALARM CLOK	QUIT	

OPTIONS	FONT	STYLE
PREVIOUS PAGE	BSW	PLAIN TEXT
NEXT PAGE	EWANS	BOLD
GOTO PAGE	HEARST	ITALIC
HIDE PICTURES	CORY	OUTLINE
PAGE BREAK	ROMA	UNDERLINE
LAST PAGE	ORMAND	
	DWINELLE	

3. Pisanie tekstu

Miejsce, w którym tekst jest pisany wskazuje kursor. Za pomocą wskaźnika kursor można ustawić w dowolnym miejscu (naciśnięcie przycisku *FIRE*). Poszczególne znaki mogą być usuwane za pomocą klawisza DEL lub SPACE. Piszący widzi tylko ok. 2/3 szerokości okna, stąd aby uniknąć męczącego oczy przechodzenia z prawej strony na lewą stronę ekranu i odwrotnie, należy prawy margines przesunąć w lewo, tak aby szerokość tekstu nie była większa niż 5 cali (patrz: formatowanie tekstu). Podczas pisania Geowrite przenosi oczywiście cały wyraz do następnego wiersza, jeżeli on nie mieści się w wierszu poprzednim. Naciśnięcie RETURN powoduje utworzenie akapitu. Tablicowanie odbywa się za pomocą klawiszy *CTRLi-I*. Pozostawienie części strony pustej (nie zapisanej) i przejście do następnej strony dokonuje się przez wybranie PAGE BREAK z menu OPTIONS. Przeglądanie zawartości strony — wybranie PREVIEW z menu FILE.

5. Poruszanie się po tekście (ekranie)

Poruszanie się po ekranie (dokumencie) w prawo — lewo odbywa się przez:

— ustawienie kursora za pomocą wskaźnika w dowolnym miejscu dokumentu,

— zaczepienie wskaźnika (naciśnięcie FIRE) na okienku reprezentującym ekran w SCROLL BOX.

Poruszanie się po ekranie w górę — w dół:

— przesuwanie o jeden wiersz — naciskanie *FIRE* na strzałkach przesuwu tekstu (okna),

— zaczepienie wskaźnika na okienku reprezentującym ekran w SCROLL BOX.

Przejście do przodu lub cofnięcie się o jedną stronę:

— wybranie *NEXT PAGE* lub *PREVIOUS PAGE* z menu OPTIONS.

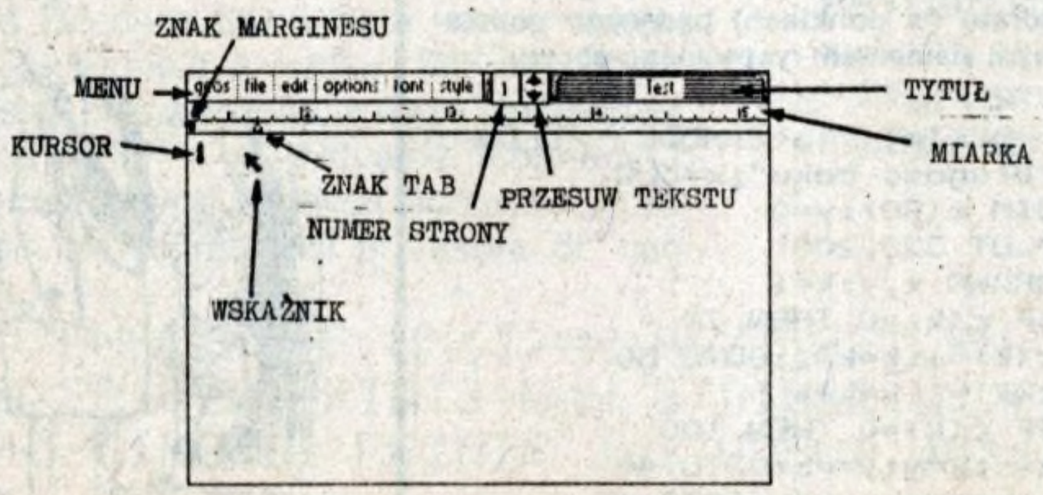
Przejście do dowolnej strony:

— wybranie GOTO PAGE z menu OPTIONS, podanie numeru strony i naciśnięcie RETURN.

6. Edycja tekstu

Wyróżnić tutaj można następujące funkcje:

— wprowadzenie tekstu do tekstu istniejącego (napisanego wcześniej) — ustawienie kursora w żądanym miejscu w tekście istniejącym. Tekst wpisywany będzie powodował „rozsuwanie" tekstu istniejącego. Jeżeli istniejący tekst (dokument) jest wielostronicowy, to również nastąpią przesunięcia w kolejnych stronach,



Rys. 1. Ekran GEOWRITE

— określenie (wybieranie) tekstu przeznaczonego do przeniesienia, kopiowania lub usunięcia — żądany tekst należy najpierw „podświetlić przesuwając wskaźnik przy wciśniętym przycisku *FIRE*. Przesuwanie poziome powoduje podświetlanie liter i słów. Przesuwanie pionowe podświetla kolejne wiersze.

Po podświetleniu tekstu (fragmentu) możliwe są następujące operacje:

— zastąpienie go innym tekstem wprowadzonym z klawiatury,

— usunięcie — naciśnięcie *DEL*,

— zmiana rodzaju lub stylu tekstu (*FONT* i *STYLE*),

— *CUT* — wycięcie fragmentu z jednoczesnym umieszczeniem go w TEXT SCRAP (wycinek tekstu),

— *COPY* — skopiowanie fragmentu do TEXT SCRAP bez usunięcia z miejsca podświetlonego,

— PASTE — przeniesienie zawartości TEXT SCRAP do tego miejsca w tekście, gdzie aktualnie znajduje się kursor.

7. **Ustawienie marginesów**

Możliwe jest dowolne ustawienie marginesów przez zaczepienie wskaźnika na symbolu marginesu (M) i przesunięcie w dowolne miejsce.

8. **Używanie różnych rodzajów i stylów pisma**

Do ustalania i zmiany rodzaju i stylu pisma służą funkcje FONT i STYLE. Dokonać tego można w dowolnej chwili, a zmian w tekście już istniejącym dokonuje się po uprzednim jego podświetleniu.

9. **Inne możliwości Geowrite**

— tablicowanie tekstu — Geowrite umożliwia zakładanie tablicy o ośmiu pozycjach. Dokonuje się tego po ustawieniu wskaźnika tablicy w żądanym miejscu w taki sam sposób, jak ustawia się margines,

— usuwanie pustego miejsca na stronie (miejsca powstałego po użyciu funkcji *PAGE BRAKE*) — ustawienie kursora w górnym lewym rogu strony następnej, naciśnięcie *DEL* i wybranie *OK* po ukazaniu się pytania, czy chcemy usunąć ostatni znak na poprzedniej stronie.

10. **Uwagi końcowe**

Maksymalna długość jednego tekstu (dokumentu) pisana za pomocą Geowrite wynosi 64 strony. Przy pisaniu długich tekstów korzystne jest posługiwanie się pismem normalnym (BSW) o wysokości 9 punktów i częste używanie funkcji *PAGE BREAK*.

Jeżeli podczas pracy w Geowrite obawiamy się przerwy w zasilaniu należy używać funkcji *UPDATE* w celu zapisania napisanych już fragmentów tekstu. Po zakończeniu pisania strony następuje jej automatyczne zapisanie na dyskietce.

W tekście pisanym za pomocą omawianego edytora można umieszczać rysunki wykonane za pomocą programu GEOPAINT zawartego również w GEOSIE.

Drukowanie dokumentu odbywa się przez wybranie *PRINT* z menu FILE. Po zakończeniu dokumentu należy go zamknąć wybierając *CLOSE* z menu FILE.

Proba pisma: font BSW, 9 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma: font ROMA , 9 pkt: plain, **bold**, *italic*; outline, underline
Proba pisma font ROMA, 12 pkt: plain, **bold**, *italic*; outline, underline
Proba pisma font ROMA, 18 pkt: plain, bold, *italic*, outline, underline
Proba pisma font ROMA, 24 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font University 6 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font University 10 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font University 12 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font University 14 pkt : plain, **bold**, *italic*; outline, underline
Proba pisma font University 18 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font University 24 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font Dwinelle 18 pkt : plain, bold, *italic*outline, underline
Proba pisma font Cory 12 pkt: plain, bold, *italic*, outline, underline
Proba pisma font Cory 24 pkt : plain, bold, *italic*, outline, underline
Proba pisma font California 10 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font California 12 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font California 14 pkt: plain, **bold**, *italic*, outline, underline
Proba pisma font California 18 pkt: plain, bold, *italic*, outline, underline

Rys. 2. Niektóre rodzaje i style pisma GEOWRITE.

11. **Ocena edytora**

Wady:

— brak numeracji stron,

— brak możliwości ustawienia odstępu między wierszami (liczby wierszy na stronie),

— konieczność posługiwania się joystickiem lub myszką,

— brak polskich liter.

Zalety:

— szybkie wybieranie poszczególnych funkcji i rozkazów,

— bardzo bogaty zestaw rodzajów i stylów pisma (rys. 2),

— możliwość łączenia tekstu z rysunkami wykonanymi za pomocą GEOPAINT.

T. CISEK

CPC • CPC • CPC • CPC • CPC • CPC • CPC • CPC

Ten program pozwala na uzyskanie ciekawego efektu animacji grafiki trójwymiarowej.

```
10    INK    0,0:INK    1,26:MODE
0:st=4:loop=14
20 FOR n=0 TO 640 STEP st
30 c=c+0.5:IF c=15.5 THEN c=1
40 PLOT 1000,1000,c:z=2*n-320
50  MOVE  z,200:DRAW   n,150:DRAW
n,40
60 MOVE z,0:DRAW n,40
70  MOVE  z,400:DRAW   n,350:DRAW
n,240
80 MOVE z,200:DRAW n,240
90 NEXT
100 INK 1,7:INK 15,0:FOR  n=2  TO
15:INK n,15:INK n-1,0:FOR t=0  TO
loop:NEXT:NEXT:GOTO 100
```

Jest to ciekawy program graficzny imitujący ruch ważki. Parametrem jest długość boku kwadratu (w punktach) będącego podstawowym elementem rysowanego obrazu.

```
10   DEFINT   a-z:MODE   1:INPUT
   "Dlugosc boku";x:CLS
20 DIM c(20):y=0
30 PLOT 320,200
40 DRAWR x,y:k=1
50 IF c(k)=0 THEN 70
60 c(k)=1:k=k+1:GOTO 50
70 c(k)=1:k=k+1
80 IF c(k)=0 THEN 100
90 z=x:x=y:y=-z:GOTO 40
100 z=x:x=-y:y=z:GOTO 40
```

Cezary SOBCZAK

— Czy mogę wypożyczyć Józia do ustawienia oprogramowania naszego komputera?

COMMODORE

PROGRAM 55

Zmiana nazwy dyskietki

Program pozwala na dowolną zmianę nazwy dyskietki. Wykonywanie tej czynności jest czasem konieczne i pomaga utrzymać ład w naszej „dyskotece".

```
10 PRINT '■■ ZMIANA NAZWY DYSKIETKI'
20 PRINT '■ NR STACJI DYSKOW NR (8-15) ?
 8 ■■■■';
30 INPUT U: IF U<8 OR U>15 THEN PRINT '■
■■': GOTO 20
40 PRINT '■ DYSK DRIVE NR (0/1) ? 0■■■';

50 INPUT D: IF D<0 OR D>1 THEN PRINT '■■
■': GOTO 40
60 PRINT '■■ UMIESC DYSKIETKE W STACJI N
R';U;' DRIVE';D
70 PRINT ' I NACISNIJ DOWOLY KLAWISZ'
80 GET X$: IF X$=''THEN 80
90 A=1: V=16: Q$=CHR$(34): P$=CHR$(160)
100 T=18: S=0: I$='I'+RIGHT$(STR$(D),1)
110 OPEN 1,U,15,I$: GOSUB 350
120 GOSUB 310: F$=H$: GET#2,X$,X$,A$,B$
130 PRINT '■■■ AKTUALNA NAZWA ';Q$;F$;Q$

140 PRINT ' SYMBOL DYSKIETKI:  ';A$;B$

150 PRINT'■ NOWA NAZWA (MAX 16 ZNAKOW) L
UB          'STOP'-UTRZYMANIE STAREJ NAZ
WY'
160 INPUT ' ?STOP■■■■■';N$: L=LEN(N$):
IF L>V THEN 130
170 IF N$='STOP'   THEN PRINT '■';: GOTO
 370
180 PRINT '■ WPISANIE NOWEJ NAZWY (T/N)
? T■■■';
190 INPUT X$: IF X$<>'T' THEN 130
200 IF L=V THEN 220
210 FOR X=L+A TO V: N$=N$+P$: NEXT
220 PRINT#1,'B-P:';2;144: PRINT#2,N$;
230 PRINT#1,'U2:';2;D;T;S: GOSUB 350
240 PRINT#1,I$: GOSUB 350: CLOSE 2
250 PRINT '■ STARA NAZWA ';Q$;F$;Q$
260 GOSUB 310: CLOSE 2: CLOSE 1
270 PRINT '■ NOWA NAZWA ';Q$;H$;Q$
280 PRINT '■ NASTEPNA DYSKIETKA (T/N) ?
T■■■';
290 INPUT X$: IF X$='T' THEN 60

300 PRINT '■';: END
310 OPEN 2,U,2,'#': GOSUB 350: H$=''
320 PRINT#1,'U1:';2;D;T;S: GOSUB 350
330 PRINT#1,'B-P:';2;144: FOR X=A TO V
340 GET#2,T$: H$=H$+T$: NEXT: RETURN
350 INPUT#1,E,M$,J,K:IF E=0 THEN RETURN
360 PRINT '■ BLAD: ';E;M$;J;K
370 CLOSE 2: CLOSE 1: END
READY.
```

Opr. Tadeusz CISEK

Poznań, dnia 17 marca 1988 r.

Można inaczej

Po zapoznaniu się z artykułem p. Janusza Morbitzera „Generowanie liczb pierwszych — sito Eratostenesa" w zeszycie 1/1988 „IKS"-a nasunęło mi się kilka uwag, o których chciałbym poinformować Redakcję i czytelników.

We wspomnianym artykule liczba $2^{11213} - 1$ została podana jako największa znana liczba pierwsza. Tymczasem jest ona zaledwie dwudziestą trzecią kolejną liczbą pierwszą Mersenne'a, a w roku 1985 odkryta została już trzydziesta kolejna liczba pierwsza Mersenne'a, mianowicie liczba $2^{216091} - 1$. Aktualne informacje o dużych liczbach pierwszych podawał niedawno „Komputer", publikując list do redakcji profesora dr. hab. Władysława Narkiewicza (zeszyt 1/88) oraz mój list do redakcji (zeszyt 9/87).

Program podany przez p. Janusza Morbitzera ma dwie wady, mianowicie działa stosunkowo wolno oraz oblicza liczby pierwsze nie większe od 7000. Napisałem krótki program w Basicu przeznaczony dla **Commodore 64**, który wykorzystuje przyspieszoną wersję sita Eratostenesa. Po uruchomieniu programu za pomocą **RUN1** należy wprowadzić dowolną liczbę naturalną *N* nie przekraczającą 58127. Po naciśnięciu klawisza **RETURN** komputer podaje na ekranie wszystkie kolejne liczby pierwsze do *N*, a czasem i dalsze.

Przypuszczam, że załączony program zainteresuje młodych czytelników „IKS"-a.

Z poważaniem

Andrzej WIĘCKOWSKI

```
0 FORJ=JTONSTEPP+P:A%(J)=N:NEXT:RETURN
1 INPUTN:PRINT2:PRINT3:N=N/3:DIMA%(N):FORI=.TON:IFA%(I)GOTO3
2 P=I*3+5-(IAND1):PRINTP:J=I:GOSUB:J=P+P-I-3:IFJ<=NTHENGOSUB
3 NEXT

READY.
```

NUTY Z KOMPUTERA (2)

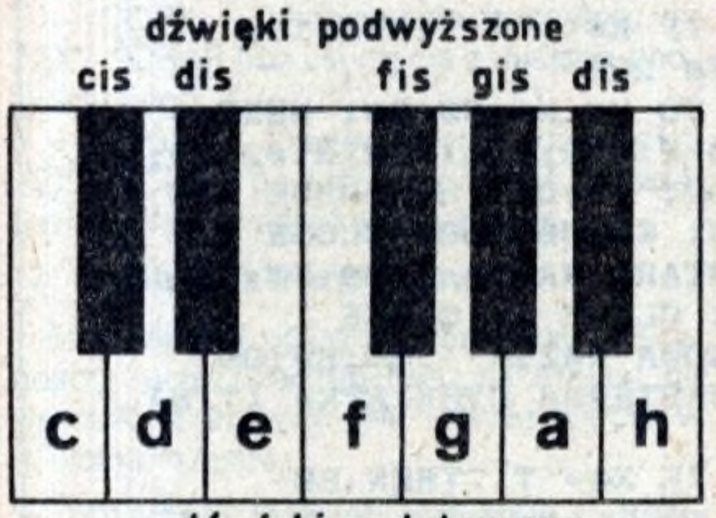
Krzysztof POŹNIAK, Michał ROLLINGER, Jarosław ZIEMBICKI

W kolejnych częściach artykułu chcemy przedstawić czytelnikom podstawowe funkcje sekwencera, zaproponować organizację programu i współpracę komputera z urządzeniami muzycznymi. Jest to bardzo rozległy problem i dlatego całość podzieliliśmy na odrębne części tematyczne. Ponieważ tylko pewna część zainteresowanych posiada ZX Spectrum (a na tym komputerze został napisany prezentowany sekwencer) postaramy się dokładnie opisać program, aby łatwo go było zaimplementować na inne komputery.

EDYTOR TAKTU

I. Informacje podstawowe:

a) Krok w takcie jest najmniejszą możliwą do wykonania jednostką rytmiczną. W zależności od stosowanego metrum i wartości rytmicznych uzyskujemy następujące ilości kroków w takcie:



rys. 1

b) Edytor dopuszcza takt składający się maksymalnie z 256 kroków i praktycznie nieograniczonej ilości ścieżek. Taka organizacja zapisu ukazuje pole edytora jako tablicę. W jego każdej komórce możemy zapisać informacje o wysokości jednego dźwięku.

c) Formy zapisu wysokości dźwięku:

1) zapis literowy — obejmuje dźwięki od C2 (C subkontry) do g7 (g siedmiokreślnego). Jest to zakres z nadmiarem, w praktyce wykorzystujemy dźwięki od A2 do c5 (klawiatura fortepianu). Ponieważ w pamięci komputera wysokość dźwięku jest zapisywana jako liczba, stąd nie możemy stosować różnych nazw dla tego samego dźwięku. Wykorzystujemy następującą skalę:

najmniejsza wartość rytmiczna :	♩	♪	𝅘𝅥𝅯	𝅘𝅥𝅰
metrum 2/4	2	4	8	16
metrum 3/4	3	6	12	24
metrum 4/4	4	8	16	32
metrum 5/4	5	10	20	40

rys. 2

Jeżeli w danym materiale nutowym występują dźwięki obniżone, to musimy zmienić je enharmonicznie na dźwięki podwyższone:

des = cis
es = dis
ges = fis
as = gis
b = dis

uwaga: ze względów graficznych dźwięki podwyższone są zapisywane jako dźwięki podstawowe z kreską nad literą.

2) zapis graficzny — jest używany do sterowania urządzeniami perkusyjnymi. Znak [ ] oznacza, że w danym kroku nastąpi wyzwolenie instrumentu

3) zapis cyfrowy — pozwala na wprowadzenie liczb z zakresu <0÷127>. Ta forma zapisu jest wygodna do pewnych specyficznych zastosowań, a jej wykorzystanie opiszemy w kolejnych częściach artykułu

uwaga: zapis literowy i cyfrowy dopuszczają dźwięki przedłużone (zapisywane jako —) oraz pauzy (oznaczane **)

d) struktura pola ekranowego edytora taktu:

1) po polu ekranowym poruszamy się kursorem za pomocą klawiszy:

klawisz <5> — lewo
klawisz <6> — dół
klawisz <7> — góra
klawisz <8> — prawo

2) ponieważ ilość kroków taktu może przekroczyć pojemność ekranu (maksymalna liczba wyświetlanych kroków wynosi 10), stąd takt dzieli się na strony. Stronicowanie pozwala nam (w zależności od długości taktu) na jego logiczny podział. Takt przegląda się stronami za pomocą klawiszy:

klawisz <4> — cofnięcie się o stronę
klawisz <9> — przejście na następną stronę

3) informacje dodatkowe:

— najwyższa linia ekranowa podaje nazwę taktu i jego stan ilościowy:
is = aktualna ilość ścieżek w takcie
ik = aktualna ilość kroków w takcie
s = numer wyświetlanej strony

— najniższa linia opisuje ścieżkę, na której znajduje się kursor

e) ścieżkę opisują następujące parametry:

1) nazwa instrumentu — ścieżka jest przypisana do jednego z obsługiwanych przez sekwencer instrumentów

2) nazwa źródła — ścieżka może obsługiwać tylko jedno źródło dźwięku w przyporządkowanym jej instrumencie

3) tryb zapisu — w zależności od rodzaju urządzenia muzycznego posługujemy się zapisem literowym, graficznym lub cyfrowym

II. Funkcje edytora:

uwaga: każdą funkcję edytora możemy przerwać naciskając klawisz <P>

klawisz <z> — umożliwia zapis dźwięku w miejscu położenia kursora, wg zadeklarowanego trybu:

a) tryb literowy — obejmuje dźwięki od C2 do g7. Najpierw wprowadza się literę (dużą lub małą), jeśli ma to być dźwięk podwyższony, to kolejnym krokiem jest wciśnięcie klawisza <i> (od przyrostka „is"); zapis kończymy wprowadzeniem cyfry

b) tryb graficzny — dopuszcza zapis uderzenia (klawisz <i>) lub pauzy (klawisz <0>)

c) tryb cyfrowy — pozwala na wprowadzenie liczby z zakresu <0÷127>. Ponieważ w polu pojedynczego dźwięku mieszczą się jedynie dwa znaki, stąd liczby od 100 wzwyż są wyświetlane w postaci dwóch ostatnich cyfr z kreską nad pierwszą cyfrą. Zakończenie wpisywania liczby uzyskujemy naciskając klawisz <SPACE>.

uwaga: w trybie literowym i cyfrowym pauzę uzyskujemy klawiszem <SPACE>, a przedłużenie dźwięku klawiszem <k>

uwaga: każda poniżej opisywana funkcja wymaga podania odpowiednich parametrów w postaci liczb. Po wprowadzeniu liczb należy wcisnąć klawisz <SPACE>. Ukaże się wtedy napis (t/n); wciśnięcie klawisza <t> wprowadzi daną informację do pamięci

klawisz <c> — umożliwia czyszczenie (wymazanie zawartości) podanej ścieżki
parametry: 1) numer ścieżki

klawisz <s> — skasowanie ścieżki o podanym numerze. Ilość ścieżek zmniejsza się o jedną
parametry: 1) numer ścieżki

klawisz <d> — dołączenie (wsunięcie) nowej ścieżki na pozycję o podanym numerze. Wszystkie ścieżki, począwszy od podanego numeru przesuną się o jedną pozycję w dół. Ich ilość wzrośnie o jedną.

