



# mikro számítógép magazin

Ara: 30 Ft



AHÁNY NYELVET BESZÉLSZ...

1988/5



# Mi az, ami legjobban foglalkoztat bennünket 1988. évben? az **ADÓ**



Kedvező ajánlat: **DIJMENTESEN** 10 munkanapig kipróbálhatja új termékünket!

**SZEMÉLYI JÖVEDELEMADÓ SZÁMÍTÓ PROGRAMRENDSZER** IBM PC kompatibilis számítógépekre, vállalatok, szövetkezetek és egyéb gazdálkodó szervezetek számára.

- Szolgáltatásai:**
- havi adóelőleg megállapítása dolgozónként (a főállású munkaviszonyból származó jövedelmek és az egyéb munkáltatói adóelőleg elírt szempontok szerinti részletezése és összesítése,
  - éves adóelszámolás elkészítése dolgozónként,

- éves adatszolgáltatás az APEH részére  
Ár: 25 000 Ft (+ ÁFA)

**KÉRJEN BŐVEBB TÁJÉKOZTATÁST, TERMÉKBEMUTATÓ!**  
OKISZ Szervezési és Számítástechnikai Vállalat  
Alkalmazástechnikai Iroda  
Budapest IX., Tűzoltó u. 79. Telefon: 138-659

**TETSZIK? BÉRÉT IS ELSZÁMOLJUK?**  
Modulárisan köréépül **Bérelszámolási és Munkaügyi egységes rendszerünk!**  
3 modul — olcsóbb ár! (220 000 Ft + ÁFA)

**OKISzoft**

Egyéb rendszereinkkel is készséggel állunk rendelkezésükre.





# mikro számítógép magazin

6. ÉVFOLYAM  
1988/5. SZÁM

## A NEUMANN JÁNOS SZÁMÍTÓGÉPTUDOMÁNYI TÁRSASÁG ÉS A KISZ KÖZPONTI BIZOTTSÁG LAPJA

### A kiadvány a Tudományszervezési és Informatikai Intézettel

### együttműködve készül

A szerkesztőbizottság  
vezetője:

Kovács Győző

A szerkesztőség  
munkatársai:

Babos János

(tervezőszerkesztő)

Bakos Tamás

(programozástechnika)

Broczkó Péter

(hírek)

Énekes Ferenc

(KISZ)

Kovács Győző

(levelezés)

Krasznai Éva

(Diákszerkesztőség)

Lindner László

(sakk)

Petróczy Judit

(könyvek)

Pinka György

(NJSZT, alkalmazások)

Simonyi Endre

Szebenszki Sándor

Szulyovszky Csaba

Tamásné Lakó Erika

Terebessy Ákosné

Varga András

(TII, iskola-számítógép)

Vizessy Mária

**Cimképünk:**  
Szentiványi Kálmán

**mikro számítógép  
magazin**



ANANYI NYELVÉZ VEZÉRLŐ

Felelős szerkesztő:  
Könyves Tóth Pál

Szerkesztőség:  
1027 Budapest, Fő u. 68.  
Telefon: 154-250

Levél cím:  
1371 Budapest  
Pf. 433.

Kiadja az Ifjúsági Lap-  
és Könyvkiadó Vállalat

Felelős kiadó:  
dr. Király G. István  
igazgató

Kiadóhivatal:  
1065 Budapest, Révay u. 16.  
Telefon: 116-660

Terjeszti a Magyar Posta  
Előfizethető a hírlapkézbesítő  
hivataloknál  
és a Posta Hírlap-előfizetési  
és Lapellátási Irodáján  
(1900 Budapest XIII.,  
Lehel út 10/A)  
vagy átutalással a 215—96 162  
pénzforgalmi jelzőszámra.

Megjelenik havonta.  
Egy szám ára 30,— Ft  
Előfizetési díj:  
egy évre 360,— Ft  
fél évre 180,— Ft  
Külföldön terjeszti  
a Kultúra,  
1389 Budapest, Pf. 149.  
és a Magyar Média  
1932 Budapest, Pf. 279.  
86—0253



Szakra Lapnyomda  
Budapest (88—0448)  
Felelős vezető:  
CsönDES Zoltán vezérigazgató