uwaga: jeśli podany numer jest o jeden większy od ilości ścieżek, to nowa ścieżka zostanie dopisana na końcu
parametry: 1) numer ścieżki
2) numer instrumentu (wg tabeli)
3) numer źródła wybranego instr. (wg tabeli)
4) tryb zapisu

rys. 3

na rys. 3 przedstawiamy stan ekranu podczas wprowadzania czwartego parametru — trybu zapisu (wybraliśmy tryb graficzny — cyfra 2) po uprzednim wyborze instrumentu: TR 505 i źródła: hi-hat (nazwy w trybie inwersyjnym)

klawisz <w> — wymiana dwóch ścieżek miejscami. Funkcja ma charakter wyłącznie porządkowy
parametry: 1 numer pierwszej ścieżki (mniejszy)

2) numer drugiej ścieżki (większy)

klawisz <**p**> — powielanie treści jednej ścieżki na drugą. Warunkiem koniecznym wykonania funkcji jest zgodność typów zapisów

parametry: 1) numer ścieżki-źródła
2) numer ścieżki-odbiorcy

klawisz <**r**> — rozszerzenie taktu na zasadzie „mnożenia" pozwala, po jego napisaniu, na wprowadzenie mniejszych wartości rytmicznych niż początkowo zamierzaliśmy. Funkcja dopisuje pauzy i zmienia długości dźwięków. Przykładowe rozszerzenie taktu razy 2 przedstawiliśmy na rysunkach: nr 4 — stan wyjściowy i nr 5 — po wykonaniu funkcji

parametry: 1) mnożnik taktu

klawisz <**R**> — pozwala na wstawienie zadanej ilości dodatkowych kroków w określone miejsce taktu

uwaga: jeżeli numer kroku, od którego będziemy dołączać nowe kroki będzie o jeden większy od ich aktualnej liczby, to nowe kroki dołączą się na końcu taktu

parametry: 1) numer kroku, od którego nastąpi wstawienie
2) ilość dołączanych kroków

klawisz <**k**> — kompresja taktu na zasadzie „dzielenia" — działa odwrotnie do funkcji <**r**>

parametry: 1) podzielnik taktu

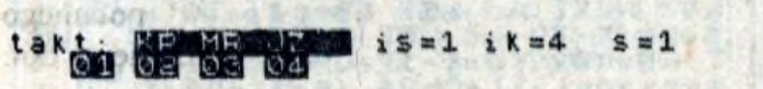
klawisz <**K**> — kasowanie grupy kroków — działa odwrotnie do funkcji <**R**>

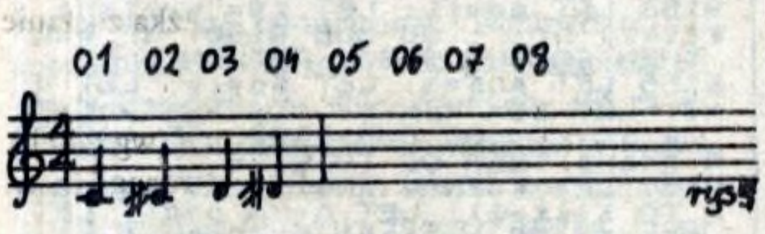
parametry: 1) numer kroku, od którego nastąpi kasowanie
2) ilość kasowanych kroków

klawisz <**i**> — określa ile kroków zawiera jedna strona taktu (w ZX Spectrum maksymalna liczba kroków na stronie wynosi 10)

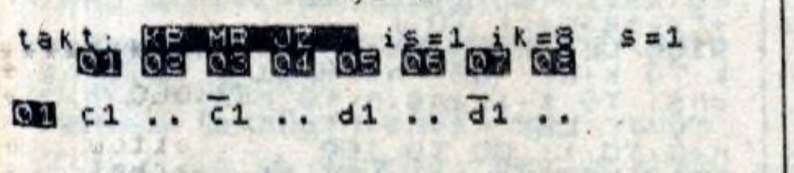
parametry: 1) ilość kroków na jednej stronie

klawisz <t> lub <**T**> — transpozycja ścieżki (w górę — klawisz <t>, dół — klawisz <**T**>) o zadany interwał (ilość półtonów)

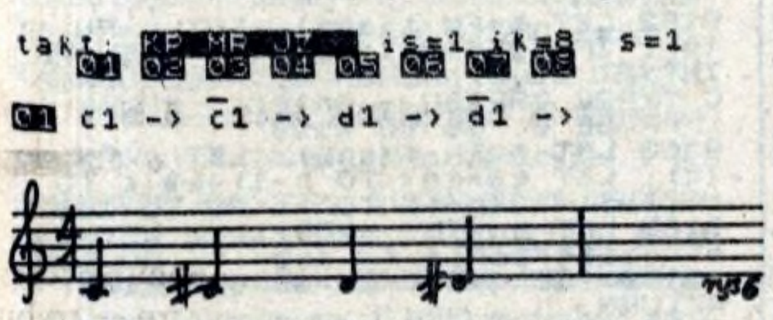




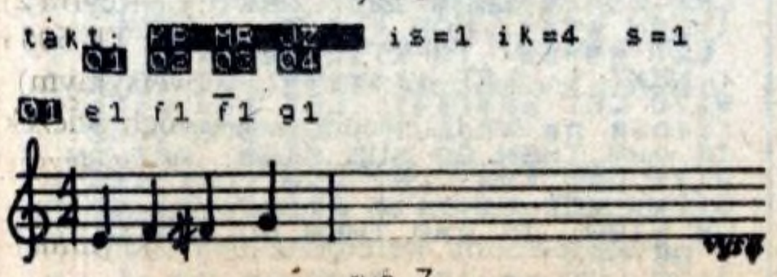
rys. 4



rys. 5



rys. 6



rys. 7

parametry: 1) numer transponowanej ścieżki
2) ilość półtonów

uwaga: transpozycja pozwala na zmianę tonacji bez żmudnego poprawiania wszystkich dźwięków

uwaga: jeżeli podczas transpozycji dźwięk przekroczy zakres C2÷g7, to na jego miejsce zostanie wprowadzona pauza

Przykładową zmianę tonacji o cztery półtony w górę przedstawiliśmy na rysunkach: nr 4 — stan wyjściowy i nr 7 — po wykonaniu transpozycji

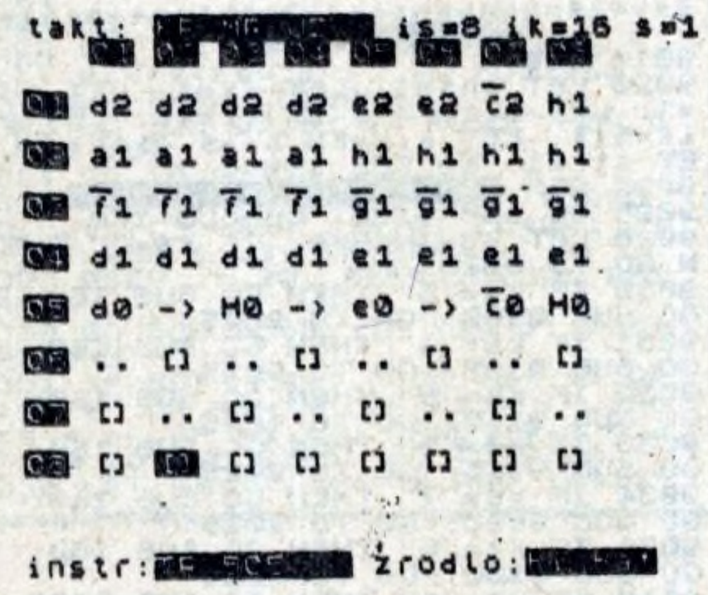
klawisz <**j**> — rozszerzenie jednostki rytmicznej — funkcja podobna do <**r**>, jednak z tą różnicą, że zachowuje długości dźwięków

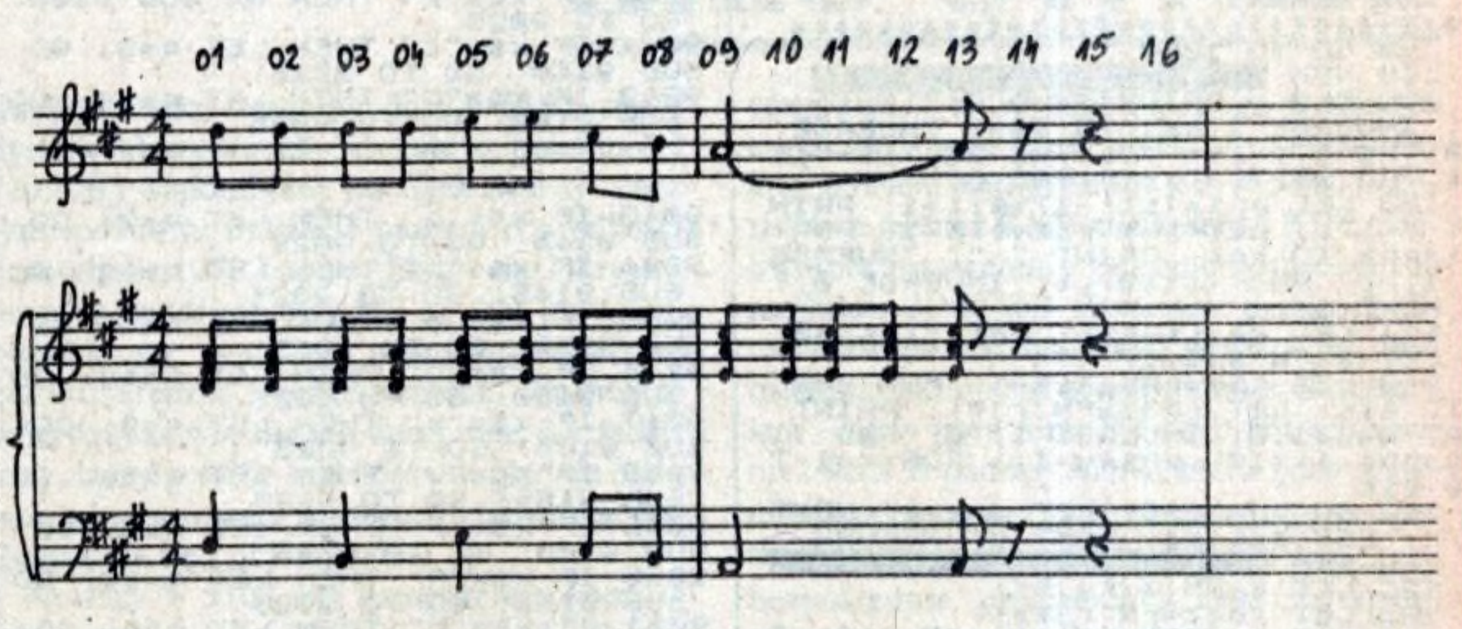
parametry: 1) mnożnik jednostki rytmicznej

Działanie tej funkcji jest uwidocznione na rysunkach: nr 4 — stan początkowy i nr 6 — sytuacja po przypisaniu wyjściowej jednostce rytmicznej dwóch kroków

uwaga: jeśli wywołujemy funkcje za pomocą dużych liter: <**C**>, <**S**>, <**D**>, <**W**>, <**P**>, to pierwszy parametr jest określony przez aktualną pozycję kursora

Na rysunku numer 8 został przedstawiony fragment utworu muzycznego w zapisie sekwencerowym i nutowym. Warto zwrócić uwagę na to, że „takt sekwencerowy" nie musi się pokrywać z taktem muzycznym.





rys. 8

3) **i$** — posiada nazwy dostępnych instrumentów muzycznych i informacje o niezależnych źródłach wchodzących w skład instrumentów

4) **s$** — przechowuje nazwy źródeł wszystkich instrumentów

b) najważniejsze zmienne proste:

ii — ilość dostępnych instrumentów (w programie przykładowo przyjęliśmy: ii = 3)

cike — całkowita ilość kroków na ekranie (dla ZX Spectrum cike = 10)

cise — całkowita ilość ścieżek na ekranie (dla ZX Spectrum cise = 8)

is — aktualna ilość ścieżek

ik — aktualna ilość kroków

pns — najmniejszy (początkowy) nr ścieżki wyświetlany na ekranie

kns — największy (końcowy) nr ścieżki wyświetlany na ekranie

pnk — najmniejszy (początkowy) nr kroku wyświetlany na ekranie

kns — największy (końcowy) nr kroku wyświetlany na ekranie

mik — maksymalna ilość kroków (ustalana funkcją <i>)

III. Zmienne programu:

a) dane edytora taktu są zapisywane w zmiennych tekstowych (wg schematów z rysunku numer 9)

1) **e$** — przechowuje dźwięki taktu (kolejno ścieżka po ścieżce)

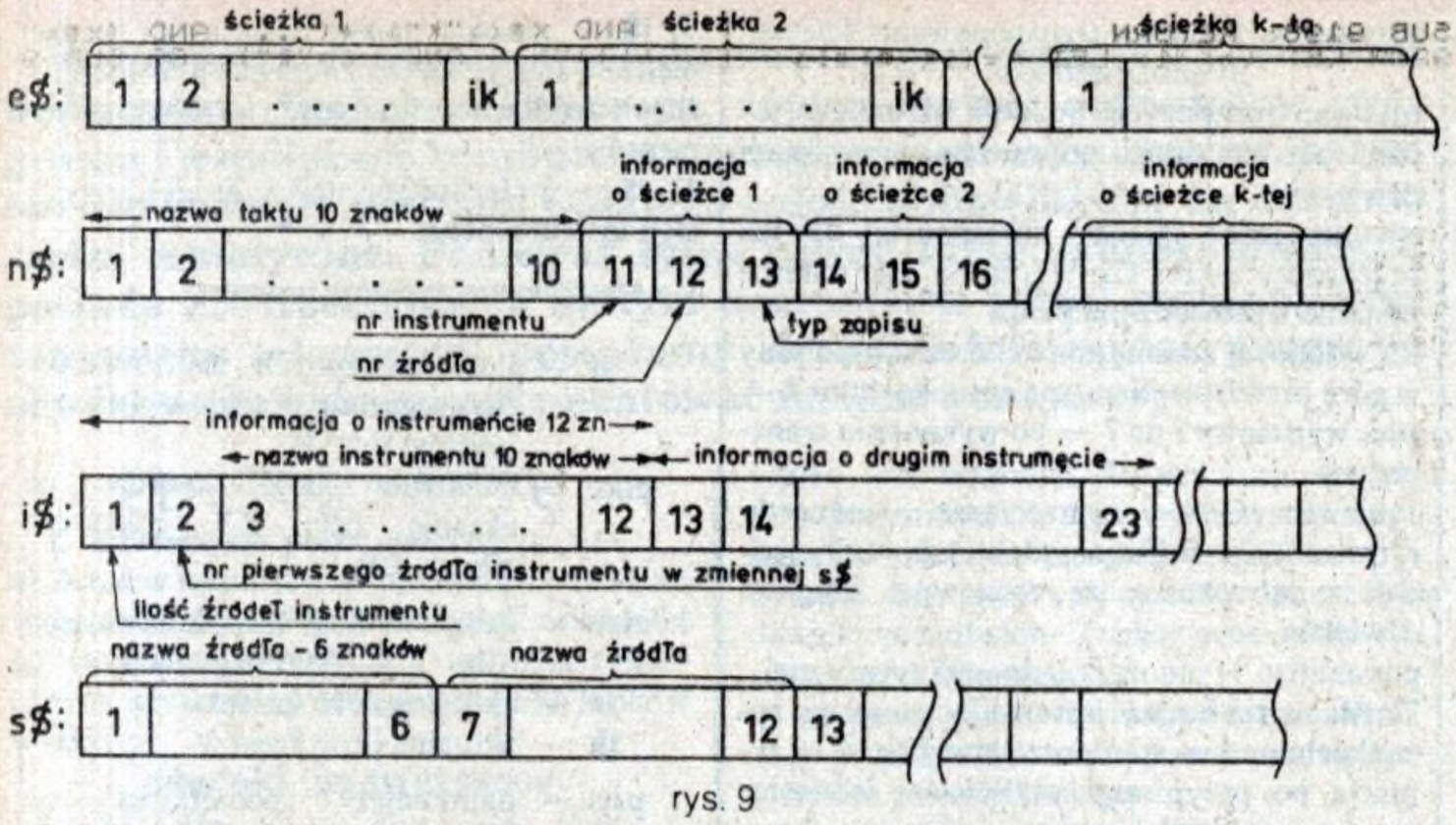
2) **n$** — zawiera nazwę taktu i informacje o każdej ścieżce

eik — ilość kroków aktualnie wyświetlanych na ekranie

eis — ilość ścieżek aktualnie wyświetlanych na ekranie

str — numer aktualnie wyświetlanej strony

kx, ky — współrzędne kursora



rys. 9

Ponieważ rozbudowa programu spowoduje pewne zmiany w prezentowanym listingu zachęcamy czytelników, którzy będą wprowadzać program do dokładnego jego odwzorowania.

```
   1 REM
************************************
        EDYTOR     TAKTU
************************************
   2 DEF FN r(x)=CODE n$(10+3*(p
ns+x-1))
   3 DEF FN m(y,x)=x*(x<y)+y*(x>
=y)
   4 DEF FN o(q)=ik*(q-1)+1
   5 DEF FN t(q)=11+(q-1)*3
   6 DEF FN a(y,x)=ik*(pns+y-2)+
mik*(str-1)+x
  10 INK 7: PAPER 0: CLS : BORDE
R 0
  15 REM Pętla testowa
  ************************************
**
  20 LET ii=3: LET i$=CHR$ 4+CHR
$ 1+"CZ 1000    "+CHR$ 6+CHR$ 5+"
TR 505     "+CHR$ 4+CHR$ 11+"DW 6
000    "
  21 LET s$="gener1gener2gener3g
ener4taktowwerbelpolk 1polk 2koc
iolhi hatgener1gener2gener3gener
4"
  30 LET n$=CHR$ 1+CHR$ 1+CHR$ 0
  31 LET n$="KP MR JZ   "+n$
  40 LET e$=CHR$ 128: LET cike=1
0: LET cise=8: LET is=1: LET ik=
1
  50 LET t$="c0c1d0d1e0f0f1g0g1a
0a1h021001234567"
  60 REM
************************************
  99 GO TO 1000
 100 REM Nowe położenie kursora
 102 CLS : PRINT AT 0,0;"takt: "
; INVERSE 1;n$( TO 10); INVERSE
0;TAB 17;"is=";is;TAB 22;"ik=";i
k;TAB 28;"s=";str;TAB 0;
 103 LET pnk=(str-1)*mik+1: PRIN
T "  ";: LET knk=pnk+eik-1: FOR
l=pnk TO knk: PRINT " "; INVERSE
 1;"0" AND (l<10);l; INVERSE 0;:
 NEXT l
 104 LET da=ik-eik: LET flash=0:
 LET a=FN a(1,1)
 110 LET kns=pns+eis-1: FOR y=1
TO eis: LET rodz=FN r(y): PRINT
AT 2*y+1,0; INVERSE 1;"0" AND ((
y+pns-1)<10);pns+y-1;: FOR x=1 T
O eik
 112 GO SUB 130: LET a=a+1: NEXT
 x: LET a=a+da: NEXT y: RETURN
 120 REM druk na pozycji kursora
 121 LET a=FN a(ky,kx)
 125 LET rodz=FN r(ky)
 130 LET q=CODE e$(a): IF rodz=0
 AND q<128 THEN LET m=INT (q/12)
: LET n=q-12*m: LET y$=(" " AND
t$(2*n+2)="0")+(" " AND t$(2*n+2
)="1"): LET x$=CHR$ (CODE t$(2*n
+1)-32*(m<3))+t$(25+m): GO TO 14
0
 131 IF rodz=0 THEN LET x$=("->"
 AND q=129)+(".." AND q<>129): L
ET y$=" ": GO TO 140
 135 IF rodz=2 THEN LET n=INT (q
/100): LET m=q-100*n: LET y$=("
" AND NOT n)+("_" AND n): LET x$
=("0" AND m<10)+STR$ m: LET y$=(
y$ AND q<=127)+(" " AND q>127):
LET x$=(x$ AND q<=127)+(".." AND
 q=128)+("->" AND q=129): GO TO
140
 137 LET y$=" ": LET x$=(".." AN
D q<>1)+("[]" AND q=1)
 140 PRINT AT 2*y,3*x;y$;AT 2*y+
1,3*x; FLASH flash;x$;: RETURN
 150 REM Usuwanie kursora
 151 LET by=pnk+ky-1: LET n=5+3*
(pns+ky): LET aki=CODE n$(n): LE
T aks=CODE n$(n+1): PRINT AT 21,
0;"instr:"; INVERSE 1;i$(3+12*(a
ki-1) TO 12*aki-1); INVERSE 0;"
zrodlo:"; INVERSE 1;s$(1+(aks-1)
*6 TO aks*6)
 153 LET x=kx: LET y=ky: LET inv
=0: LET flash=1: GO TO 120
 160 REM Usuwanie kursora
 162 LET x=kx: LET y=ky: LET fla
sh=0: GO TO 120
 170 REM aktualizacja wskaznikow
 172 LET str=FN m(str,INT (ik/mi
k)+1): LET pnk=mik*(str-1)+1
 174 LET eik=FN m(ik-mik*(str-1)
,mik)
 176 LET kx=FN m(kx,eik)
 178 GO TO 100
9000 REM
************************************
        EDYTOR TAKTU
9010 CLS
9020 LET pns=1: LET kx=1: LET ky
=1: LET mik=FN m(ik,cike): LET e
ik=mik: LET eis=FN m(is,cise): L
ET str=1: GO SUB 100
9024 REM Główna petla edytora
9025 GO SUB 9198: GO SUB 150
9026 LET x$=INKEY$: IF x$="" THE
N GO TO 9026
9030 IF x$="5" THEN GO SUB 160:
GO SUB 9100: GO TO 9025
9031 IF x$="4" THEN GO SUB 160:
GO SUB 9105: GO TO 9025
9032 IF x$="8" THEN GO SUB 160:
GO SUB 9110: GO TO 9025
9033 IF x$="9" THEN GO SUB 160:
GO SUB 9115: GO TO 9025
9034 IF x$="7" THEN GO SUB 160:
GO SUB 9120: GO TO 9025
9036 IF x$="6" THEN GO SUB 160:
GO SUB 9130: GO TO 9025
9040 IF x$="z" THEN GO SUB 9300:
 GO TO 9025
9041 IF x$="c" THEN LET q=0: GO
SUB 9140: GO TO 9025
9042 IF x$="C" THEN LET q=by: GO
 SUB 9140: GO TO 9025
9043 IF x$="s" THEN LET q=0: GO
SUB 9145: GO TO 9025
9044 IF x$="S" THEN LET q=by: GO
 SUB 9145: GO TO 9025
9045 IF x$="d" THEN LET q=0: GO
SUB 9150: GO TO 9025
9046 IF x$="D" THEN LET q=by: GO
 SUB 9150: GO TO 9025
9047 IF x$="w" THEN LET q=0: GO
SUB 9155: GO TO 9025
9048 IF x$="W" THEN LET q=by: GO
 SUB 9156: GO TO 9025
9049 IF x$="p" THEN LET q=0: GO
SUB 9160: GO TO 9025
9050 IF x$="P" THEN LET q=by: GO
 SUB 9161: GO TO 9025
9051 IF x$="r" THEN LET q=0: GO
SUB 9165: GO TO 9025
9052 IF x$="R" THEN LET q=0: GO
SUB 9170: GO TO 9025
9053 IF x$="k" THEN LET q=0: GO
SUB 9175: GO TO 9025
9054 IF x$="K" THEN LET q=0: GO
SUB 9200: GO TO 9025
9055 IF x$="i" THEN LET q=0: GO
SUB 9205: GO TO 9025
9056 IF x$="t" THEN LET q=0: GO
SUB 9210: GO TO 9025
9057 IF x$="T" THEN LET q=0: GO
SUB 9215: GO TO 9025
9058 IF x$="j" THEN LET q=0: GO
SUB 9220: GO TO 9025
9060 GO TO 9025
9099 REM Procedury wykonawcze
9100 IF kx>1 THEN LET kx=kx-1: G
O TO 153
9102 IF pnk>1 THEN LET kx=mik: G
O TO 9107
9103 BEEP .1,20: GO TO 153
9105 IF pnk=1 THEN BEEP .1,20: G
O TO 150
9107 LET str=str-1: LET eik=mik:
 GO TO 100
9110 IF kx<eik THEN LET kx=kx+1:
 GO TO 153
9112 IF knk<ik THEN LET kx=1: GO
 TO 9117
9114 BEEP .1,20: GO TO 153
9115 IF knk>=ik THEN BEEP .1,20:
 GO TO 150
9117 LET str=str+1: LET eik=FN m
(mik,ik-knk): LET kx=FN m(kx,eik
): GO TO 100
9120 IF ky>1 THEN LET ky=ky-1: G
O SUB 150: RETURN
9122 IF pns>1 THEN LET pns=pns-1
: GO TO 100
9123 BEEP .1,20: GO TO 150
9130 IF ky<eis THEN LET ky=ky+1:
 GO TO 150
9132 IF kns<is THEN LET pns=pns+
1: GO TO 100
9133 BEEP .1,20: GO TO 150
9140 LET md=is: LET z$="czyszcze
nie sciezki nr:": GO SUB 9190: I
F q=0 THEN GO SUB 9198: RETURN
9141 LET k=FN o(q): LET m=k+ik-1
: FOR l=k TO m: LET e$(l)=CHR$ 1
28: NEXT l: GO TO 100
9145 LET md=is: LET z$="skasowan
ie sciezki nr:": GO SUB 9190: IF
 q=0 THEN GO SUB 9198: RETURN
9146 IF is=1 THEN GO TO 9141
9147 LET ky=ky-(is<=cise)*(ky>1)
: LET pns=pns-(is>cise)*(is=pns+
cise-1 OR q<pns)
9148 LET k=FN o(q): LET e$=e$( T
O k-1)+e$(k+ik TO ): LET k=FN t(
q): LET n$=n$( TO k-1)+n$(k+3 TO
 ): LET is=is-1: LET eis=FN m(is
,eis): GO SUB 100: IF is<cise TH
EN PRINT AT 2*is+2,0;TAB 31;" ";
TAB 31;" "
9149 GO TO 9198
9150 LET md=is+1: LET z$="dolacz
. sciez. nr:": GO SUB 9190: IF q
=0 THEN GO TO 9198
9151 LET ans=q: GO SUB 9180: LET
 md=ii: LET q=0: LET z$="podaj n
r instumentu:": GO SUB 9190: IF
q=0 THEN GO TO 100
9152 LET ani=q: PRINT AT 3+q,2;
INVERSE 1;q;"-";i$(12*(q-1)+3 TO
 12*q): LET aai=1+12*(q-1): LET
ais=CODE i$(aai TO aai): LET asp
=CODE i$(aai+1 TO aai+1): GO SUB
 9185: LET md=ais: LET q=0: LET
z$="podaj nr zrodla:": GO SUB 91
90: IF q=0 THEN GO TO 100
9153 PRINT AT 3+q,20; INVERSE 1;
;q;"-";s$(6*(asp+q-2)+1 TO 6*(as
p+q-1)): LET ansi=q: LET md=3: L
ET q=0: LET z$="1-liter 2-graf 3
-cyfr": GO SUB 9190: IF q=0 THEN
 GO TO 100
9154 LET k=FN o(ans): LET z$="":
 FOR n=1 TO ik: LET z$=z$+CHR$ 1
28: NEXT n: LET e$=e$( TO k-1)+z
$+e$(k TO ): LET k=FN t(ans): LE
T n$=n$( TO k-1)+CHR$ ani+CHR$ (
asp+ansi-1)+CHR$ (q-1)+n$(k TO )
: LET eis=eis+(is<cise): LET is=
is+1: GO TO 100
9155 LET md=is: LET z$="wymiana
sciez. nr:": GO SUB 9190: IF q=0
 THEN GO SUB 9198: RETURN
9156 LET ans=q: LET md=is: LET q
=0: LET z$="wym sc. "+STR$ ans+"
ze sc.:": GO SUB 9190: IF q=0 O
R q<ans THEN GO TO 9198
9157 LET k=FN o(ans): LET z$=e$(
k TO k+ik-1): LET n=FN o(q): LET
 e$=e$( TO k-1)+e$(n TO n+ik-1)+
e$(k+ik TO ): LET e$=e$( TO n-1)
+z$+e$(n+ik TO )
9158 LET k=FN t(ans): LET z$=n$(
k TO k+2): LET n=FN t(q): LET n$
=n$( TO k-1)+n$(n TO n+2)+n$(k+3
 TO ): LET n$=n$( TO n-1)+z$+n$(
n+3 TO ): GO TO 100
9160 LET md=is: LET z$="powiel.
sciez. nr:": GO SUB 9190: IF q=0
 THEN GO SUB 9198: RETURN
9161 LET ans=q: LET md=is: LET q
=0: LET z$="powiel. sc. "+STR$ a
ns+" na sc.:": GO SUB 9190: IF q
=0 THEN GO SUB 100: GO SUB 9198:
 RETURN
9162 LET k=FN t(ans): LET n=FN t
(q): IF n$(k+2)<>n$(n+2) THEN PR
INT AT 19,0; FLASH 1;"NIEZGODNOS
C TYPOW ZAPISU - Enter": PAUSE 1
: PAUSE 0: GO TO 9198
9163 LET k=FN o(ans): LET n=FN o
(q): LET e$=e$( TO n-1)+e$(k TO
k+ik-1)+e$(n+ik TO ): GO TO 100
9165 LET md=INT (256/ik): LET z$
="rozszerzenie t. razy:": GO SUB
 9190: IF q=0 THEN GO SUB 9198:
RETURN
9166 LET z$="": FOR k=1 TO q-1:
LET z$=z$+CHR$ 128: NEXT k: LET
l=LEN e$*q: FOR k=1 TO l STEP q:
 LET e$=e$( TO k)+z$+e$(k+1 TO )
: NEXT k: LET ik=ik*q: GO TO 170
9170 LET md=ik+1: LET z$="rozsze
rzenie na kroku:": GO SUB 9190:
IF q=0 THEN GO SUB 9198: RETURN
9171 LET k=q: LET q=0: LET md=12
8-ik: LET z$="ile krokow:": GO S
UB 9190: IF q=0 THEN GO SUB 9198
: RETURN
9172 LET z$="": FOR m=1 TO q: LE
```

```
T z$=z$+CHR$ 128: NEXT m: LET i=
(ik+q)*is: FOR k=k-1 TO i STEP i
k+q: LET e$=e$( TO k)+z$+e$(k+1
TO ): NEXT k: LET ik=ik+q: GO TO
 170
9175 LET md=ik: LET z$="kompresj
a taktu razy:": GO SUB 9190: IF
q=0 THEN GO SUB 9198: RETURN
9176 LET k=INT (ik/q): IF ik/q<>
k THEN PRINT AT 19,0; FLASH 1;"
PODZIAL  NIEPRAWIDLOWY - Enter "
: PAUSE 1: PAUSE 0: GO TO 9198
9178 LET ik=k: FOR k=1 TO ik*is:
 LET e$=e$( TO k)+e$(q+k TO ): N
EXT k: GO TO 170
9179 REM druk zestawu instrumen.
9180 LET ax=0: LET k=0: GO SUB 9
183: GO SUB 9184: GO SUB 9184: P
RINT AT 2,3;"INSTRUMENT": GO SUB
 9184: FOR k=4 TO ii+3: PRINT AT
 k,0;"* ";k-3;"-";i$(12*(k-4)+3
TO 12*(k-3));TAB 15;"*": NEXT k:
 GO SUB 9184: GO SUB 9183
9182 RETURN
9183 PRINT AT k,ax;"**********
****": LET k=k+1: RETURN
9184 PRINT AT k,ax;"*";TAB ax+15
;"*": LET k=k+1: RETURN
9185 LET ax=16: LET k=0: GO SUB
9183: GO SUB 9184: GO SUB 9184:
PRINT AT 2,20;"ZRODLO:": GO SUB
9184: FOR k=4 TO ais+3: PRINT AT
 k,16;"*   ";k-3;"-";s$(6*(asp+k
-5)+1 TO 6*(asp+k-4));TAB 31;"*"
: NEXT k: GO SUB 9184: GO SUB 91
83
9186 RETURN
9189 REM
9190 GO SUB 9198: PRINT AT 19,0;
 INVERSE 1;z$;: PRINT ;" ";: IF
q<>0 THEN PRINT ;q;: LET x$=" ":
 GO TO 9195
9191 LET q=0
9192 LET x$=INKEY$: IF x$="" THE
N GO TO 9192
9193 IF x$="p" THEN LET q=0: RET
URN
9194 IF NOT (x$>="1" AND x$<="9"
 OR x$=" " AND q<>0 OR x$="0" AN
D q<>0) THEN GO SUB 9326: GO TO
9192
9195 IF x$=" " THEN PRINT ;" ";
FLASH 1;"(t/n)": PAUSE 1: PAUSE
0:: LET x$=INKEY$: LET q=q*(x$="
t"): BEEP .1,30: RETURN
9196 LET q=q*10+VAL x$: IF q>md
THEN LET q=0: GO SUB 9326: GO TO
 9190
9197 PRINT INVERSE 1;x$;: GO TO
9192
9198 REM zaczernia ostatni. spit.
9199 PRINT AT 19,0;TAB 31;" ": R
ETURN
9200 IF ik=1 THEN RETURN
9201 LET md=ik: LET z$="kasowani
e od kroku:": GO SUB 9190: IF q=
0 THEN GO TO 9198
9202 LET k=q: LET md=ik-q+1: LET
 q=0: LET z$="ile krokow kasowac
:": GO SUB 9190: IF q=0 THEN GO
SUB 9198: RETURN
9203 LET i=k-1: LET n=(ik-q)*is:
 FOR m=i TO n STEP ik-q: LET e$=
e$( TO m)+e$(q+m+1 TO ): NEXT m:
 LET ik=ik-q: GO TO 170
9205 LET md=cike: LET z$="ilosc
krok. na stronie:": GO SUB 9190:
 IF q=0 THEN GO TO 9198
9207 LET mik=q: GO TO 170
9210 LET md=is: LET z$="nr sciez
ki transpon.:": GO SUB 9190: IF
q=0 THEN GO TO 9198
9211 LET k=q: LET q=0: LET md=12
7: LET z$="tronspozycja w gore o
:": GO SUB 9190: IF q=0 THEN GO
TO 9198
9212 LET i=FN o(k): FOR m=i TO i
k+i-1: LET n=CODE e$(m): IF n>12
7 THEN GO TO 9214
9213 LET n=n+q: IF n>127 OR n<0
THEN LET n=128
9214 LET e$(m)=CHR$ n: NEXT m: G
O TO 100
9215 LET md=is: LET z$="nr sciez
ki transpon.:": GO SUB 9190: IF
q=0 THEN GO TO 9198
9216 LET k=q: LET q=0: LET md=12
7: LET z$="tronspozycja w dol o:
": GO SUB 9190: IF q=0 THEN GO T
O 9198
9217 LET q=-q: GO TO 9212

9220 LET md=INT (256/ik): LET z$
="rozsz. j. rytm. razy:": GO SUB
 9190: IF q=0 THEN GO SUB 9198:
RETURN
9222 LET x$="": LET z$="": FOR k
=1 TO q-1: LET x$=x$+CHR$ 129: L
ET z$=z$+CHR$ 128: NEXT k: LET i
=LEN e$*q: FOR k=1 TO i STEP q:
LET e$=e$( TO k)+(z$ AND e$(k)=C
HR$ 128)+(x$ AND e$(k)<>CHR$ 128
)+e$(k+1 TO ): NEXT k: LET ik=ik
*q: GO TO 170
9300 REM zapis cane
9310 PRINT AT 19,0; INVERSE 1;"z
apis ";"literowy" AND NOT rodz;"
graficzny" AND rodz=1;"liczbowy"
 AND rodz=2
9320 LET m=CODE e$(FN a(ky,kx)):
 LET y$="  ": LET cis=0: GO SUB
9325
9321 GO SUB 9330+(10*rodz)
9322 LET e$(FN a(ky,kx))=CHR$ m:
 LET powr=(x$="p"): GO SUB 160:
IF powr THEN RETURN
9323 GO SUB 9110: IF ky=1 THEN G
O SUB 150
9324 GO TO 9320
9325 PRINT AT ky*2,3*kx;" " AND
NOT cis;"_" AND cis;AT ky*2+1,3*
kx; FLASH 1;y$: RETURN
9326 BEEP .1,20: BEEP .1,30: RET
URN
9330 GO SUB 9365: IF x$="p" THEN
 RETURN
9331 IF NOT (x$=" " OR x$="k" OR
 x$="a" OR x$="A" OR x$>="c" AND
 x$<="h" OR x$>="C" AND x$<="H")
 THEN GO SUB 9326: GO TO 9330
9332 LET y$=((x$+" ") AND (x$<>"
" AND x$<>"k")+(".," AND (x$="
")+("->" AND x$="k"): GO SUB 9
325: IF x$=" " OR x$="k" THEN LE
T m=128+(x$="k"): GO TO 9338
9333 GO SUB 9365: IF x$="p" THEN
 RETURN
9334 LET n=(y$(1)<="H"): IF NOT
(x$>="0" AND x$<="2" OR NOT n AN
D x$>="3" AND x$<="7" OR x$="i"
AND (y$(1)<>CHR$ (CODE "e"-n*32)
) AND (y$(1)<>CHR$ (CODE "h"-n*3
2))) THEN GO SUB 9326: GO TO 933
3
9335 IF x$="i" THEN LET cis=NOT
cis: GO SUB 9325: GO TO 9333
9336 FOR m=0 TO 11: IF y$(1)<>CH
R$ (CODE t$(2*m+1)-n*32) THEN NE
XT m: STOP
9337 LET m=m+cis+12*((2-VAL x$)*
n+(3+VAL x$)*NOT n): IF m>=128 T
HEN GO SUB 9326: LET m=128
9338 RETURN
9340 GO SUB 9365: IF x$="p" THEN
 RETURN
9341 IF x$<>"1" AND x$<>"0" THEN
 GO SUB 9326: GO TO 9340
9342 LET m=(x$="1"): RETURN

9350 GO SUB 9365: IF x$="p" THEN
 RETURN
9351 IF (x$<"0" OR x$>"9") AND x
$<>"p" AND x$<>"k" AND x$<>" " T
HEN GO SUB 9326: GO TO 9350
9352 IF x$=" " OR x$="k" OR x$="
0" THEN LET m=128*(x$=" ")+129*(
x$="k"): RETURN
9353 LET y$=" "+x$: GO SUB 9325
9354 GO SUB 9365: IF x$="p" THEN
 RETURN
9355 IF (x$<"0" OR x$>"9") AND x
$<>"p" AND x$<>" " THEN GO SUB 9
326: GO TO 9354
9356 IF x$=" " THEN LET m=VAL y$
(2): RETURN
9357 LET y$=y$(2)+x$: GO SUB 932
5: IF VAL y$*10>127 THEN LET m=V
AL y$: RETURN
9358 GO SUB 9365: IF x$="p" THEN
 RETURN
9359 IF (x$<"0" OR x$>"9") AND x
$<>" " THEN GO SUB 9326: GO TO 9
358
9360 IF x$="p" THEN RETURN
9362 IF x$=" " THEN LET m=VAL y$
: RETURN
9363 LET cis=1: LET y$=y$(2 TO 2
)+x$: LET m=VAL y$+100*cis: IF m
<128 THEN RETURN
9364 IF m>127 THEN LET m=CODE e$
(a): LET y$="  ": LET cis=0: GO
SUB 9325: GO TO 9350
9365 REM wczytanie znaku
9366 LET x$=INKEY$: IF x$="" THE
N GO TO 9366
9367 RETURN
```

Krzysztof POŹNIAK
Michał ROLLINGER

...ze świata

Komputery przenośne zyskują w świecie coraz większą popularność. Wprawdzie ich moc obliczeniowa zazwyczaj jest mniejsza od mocy komputerów stacjonarnych, ale za to można z nich korzystać na przykład w czasie podróży. W produkcji takich komputerów prym wiedzie japońska firma Toshiba, która swymi ostatnimi produktami udowadnia, że komputer przenośny wcale nie musi być gorszy od stacjonarnego.

Weźmy Toshibę T 1000. Jego wymiary 32×5×28 cm są niezwykłe nawet w klasie mikrokomputerów przenośnych. Pamięć RAM ma pojemność 512 KB, a wewnątrz obudowy można umieścić dodatkowe 768 KB pamięci. W komputerze zastosowany został monochromatyczny ekran ciekłokrystaliczny o rozdzielczości 640 na 200 punktów. Odpowiada to 25 liniom tekstu po 80 znaków. W 82-klawiszowej klawiaturze 10 klawiszy wydzielonych zostało jako funkcyjne i do sterowania kursorem. Standardowe wyposażenie, to łącze szeregowe RS-232C, łącze równoległe i gniazdo RGB dla dodatkowego monitora kolorowego. T 1000 może być zasilany z baterii Ni-Cd pozwalających na 4-godzinną pracę bez ładowania. Nowsza wersja tego komputera, T 1200 uzupełniona została o dysk twardy o pojemności 20 MB.

Taki sam dysk wraz z kontrolerem RLL wprowadziła Toshiba do modelu T 3100/20. Wewnątrz tego komputera umieszczono procesor 80 286, 640 KB pamięci RAM (z możliwością rozszerzenia jej do 2,6 MB), stację dysków elastycznych 3,5 cala o pojemności 720 KB i wspomniany już dysk twardy. Poza tym można tu umieścić modem o prędkości przesyłania 1200 bodów. Ekran ciekłokrystaliczny o rozdzielczości 640 na 400 punktów, pozwala na uzyskanie monochromatycznej grafiki średniej rozdzielczości. Fachowcy twierdzą, że właśnie ten komputer, jak żaden inny, łączy w sobie szybkość i względnie dużą moc przetwarzania z wygodą, łatwością posługiwania się i przenoszenia.

Nowszy T 5100 jest komputerem kompatybilnym z rodziną IBM PC/XT/AT. Wyposażony jest w procesor 80386 z przełączanym zegarem 8/16 MHz, 2 MB pamięć RAM (z możliwością rozszerzenia do 4 MB), gniazdo do zamontowania koprocesora matematycznego 80387, stację dysków 3,5 cala i 1,44 MB oraz twardy dysk 40 MB. Model dodatkowo wyposażony jest w wyjście szeregowe i równoległe, gniazdo do podłączenia zewnętrznego monitora kolorowego oraz zewnętrznej stacji dysków 5,25 cala. Dzięki temu wyposażeniu T 5100 kosztuje jedynie nieco ponad... 6 tys. dolarów amerykańskich.

PC, to skrót oznaczający komputer przenośny — portable computer. PC może też oznaczać komputer kieszonkowy — pocket computer. I w wypadku jednego z najnowszych wytworów firmy Sharp — oznacza. Takim bowiem komputerem jest PC-1600. Posiada on pamięć RAM wielkości 16 KB z możliwością rozszerzenia za pomocą specjalnych modułów do 80 KB. Zawartość pamięci podtrzymywana jest bateriami przez 5 lat, jeśli moduł znajduje się w komputerze, a przez 2 lata — gdy poza nim. PC-1600 posiada również 96 KB pamięć ROM, która poza systemem operacyjnym zawiera dialekt języka Basic. Standardowy interfejs, RS-232 umożliwia bezpośrednie połączenie z innymi typami mikrokomputerów. Dodatkowy optyczny interfejs przeznaczony jest do szybkiej komunikacji optycznej. Zastosowany w komputerze wyświetlacz ciekłokrystaliczny może pracować w systemie alfanumerycznym (4 wiersze po 26 znaków) lub graficznym (156×32 punkty). Zasilanie — z baterii umożliwiających 25-godzinną pracę bez przerwy.

Do tego komputera firma Sharp produkuje sprzęt peryferyjny. Są to czterokolorowy printer (ploter drukujący na papierze formatu A-4, stacja 2,5-calowych dysków elastycznych o pojemności 64 KB i interfejs do podłączenia standardowego magnetofonu kasetowego.

Jerzy RAJCH

JAK WYKORZYSTAĆ DODATKOWĄ PAMIĘĆ ATARI

Od dłuższego czasu wśród użytkowników komputerów Atari krąży program RAMdysk (przedstawiany również na łamach „IKS-a"). Symuluje on pracę bardzo szybkiej pamięci zewnętrznej, wykorzystującej dodatkową pamięć RAM. Poniższy program, który zaczerpnięty został z miesięcznika „COMPUTE!", przeznaczony jest dla Atarii 130XE oraz innych komputerów serii XE i XL mających pamięć rozszerzoną do 128KB. Oczywiście program będzie pracował poprawnie również na komputerach mających jedynie (a może aż) 64KB, ale nie będzie pozwalał na całkowite wykorzystanie jego możliwości.