INDEX: 25 629  
ISSN 0236-6088

### Tartalom

Értelmiség '88	2
Nyitott rendszer — nyitott egyetem	6
Kis magyar tragédia . . . ?	10
Mi és a számítógép	12
Számítógép és művészet . . .	12
Mit tud a C nyelv?	20
Merre tart a világ?	22
Egy sarrakkal olcsóbb!	26
Közprogramok	27
Rendszerfejlesztési eszközök	28
Mikrokalausz	30
Adok-veszek-cserélek	33
Petike adófékönyvet egyeztet	34
Ki ad magyarázatot?	36
inFORM	37
Olvastunk . . .	38
Új föld	40
Ahány nyelvet beszélsz, annyi ember vagy	45

### ISKOLA-SZÁMÍTÓGÉP

TechnoMIR	3
Az ABC 80 programjának beolvasása HT-be	4
Elsősorban Primósoknak	5
Elektronikus zene C64-en	5

### DIÁKROVAT

Primo-aprórság	7
Hét kicsi kecske	7
TOP-lista	7
Kilencárnyalatú nyomtatás	8

### PROGRAMOZÁSTECHNIKA

BASIC és gépi kód	14
Számítógépes grafika Pascalban	16
SzellemeK forgatása	18

### PROGRAMOK

A billentyűkezelő rutin sajátosságai	32
--------------------------------------	----

### TERMÉKISMERTETŐ

Tele-Script telefonáló és szövegszerkesztő rendszer	42
---	----

### SAKK

Király plusz gyalog a király ellen II.	43
--	----

### AZ OLVASÓ ÍRJA

	44
--	----

### KÖNYVEK

	46
--	----

### HÍREK, ÉRDEKESSÉGEK

	47
--	----

### PONTVADÁSZAT

	48
--	----



*„Legfőbb nemzeti értékünk, amivel felkészülhetünk a nemrég kezdődött időszak történelmi erőpróbájára, a kreativitás gyors emelése. Ebben a vonatkozásban követtük el valószínűleg a leg-hosszabb távon ható hibákat. Az erőforrások szegénységével küzdő rendszer tartalékok felélésére kényszerül, ami azonban a következő periódusban sokkal nagyobb befektetést igényel és sokkal hosszabb időszak alatt helyreállítható károkat okoz. (...) A következő évtizedek adója a jelen szemben a kreatív értelmiségi munka devalválódásából fog fakadni. (Vámos Tibor: Hazánk és a műszaki haladás. Az MTA 1987. évi 141. közgyűlésének központi előadása)*

utazni kellene és mindez a kezdő „felbrutósított” 4-6000 forint fizetésből; nem csoda, ha a pályakezdő fiatal ember a főállásos munkáján után maszkol, még talán az a leg-szerencsésebb megoldás, ha cikkeket vagy könyvet ír, mert a művét csak sok tanulás árán tudja színvonalasan elkészíteni. Arról persze már ne beszéljünk, hogy a honorárium milyen.

Egyesek visszakérdeztek: „Mondd, ha tanulok, akkor mi van? Magasabb lesz a fizetésem, esetleg magasabb beosztásba helyeznek? Te tényleg azt hiszed, hogy bárkit a főnökeim közül érdekel az, hogy én elvégeztem egy továbbképző tanfolyamot és bizonyítványt szereztem?” Valóban, én jól tudom, hogy a jelenlegi merev bértáblázatok mindenre alkalmasak, csak arra nem, hogy a magasabb tudást elismerjék.

Végül a válaszadók igen nagy csoportjának volt az a véleménye, hogy „szeretnék tanulni, de hol? Iratkozzak be egy továbbképző kurzusra, ahol az előadott és vizsgaköteles tárgyaknak már legalább a felét tanultam, most újra tanuljam meg egy másik előadó szájá ez szerint? Szerintem ez nem elpocskélt idő? Én csak néhány tantárgyra vagyok kíváncsi, azt szívesen megtanulnám, de mutass nekem egy intézményt, ahol az általam kiválasztott tantárgyak tanulására lehetőséget adnak!”

Valóban — ellentétben az iparilag fejlett országokkal, de egyre több fejlett országgal is — se nálunk, se a többi szocialista országban nincs bevezetett nyílt tanulási rendszer, vagy ahogy újabbban nevezzük, távtanulás (distance learning), tehát egy olyan állami közoktatási rendszer, amelyik az egyéni tanulás és a vizsgázás, tehát a bizonyítvány-, illetve a képesítésszerzés lehetőségét individuális módon biztosítja. Ebben a rendszerben a tanuló választja ki a tantárgyakat is.