Dlaczego, aby wykorzystać całe 128KB komputera (np. w Atari 130XE) należy użyć specjalnego programu — na przykład takiego, jak ten? Dzieje się tak ze względu na strukturę wewnętrzną komputera. Maszyny Atarii serii XE oraz XL są urządzeniami mającymi procesory ośmiobitowe z szesnastobitową szyną danych. Mogą więc adresować jednocześnie 64KB (adresy od 0 do 65535). Dlatego, aby wykorzystać dodatkowy bank pamięci, należy napisać program (w języku maszynowym, gdyż tylko on zapewnia odpowiednią szybkość) przełączający owe banki. W podobny sposób działa właśnie poniższy program.

Teraz parę słów o inicjacji programu. Jest on napisany w języku AtariBasic. Uruchamiamy go rozkazem **RUN** — komputer ładuje procedurę maszynową; umieszcza ją na szóstej stronie pamięci (adresy od 1536 do 1780). Po zgłoszeniu się interpretera Basic'a komunikatem **READY**, możemy przystąpić do właściwego wykorzystania programu. W wypadku pojawienia się komunikatu o błędzie, należy program wydrukować oraz wyszukać i poprawić błędne linie (lub częściej dane w liniach DATA). Mam nadzieję, że sumy kontrolne w znacznym stopniu zmniejszą prawdopodobieństwo pojawienia się błędu w wydruku.

Program możemy wykorzystać do umieszczenia kilkunastu, a nawet kilkudziesięciu np. pamięci ekranów (tj. obszarów pamięci, w których komputer przechowuje dane dotyczące obrazu) w dodatkowych 64 kilobajtach pamięci. **Dokładniej, mając do dyspozycji komputer Atarii 130XE oraz poniższy program, możemy zgromadzić (jednocześnie) osiem obrazów w trybie graficznym 8 lub np. sześćdziesiąt osiem obrazów w trybie tekstowym 0. Szybkość przesyłania danych z dodatkowej pamięci wynosi 9000 bajtów w ciągu sekundy**.

Wiedząc, iż jeden ekran w trybie 0 zajmuje 992 bajty, dokonujemy zdumiewającego odkrycia — będzie on transmitowany ok. 0,1 sekundy. Wydaje mi się, że jest to zadowalający wynik.

Program realizuje cztery następujące funkcje:

a) odczytuje zawartość dowolnej komórki pamięci dodatkowej (ang. PEEK),

b) zapisuje dowolną komórkę pamięci dodatkowej wartości od 0 do 255 (ang. POKE),

c) przepisuje dowolny obszar pamięci do obszaru rozszerzenia,

d) transmituje w kierunku odwrotnym niż określony w podpunkcie c.

Poszczególne funkcje wywołujemy za pomocą instrukcji USR. Dla tych czytelników, którzy jeszcze nie wiedzą do czego służy ta komenda, przypominam, iż przekazuje ona sterowanie do procedury maszynowej umieszczonej pod określonym adresem.

Czas na szczegółowe omówienie wszystkich możliwości programu.

ad. a) Aby odczytać wartości którejkolwiek komórki pamięci dodatkowej należy wpisać w trybie bezpośrednim bądź wykorzystać we własnym programie następujące linie:

X = USR (AUXBYTE, *ad*)

gdzie zmienna *'ad'*, mogąca przyjmować wartość w zakresie od 0 do 65535, określa adres komórki, której zawartość zamierzaliśmy zbadać. Po wykonaniu tej sekwencji otrzymamy liczbę od 0 do 255, określającą stan komórki pamięci.

ad. b) Wpisanie pewnej wartości liczbowej do komórki dodatkowego banku pamięci również nie nastręcza naszemu programowi żadnych trudności, linia:

X = USR (AUXBYTE, *ad*, *wart*)

zapewnia wprowadzenie do komórki o adresie określonym zmienną *'ad'* wartości, którą zawiera zmienna *'wart'*. UWAGA! Zmienna numeryczna *'ad'* musi należeć tak jak i w poprzednim wypadku do przedziału 0—65535, a zmienna *'wart'* od 0—255.

— Przydałby się naszemu panu komputer...

ad. c) Znacznie bardziej skomplikowana jest operacja przesłania bloku danych. Wygląda ona następująco:

X = USR (AUXDUMP, *źródło, cel, dług*).

A oto znaczenie poszczególnych zmiennych:

'źródło' — określa początek bloku w pamięci podstawowej, przygotowanego do przesłania; możemy nadać tej zmiennej wartość od 0 do 6553.

'cel' — określa adres w dodatkowym bloku pamięci, tj. adres, pod którym znajdą się dane po przesłaniu. Może przyjmować wartości takie, jak poprzednia zmienna.

'dług' — „mówi" komputerowi, jak długi jest przesyłany blok.

ad. d) jest to ostatnie (niestety) udogodnienie wnoszone przez ten program, umożliwia ono powrót danych z pamięci dodatkowej do podstawowego bloku 64KB. Rozkaz, który przeprowadza tę transmisję ma postać bardzo podobną do postaci poprzedniego rozkazu, jedyna różnica polega na użyciu innej stałej programowej — zamiast AUXDUMP należy użyć AUXLOAD. Linia ta wygląda następująco:

X = USR (AUXLOAD, *źródło, cel, dług*).

Należy także zmienić kolejność występowania zmiennych *'źródło'* i *'cel'*, w tym wypadku pierwsza oznacza adres bloku w pamięci dodatkowej, a druga adres pamięci, pod którym dane zostaną umieszczone.

Na koniec pragnę zaprezentować jeden z praktycznych przykładów programu, o którym była mowa. Będzie on przesyłał dane obrazu w grafice 8 do pamięci dodatkowej i z powrotem (tj. do podstawowego bloku 64KB).

Przede wszystkim należy określić adres początku pamięci obrazu; jest on przechowywany w komórkach 88 (LSB — młodszy bajt) i 89 (MSB — starszy bajt). Znając zawartości tych komórek, możemy w prosty sposób odnaleźć poszukiwany początek pamięci obrazu.

pocz = REEK (88) + 256 x PEEK (89)

Będzie nam jeszcze potrzebna informacja, jak duży obszar pamięci wykorzystany jest w celu zapamiętania obrazu w trybie

graficznym bądź tekstowym. W wypadku trybu numer 8 wynosi on 8138 bajtów. Informacje o pozostałych trybach znajdują się poniżej:

tryb 0	(tekstowy)	— 992	bajty
tryb 1	(tekstowy)	— 672	bajty
tryb 2	(tekstowy)	— 420	bajtów
tryb 3	(graficzny)	— 432	bajty
tryb 4	(graficzny)	— 696	bajtów
tryb 5	(graficzny)	— 1176	bajtów
tryb 6	(graficzny)	— 2184	bajty
tryb 7	(graficzny)	— 4200	bajtów
tryb 8, 9, 10, 11	(graficzne)	— 8138	bajtów
tryb 12	(graficzny)	— 1152	bajty
tryb 13	(graficzny)	— 660	bajtów
tryb 14	(graficzny)	— 4296	bajtów
tryb 15	(graficzny)	— 8138	bajtów

Te wiadomości w zupełności wystarczą, by dokonać transmisji danych obrazu:

X = USR (AUXDUMP, *pocz.*, 0,8138)

Liczba 0 określa, pod jakim adresem bloku rozszerzenia pamięci znajdzie się początek przesłanego obszaru danych. Aby dane te otrzymać z powrotem wpisujemy następującą linię:

X = USR (AUXLOAD,0 *pocz.*, 0,8112)

Dzięki niej zapamiętany obrazek wraca do podstawowego bloku pamięci.

```
DB 10 AUXBYTE=1624
GK 20 AUXDUMP=1655
YB 30 AUXLOAD=1718
IX 40 FOR I=1536 TO 1780:READ A:P
   OKE I,A:X=X+A:NEXT I:IF X<>353
   14 THEN PRINT CHR$(125);"BLAD
   W LINIACH 50-220"
DR 50 DATA 160,0,173,1,211,41,195
   ,133,217,165,216,41,192,74
VN 60 DATA 74,74,74,9,32,5,217,14
   1,1,211,165,216,41,63
VK 70 DATA 9,64,133,215,165,212,1
   45,214,165,217,9,48,141,1
OS 80 DATA 211,96,160,0,173,1,211
   ,41,195,133,217,165,216,41
VJ 90 DATA 192,74,74,74,74,9,32,5
   ,217,141,1,211,165,216
LK 100 DATA 41,63,9,64,133,215,17
   7,214,133,212,165,217
NM 110 DATA 9,48,141,1,211,96,104
   ,133,245,198,245,104,133,216
XK 120 DATA 104,133,214,165,245,2
   08,6,32,44,6,132,213,96,104
UY 130 DATA 104,133,212,32,0,6,24
   ,144,243,104,104,133,225,104
CK 140 DATA 133,224,104,133,216,1
   04,133,214,104,133,227,104,133
   ,226
XS 150 DATA 160,0,132,229,132,228
   ,177,224,133,212,32,0,6,230
FB 160 DATA 224,208,2,230,225,230
   ,214,208,2,230,216,230,228,208
DU 170 DATA 2,230,229,165,228,197
   ,226,208,225,165,229,197,227,2
   08
FY 180 DATA 219,96,104,104,133,21
   6,104,133,214,104,133,225,104,
   133
RB 190 DATA 224,104,133,227,104,1
   33,226,160,0,132,229,132,228,3
   2
LL 200 DATA 44,6,165,212,145,224,
   230,224,208,2,230,225,230,214
YX 210 DATA 208,2,230,216,230,228
   ,208,2,230,229,165,228,197,226
AG 220 DATA 208,225,165,229,197,2
   27,208,219,96
```

Mam nadzieję, że program ten stanie się użytecznym narzędziem dla wszystkich, którzy chcieliby wykorzystać dotychczas niedostępne i „marnujące się" kilobajty pamięci w swoich komputerach.

Na podst. miesięcznika COMPUTE!
opracował: **Piotr JASINIEWSKI**

GIEŁDA POMYSŁÓW

Drukowanie minikatalogu dyskietki

Rafał FOLTYNIEWICZ

```
10 'DRUKOWANIE miniKATALOGU DYSKIETKI
20 MODE 1
30 INPUT "Wlasciciel ";w$:d=LEN(w$)
40 IF d>11 THEN 30
50 w$=w$+SPACE$(11-d)+" *"
60 'inicjalizacja drukarki
70 RESTORE 600
80 FOR i=1 TO 8
90 READ zn:PRINT #8,CHR$(zn);:NEXT i
100 INPUT "Numer i strona dyskietki ";o$:d=LEN(o$)
110 IF d>10 THEN 100
120 o$="* "+o$+SPACE$(10-d)
130 CLS:CAT
140 'kopiowanie linii z ekranu
150 l=3:z$=""
160 WHILE z$<>" "
170 l=l+1:LOCATE 1,l:GOSUB 400
180 WEND
190 f$=""
200 FOR i=1 TO 9
210 LOCATE i,l+1:GOSUB 400
220 f$=f$+z$
230 NEXT i
240 'drukowanie naglowka
250 GOSUB 500
260 PRINT #8,o$;"  *  ";f$;"  *  ";w$:GOSUB 500
270 'drkowanie katalogu
280 FOR k=4 TO l-1:l$="* "
290 FOR i=1 TO 40
300 LOCATE i,k:GOSUB 400
310 l$=l$+z$
320 NEXT i
325 l$=l$+" *"
330 PRINT #8,l$:l$="* "
340 NEXT k:GOSUB 500
350 LOCATE 5,l+3:PRINT "Kolejna dyskietka ? (t/n)"
360 c$=INKEY$:IF c$="" THEN 360
370 IF UPPER$(c$)="T" THEN 60
380 END
400 'kopiowanie znaku z ekranu
410 z$=COPYCHR$(#0):RETURN
500 'drukowanie ramki
510 PRINT #8,STRING$(44,"*")
520 RETURN
600 'znaki dla drukarki
610 DATA 27,64,27,49,15,27,83,0
```

Obracanie ekranu

```
5 'OBRACANIE EKRANU  1987.12.30
10 'kod maszynowy
20 MEMORY &9FFF:adres=&A000
30 RESTORE 210
40 FOR i=1 TO 14
50 suma=0:READ a$,b$
60 FOR k=1 TO 31 STEP 2
70 wartosc=VAL("&" + MID$(a$,k,2))
80 POKE adres,wartosc
90 suma=suma+wartosc:adres=adres+1
100 NEXT k
110 IF suma<>VAL("&"+b$) THEN PRINT "Zle dane w
linii nr: ";200+i*10:STOP
120 NEXT i
130 SAVE "obroc",b,&A000,&D7
140 END
200 '
210 DATA CD11BC3807280A21A2A0180821ADA018,514
220 DATA 03219BA02256A0DD21C9A0FD21CEA0CD,837
230 DATA 0BBCE5DD46000E000922D6A0D121FF0F,67E
240 DATA 19113000A7ED527CE60F67D608300411,53B
250 DATA 0008191100F01922D8A02AD6A0ED5BD8,695
260 DATA A001D007C5CDADA0C10B78B12806C5CD,80C
270 DATA 85A018F1DD23FD23FD23010008 2AD6A0,717
280 DATA 0922D6A07CE6F0FEE0C82AD8A0A7ED42,A11
290 DATA 22D8A018C523DD7E01BC2003DD66001B,633
300 DATA FD7E00BAC0FD56011EFFC90608CB06EB,7F9
310 DATA 10FB1A060817CB1E10FB1712C97E47CD,5C2
320 DATA BDA0F51A47F11278CDBDA077C9E655CB,99E
330 DATA 274F78E6AACB3FB1C9C0C8D0D8E0F7FF,B08
340 DATA EFF7E7EFDFE700000000000000000000,582
```

Lilavati (6)

„PIĘTNASTKA"

Danuta KWASIŻUR,
Mieczysław SKONIECZNY

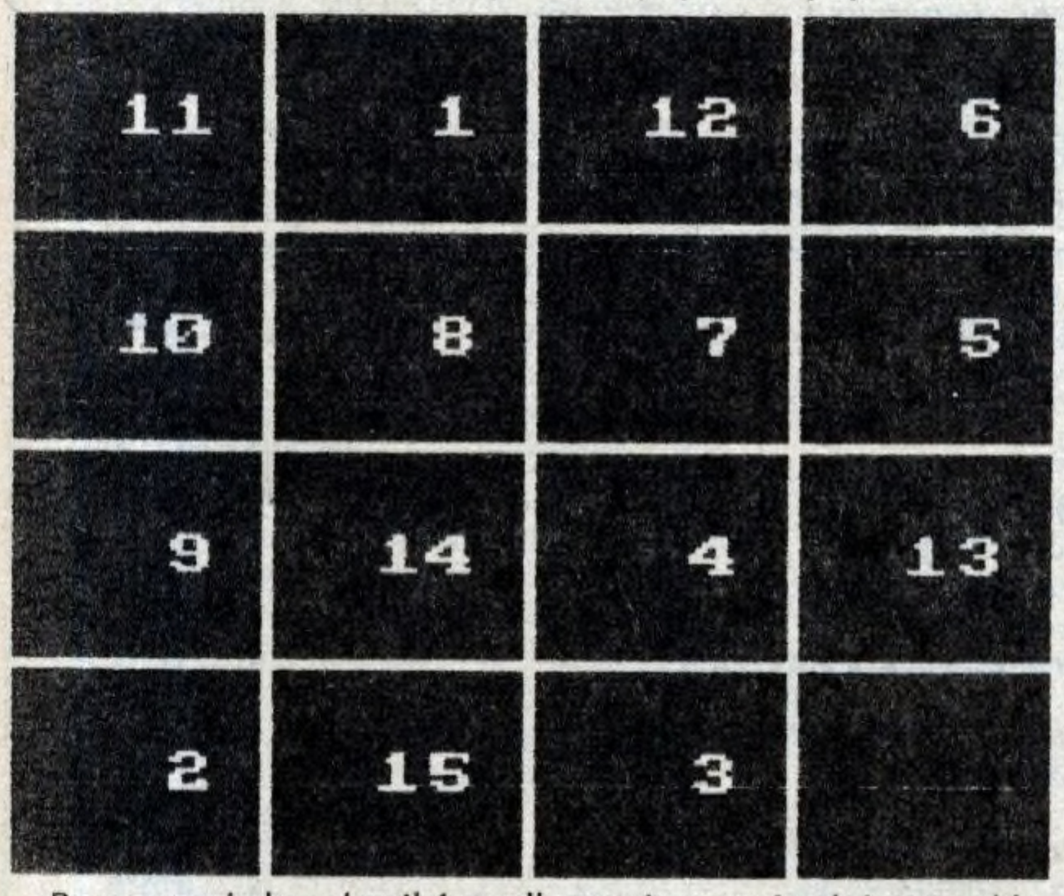
W szóstym odcinku cyklu poświęconego mikrokomputerowej realizacji rozrywek matematycznych, zgromadzonych przez Szczepana Jeleńskiego w „Lilavati", chcemy zaprezentować znaną grę o nazwie „piętnastka". Gra ta znana również pod angielską nazwą „Fifteen puzzle" pochodzi ze Stanów Zjednoczonych, gdzie wynalazł ją pewnien głuchoniemy w roku 1878. Szybko zyskała popularność — również w Europie. Jako ciekawostkę można podać, że swego czasu władze rządowe USA musiały wydać zakaz posiadania tej gry przy sobie przez urzędników w godzinach pracy.

Zasady gry są bardzo proste. Są one przedstawione na ekranie mikrokomputera po zainicjowaniu programu w następującej postaci:

```
Gra składa się z ramki, w której mieści
się piętnaście ruchomych kwadracików —
po 4 w każdej kolumnie z wyjątkiem osta-
tniego szeregu, w którym miejsce odpo-
wiadające szesnastemu kwadracikowi jest
puste. Na kwadracikach widnieją numery
od 1 do 15. Korzystając z owego jednego
wolnego miejsca należy kwadraciki prze-
suwać dopóty, dopóki nie zostaną upo-
rządkowane w kolejności od 1 do 15,
a ostatni kwadracik będzie pusty.

STEROWANIE KLAWISZAMI:

↑,↓,←,→

NACISNIJ DOWOLNY KLAWISZ
```

Następnie pojawi się plansza gry z losowym, generowanym przez program, rozstawieniem numerów.

Przykładowe rozstawienie prezentuje poniższy rysunek:



Przesuwanie kwadracików odbywa się za pośrednictwem klawiatury funkcyjnej (4 klawisze ze strzałkami). W sytuacji, gdy przesunięcie jest niemożliwe, program informuje o tym gracza komunikatem:

PRZESUNIĘCIE JEST NIEMOŻLIWE

Po każdym przesunięciu program bada, czy gra zakończyła się sukcesem. Jeżeli ustawienie ponumerowanych kwadracików jest prawidłowe, na ekranie pojawia się komunikat:

BRAWO! UDAŁO SIĘ

i program kończy działanie.

Wkrótce po rozpowszechnieniu się „Piętnastki" pojawiły się liczne jej teorie. Zwłaszcza dwaj znani matematycy, Johnson i Story podali sporo ciekawego materiału naukowego dotyczącego tej gry. Udowodnili oni, że w zależności od ustawienia początkowego można przewidzieć rozwiązalność lub nierozwiązalność zadanego układu.

Dodamy tylko, że nie zawsze możliwe jest osiągnięcie układu kwadracików gwarantującego sukces w grze. Niejednokrotnie w szeregu najniższym zamiast kolejności 13, 14, 15 otrzymuje się ustawienie 13, 15, 14. Nie radzimy wtedy mozolić się dalej nad przestawieniem kwadracików — najlepiej nacisnąć klawisz BREAK i rozpocząć grę od początku.

```
10 MODE 1:INK 0,26:INK 1,0:INK 2,24:INK 3,14:BORDER 0:PAPER 1:CLS
20 PEN 2:PRINT"Gra sklada sie z ramki, w ktorej miesci sie pietnascie
ruchomych kwadracikow-  po 4 w kazdej kolumnie z wyjatkiem os
ta-tniego szeregu, w ktorym miejsce odpo-  wiadajace szesnastemu kwadr
acikowi jest"
30 PRINT "puste. Na kwadracikach widnieja numery  od 1 do 15. Korzysta
jac z owego jednego wolnego miejsca nalezy kwadraciki prze- su
wac dopoty, dopoki nie zostana upo-  rzadkowane w kolejnosci od 1 do
15,     a ostatni kwadracik bedzie pusty."
40 PRINT:PRINT "STEROWANIE KLAWISZAMI:":PRINT:PRINT CHR$(240);",";CHR$
(241);",";CHR$(242);",";CHR$(243)
50 LOCATE 1,20:PRINT "NACISNIJ DOWOLNY KLAWISZ"
60 IF INKEY$="" THEN GOTO 60
70 CLS
80 WINDOW £1,11,30,6,25:PAPER £1,2:CLS £1
90 WINDOW £2,1,40,1,1:PAPER £2,1:CLS £2:PEN £2,0
95 REM   'rysowanie planszy gry'
100 GRAPHICS PEN 0:FOR k=0 TO 3
110 PLOT 160+k*80,0:DRAWR 0,318
120 PLOT 238+k*80,0:DRAWR 0,318
130 NEXT k
140 FOR k=0 TO 3
150 PLOT 160,K*80:DRAWR 318,0
160 PLOT 160,78+K*80:DRAWR 318,0
170 NEXT K
180 DIM TABL(16)
190 GOSUB 420
200 TABL(16)=0
210 PAPER 2:PEN 1
220 FOR K=1 TO 15
230 X=13-INT((K-1)/4)*20+(K-1)*5:Y=8+INT((K-1)/4)*5
240 LOCATE X,Y:PRINT USING "££";TABL(K)
250 NEXT K
255 REM   'sterowanie przesuwaniem kwadracikow'
260 I=16
270 A$=INKEY$
280 IF A$="" THEN GOTO 270
290 IF ASC(A$)=240 THEN GOTO 340
300 IF ASC(A$)=241 THEN GOTO 360
310 IF ASC(A$)=242 THEN GOTO 380
320 IF ASC(A$)=243 THEN GOTO 400
330 GOTO 270
335 REM  'kontrola poprawnosci przesuniecia'
340 IF I=13 OR I=14 OR I=15 OR I=16 THEN GOTO 500
350 TABL(I)=TABL(I+4):TABL(I+4)=0:GOSUB 510:LOCATE X,Y:PRINT USING "££
";TABL(I):I=I+4:GOSUB 510:LOCATE X,Y:PRINT "  ":GOSUB 520:GOTO
 270
360 IF I=1 OR I=2 OR I=3 OR I=4 THEN GOTO 500
370 TABL(I)=TABL(I-4):TABL(I-4)=0:GOSUB 510:LOCATE X,Y:PRINT USING "££
";TABL(I):I=I-4:GOSUB 510:LOCATE X,Y:PRINT "  ":GOSUB 520:GOTO
 270
380 IF I=4 OR I=8 OR I=12 OR I=16 THEN GOTO 500
390 TABL(I)=TABL(I+1):TABL(I+1)=0:GOSUB 510:LOCATE X,Y:PRINT USING "££
";TABL(I):I=I+1:GOSUB 510:LOCATE X,Y:PRINT "  ":GOSUB 520:GOTO
 270
400 IF I=1 OR I=5 OR I=9 OR I=13 THEN GOTO 500
410 TABL(I)=TABL(I-1):TABL(I-1)=0:GOSUB 510:LOCATE X,Y:PRINT USING "££
";TABL(I):I=I-1:GOSUB 510:LOCATE X,Y:PRINT "  ":GOSUB 520:GOTO
 270
415 REM  'generowanie kolejnosci kwadracikow na planszy'
420 FOR I=1 TO 15
430 LOS=1+INT(RND*15)
440 FOR J=1 TO I-1
450 IF TABL(J)=LOS THEN GOTO 430
460 NEXT J
470 TABL(I)=LOS
480 NEXT I
490 RETURN
500 CLS £2:PRINT £2,"PRZESUNIECIE NIEMOZLIWE":FOR T=1 TO 1000:NEXT T:C
LS £2:GOTO 270
510 X=13-INT((I-1)/4)*20+(I-1)*5:Y=8+INT((I-1)/4)*5:RETURN
520 IF TABL(16)>0 THEN RETURN
530 FOR J=1 TO 15
540 IF NOT TABL(J)=J THEN RETURN
550 NEXT J
560 PRINT £2,"BRAWO! UDALO SIE!"
570 FOR a=0 TO 26:BORDER a:FOR t=1 TO 100:NEXT t:NEXT a:END
```

UWAGI PROGRAMOWE

1. Program opracowano w języku BASIC mikrokomputera AMSTRAD CPC 6128.

2. Tablica TABL służy do przechowywania aktualnego układu kwadratów.

3. Instrukcje 100—170 realizują rysowanie planszy gry.

4. Instrukcje 260—410 sterują przesuwaniem kwadracików.

5. Generowanie początkowego ustawienia ponumerowanych kwadratów realizuje podprogram 420—490.

6. Instrukcje 520—570 badają czy gra zakończyła się sukcesem. Jeśli tak, to generują stosowny komunikat i kończą pracę programu.

Oprogramowanie układów transputerowych

Początkowo programowanie komputerów polegało na zapisaniu sekwencji rozkazów opisującej algorytm rozwiązania danego problemu w postaci numerycznych kodów maszynowych, zrozumiałych bezpośrednio przez procesor. Ta metoda programowania, pracochłonna i nieefektywna była obarczona szczególną podatnością na błędy. Mała czytelność stosowanego zapisu programu powodowała znaczne trudności w czasie procesów uruchomieniowych. Wkrótce udoskonalono metodologię programowania przez wprowadzenie języków niskiego poziomu (asemblery) oraz problemowo (języki wysokiego poziomu) — FORTRAN, ALGOL, COBOL.

Języki wysokiego poziomu stały się bardzo wygodnym narzędziem dla programisty, gdyż nie wymagały znajomości organizacji używanego sprzętu. Jednakże w programach, w których czas wykonywania był elementem krytycznym, w szczególności dotyczy to oprogramowania systemowego, języki wysokiego poziomu nie były przydatne, ze względu na swą nadmiarowość. Do najbardziej rozpowszechnionych obecnie języków zaliczyć należy BASIC, PASCAL, ADA, LISP, PROLOG i LOGO, ale wskutek zmiany charakteru programowania, zbliżania go do modelowania systemowego istnieje dziś duże zainteresowanie językami typu FORTH, PL/M, C i MODULA. Języki te mając konstrukcję języka wysokiego poziomu, wzbogacone są o zestaw instrumentów, które nadają im pewne własności „asemblerowe".

Transputer a OCCAM

W pierwszych zastosowaniach, język maszynowy transputera wykorzystywał nieznaczną ilość instrukcji, których funkcyjna natura określała działanie. Instrukcje te były jednobajtowe, podzielone na cztery „bity funkcji" oraz cztery „bity danych". W ten sposób podstawowy zbiór instrukcji liczył 13 funkcji (*Load Constant, *Add Constant, *Load Local, *Store Local, *Load Local Pointer, *Jump, *Store Non Local, *Conditional Jump, *Call, *Load Non Local, *Prefix, *Negative Prefix i *Operate). Dziesięć pierwszych funkcji, to operacje obliczeniowe. Pozostałe trzy są szczególnie ciekawe, gdyż zezwalają na budowę słów 32-bitowych złożonych z 4-bitowych danych (*Prefix, *Negative Prefix) oraz powodują interpretowanie słowa 8-bitowego jako kodu funkcji (*Operate). Tym prostym sposobem INMOS formalnie zapewniła sobie znaczne możliwości rozbudowywania listy rozkazów. Konwencjonalne języki programowania nie są zbyt przydatne do konstruowania programów przeznaczonych dla współbieżnej pracy układów wieloprocesorowych.

Równolegle z pracami badawczo-projektowymi nad transputerami rozpoczęto tworzenie odpowiednich języków programowania. OCCAM — specjalizowany język wysokiego poziomu powstawał pod kątem przyszłych zastosowań w środowisku transputerowym, stąd też jest on wewnętrznie dostosowany do architektury i możliwości tych układów. OCCAM tworzy szkielet koncepcyjno-konstrukcyjny projektowanego współbieżnego systemu w analogiczny sposób, jak algebra Bool'a czyni to w stosunku do systemów elektronicznych złożonych z bramek logicznych. OCCAM wyróżnia głęboka spójność z modelem współbieżnie pracujących wielu procesów, komunikujących się za pomocą dwuprzewodowych łączy lub ich odpowiedników programowych — kanałów. Dwojakie rozumienie kanału przez język sprowadza się w efekcie do nierozróżniania ich fizycznej implementacji. Z punktu widzenia programisty nie ma znaczenia lokalizacja pamięci i miejsce wykonywania poszczególnych procesów. Wzajemna komunikacja między procesami odbywa się automatycznie i nie ma potrzeby stosowania specjalnych procedur zapewniających kontrolę jej poprawności oraz właściwą synchronizację wewnętrzną. Dla transputera T-800 powstała nowa, znacznie wzbogacona wersja OCCAM-2, oparta na 16-, 32- i 64-bitowych danych. Zastosowane w niej mechanizmy sprzętowe zapewniają wykonywanie operacji zmiennoprzecinkowych z podobną szybkością, jak analogicznych operacji na liczbach całkowitych. Zasadniczą zmianę wprowadza to, że kanały mogą przesyłać dane zdefiniowane strukturalnie zamiast 16-bitowych wartości. Oznacza to, że łańcuchy, macierze oraz skomplikowane rekordy Pascala i struktury języka C mogą tworzyć wyjście pojedynczych operacji. Opracowano specjalne wersje kompilatorów: Pascala, C, Fortranu, Prologu i Macro Assemblera przeznaczone dla układów transputerowych. Moduły pisane w tych językach po rekompilacji mogą być powiązane programem napisanym w OCCAM, który wywołuje te moduły jako procesy. Dotychczas nie ma systemu operacyjnego dla urządzeń bazujących na układach transputerowych. Niektóre z firm proponują stosowanie techniki nakładek systemowych, a także adaptację systemu UNIX.

OCCAM - 2

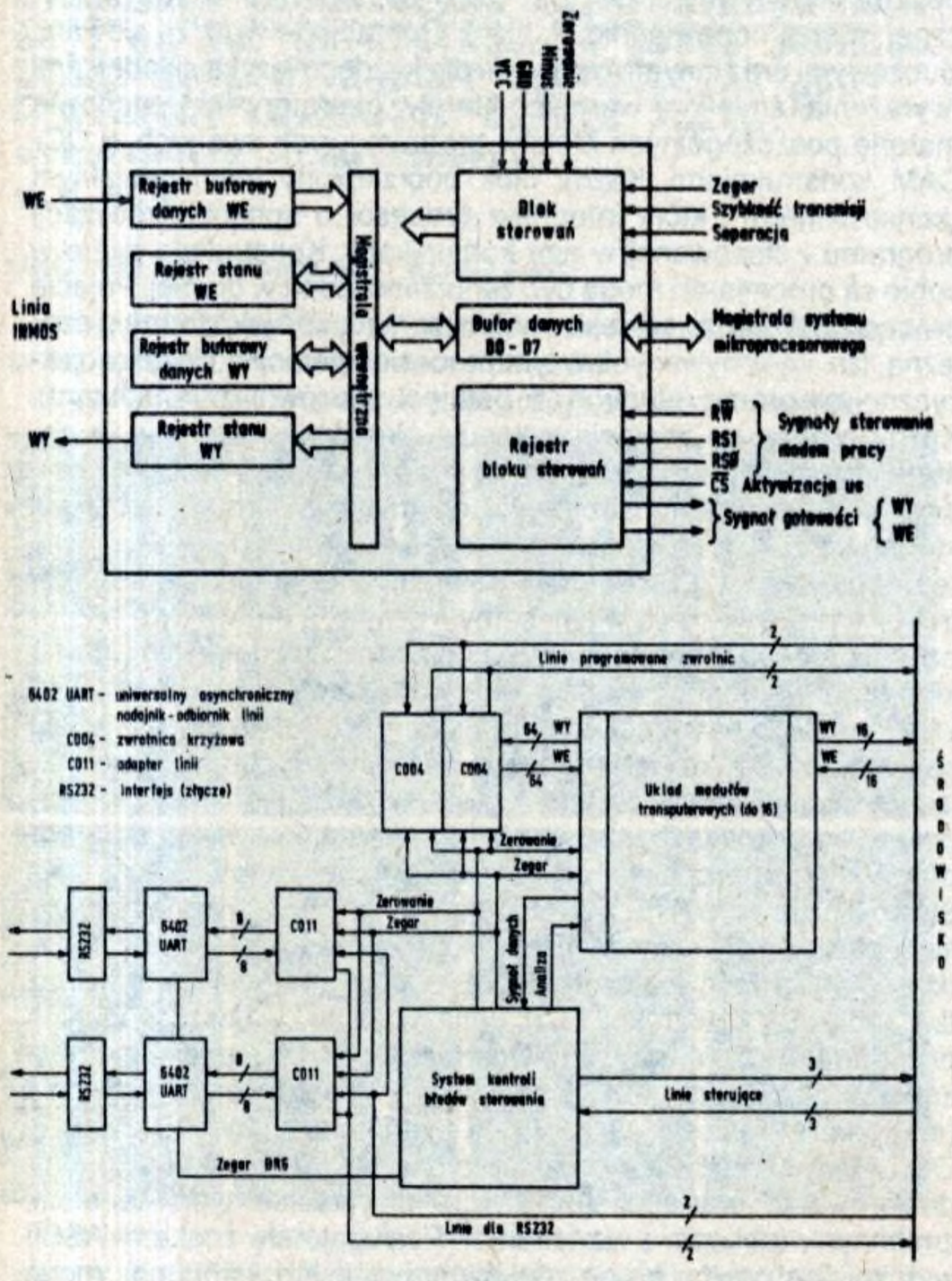
Jądrem koncepcji języka OCCAM są pojęcia procesu i kanału. W OCCAM występują trzy podstawowe operacje zwane też procesami prymitywnymi (INPUT, OUTPUT, ASSIGN — w symbolicznej notacji odpowiednio: ?, !, :=). Operacje te wraz ze słowami kluczowymi oraz innymi, typowymi dla każdego języka składnikami (wyrażenia, zmienne, wartości, literały, operatory itp.), stanowią materię poszczególnych bloków programowych zwanych w OCCAM konstrukcjami. Każdy blok poprzedzony jest specjalnym „konstruktorem", który informuje procesor o sposobie realizacji programu i stosowanej w nim komunikacji. Konstrukcje same w sobie są procesami i mogą być zanurzane jedna w drugiej. Pojęcie procesu jest bardzo szerokie i należy je rozumieć jako formę niezależną, tzn. wykonywanych w tym procesie operacji od lokalizacji fizycznej programu i danych (tj. pamięci, jak również transputera). Każdy proces po inicjacji wykonuje określone instrukcje, a następnie zostaje zatrzymany lub się kończy. W OCCAM-2 wyróżnia się 4 podstawowe konstruktory. Konstruktor SEQ określa blok, w którym instrukcje są realizowane jedna po drugiej. PAR umożliwia realizację współbieżności, tzn. poszczególne komponenty konstrukcji wykonywane są w tym samym czasie, a prawidłowe zakończenie procesu może nastąpić po uprzednim zakończeniu wszystkich jego elementów. Konstrukcję warunkową rozpoczyna IF. W zależności od wartości logicznej warunków wykonywana jest sekwencja instrukcji umieszczona bezpośrednio po warunku mającym wartość TRUE. Ostatni konstruktor ALT jest związany bezpośrednio z komunikacją wyrażając alternatywę wykonania odpowiednich części bloku od gotowości kanału komunikacyjnego. Słowo kluczowe WHILE z występującym łącznie warunkiem umożliwia powtarzanie sekwencji instrukcji do czasu, gdy warunek będzie miał wartość logiczną FALSE. Tworzenie pętli umożliwi też tzw. replikator konstrukcji FOR. Pętla może być w poszczególnych przebiegach różna, gdyż FOR odnosi się do występującej z nim konstrukcji. Selektor CASE zestawia w programie rozgałęzienia (analogicznie jak np. w BASIC-u instrukcja IF... WHEN...ELSE). OCCAM-2 umożliwia tworzenie procedur, definiowanie funkcji, deklarowanie wartości zmiennych, przypisanie typu kanałom, zmiennym, tablicom i wartościom. Posługuje się znakami ASCII wyróżniając pewną grupę znaków specjalnych. Łańcuchy mogą być tworzone ze wszystkich znaków z wyjątkiem: *,',". W wyrażeniach stosuje się operatory opisane w tabeli. Niektóre z nich mają swoje odpowiedniki w postaci znaków (pary znaków). Gdy łańcuch zawiera parę znaków, z których pierwszy jest * , to jego wartość jest równa 0, a znaczenie opisuje drugi znak.

Do prymitywnych typów w OCCAM zalicza się: INT, INT32, INT 64, REAL32, REAL64, BYTE, TIMER oraz CHAN OF type. Wszystkim zmiennym wyrażeniom i wartościom przypisywany jest typ, który może należeć do prymitywów albo być typem tablicy, rekordu lub struktury. CHAN OF protocol (type określa kanał i specyfikuje protokół komunikacji). TIMER jest typem określanym dla zmiennych, które w programie (procesie) odmierzają czas. Forma [*n*] BYTE przedstawia łańcuch o n znakach, a [*n*]INT opisuje

Operator	Typ operandu	Opis
+, -, *, /, \ (REM)	INTEGER, REAL	op. arytmetyki
PLUS, MINUS, TIMER, AFTER	INTEGER	op. arytmetyki modu
=, <>	wszystkie typy	op. relacji
>, <, <=, >=	INTEGER REAL	op. relacji
AND, OR, NOT	BOOLEAN	op. logiczne
>< , /\ (BITAND), \/ (BITOR, ~ (BITNOT)	INTEGER	op. logiczne bitów
>>, <<	INTEGER	op. przesunięcia

Reprezentacja łańcuchowa znaków specjalnych				
*c - CR	*n - LF	*t - TAB	*s - Spacja	*' - '
*" - "	** - *	*= - screen HEX		

tablicę o *n* elementach typu INT. Przy definiowaniu procedur i funkcji przypisuje się im nazwy oraz parametry formalne ujęte w nawiasach. Wywoływanie tych struktur dokonywane jest przez nazwę z listą parametrów. W OCCAM-2 można uwzględnić konfigurację systemu, tym samym zapewniając prawidłowy rozdział programu na poszczególne transputery. Konfiguracja nie wpływa na samą logikę zachowań programowych. Przez użycie formuły PLACED PAR określamy lokalizację procesu w pamięci lub przypisujemy konkretny procesor jako wykonawcę procesu. Używane zmienne i zegar muszą być deklarowane oddzielnie dla każdej lokalizacji procesu. Formuły PRI PAR i PRI ALT nadają priorytety procesom wykonywanym przez ten sam transputer. Formuła PLACE.....AT... określa adres fizyczny zmiennej, kanału, tablicy lub portu I/O w pamięci. Oto ilustracja programowa.




```
Val ekran IS #B800: —— nadanie wartości zmiennej ekranu
[32000] BYTE ekran_bufor: —— deklaracja tablicy bajtowej
PLACE ekran_bufor AT ekran: —— lokalizacja bufora w pamięci
  . . . . . .
ekran_bufor 57: = FF —— 57 element tablicy przyjmuje wartość FF
  . . . . . .
```

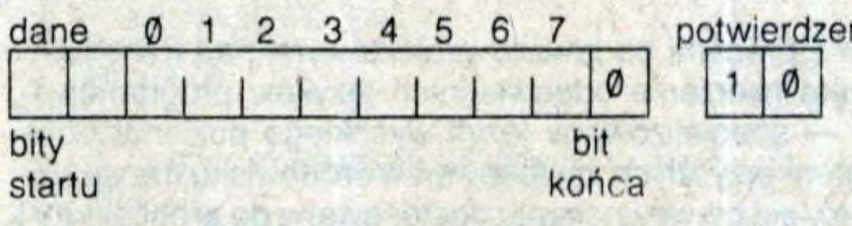
Poniższy fragment programu ilustruje przydział procesów w procedurze sortującej elementy.

```
PLACED PAR i = 0 FOR numer
 PROCESOR i —— Lokalizacja
    input     IS p[i]: —— przypisanie
    output    IS p[i + 1]:
    INT max: —— deklaracja typu zmiennej max
SEQ
   input ? max
   SEQ j = 0 FOR numer -1 —— przemieszczenie danych
     INT następny : —— deklaracja typu zmiennej następny
     SEQ
       input ? następny
       IF
         następny <= max
           output ! następny
         następny > max
           SEQ
             output ! max
             max := następny
   output ! max
..... —— procedury kopiowania posortowanych elementów
.....
```

Transputery zawierają w swojej architekturze zegar. Umożliwia to bezpośrednie i niezależne modelowanie czasu dla danego procesu, tzn. pracę w czasie rzeczywistym. Zmienne typu TIMER stosowane są do wewnętrznych pomiarów czasu lub ustalania harmonogramu czynności. Poniższy przykład ilustruje realizację opóźnienia PROC czekaj (VAL INT cykl_op) —— procedura z parametrem typu INT

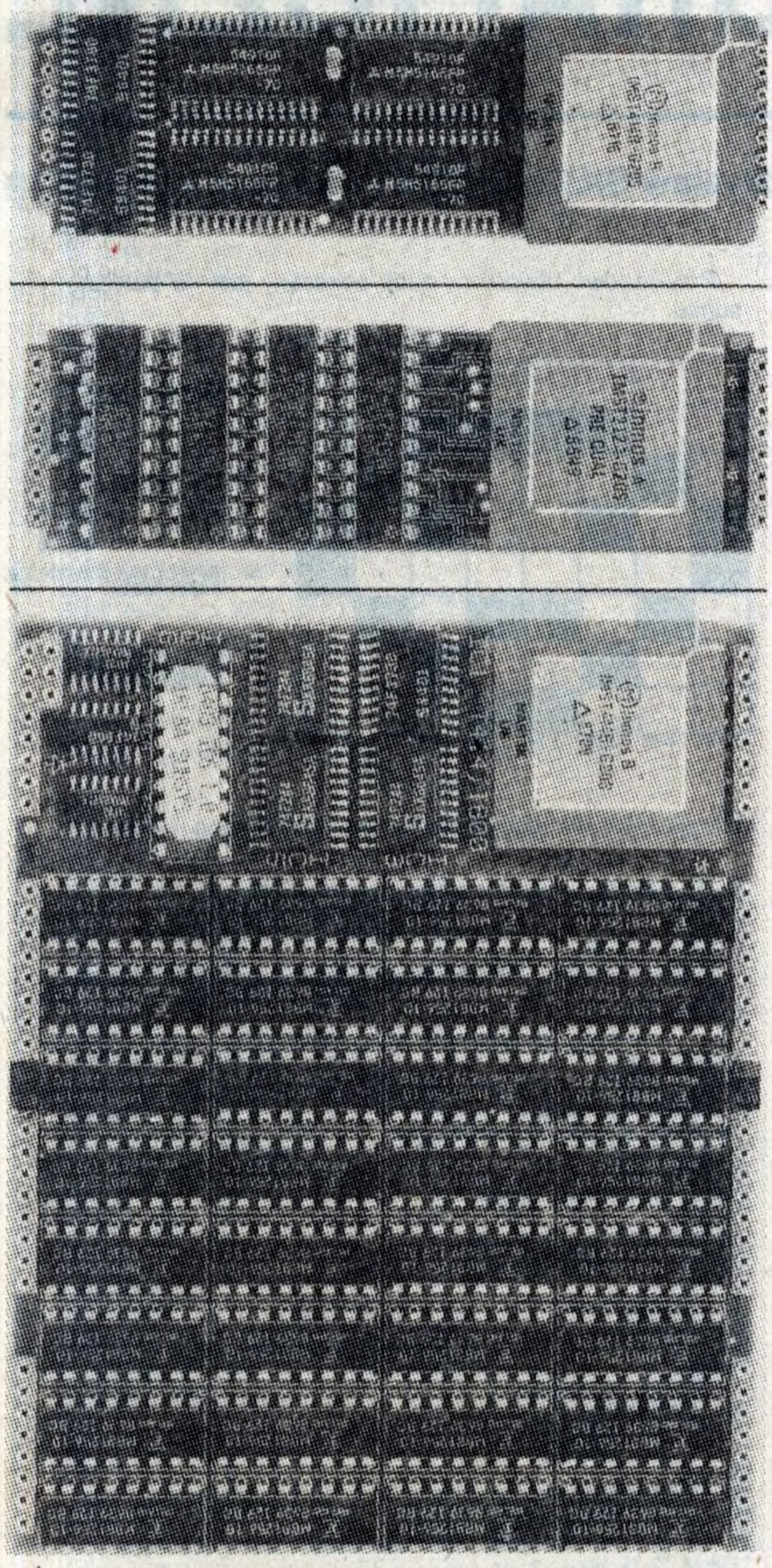
```
  TIMER zegar : —— deklaracja typu zmiennej zegar
  INT aktualny_czas : —— deklaracja typu zmiennej aktualny_czas
  SEQ
    zegar ? aktualny_czas
    zegar ? AFTER aktualny_czas PLUS cykl_op
:
INT y : —— własny proces
SEQ
  input1 ? y —— y przyjmuje wartość z kanału input1
  czekaj (1000)
  output1 ! y —— wyślij wartość zmiennej y do kanału output1
  .....
  .....
```

Niezmiernie trudno jest w zwięzły sposób przedstawić język, jego strukturę, formuły zapisu i cechy charakterystyczne, dlatego ograniczyłem się do omówienia podstawowych zasad i wiadomości o OCCAM — 2. Zainteresowanych niestety muszę odesłać do opracowań aplikacyjnych firmy INMOS lub wydawnictw zagranicznych. Na zakończenie chciałbym przedstawić kilka uwag o komunikacji. Będzie to krótki wstęp do następnego odcinka poświęconego zastosowaniom transputerów. Komunikacja między procesami odbywa się podobnie do komunikacji sprzętowej. Jeśli pewien kanał jest używany jako wejście jednego procesu oraz jako wyjście w drugim procesie, to transmisja może się odbywać tylko wtedy, gdy oba procesy są gotowe. Wartość, która będzie przesyłana, jest kopiowana z procesu wysyłającego do przyjmującego. W ten sposób interpretuje się współbieżną pracę procesów w jednym transputerze. Kanałem w tym wypadku jest pamięć. W wypadku sieci transputerów, każdy z nich wykonuje proces w nim ulokowany. Transmisja dokonuje się bezpośrednio przez parę jednokierunkowych przewodów. Dane są przesyłane szeregowo wg poniższego protokołu.