E hasábkokon már nagyon sokszor írtam a távtanulásról. Sajnos ezen a téren vajmi kevés történt az elmúlt időszakban, és ami történt, az is csak az informatikában. Ide sorolom a Gödöllői Agrártudományi Egyetem és az LSI ATSZ hamarosan induló nyílt tagozatát és azt a távtanulási rendszert is, amelyről a megállapodás a cikk írásának idején van születőben a KISZ KB, az AISH, a BME, a KKM, a TTK, valamint az MTV, a Rádió és a SZÁMALK között, az utóbbinak a gesztorálásával. Ebben, a SZÁMALK Távtanulási Központ által szervezett rendszerben a tankönyv mellett a három legfontosabb közvetítő a televízió, a rádió és a számítógépes oktatási program. Az persze köztudott, hogy minden médiát alkalmazó oktatási rendszer rendkívüli módon beruházásigényes, hiszen a jelenlegi árak mellett egy 16 x 20 perces videofilm vagy egy 10-15 órás számítógépes oktatási program millióba kerül. Márpedig a tananyagkészítés költségeit az árak színvonalához, a tanulók által fizetett költségeket pedig a kereseti színvonalhoz kell igazítani, ami önmagában is óriási feszültséget jelent.

A legnagyobb múltú távtanulási intézményben, az 1963-ban alapított British Open University (OU) ma kb. 100 ezer diák tanul évente. Egy-egy diák felvételeként általában két tantárgyat vesz fel, ami azt jelenti, hogy minden félévben 200 ezer tantárgyat után fizetnek díjat a hallgatók. A beiratkozási díjakból az OU évi árbevétele kb. 20 millió font. Az évi költségük sokkal több, a hiányzó cca. háromnegyed részt az állam pótolja. Felmerül a kérdés, hogy érdemes-e Angliának ezt az irdatlan összeget a tömegoktatásra fordítania. Azt mondták — és nemcsak az OU munkatársai —, hogy nemcsak érdemes, hanem kötelező is. Hiszen

— Anglia fejlődését, a világcsonk való versenyképességét csak jól képzett értelmiséggel lehet biztosítani,

— az angol gazdaság egyre gyakoribb foglalkoztatási válságait — lásd bányászati, kohászati, legutóbb a nyomdaipar —, amelyeket nem is kis mértékben a technika fejlődése okoz, csak egy folyamatosan üzemelő, korszerű távtanulási rendszerrel lehet túlélni,

— a gazdasági problémákra csak sokoldalúan művelt emberek képesek hatékony megoldást találni, így a társadalom minden tagja részére — és nem is csak a szakma továbbképzést — az általános műveltség megszerzését és gyarapítását kell biztosítani,

— végül nem elhanyagolható argumentum a távtanulási rendszer mellett, hogy bizonyos szakmákban a hagyományosnak a negyedéért, máshol (például bölcsészet) a tizedéért vagy még annál is kevesebb költséggel tud ez a rendszer diplomás szakembereket előállítani.

Nem árt a távtanulás ellenzőinek az érveit is összegyűjteni, még akkor sem, ha ezek könnyűszerrel cáfolhatók. Az ellenzők ugyanis

— az oktatás dehumanizálását (technológizálását) roppant veszélyesnek tartják. Az évszázados tanár—diák kapcsolat megszűnése — úgymond — a tudás szintje gépi megmérése (tesztvizsga) véleményük szerint egyben a képzés színvonalának katasztrofális csökkenését is jelenti,

— azt mondják, hogy minek annyi tanult ember!? Ha jó a gazdasági vezetés, akkor pontosan meg tudják mondani, hogy mennyi mérnökre, tanárra, orvosra van szükség, a túl sok „okos fej” szakember-inflációhoz vezet; ha mindenki tanul, akkor ki végzi majd a fizikai munkát?

— a távtanulásba sokat kell befektetni, mire el lehet kezdeni a tanulást. Szegények vagyunk, örülünk, hogy élünk, nem kell nekünk ilyen ütemben követni a gazdaságilag fejlett országokat stb.

Talán megengedi a kedves olvasó, hogy ezekre az érveknek, hiszen a felelet annyira triviális, ne válaszoljak. Ha a felet a válasz mégsem kézenfekvő, akkor baj van — sajnos nem Angliában.