Protokół zapewnia prawidłową komunikację między transputerami z różną długością słowa. Linie komunikacyjne nie są czułe na fazę zegara. Transputerowa rodzina zawiera adaptery zapewniające sprzężenia transputerowych linii komunikacyjnych z urządzeniami nietransputerowymi. Adapter IMS C012 przedstawiony na rysunku jest uniwersalnym interfejsem sprzężenia między interfejsami współpracujących urządzeń, zapewniającym pełny duplex i dużą szybkość transmisji (10 lub 20 Mbit/s). Może być używany w połączeniach między systemami transputerowymi, do połączeń transputera z kontrolerami peryferiali f-my INMOS, systemem I/O lub układami mikroprocesorowymi INTEL, MOTOROLA i innych firm. Adapter ten przyjmuje z linii komunikacyjnej INMOS szeregowo i przekształca je w postać 8-bitowego słowa lub odwrotnie. Służą w tym celu rejestry we/wy danych. Kontrolę zapewnia system sterujący, który bada słowo stanu rejestru sterującego oraz rejestry statusu stowarzyszone z rejestrami we/wy danych. 8-bitowe dane są wysyłane, bądź pobierane z bufora danych D0—D7

od strony magistrali systemowej. Do bloku sterującego doprowadzone są sygnały zasilania, zegar oraz sygnał separacji i sygnał wyboru szybkości transmisji. Rejestr stanu bloku sterującego zatrzaskuje informacje o trybie pracy (bity RW, RS1, RSØ) oraz o gotowości linii komunikacyjnej INMOS. Na kolejnym rysunku przedstawiona została karta firmy Transtech TSMB-16. Jest ona kartą umożliwiającą budowanie sieci złożonej z modułów transputerowych (do 16 modułów z transputerem T-414 lub T-800 z 64 MB



DRAM). W skład jej wchodzą adaptery linii z 2 interfejsami RS232 oraz 2 moduły INMOS IMS CØØ4 — które stanowią programowalną łącznicę z zespołem 32 linii we i 32 linii wy. Programowanie przełącznicy odbywa się poprzez 2 linie zwane liniami konfiguracji, ponadto może ono się odbywać w sposób dynamiczny. Możliwe jest łączenie kaskadowe modułów CØØ4. Szybkość transmisji wynosi 10 lub 20 Mb/s zapewniając maksymalne opóźnienie do 2 bitów. Karta TSMB ma także 2 linie (z 8 możliwych, wybieranych programowo), które są wyprowadzone na zewnątrz (backplane). W ten sposób można do niej dołączać inne karty modułów transputerowych firmy Transtech lub INMOS i kompatybilne z nimi. TSMB-16 znacznie zwiększa możliwości budowania na jej podstawie wielkich systemów transputerowych i mikroprocesorowych. Jest ona wyposażona w niezbędne sygnały sterujące i testujące, zapewniając właściwy poziom współpracy wszystkich elementów sieci.

W.M.S

Przedstawiamy zaktualizowaną listę (por. „IKS" nr 1/87 oraz nr 12/87) skrótów, które związane są z mikrokomputerem, łącznie z ich oryginalną interpretacją. Czekamy na inne, Waszym zdaniem, ważne skróty.

W jednym z najbliższych numerów „IKS-a" postaramy się przedstawić uaktualnioną listę skrótów wraz z ich tłumaczeniem (interpretacją) w języku polskim.

SLAM - SCAN-LINE ACCESS MEMORY,
SLIC - SELECTIVE LISTING COMBINATION,
SLT - SOLID LOGIC TECHNOLOGY,
SNA - SYSTEM NETWORK ARCHITECTURE,
SNI - SERIAL NETWORK INTERFACE
SNR - SIGNAL-TO-NOISE RATIO,
SOH - START OF HEADER,
SOL - SIMULATION ORIENTED LANGUAGE,
SP - SPACE CHARACTER,
SP - STACK POINTER,
SPOOL - SIMULTANEOUS PERIPHERAL OPERATIONS OFF-LINE,
SPP - SPECIAL-PURPOSE PROCESSOR,
SPRS - SIGNAL PROCESSING ROUTER/SCHEDULER,
SPT - SHORTEST PROCESSING TIME,
SQL - STANDARD QUERY LANGUAGE,
SR - SEQUENCE REGISTER,
SRAM - STATIC RAM,
SRQ - SERVICE REQUEST,
SSI - SMALL SCALE INTEGRATION,
SST - SYNCHRONOUS SYSTEM TRAP,
STX - START OF TEXT,
SYN - SYNCHRONOUS IDLE,
SYSIN - SYSTEM INPUT,
TBF - TIME BETWEEN FAILURES,
TCC - TRANSMISSION CONTROL CHARACTER,
TCP - TRANSMISSION CONTROL PROTOCOL,
TDM - TIME DIVISION MULTIPLEXING,
TDMA - TIME DIVISION MULTIPLE ACCESS,
TELEX - TELEPRINTER-EXCHANGE,
TPI - TRACKS PER INCH,
TRL - TRANSISTOR-RESISTOR LOGIC,
TS - TRANSPORT STATION,
TSL - THREE-STATE-LOGIC,
TSS - TIME-SHARING SYSTEM,
TSS - TASK STATE SEGMENT,
TTD - TEMPORARY TEXT DELAY,
TTL - TRANSISTOR-TRANSISTOR LOGIC,
UART - UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER,
UC - UPPER CASE,
UDC - UNIVERSAL DECIMAL CODE,
UPC - UNIVERSAL PRODUCT CODE,
US - UNIT SEPARATOR,
USART - UNIVERSAL SYNCHRONOUS/ASYNCHRONOUS RECEIVER-
-TRANSMITTER,
VC - VIRTUAL CALL,
VDU - VIDEO DISPLAY UNIT,
VDU - VISUAL DISPLAY UNIT,
VGA - VIDEO GATE ARRAY,
VIC - VIDEO INTERFACE CHIP,
VLSI - VERY LARGE SCALE INTEGRATION,
VOLSER - VOLUME SERIAL NUMBER,
VSAM - VIRTUAL STORAGE ACCESS METHOD,
VRC - VERTICAL REDUNDANCY CHECK,
VT - VERTICAL TAB,
VTAM - VIRTUAL TELECOMMUNICATION ACCESS METHOD,
VTOC - VOLUME TABLE OF CONTENTS,
VW-GRAMMAR - VAN WIJNGAARDEN GRAMMAR,
WACK - WAIT BEFORE TRASMITTING POSITIVE ACKNOWLEDGE-
MENT,
WADEX - WORD AND AUTHOR INDEX,
WE - WRITE ENABLE,
WPR - WRITE PERMISSION RING,

— *Komputer prosi Kowalskiego.*

SAMI TWORZYMY SYMBOLE GRAFICZNE

Mieczysław SKONIECZNY , Robert ZUGEHOR

Początkującemu użytkownikowi mikrokomputera wiele trudności nastręcza definiowanie własnych symboli graficznych.

Prezentowany program ma na celu wyjaśnienie tego problemu na przykładzie mikrokomputera AMSTRAD CPC 6128. Program kreśli na ekranie powiększoną siatkę znaku, złożoną z 64 punktów. Następnie prezentowana jest: postać jednego wiersza znaku graficznego oraz odpowiadająca mu binarna wartość bajtu. Ma to na celu wyjaśnienie zależności pomiędzy wypełnieniem odpowiedniego pola wiersza znaku graficznego a jego interpretacją liczbową i binarną. Wypełniając pixelami kolejne pola, obserwujemy zmianę obliczanych wartości bajtu. Tę sytuację prezentują poniższe rysunki:

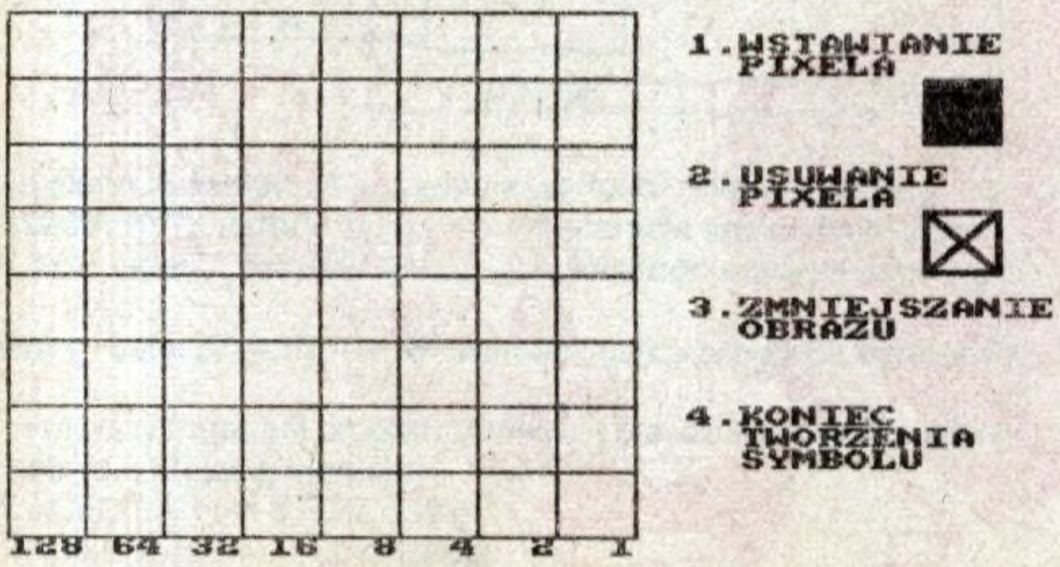
rys. 1

rys. 2

Dalsza część programu pozwala na tworzenie całego znaku graficznego. Możliwe są następujące opcje, których wybór następuje przez kolejne przyciskanie klawisza SPACJA (akceptacja opcji — ENTER):

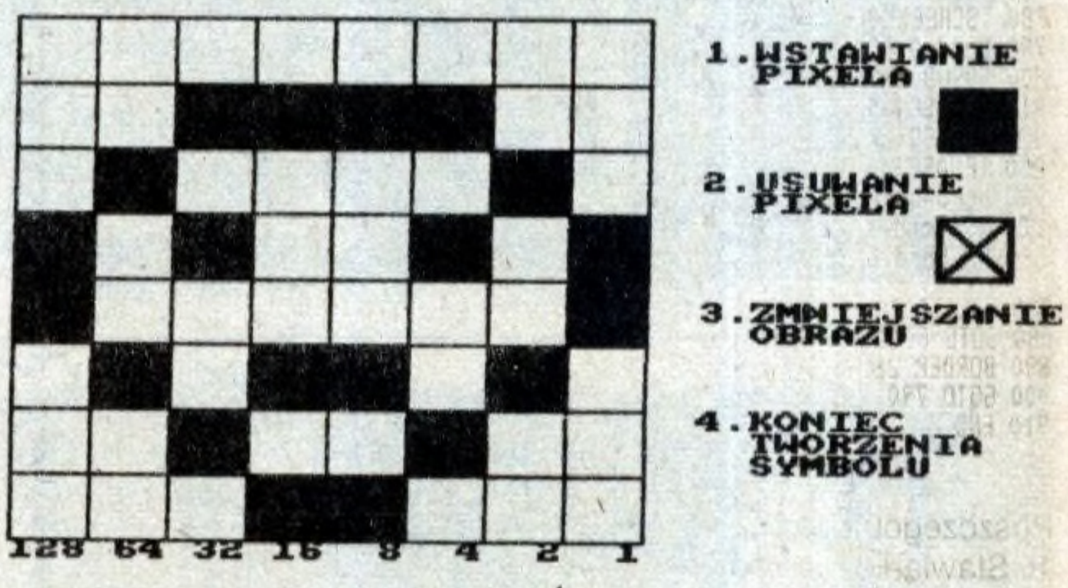
1. Wstawienie pixela
2. Usuwanie pixela
3. Zmniejszanie obrazu
4. Koniec tworzenia symbolu

Początkową fazę tego etapu przedstawia poniższy rys.



rys. 3

Przykładowo utworzony znak pokazany jest na poniższym rysunku.



rys. 4

Wersja źródłowa tego programu ma następującą postać:

```
10 INK 0,24:INK 1,6:INK 2,0:BORDER 24:PAPER 0:MODE 0
20 t2$="      GRAFIKA          ":t3$=" MIKROKOMPUTEROWA  "
30 PEN 1
40 FOR x=1 TO 20
50 LOCATE 1,13:PRINT MID$(t2$,21-x,x);:LOCATE 21-x,14:PRINT MID$(t3$,1,x);
:NEXT
60 GOSUB 2700

70 GOSUB 2550
80 MODE 1:INK 1,6:INK 0,0:INK 2,9:INK 3,26:PEN 0:PAPER 3:BORDER 26:CLS
90 GRAPHICS PEN 0:PEN 1
100 WINDOW £1,12,29,11,14:PAPER £1,0
110 DIM c(8)
120 FOR x=127 TO 511 STEP 48
130 PLOT x,400:DRAWR 0,-384
140 NEXT
150 FOR y=399 TO 0 STEP -48
160 PLOT 127,y:DRAWR 384,0
170 NEXT
180 !SCREENCOPY,2,1
190 a$=INKEY$
200 CLS £1:PEN £1,1:LOCATE £1,6,2:PRINT £1,"POSTAC":LOCATE £1,7,3:PRINT £1,
"ZNAKU"
210 FOR t=1 TO 300:NEXT
220 !SCREENCOPY,1,2
230 FOR t=1 TO 300:NEXT
240 IF a$="" GOTO 190
250 CLS
260 LOCATE 3,5:PRINT"0  0  0  0  0  0  0  0  = 00000000":GOSUB 2730
270 LOCATE 2,3:PRINT"POSTAC BAJTU"
280 LOCATE 2,9:PRINT"WIERSZ ZNAKU GRAFICZNEGO":LOCATE 2,13:PRINT
"128 64 32 16  8  4  2  1"
290 GOSUB 2760
300 !SCREENCOPY,2,1
310 PEN 0:CLEAR INPUT
320 !SCREENCOPY,1,2
330 a$=INKEY$
340 FOR j=1 TO 3:LOCATE 26,9+j:PRINT CHR$(201);CHR$(201);CHR$(201);:NEXT
350 FOR t=1 TO 200:NEXT
360 FOR j=1 TO 3:LOCATE 26,9+j:PRINT"   ";:NEXT
370 IF a$="" GOTO 330
380 IF a$=" " GOTO 660
390 px=2:kx=23:x=26:y=10
400 a$=INKEY$
410 IF a$="" THEN !SCREENCOPY,1,2:GOSUB 2470:GOSUB 2510:GOTO 400
420 IF ASC(a$)=13 GOTO 520
430 IF ASC(a$)=32 GOTO 660
440 IF ASC(a$)<242 OR ASC(a$)>243 GOTO 400
450 IF ASC(a$)=242 THEN x=x-3
460 IF ASC(a$)=243 THEN x=x+3
470 IF x<px THEN x=px
480 IF x>kx THEN x=kx
490 !SCREENCOPY,1,2
500 GOSUB 2510
```

```
510 GOTO 400
520 IF x>kx GOTO 330
530 c(INT(x/3)+1)=1:PEN 1
540 FOR i=1 TO 8
550 LOCATE 3*(i-1)+2,5:PRINT c(i);:NEXT:GOSUB 2730
560 LOCATE 29,5
570 FOR i=1 TO 8
580 PRINT MID$(STR$(c(i)),2,1);:NEXT
590 !SCREENCOPY,2,1:LOCATE 1,15:s=0:PEN 0
600 FOR i=1 TO 8
610 m=c(i)*2^(8-i):s=s+m:PRINT MID$(STR$(m),2,3);
620 IF i=8 THEN PRINT"="; ELSE PRINT"+";
630 NEXT
640 PRINT s
650 PEN 0:GOTO 330
660 !SCREENCOPY,1,2:ERASE c
670 GOSUB 2700:CLS
680 LOCATE 1,25:PRINT"128 64 32 16  8  4  2  1";
690 GRAPHICS PEN 0
700 FOR x=0 TO 384 STEP 48
710 PLOT x,400:DRAWR 0,-384
720 NEXT x
730 FOR y=399 TO 0 STEP -48
740 PLOT 0,y:DRAWR 384,0
750 NEXT y
760 !SCREENCOPY,3,1
770 GOSUB 2320:GOSUB 2370:GOSUB 2420:GOSUB 2440
780 !SCREENCOPY,2,1
790 bord=0:GOSUB 2320:GOSUB 2370:GOSUB 2420:GOSUB 2440
900 GOSUB 2700
810 IF ASC(a$)<>32 AND ASC(a$)<>13 GOTO 800
820 IF ASC(a$)=13 AND bord<>0 GOTO 890
830 IF ASC(a$)=13 AND bord=0 THEN GOTO 800
840 bord=bord+1
850 IF bord>4 THEN bord=1
860 ON bord GOSUB 2800,2820,2840,2860
870 LOCATE 38,24:PRINT bord:PAPER 3
880 GOTO 800
890 BORDER 26:ON bord GOSUB 1000,1310,1620,2020
900 GOTO 790
910 END
```

Poszczególne funkcje realizują następujące podprogramy:

1. Stawianie punktu w dowolnym miejscu siatki znaku.

```
1000 PEN 0:CLEAR INPUT
1010 !SCREENCOPY,1,3
1020 a$=INKEY$
1030 FOR j=1 TO 3:LOCATE 25,21+j:PRINT CHR$(201);CHR$(201);CHR$(201);:NEXT
1040 FOR t=1 TO 200:NEXT
1050 FOR j=1 TO 3:LOCATE 25,21+j:PRINT"   ";:NEXT
1060 IF a$="" GOTO 1020
1070 IF a$=" " GOTO 1290
1080 px=1:kx=22:py=1:ky=22:x=25:y=22
1090 a$=INKEY$
1100 IF a$="" THEN !SCREENCOPY,1,3:GOSUB 2470:GOSUB 2510:GOTO 1090
1110 IF ASC(a$)=13 GOTO 1250
1120 IF ASC(a$)=32 GOTO 1290
1130 IF ASC(a$)<240 OR ASC(a$)>243 GOTO 1090
1140 IF ASC(a$)=240 THEN y=y-3
1150 IF ASC(a$)=241 THEN y=y+3
1160 IF ASC(a$)=242 THEN x=x-3
1170 IF ASC(a$)=243 THEN x=x+3
1180 IF x<px THEN x=px
1190 IF x>kx THEN x=kx
1200 IF y<py THEN y=py
1210 IF y>ky THEN y=ky
1220 !SCREENCOPY,1,3
1230 GOSUB 2510
1240 GOTO 1090
1250 IF x>kx GOTO 1020
1260 !SCREENCOPY,3,1
1270 a(INT(x/3)+1,INT(y/3)+1)=1
1280 GOTO 1020
1290 !SCREENCOPY,1,3
1300 RETURN
```

Sterowanie ruchem „kostki" odbywa się za pomocą klawiszy funkcyjnych (↑,↓,→,←). Umieszczenie „kostki" w wybranym miejscu — ENTER, przejście do innej funkcji programu — SPACJA.

2. Usuwanie pixela z dowolnego pola.

```
1310 PEN 1:CLEAR INPUT
1320 !SCREENCOPY,1,3
1330 a$=INKEY$
1340 FOR j=1 TO 3:FOR i=1 TO 3:LOCATE 24+j,21+i:PRINT CHR$(201+3*(i-1)+j);
:NEXT i:NEXT j
1350 FOR t=1 TO 200:NEXT
1360 FOR j=1 TO 3:LOCATE 25,21+j:PRINT"   ";:NEXT
1370 IF a$="" GOTO 1330
1380 IF a$=" " GOTO 1600
1390 px=1:kx=22:py=1:ky=22:x=25:y=22
1400 a$=INKEY$
1410 IF a$="" GOTO 1400
1420 IF ASC(a$)=13 GOTO 1560
1430 IF ASC(a$)=32 GOTO 1600
1440 IF ASC(a$)<240 OR ASC(a$)>243 GOTO 1400
1450 IF ASC(a$)=240 THEN y=y-3
1460 IF ASC(a$)=241 THEN y=y+3
1470 IF ASC(a$)=242 THEN x=x-3
1480 IF ASC(a$)=243 THEN x=x+3
1490 IF x<px THEN x=px
1500 IF x>kx THEN x=kx
1510 IF y<py THEN y=py
1520 IF y>ky THEN y=ky
1530 !SCREENCOPY,1,3
1540 FOR j=1 TO 3:FOR i=1 TO 3:LOCATE x-1+j,y-1+i:PRINT CHR$(201+3*(i-1)+j)
::NEXT j:NEXT j
1550 GOTO 1400
1560 IF x>kx GOTO 1330
1570 PEN 0:LOCATE x,y:PRINT CHR$(211);CHR$(212);CHR$(212);:LOCATE x,y+1:
PRINT CHR$(213);"  ";:LOCATE x,y+2:PRINT CHR$(213);"  ";:PEN
 1:!SCREENCOPY,3,1
1580 a(INT(x/3)+1,INT(y/3)+1)=0
1590 GOTO 1330
1600 !SCREENCOPY,1,3
1610 RETURN
```

Sterowanie symbolem „usuwania" odbywa się jak wyżej.

3. Pomniejszanie obrazu.

```
1620 !SCREENCOPY,1,3
1630 skala=1
1640 a$=INKEY$
1650 IF a$="" GOTO 1640
1660 IF a$=" " GOTO 2000
1670 IF ASC(a$)<240 OR ASC(a$)>241 GOTO 1640
1680 IF ASC(a$)=241 THEN skala=skala*2
1690 IF ASC(a$)=240 THEN skala=skala/2
1700 IF skala<1 THEN skala=1
1710 IF skala>16 THEN skala=32
1720 CLS
1730 IF skala =1 THEN !SCREENCOPY,1,3:GOTO 1640
1740 GRAPHICS PEN 0
1750 IF skala=16 GOTO 1830
1760 IF skala=32 GOTO 1940
1770 FOR x=0 TO 384/skala STEP 48/skala
1780 PLOT x,400:DRAWR 0,-384/skala
1790 NEXT
1800 FOR y=399 TO 399-(8*48/skala) STEP -48/skala
1810 PLOT 0,y:DRAWR 384/skala,0
1820 NEXT y
1830 FOR i=1 TO 8
1840 FOR j=1 TO 8
1850 IF a(i,j)=0 GOTO 1910
1860 poczk=(i-1)*48/skala:konk=poczk+48/skala
1870 poczl=399-(j-1)*48/skala:konl=poczl-48/skala
1880 FOR k=poczk TO konk
1890 PLOT k,poczl:DRAWR 0,-48/skala
1900 NEXT
1910 NEXT j
1920 NEXT i
1930 GOTO 1640
1940 FOR i=0 TO 7
1950 FOR j=0 TO 7
1960 IF a(i+1,j+1)=0 THEN GRAPHICS PEN 3 ELSE GRAPHICS PEN 0
1970 PLOT 2*i,399-2*j
1980 NEXT j:NEXT i
1990 GOTO 1640
2000 !SCREENCOPY,1,3
2010 RETURN
```

Znak jest przedstawiany w 6 kolejnych, pomniejszanych skalach aż do naturalnej wielkości.