Az ellenkezőjét reméli

KOVÁCS GYÖZÖ

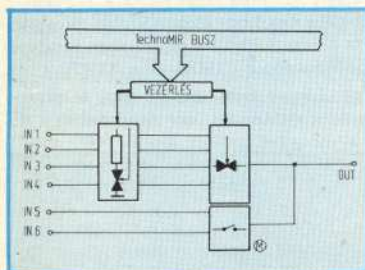


# TechnoMIR

## Analóg modulok

Hogyan sokszorozható meg egy rendszer?

Az analóg sokszorozó (AMUX) modul (1. ábra) alapvető feladata, hogy vezérlő-programtól függően a hat bemeneti pontot (IN1-től IN6-ig) összekösse a kimeneti ponttal (OUT). Ennek a vezérelhető kapcsolófunkciót ellátó modulnak hat bemenete közül négy (IN1—IN4) a CD 4066 CMOS analóg kapcsoló IC-n keresztül



1. ábra

igen gyorsan köthető össze a kimenettel. Az IN5 és IN6 bemenetek reed relén keresztül kapcsolhatók, így ezek lassúbbak, de jobban terhelhetők: max. 25 mA áramot viselnek el károsodás nélkül. A kapcsolható feszültség  $0 V < U < +10 V$  minden bemeneten.

A modul felhasználhatóságát növeli, hogy az IN1—IN4 bemenetek nemcsak aktív érzékelőket képesek fogadni, hanem a programtól függően passzív érzékelők jeleit is el tudják juttatni a kimenetre. Ez a passzív érzékelő 1 mA-es árammal történő táplálásával érhető el.

1. táblázat

Bemenet	d2 bit	d1 bit	d0 bit
IN1	0	0	0
IN2	0	0	1
IN3	0	1	0
IN4	0	1	1
IN5	1	0	0
IN6	1	0	1

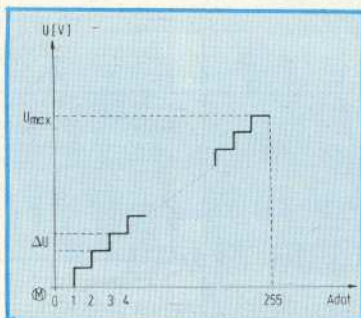
Bemenet Bit

IN4	d4
IN3	d5
IN2	d6
IN1	d7

2. táblázat

A modul a következőképpen programozható. Az AMUX HT címe 25, így HT gépen OUT 25, SZÁM utasítással lehet programozni. A SZÁM értékének megváltoztatásával változtatható a modul állapota. Ez az érték a következő bitenkénti információkat hordozza:

- d0 — d2 bitek határozzák meg, hogy melyik bemenet kapcsolódik a kimenetre (1. táblázat).
- d3 bit „1” állapota az összes bemenet leltiltását eredményezi, azaz egyik bemenet jele sem kerül a kimenetre. A bit „0” állapota az engedélyezés.



2. ábra

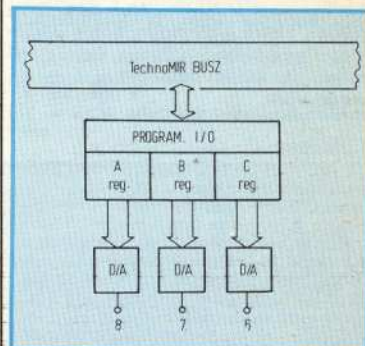
d4 — d7 bitekkel az IN1—IN4 bemenetek táplálását lehet beállítani (2. táblázat). A megfelelő bit „1” értéke táplált, a „0” érték nem táplált bemenetet eredményez. A tápláláskijelölés és a bemenetengedélyezés egymástól független, csak az jelenik meg a kimeneten, amit a d0 — d2 bitek meghatároznak.

Végül egy figyelmeztetés: az AMUX mellett a POW modul is alkalmazni kell!

## D/A modul

A D/A modul olyan programozható digitális-analóg átalakító, amely a számítógépből érkező digitális, 8 bites adatot (0—255 közötti szám) analóg jellé, meghatározott feszültséggé alakítja. Egy D/A konverter transzfer karakterisztikája látható a 2. ábrán.