Pomniejszanie — klawisz ↓

Powiększanie — klawisz ↑

4. Koniec tworzenia symbolu.

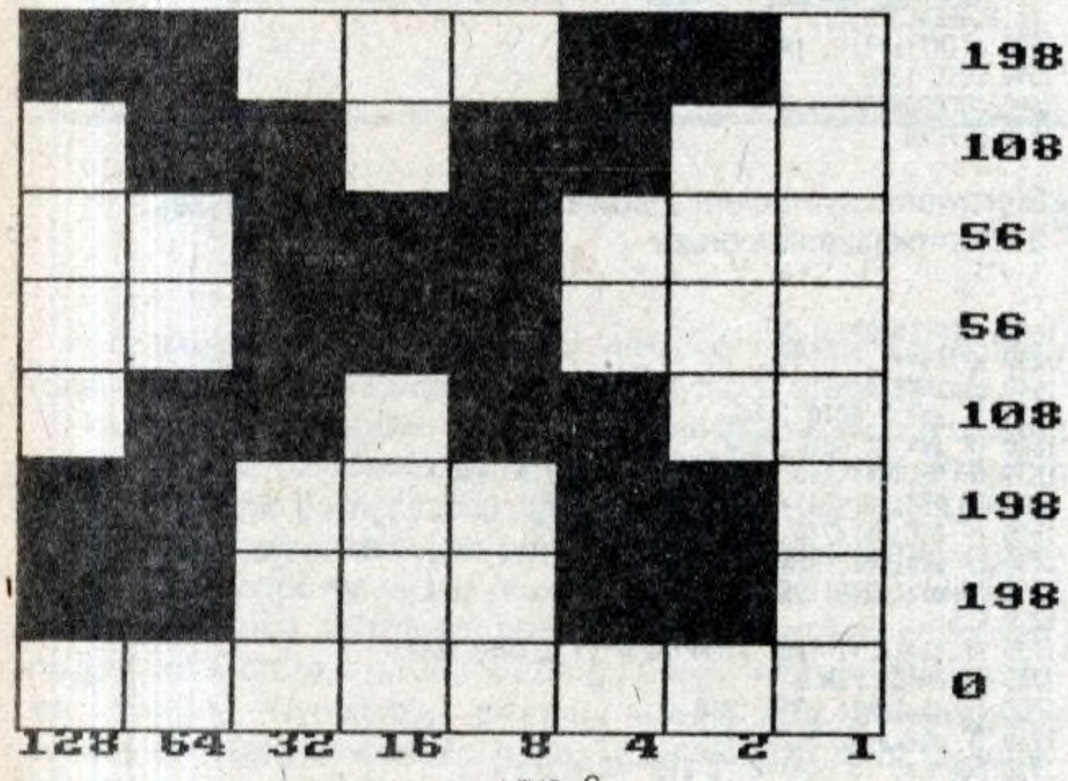
```
2020 !SCREENCOPY,1,3
2030 PEN 0:DIM b(8)
2040 FOR i=1 TO 8
2050 licz=0
2060 FOR j=1 TO 8
2070 licz=licz+a(j,i)*2^(8-j)
2080 NEXT j
2090 LOCATE 26,3*(i-1)+2:PRINT licz;:b(i)=licz
2100 NEXT i
2110 GOSUB 2700
2120 CLS
2130 FOR i=0 TO 7
2140 FOR j=0 TO 7
2150 IF a(i+1,j+1)=0 THEN GRAPHICS PEN 3 ELSE GRAPHICS PEN 0
2160 PLOT 2*i,399-2*j
2170 NEXT j:NEXT i
2180 LOCATE 3,1:PEN 1:PRINT"SYMBOL GRAFICZNY":LOCATE 3,11:PRINT"INSTRUKCJE
 PROGRAMU":PEN 0
2190 LOCATE 1,12:PRINT"...":LOCATE 1,13:PRINT"...":LOCATE 1,14:PRINT"1000
SYMBOL AFTER n"
2200 LOCATE 1,15:PRINT"1010 SYMBOL n";
2210 FOR i=1 TO 8
2220 PRINT",";MID$(STR$(b(i)),2,3);
2230 NEXT i
2240 GOSUB 2700
2250 CLS:LOCATE 1,12:PRINT"            NOWY SYMBOL (T/N)"
2260 a$=INKEY$
2270 IF a$=""GOTO 2260
2280 IF UPPER$(a$)="T" THEN ERASE a:ERASE b:CLS:GOTO 680
2290 IF UPPER$(a$)="N" THEN !SCREENCOPY,1,3:ERASE b:GOTO 2310
2300 GOTO 2260
2310 RETURN
```

Z prawej strony siatki znaku wyświetlane są na wysokości odpowiednich wierszy ich wartości liczbowe. Następnie jest pokazany fragment programu definiujący dany symbol graficzny.

Tę sytuację prezentują poniższe rysunki.

```
X SYMBOL GRAFICZNY

  INSTRUKCJE PROGRAMU
...
1000 SYMBOL AFTER n
1010 SYMBOL n,198,108,56,56,108,198,198,
0
```

rys. 5



rys. 6

Dodatkowo są wykorzystywane następujące podprogramy pomocnicze:

```
2320 FOR i=1 TO 3:FOR j=1 TO 3
2330 LOCATE 35+j,3+i:PEN 0:PRINT CHR$(201);
2340 NEXT j:NEXT i
2350 LOCATE 27,2:PRINT"1.WSTAWIANIE":LOCATE 27,3:PRINT"  PIXELA";
2360 RETURN
2370 FOR i=1 TO 3:FOR j=1 TO 3
2380 LOCATE 35+j,9+i:PEN 1:PRINT CHR$(201+3*(i-1)+j);
2390 NEXT j:NEXT i
2400 LOCATE 27,8:PRINT"2.USUWANIE ":LOCATE 27,9:PRINT"  PIXELA";
2410 RETURN
2420 LOCATE 27.14:PEN 2:PRINT"3.ZMNIEJSZANIE";:LOCATE 27,15:PRINT"  OBRAZU";
2430 RETURN
2440  WINDOW £1,26,40,18,22:PAPER £1,0:CLS £1
2450 LOCATE £1,2,2:PEN £1,3:PRINT £1,"4.KONIEC":LOCATE £1,2,3:PRINT£1,
"  TWORZENIA";:LOCATE £1,2,4:P
RINT£1,"  SYMBOLU"
2460 RETURN
2470 FOR i=0 TO 2
2480 LOCATE x,y+i:PRINT "   ";
2490 NEXT i
2500 RETURN
2510 FOR i=0 TO 2
2520 LOCATE x,y+i:PRINT CHR$(201);CHR$(201);CHR$(201);
2530 NEXT i
2540 RETURN
2550 SYMBOL AFTER 201
2560 SYMBOL 201,255,255,255,255,255,255,255,255
2570 SYMBOL 202,255,255,240,248,220,206,199,195
2580 SYMBOL 203,255,255,0,0,0,0,0,129
2590 SYMBOL 204,255,255,15,31,59,115,227,195
2600 SYMBOL 205,193,192,192,192,192,192,192,193
2610 SYMBOL 206,195,231,126,60,60,126,231,195
2620 SYMBOL 207,131,3,3,3,3,3,3,131
2630 SYMBOL 208,195,199,206,220,248,240,255,255
2640 SYMBOL 209,129,0,0,0,0,0,255,255
2650 SYMBOL 210,195,227,115,59,31,15,255,255
2660 SYMBOL 211,255,128,128,128,128,128,128,128
2670 SYMBOL 212,255
2680 SYMBOL 213,128,128,128,128,128,128,128,128
2690 RETURN
2700 a$=INKEY$
2710 IF a$="" GOTO 2700
2720 RETURN
2730 PLOT 15,351:DRAWR 384,0:PLOT 15,303:DRAWR 384,0
2740 FOR x=15 TO 399 STEP 48:PLOT x,351:DRAWR 0,-48:NEXT
2750 RETURN
2760 PLOT 15,255:DRAWR 384,0:PLOT 15,207:DRAWR 384,0
2770 FOR x=15 TO 399 STEP 48:PLOT x,255:DRAWR 0,-48:NEXT
2780 RETURN
2800 BORDER 0:PEN 0
2810 RETURN
2820 BORDER 6:PEN 1
2830 RETURN
2840 BORDER 9:PEN 2
2850 RETURN
2860 BORDER 26:PAPER 0:PEN 3
2870 RETURN
```

Realizacja programu odbywa się z wykorzystaniem dodatkowych 64 KB. Niezbędne jest więc uprzednie załadowanie programu BANKMANAGER.

Robert ZUGEHOR

Mieczysław SKONIECZNY

Spearmen i Pearson powiedzą Ci prawdę

Mariusz KUBIAK

W czasie jednych z ostatnich ćwiczeń z podstaw informatyki w Wojskowej Akademii Politycznej byłem uczestnikiem atrakcyjnego, a przede wszystkim dającego wiele do myślenia eksperymentu (testu) z wykorzystaniem komputera.

Otóż grupa słuchaczy drugiego roku Wydziału Nauk Pedagogicznych w trakcie zajęć z wykorzystaniem sieci mikrokomputerów BBC miała m.in. za zadanie sprawdzenie możliwości zastosowania mikrokomputerów do obliczeń statystycznych. Wykorzystywane były do tego celu programy na „rachowanie" współczynników Spearmana i Pearsona. Jako przykładowy cel badań (by nie były zbyt odległe od rzeczywistości) przyjęliśmy szukanie zależności między naszymi wynikami nauki uzyskanymi w trzeciej sesji egzaminacyjnej, a następującymi zmiennymi: wiekiem, odległością od miejsca stałego zamieszkania, ilością wyjazdów do domu w ciągu miesiąca, ilością posiadanych dzieci, dochodami na osobę w rodzinie, ilością wypalanych papierosów w czasie dnia.

Oczywiście, aby wyniki testu w sposób maksymalnie obiektywny obrazowały wielkość wpływu poszczególnych zmiennych na ową średnią egzaminacyjną, każdy z nas musiał zdobyć się nie tylko na szczere przeprowadzenie „rachunku sumienia", lecz także ujawnienie swoich słabości do palenia papierosów.

Po wprowadzeniu danych do komputera, a następnie wypisaniu na tablicy wyników, najbardziej zdziwiony i zaskoczony był prowadzący ćwiczenie. Co się bowiem okazało? Ano to, że im słuchacze częściej wyjeżdżają do rodziny i im dalej mieszkają, a także... im więcej palą papierosów, tym lepsze osiągają wyniki w nauce. Natomiast wpływ pozostałych zmiennych okazał się nieco mniejszy, ale także bardzo istotny. Wobec tak nieoczekiwanych wyników sprawdziliśmy dodatkowo na mikrokomputerze raz jeszcze rezultaty testu, licząc — przynajmniej niektórzy z nas — na błąd maszyny. Niestety, mikrokomputer okazał się nieprzekupnym, perfekcyjnym „mózgiem".

Kończąc to krótkie sprawozdanie z ćwiczeń komputerowych, chciałbym zaznaczyć, że moim zdaniem przydatność komputera do pracy dydaktyczno-wychowawczej czy naukowej niewspółmiernie wzrasta, jeśli jest on wykorzystywany do rozwiązywania problemów bezpośrednio dotyczących człowieka, który go obsługuje, ewentualnie zaspokaja jego ciekawość poznawczo-naukową. W każdym razie jeszcze długo po ćwiczeniu komentowaliśmy w swoim gronie jego wyniki, wielu z nas chętniej sięgnęło po czasopisma i specjalistyczną literaturę komputerową, a niektórzy poważnie myślą o zakupie „Spectrum" czy „Atari".

Po raz kolejny przekonaliśmy się, że komputer to nie tylko gry i zabawy wymagające pewnej sprawności manualnej, lecz również cenna pomoc w zdobywaniu wiedzy nie tylko o otaczającej nas rzeczywistości, ale także o nas samych.

INFORMATYCZNY SŁOWNIK ANGIELSKO-POLSKI

END OF MEDIUM CHARACTER - znak fizycznego zakończenia nośnika,
END OF MESSAGE - koniec komunikatu,
END OF RECORD - koniec zapisu (rekordu),
END-OF-RECORD FLAG - identyfikator końca zapisu,
END OF REEL - koniec szpuli,
END OF TEXT CHARACTER - znak zakończenia tekstu (treści),
END OF THE TAPE - znacznik końca taśmy,
END OF TRANSMISSION - koniec transmisji,
END OF TRANSMISSION CHARACTER - znak zakończenia transmisji,
END OF TRANSMISSION BLOCK CHARACTER - znak zakończenia transmisji bloku,
END OFFICE - lokalna centrala telefoniczna,
ENDPOINT - punkt końcowy,
END-TO-END PROTOCOL - protokół końcowy,
ENERGIZE - wzbudzać, zasilać, także: zapalać,

ESCAPE SEQUENCE - sekwencja zarządzająca,
ESCAPING KEY - klawisz powodujący posuw papieru,
ESD - patrz: EXTERNAL SYMBOL DICTIONARY,
ESS - patrz: ELECTRIC SWITCHING SYSTEM,
ESSENCE - istota (rzeczy),
ESSENTIAL - istotny, zasadniczy,
ESTABLISH - ustanowić, założyć,
ESTABLISH A RELATIONSHIP - ustalić zależność,
ESTABLISHMENT - firma, przedsiębiorstwo,
ESTIMATE - oszacowanie, szacunek, ocena, także: szacować, oznaczać,
ESTIMATION - ocena, oszacowanie, także: estymacja,
ETB - patrz: END OF TRANSMISSION BLOCK CHARACTER,
ETCHING - trawienie, wytrawianie,
ETX - patrz: END OF TEXT CHARACTER,
EU - patrz: EXECUTION UNIT,
EURONET - sieć komputerowa EURONET,
EUROPEAN ACADEMIC RESEARCH NETWORK - Międzynarodowy Skomputeryzowany System Wymiany Informacji Naukowej,
EUROPEAN COMPUTER MANUFACTURES ASSOCIATION - Europejskie Stowarzyszenie Producentów Sprzętu Komputerowego,
EUROPEAN INFORMATION NETWORK - sieć komputerowa EIN,
EVADE - unikać, obchodzić,
EVALUATE - obliczać (np. wartość wyrażenia arytmetycznego), oceniać,
EVALUATION - wyznaczanie wartości, ocena, obliczanie,
EVALUATION SYSTEM - system oceny,
EVEN - parzystość, także: równać, wyrównywać,
EVEN NUMBER - liczba parzysta,
EVEN-ODD - parzysty-nieparzysty,
EVEN PARITY BIT - bit parzystości,
EVEN PARITY CHECK - kontrola parzystości,
EVENT - zaszłość, także: zdarzenie, wydarzenie,
EVENT-DRIVEN - zarządzający przerwaniami w trybie przerwania,
EVENT FLAG - znak wystąpienia zdarzenia,
EVENT MODE INPUT - wprowadzenie według kolejki, wprowadzenie z buforowaniem,
EVENT TRAPPING - obsługa przerwania,
EVIDENCE - dowód, także: świadczyć, udowadniać,
EVOLUTION - rozwijanie, rozwój, ewolucja, także: pierwiastkowanie,
EVOLUTIONARY SYSTEM - system rozwijający się,

Słownik na dysku

Od kilku miesięcy zawsze na tych samych kolumnach „IKSa" publikujemy kolejne odcinki informatycznego słownika angielsko-polskiego. Cieszy nas, że ma on liczną grupę odbiorców.

Tym przyjemniej jest nam poinformować, o udanym oprogramowaniu niemal 16-tysięcznego zbioru terminów angielskich i ich polskich odpowiedników. Oprogramowania słownika dokonała firma DORAN, instalując je jako integralną część edytora tekstów, rozbudowującą jego możliwości o funkcję „tłumacza". Mieliśmy okazję korzystać ze słownika w nowej postaci, jest prosty w obsłudze, a szybkość dostępu do każdego z haseł jest mniejsza niż 1/2 sekundy.

Tyle wiemy. O bliższe informacje radzimy się zwracać listownie do firmy DORAN, której specjalnością jest oprogramowanie. 03-756 W-wa ul. Siedlecka 1/15 m. 74.

EVOLVE - wymyśleć, wykombinować,
EXACT - dokładny, ścisły,
EXACT DIVISION - dzielenie bez reszty,
EXACTITUDE - dokładność, ścisłość,
EXAMINE - badać, rozpatrywać,
EXAMPLE - przykład,
EXCEED - przekraczać (np. pojemność pamięci),
EXCEED CAPACITY CHECK - kontrola przepełnienia,
EXCEEDING - niezmierzony, nadzwyczajny,
EXCELLENT - doskonały, znakomity,
EXCEPT - wykluczać, wyłączać, także: z wyjątkiem,
EXCEPTION - wyjątek,
EXCEPTION CONDITION - wydarzenie wyjątkowe,

EXCEPTION HANDLER - program obsługi zdarzenia wyjątkowego,
EXCEPTION HANDLING - obsługiwanie zdarzenia wyjątkowego, (przerwanie wewnętrzne),
EXCEPTIONAL - wyjątkowy,
EXCESS - nadmiar, nadwyżka, przekroczenie,
EXCESS-THREE CODE - kod z nadmiarem trzy, kod Stibitza,
EXCHANGE - wymiana, zamiana, także: wymieniać, zamieniać,
EXCHANGEABLE - wymienny,
EXCHANGEABLE DISK - dysk wymienny,
EXCHANGEABLE DISK STORE - pamięć na dysku wymiennym,
EXCITATION - pobudzenie, wzbudzenie,
EXCITE - wzbudzać, pobudzać,
EXCLAMATION MARK - znak wykrzyknika, "!",
EXCLUDE - wykluczać, wyłączać,
EXCLUSIVE LOCKING - blokada wybiórcza, także: blokowanie dla zapisu,
EXCLUSIVE OR - "LUB" wykluczające,
EXCLUSIVE SEGMENTS - wzajemnie wykluczające się moduły,
EXCLUSIVE USAGE MODE - tryb wyłączności,
EXE - patrz: EXECUTE,
EXE FILE - moduł ładowania, moduł wprowadzania,
EXECUTE - wykonywać,
EXECUTE ONLY PROGRAM - program bez opisów, program bez komentarzy,
EXECUTE PROGRAM - program wykonawczy,
EXECUTING - wykonanie, także: uruchomienie,
EXECUTION - wykonanie,
EXECUTION CYCLE - cykl wykonania,
EXECUTION ENVIRONMENT - warunki (środki) wykonania programu,
EXECUTION TIME - czas wykonania, czas obliczeń,
EXECUTION UNIT - układ wykonawczy,
EXECUTIVE - program zarządzający, egzekutor, także: system operacyjny,
EXECUTIVE DEVICE - człon operacyjny (komputera analogowego),
EXECUTIVE DIRECTIVE - dyrektywa (operacja) systemu operacyjnego,
EXECUTIVE INSTRUCTION - instrukcja (rozkaz) systemu operacyjnego,
EXECUTIVE MODE - priorytetowy tryb pracy,
EXECUTIVE PROGRAM - program organizacyjny, program sterujący, program wykonawczy, program nadzorczy, program zarządzający,
EXECUTIVE REPLIES - informacje EXECUTIVE'a umieszczane w słowach odpowiedzi,
EXECUTIVE RESIDENT - część systemu operacyjnego, rezydująca na stałe w pamięci operacyjnej,
EXECUTIVE SUPERVISOR - program zarządzający systemu operacyjnego,
EXECUTIVE SYSTEM - system operacyjny,
EXECUTIVE SYSTEM UTILITY - systemowy program serwisowy,
EXEMPLIFY - ilustrować (na przykładzie),
EXERCISE CONTROL OVER ... - sprawować kontrolę nad ..., nadzorować,
EXERCISER - program testujący, program kontrolny,
EXHAUSTED ARGUMENT - parametr wyczerpany (oznacza przeanalizowanie wszystkich wariantów problemu),
EXHAUSTIVE SEARCH - pełne przeszukanie,

EXHIBIT - wykazywać (cechę), wskazywać,
EXIST - istnieć,
EXISTENCE - istnienie, egzystencja,
EXIT - wyjście,
EXIT CONDITIONS - warunki wyjścia, także: realizacja warunków po wyjściu z podprogramu,
EXJUNCTION - "LUB" wykluczające,
EXPAND - rozszerzać, rozwijać (np. wzór matematyczny),
EXPANDABILITY - możliwość rozszerzania (np. pojemności pamięci),
EXPANDABLE SYSTEM - system rozszerzalny, system dostosowany do rozbudowy,
EXPANDER - ekspander (sygnału),
EXPANDING - rozszerzanie (się), rozprężanie (się), także: rozwijanie (funkcji),
EXPANSION - rozwinięcie, rozszerzenie, rozbudowa, uaktywnienie,
EXPANSION BOX - blok rozbudowy,
EXPANSION BUS - szyna rozszerzenia,

EXPANSION SLOT - złącze do rozbudowy funkcjonalnej,
EXPECTATION - wartość oczekiwana (zmiennej losowej), wartość przeciętna, także: oczekiwanie,
EXPECTED VALUE - wartość oczekiwana (zmiennej losowej), nadzieja matematyczna, wartość przeciętna,
EXPEL - wydalać, usuwać,
EXPENDABLE - jednorazowego użytku,
EXPENDITURE - wydatki, rozchód,
EXPENDITURE QUOTA - normy zużycia,
EXPENSES - wydatki,
EXPERIENCE - doświadczenie (nabyte),
EXPERIMENT - doświadczenie, eksperyment, także: przeprowadzać doświadczenie, przeprowadzać eksperyment,
EXPERIMENTAL - doświadczalny,
EXPERIMENTAL CENTRE - ośrodek doświadczalny,
EXPERIMENTAL CONDITIONS - warunki doświadczalne,
EXPERIMENTAL ERROR - błąd doświadczenia,
EXPERIMENTATION - wykonywanie doświadczeń, przeprowadzanie doświadczeń,
EXPERT OPINION - ekspertyza,
EXPERT SYSTEM - system ekspertowy, system ekspercki,
EXPIRATION DATE - termin upływu (np. ważności), data ważności,
EXPLANATION - objaśnienie, wyjaśnienie,
EXPLANATION FACILITIES - środki umotywowania, środki uzasadniania (dot. systemów ekspertowych),
EXPLANATORY - wyjaśniający, tłumaczący,
EXPLICIT - jawny, jawnie, także: jasny, sprecyzowany, wyraźny, czytelny,
EXPLODED VIEW - przedstawienie częściowe, także: obraz częściowy,
EXPLORATION - poszukiwanie, badanie,
EXPONENT - cecha, wykładnik potęgi,
EXPONENTIAL - wykładniczy,
EXPONENTIALLY - wykładniczo,
EXPONENTIAL NOTATION - zapis liczb w postaci wykładniczej,
EXPONENTIATE - podnosić do potęgi,
EXPONENTIATION - potęgowanie,
EXPORTED - przesyłany (eksportowany),
EXPRESS - wyrażać się,
EXPRESSION - wyrażenie,
EXTEND - rozciągać, przedłużać, rozszerzać,