A TechnoMIR rendszer D/A modulja három, egymástól függetlenül használható átalakítót tartalmaz (3. ábra). Az egyes ki-



3. ábra

Kimenetek:	6.	7.	8.
HT cím	12	13	14
Utasítás	OUT 12, X	OUT 13, X	OUT 14, X
U(ki)	-10V +10V	0V +10V	0V +10V
I(ki)max	50 mA	25 mA	25 mA
Megjegyzés:	0KX<255 egész szám		

3. táblázat

menetek mindaddig az előzőleg beállított feszültséget szolgáltatják, míg a bemeneten található I/O felület megfelelő regisztereibe más adatot nem írunk. Az I/O rendszert a PB modul kapcsán már részleteztük.

Itt is a kontrollregisztert kell először feltölteni. HT címe 15, így HT-n a megfelelő parancs OUT 15,128. Ez C64-en POKE 56848,128. A további tudnivalók a 3. táblázatrol leolvashatók.

KIRÁLY LÁSZLÓ—ALBU LÁSZLÓ



# AZ ABC 80 programjainak beolvasása HT-be

Bizonyára nem egyedül vagyok az országban, aki még az ABC 80-on tanulta a BASIC alapjait, és fiókja mélyén ma is őriz egy olyan szalagot, amelyen erre a gépre írt programok vannak. Programok tehát lennének, ám ABC 80-as számítógép nincs a közelben. Más gépen pedig nem lehet felhasználni ezeket a programokat. Ezen a problémán próbálok most segíteni.

Az ABC 80 kétféle programfajlt tud készíteni: az egyiket a LIST paranccsal lehet felvenni és forrásnyelvi alakban tárolja a programot, ahogyan a billentyűzetről kapta, a másikat pedig a SAVE paranccsal lehet felvenni, és egy tömörített kódot tartalmaz. A módszer, melyet ismertetek, mind a két fajltat beolvassa, de közvetlenül csak a LIST paranccsal felvett lehet listázni. Ez érthető, hiszen a tömörített kódot először még vissza kellene alakítani forrásnyelvi alakra. A közölt számok hexadecimális rendszerűek.

Tekintsük át a program tárolásának módját! Az ABC az adatokat blokkokban veszi fel, melyek felépítése a következő:

- 64 db 00 bájtt
- 3 db 16 bájtt
- 1 db 02 bájtt
- 256 db adatbájtt
- 1 db 03 bájtt
- 2 db ellenőrző összeg, kétbájtos (adatbájttok + 03 bájtt)
- 1 db 00 bájtt

Az adatbájttok felépítése a tartalomnak megfelelően változik. A program legelső blokkjának szerkezete:

- 3 db FF bájtt
- 11 db azonosító (8 bájttév + BAS vagy BAC mutató)
- 242 db 00 bájtt

A program egy általános blokkjának adatbájtt-felépítése:

- 1 db 00 bájtt
- 2 db blokkszám, kétbájttos
- 252 db szöveg
- 1 db 03 bájtt

A program utolsó blokkjának adatbájtt-felépítése:

- 1 db 00 bájtt
- 2 db blokkszám, kétbájttos
- 6 db 00 bájtt
- 247 db, az előző blokkból még a puffertben maradt adat.

Tudni kell még azt is, hogy a szalagon a bitek a nevük szerinti növekvő sorrendben vannak, tehát az első a 0., az utolsó a 7. bit.

Ezek után tekintünk az 1. programot, amely egy bitet olvas be az A regiszter 7. bitjébe úgy, hogy közben a többi bitet jobbra eltolja. Látható a programban két darab időálló: a TIME1 és a TIME2. Ezek

## 1. program

```

BITOLV :PUSH BC      BC regiszterpár megőrzése
        PUSH AF      AF regiszterpár megőrzése
HUR1   :IN  A,(OFFH)  A-ba a port értéke
        RLA          a 7. bit a c-be
        JR  NC,HUR1  várakozás az órajelre, ha c=0
        LD  B,TIME1  B-be az első időálló
HUR2   :DJNZ HUR2    várakozás
        CALL 021EH   a vezérlőkód állítása
        LD  B,TIME2  B-be a második időálló
HUR3   :DJNZ HUR3    várakozás
        IN  A,(OFFH) A-ba a port értéke
        LD  B,A      beolvasott érték megőrzése B-ben
        POP  AF      AF regiszterpár vissza a zsákból
        RL  B        a vett bit c-be
        RRA        a vett bit beforgatása A-ba
        PUSH AF     AF regiszterpár megőrzése
        CALL 021EH   a vezérlőkód állítása
        POP  AF     AF regiszterpár vissza a zsákból
        POP  BC     BC regiszterpár vissza a zsákból
        RET        visszatérés
  