EXTEND ACTIVITY - rozszerzać działalność,
EXTENDED ASCII - rozszerzony kod ASCII,
EXTENDED BACKUS-NAUR FORM - rozszerzona notacja Backusa-Naura,
EXTENDED BCD-CODE - patrz: EXTENDED BINARY-CODED DECIMAL CODE,
EXTENDED BINARY-CODED DECIMAL INTERCHANGE CODE - rozszerzony kod binarny do kodowania liczb dziesiętnych,
EXTENDED BINARY-CODED DECIMAL CODE - patrz: EXTENDED BINARY-CODED DECIMAL INTERCHANGE CODE,
EXTENDED BINARY INTERCHANGE CODE - kod 8-bitowy bez bitu kontrolnego (dający 256 możliwości),
EXTENDED CODE - patrz: EXTENDED BINARY INTERCHANGE CODE,
EXTENDED INSTRUCTION SET - repertuar rozkazów rozszerzony,
EXTENDED-PRECISION - podwyższona precyzja,
EXTENDER CARD - karta ekspandera,
EXTENDER CIRCUIT - ekspander,
EXTENDER INPUT - wejście ekspandera,
EXTENDING A FILE - rozszerzanie pliku (zbioru), powiększanie pliku,
EXTEND THE RANGE - zwiększyć zasięg,
EXTENSIBLE - rozszerzony, także: odkryty,
EXTENSIBLE ADDRESSING - adresacja w trybie rozszerzonym, adresacja rozszerzona,
EXTENSIBLE LANGUAGE - język rozszerzony,
EXTENSIBLE REGISTER - rejestr rozszerzony,
EXTENSION - rozwinięcie, rozszerzenie, rozbudowanie, także: zakres (pojęcia),
EXTENSION BUCKET - porcja (blok programowy) w obszarze przepełnień drugiego stopnia,
EXTENSION BUCKET HEADER - etykieta porcji dodatkowej (bloku dodatkowego),
EXTENSION REGISTER - rejestr rozszerzenia,
EXTENSIVE - obszerny, rozległy,
EXTENSIVE DAMAGE - poważne uszkodzenie,
EXTENT - podobszar, zasięg, także: stopień,
EXTERNAL ARITHMETIC - procesor arytmetyczny (dodatkowy),
EXTERNAL DATA REPRESENTATION - zewnętrzna reprezentacja danych,
EXTERNAL DELAY - strata czasu (np. komputera) spowodowana przyczynami zewnętrznymi,
EXTERNAL DEVICE - urządzenie zewnętrzne, urządzenie peryferyjne,
EXTERNAL ENVIRONMENT - warunki eksploatacji,
EXTERNAL INTERRUPT - przerwanie zewnętrzne,
EXTERNAL KEY - klucz zewnętrzny (w relacyjnych bazach danych oznacza dostęp do danych),
EXTERNAL MEMORY - pamięć zewnętrzna,
EXTERNAL PERFORMANCE - rzeczywista szybkość działania,
EXTERNAL REFERENCE - odsyłacz zewnętrzny,
EXTERNAL STORAGE - pamięć zewnętrzna,
EXTERNAL STORE - patrz: EXTERNAL STORAGE,
EXTERNAL SYMBOL DICTIONARY - słownik symboli zewnętrznych, także: lista symboli do tłumaczenia programów,
EXTRA-LARGE-SCALE INTEGRATION - bardzo duża skala integracji,
EXTRA SEGMENT - segment pomocniczy, segment dodatkowy,

EXTRACT - wybór, wyciąg, wypis, także: wydobywać, wybierać, wydostawać,
EXTRACT INSTRUCTION - rozkaz maskowania,
EXTRACTION - wydzielanie, maskowanie,

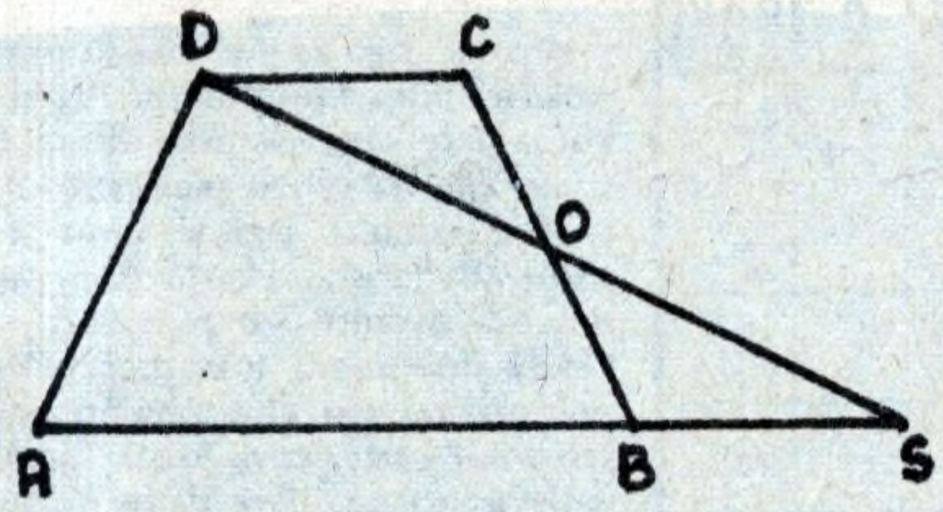
F

FALLBACK - rezerwa awaryjna, przejście na awaryjny tryb pracy, zmiana trybu pracy lub konfiguracji sprzętu w celu uniknięcia niesprawności,
FALLOUT - znikanie, zanikanie, przepadanie,
FALSE - fałszywy, także: fałsz (pojęcie używane w logice),
FALSE ACCEPTANCE - przyjęcie fałszywej hipotezy,
FALSEHOOD - przekłamanie, fałszywy wniosek, fałszywa decyzja,
FALSE REJECTION - odrzucenie hipotezy fałszywej,
FALSE RETRIEVAL - uzyskanie niepotrzebnej informacji w automatycznym systemie wyszukiwania,
FALSE STATEMENT - twierdzenie fałszywe,
FAMILY - rodzina,
FAMILY OF CHARACTERISTICS - rodzina charakterystyk,
FAMILY OF CURVES - rodzina krzywych,
FANFOLD - druk ciągły składany w "harmonię",
FANFOLD FORM - formularze ciągle złożone w postaci harmonii,
FAN-IN - obciążalność wejściowa (bramki logicznej),
FAN-OUT - obciążalność wyjściowa (bramki logicznej),
FAR CALL - wywołanie długie, wywołanie odległe,
FAR END CROSS TALK - przesłuch skrośny,
FAR PLANE - daleki plan (w grafice komputerowej określenie płaszczyzny ograniczającej przestrzeń zobrazowania na ekranie),
FAR-REACHING - dalekosiężny,
FAST - szybki, szybko,
FAST-ACCESS STORAGE - pamięć szybka, pamięć o dostępie szybkim,
FAST-ACTING - szybki, szybko działający,
FAST LINE PRINTER - szybka drukarka wierszowa,
FAST MEMORY - szybka pamięć (operacyjna),
FAT - patrz: FILE ALLOCATION TABLE,
FATAL ERROR - błąd uniemożliwiający kontynuację programu,
FATHER FILE - początkowa (wyjściowa) wersja pliku (zbioru),
FAULT - 1. błąd, 2. uszkodzenie, zakłócenie, niesprawność, 3. wada (fizyczna wada sprzętu),
FAULT CLEARING - likwidacja zakłóceń, usuwanie uszkodzeń,
FAULT DETECTOR - wskaźnik uszkodzeń,
FAULT FINDING - wykrywanie uszkodzeń, także: usuwanie zakłóceń,
FAULTLESS - bezusterkowy, bezbłędny,
FAULTLESS TAPE - taśma sprawdzona, taśma bezbłędna,
FAULT LOCATION - lokalizacja uszkodzeń,
FAULT SIMULATOR - symulator uszkodzeń, symulator zakłóceń,
FAULT TIME - czas przestoju z powodu uszkodzenia,
FAULT-TOLERANT SYSTEM - system odporny na uszkodzenia,
FAULT TRACING - wyszukiwanie uszkodzeń,
FAULT TREE GATE - element drzewa niesprawności (usterek),
FAULTY - 1. błędny, 2. uszkodzony, zakłócony, niesprawny, 3. wadliwy,
FAULTY OPERATION - działanie wadliwe, działanie błędne,
FAX - patrz: FACSIMILE,
FAY - przylegać, dokładnie pasować,
FC - patrz: FONT-CHANGE CHARACTER,
FCB - patrz: FILE CONTROL BLOCK,
FCFS - patrz: FIRST-COME FIRST-SERVED,
FCP - patrz: FILE CONTROL PROCESSOR,

LIGA MYŚLĄCYCH

ZADANIE 1

Trapez ABCD na poniższym rysunku jest równoramienny. Punkt O jest środkiem boku BC, a punkt S punktem wspólnym prostej AB i prostej OD. Które z tych figur mają równe pola?



ZADANIE 2

Znaleźć najmniejszą liczbę naturalną n>1, dla której suma kwadratu kolejnych liczb naturalnych od 1 do n byłaby kwadratem liczby naturalnej.

ZADANIE 3

Siostra jest 3 lata młodsza od brata. Brat ma obecnie 2 razy tyle lat, ile miała siostra wtedy, kiedy brat miał tyle, ile siostra ma teraz. Ile lat ma siostra, a ile brat?

ZADANIE 4

Jeżeli cyfrę dziesiątek pewnej liczby dwucyfrowej zwiększymy o 4, a jej cyfrę jedności zmniejszymy o 2, to otrzymamy liczbę mniejszą od 86. Jeżeli zaś cyfrę dziesiątek tej liczby zmniejszymy o 2, a cyfrę jedności powiększymy o 1, to otrzymamy liczbę większą od 27. Jaka to liczba?

ZADANIE 5

Pociąg wyjechał ze stacji A do stacji C mijając po drodze stację B. Odległość pomiędzy stacjami A i B pociąg przejechał z pewną stałą prędkością, a odległość pomiędzy stacjami B i C z prędkością mniejszą od niej o 25 proc. W drodze powrotnej od stacji C do B pociąg jechał z prędkością taką samą, z jaką w pierwszą stronę jechał od stacji A do B, a ze stacji B do A z prędkością o 25 proc. mniejszą. Ile godzin jechał pociąg ze stacji A do C, jeżeli wiadomo, że odległość od stacji A do B przejechał w tym samym czasie co odległość od stacji B do C oraz że w kierunku od stacji A do C pociąg jechał o 5/12 godziny krócej niż z powrotem (tj. od C do A)?

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

TEŻ KOMPUTERY...

DOKOŃCZENIE ZE STR. 32

komputer od razu przełoży to na język włoski, francuski, niemiecki lub hiszpański i poda syntetycznym komputerowym głosem. Niezbyt miłym, ale zrozumiałym dla kogoś, kto zna któryś z tych języków.

„Voice" tłumaczy zdania w wybranym z tych pięciu języków. Jego pamięć zawiera ponad 2 tys. prostych zdań stanowiących zazwyczaj podstawowy zasób tzw. rozmówek dla turystów. Niedostatkiem tego zmyślnego urządzenia jest to, iż robi to tylko w jedną stronę. Nie można go natychmiast wykorzystać do przekładania odpowiedzi na język zrozumiały dla jego posiadacza. Za to „Voice" ma płaski ekran, na którym można wyświetlać 16 wierszy tekstu w wybranym języku.

Zdaniem prezesa firmy APT, Steve'a Rondela, natychmiastowe tłumaczenie w obie strony technologicznie jest całkiem realne. Lecz wówczas urządzenie byłoby zbyt masywne, by mogło służyć w wersji przenośnej, a poza tym kosztowałoby zbyt dużo. Obecnie ustalona cena — 1500 dolarów w Stanach Zjednoczonych — nie wydaje się zbyt wygórowana, a czas powinien sprzyjać rozwiązaniu problemów gabarytowych.

Firma Sharp dla nieco innych celów takie problemy już rozwiązała. Zbudowała mikrokomputer PC-1100, przy czym skrót PC nie oznacza Personal Computer, lecz Pocket Computer, czyli komputer kieszonkowy. Łączy on w sobie funkcje minikalkulatora i podręcznej bazy danych. Mimo cech podręcznego notatnika i terminarza PC-1100 jest mniejszy od typowego kalendarzyka, jego wymiary bowiem wynoszą 14x8x1 cm.

Wymienne karty pamięci RAM oraz możliwość programowania w uproszczonym języku Basic otwierają przed tym zminiaturyzowanym urządzeniem duże możliwości zastosowania. Miniekranik zbudowany z wyświetlaczy ciekłokrystalicznych umożliwia wyświetlanie dwóch linii po 16 znaków w każdej. Specjalna funkcja przeszukująca dane umożliwia dostęp do ponad setki zapamiętanych nazwisk i numerów telefonów. Wyszukane nazwisko wyświetlane jest w górnej części ekranu, inne dane w dolnej. PC-1100 umożliwia również zapamiętanie innych informacji takich, jak czas odjazdów pociągów czy terminy spotkań. Do pamięci urządzenia można wprowadzić także wszystkie dane personalne właściciela. Wielostronność zastosowań ułatwiają wymienne karty pamięci RAM w pakietach 2, 4 lub 8 KB.

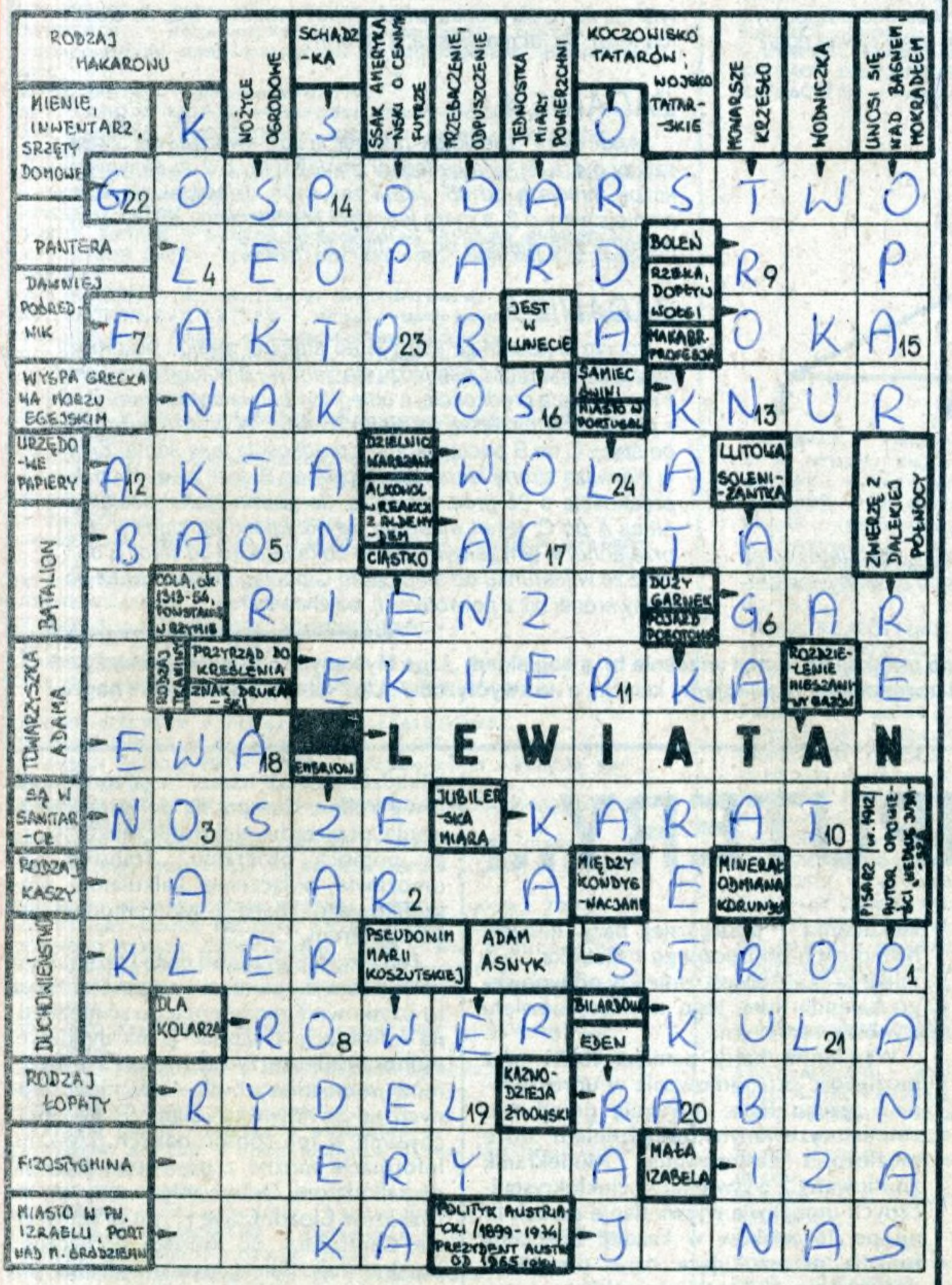
Oprogramowane kalkulatory są partnerem dla dużych komputerów, na przykład Atari lub IBM. W umysłach konstruktorów zrodził się pomysł, by je udoskonalić, wykorzystując do zapisu i edycji programów, dostępnych na kalkulatorze, obszaru pamięci komputerów personalnych. Jeden z nich, Atari ST ma zamkniętą stację dyskietek i ekran monitora mieszczący kilkadziesiąt liczb naraz. Ta zaleta skłoniła firmę Yellow Computing do wyprodukowania przyrządu elektronicznego, który za pomocą programu „Transfile ST" umożliwia połączenie kalkulatora kieszonkowego Sharp z wymienionym już komputerem.

Ten interfejs przenosi dane i programy z kalkulatora do pamięci komputera. Przez to użytkownik może opracować niezbędną informację i zapisać ją na dyskietce. Jednocześnie korzystać można z programów pomocniczych dla Atari, niedostępnych na zwykłym kalkulatorze. Po opracowaniu w ten sposób danych uzyskane informacje można z powrotem przesłać do kalkulatora. Odbywa się to przy użyciu rozkazów Cload, Csave i Print przez wejście Centronics, do którego podłączony jest interfejs. Zaletą tego urządzenia jest ominięcie zawodnej pamięci taśmowej i możliwość przeglądu zawartości wnętrza Sharpa.

Czego to ludzie nie wymyślą. Entuzjasta komputerów z Sydney, Barry Tucker, wpadł na pomysł wydawania gazety bez... jej drukowania. Takie pismo, dotyczące gier liczbowych, sprzedawane jest na dyskietkach. Nosi tytuł „Hazardzista" i przeznaczone jest — na razie — na komputery Amstrad serii PCW. Jego cena wynosi 6,5 dolara australijskiego, czyli prawie tyle samo, co normalne wydawnictwo tego typu. Cena obejmuje również koszt zwrotnej przesyłki. Klienci przysyłają swoje dyskietki, a Tucker kopiuje na nich swój magazyn i odsyła czytelnikom. Liczba „czytelników" rośnie, kolejne „nakłady" mają coraz więcej odbiorców.

J. RAJCH

KRZYŻÓWKA NR 6



Litery z kratek ponumerowanych od 1 do 24 utworzą 4 terminy związane z komputerami, które wystarczy podać jako rozwiązanie zadania przesyłając pod adresem redakcji na kartkach pocztowych w terminie do końca września (naklejając kupon „IKS-a".) Wśród autorów prawidłowych odpowiedzi rozlosujemy bony pieniężne i nagrody książkowe.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

PROLOG FORTRAN PASCAL ALGOL

TEŻ KOMPUTERY

Jerzy RAJCH

Wydaje się, że nie ma końca zastosowaniom komputera. Przed kilkoma laty amerykańska firma Intel wypuściła na rynek pierwszą bitową kostkę pamięci. Dzięki temu skonstruowana została karta zastępująca dowód osobisty, prawo jazdy lub każdy dokument tożsamości. Rzecz polegała na tym, że w obwód scalony sztywnej karty, czyli kostki pamięci, wpisywano wszelkie dane personalne. Plus takiej karty polega na tym, że dane nie dają się zniekształcić ani sfałszować.

Zwiększenie pojemności pamięciowej kostki do 32 bitów — w perspektywie do 64 bitów — doprowadziło do rozszerzenia zastosowań karty. Można już w niej zapisać nie tylko nazwisko, wiek i stan konta bankowego, ale także dane samochodu czy sposób otwarcia szyfrowanego sejfu. Taka utwardzona plastykiem wizytówka — znana w świecie jako „chipcard" — może być prawem do posiadania broni, książeczką zdrowia, legitymacją związkową i klubową, a nawet nie podrabianym kluczem do domu czy samochodu.

Aby wyposażyć obywatela w pamięciowy plastykowy dowód tożsamości, nie wystarczy wyprodukować dla każdego kartę i wprowadzić do niej niezbędne dane. Trzeba jeszcze stworzyć do tego niezbędny system, który zastąpiłby dotychczasowe metody zarządzania i rozliczeń. Robią to już niektóre zachodnie banki, gdzie wszelkie operacje księgowo-rozliczeniowe dokonywane są tylko w pamięci komputerów, a użytkownicy korzystają z nich za pomocą swych kart. Zdaniem futurologów taka karta-wizytówka będzie niebawem do życia niezbędna i posiadać ją będzie dosłownie każdy.

Inne zastosowanie komputera proponuje amerykańska firma Advanced Products and Technologies. Już w tym roku zamierza sprzedawać przenośne urządzenia elektroniczne tłumaczące natychmiast mowę z jednego języka na drugi. „Voice" — bo tak nazywa się ów komputerowy tłumacz osobisty, waży niespełna 2 kg i zasilany jest z baterii. Gdy jego użytkownik powie coś, przykładowo po angielsku,

DOKOŃCZENIE NA STR. 31

„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. 1444 Nr ind. 361682. U-23