```

```

BYTOLV :LD  B,0BH   B-be a bitek száma
HUR4   :CALL BITOLV egy bit olvasása
        DJNZ HUR4   ugrás a többi bit beolvasására
        RET        visszatérés
  
```

## 2. program

```

FEJDOLV :XOR  A      A regiszter nullázása
HUR5    :CALL BITOLV egy bit olvasása
        CP  16H     A=16H ?
        JR  NZ,HUR5 ugrás, ha nem egyezik meg
HUR6    :CALL BYTOLV egy bájtt olvasása
        CP  16H     A=16H ?
        JR  Z,HUR6  ugrás, ha megegyezik
        CP  02H     A=02H ?
        JR  NZ,HUR5 ugrás, ha nem egyezik meg
        RET        visszatérés
  
```

## 3. program

## 4. program

```

7090 DD 21 00 80 AF CD 12 02 CD 87 70 DD E5 CD 6E 70
7010 CD 6E 70 CD 6E 70 0E FD CD 76 70 CD 6E 70 CD 6E
7020 70 CD 6E 70 CD 6E 70 E1 7E A7 20 0C CD FB 01 21
7030 00 80 CD C9 01 7E 23 CD 3A 03 FE 0D 20 F7 3A 01
7040 3B A7 20 F1 3A 04 3B E6 40 C2 00 43 1B F0 C5 FE
7050 DB FF 17 30 FB 06 2B 10 FE CD 1E 02 06 4B 10 F5
7060 DB FF A7 F1 CB 10 1F F5 CD 1E 02 F1 C1 C9 06 0B
7070 CD 4E 70 10 FB C9 11 00 3C CD 6E 70 12 00 77 00
7080 13 DD 23 00 20 F3 C9 AF CD 4E 70 FE 16 20 F9 CD
7090 6E 70 FE 16 28 F9 FE 02 20 EE C9
  
```



# Elsősorban primösoknak

beállításával érhetjük el, hogy hibátlanul olvassa a HT a szalagot. Többféle időállandóval is kipróbáltam a programot, és kétféle HT-n is. Azt tapasztaltam, hogy az egyik HT-n — az általam is jobb magnójúnak tartottn — mindkét időállandóra egy-egy intervallum adódott, amelyekben belüli tesztoleges értékkel hibátlanul olvasott. A másik HT-n nem lehetett beállítani az időállandókat. Az általam kapott értékek általa: TIME1=40 (Dec) és TIME2=75 (Dec).

Nem akarom elhallgatni azt sem, hogy még a jobbik HT-vel sem tudtam a legelső blokkot elolvasni, azaz a program neve titokban maradt. Ám ha az időállandókat csökkentettem, az első blokkot is elolvastam, kiírta a nevet, de a programot olvashatatlannal vette le. Így először el lehet olvasni a program nevét, majd az időállandókat beállítva elolvashatjuk magát a programot is.

A 2. program egy bájtot olvas az A regiszterbe, az 1. program felhasználásával. A 3. program a szinkronizáció miatt kell: megvárja, míg egy blokk érkezik. Ezzel a három alpprogrammal már bárki írhat egy ABC programszalag-olvasót.

A 4. program 7000 kezdőcímrre van szerelve. A program feltételezi, hogy az EDI is a memóriában van. Használata a következő. 7000 címről indítva bekapcsolja a belső magnót és olvassa a szalagot. Ha talál ABC programot, akkor a 8000 címűtől kezdve tárolja, és a képernyő első negyedében is látható az, amit a memóriában letárol. Ha úgy látjuk, hogy értelmetlenség, akkor módosítsunk az időállandókon. Olvasás közben a program figyel, hogy az utolsó blokkot olvasta-e be. Ha igen, kikapcsolja a magnót, letölti a képernyőt és listázza kezd. Az A, B, C, D, E, F, G, H billentyűk bármelyikének lenyomására listáz egy sort, majd újra megnézi a billentyűzetet. A V billentyű hatására az EDI-be ugrik, innen akár újra is lehet indítani. Segítségül néhány cím:

7000 a program indítása

702F a listázás megkezdése

704A az EDI címre, ezt átírva a V betűre nem az EDI-be megy

704E egy bitet olvas

706E egy bájtot olvas

7076 egy blokk adatbájtjait olvassa

7087 megvárja a blokk elejét

## SZOLDATTI JÓZSEF

A rovszerkesztő megjegyzése. A Tudományos Szervezési és Informatikai Intézetnek van egy olyan illetője, amelynek segítségével ABC 80-ról HT 1080Z-be át lehet vinni BASIC programállományokat. Az intézet kívánságra szívesen végez másolat szabványos HT kazettára. Természetesen az előző BASIC miatt itt is szükség van némi utólagos programmódosításra.

A matematikafakultáció keretében sokan tanulják az integrálszámítást. Házi feladataik készítését megkönnyítheti az 1. lista látható program.

A program működése során megkérdezi, hogy a 70-es sorban megadott F függvényt mely intervallumon integrálja. Ezt követi a számolás alsó és felső közelítéssel. Amikor a két közelítés eltérése kisebb egy adott értéknél, akkor a gép befejezteknek tekinti a számolást és kiírja az így kapott végeredményt.

A programban a következő változókat használta:

— A és B: az intervallum alsó, illetve felső határa

— N: az adott intervallumban a részintervallumok száma (a számolás folyamán tart a végtelehez)

— H: egy részintervallum nagysága

```
10 REM * INTEGRÁLSZÁMÍTÁS *
20 CLS
30 INPUT "AZ INTERVALLUM HATÁRAI":A,B
40 N=1:E=6E-3
50 H=(B-A)/N
60 FOR X=A TO B STEP H
70 TA=TA+H*F(X):TF=TF+H*F(X)+H
80 NEXT X
90 TA=INT(TA*1000)/1000
100 TF=INT(TF*1000)/1000
110 IF ABS(TF-TA)<=E THEN 130
120 TF=0:TA=0:N=N+1:GOTO 50
130 PRINT "A GÖRBE ALATTI TERÜLET:";
140 PRINT (TF+TA)/2*"EGYSÉG"
150 END
```

### 1. lista

```
10 REM * GYORS KÖR SZÁMÍTÁS *
20 CLS
30 INPUT "A KÖR SUGARÁRA":R
40 INPUT "A KÖR KÖZÉPPONTJA":OA,OB
50 K=3.141592653589793
60 FOR I=1 TO 629 STEP K
70 X=A(I)+R*OA:Y=B(I)+R*OB
80 SET(X,Y):NEXT I
90 END
```

### 2. lista

```
10 REM * MAX. KÖR KOORDINÁTÁI *
20 DIM A(630),B(630)
30 FOR I=0 TO 2*PI STEP 0,01
40 J=I+1
50 A(J)=COS(I)*R5:B(J)=SIN(I)*R5
60 NEXT I
```

### 3. lista

```
10 S=54272:FOR I=0 TO 24:POKES+I,0:NEXT
20 POKES+24,143:POKES+17,8:POKES+6,250
30 FOR I=1 TO 3:PRINT "TETEL":POKES+15,T-1
40 FOR Q=22 TO 56 STEP 2:POKES+4,17
50 POKES+18,16:POKES+14,2:4*Q:Q=3*Q
60 Q=Q+FEEL(Q/27)*3,5*Q:Q=INT(Q/256)
70 R=Q-F*256:POKES,R:POKES+1,F
80 IFEEK(203)GOTO 4 THEN I=0
90 GOTO 50
100 GETTF:IFTB=""THEN I=0
110 POKES+4,16:NEXT NEXT
```

Ez a rövid program egyszerűsége ellenére sokat tud. Futtatása során bárki izléseinek megfelelő elektronikus zenét komponálhat.

A mű háromtételre lesz (30-as sor), és nagyjából követi a szimfóniák tempóviszo-

— E: a két közelítés eltéréseinek értéke (minél kisebb, annál pontosabb a számolás)

— TA, TF: az alsó, illetve felső közelítés. Mint már említettem, az adott függvényt a 70-es sorba kell beírni az F(x), illetve F(x+H) helyére, tehát a független változót kell H-val növelni.

A 2. lista egy gyors körrajzoló program. Számos BASIC programban lehet szükség kör rajzolására, ezért jól szolgálhat egy gyors változat. A módszer jelentősen csökkenti a körrajzolásra fordított időt, ezért olyankor célszerű alkalmazni, amikor szerepe van a futási időnek. Hátránya azonban, hogy több helyet foglal el a memóriában, mint a hagyományos körrajzoló program.

Ez a program a következők alapján működik:

— Az A(I), B(I) változóknak tárolja a lehető legnagyobb kör koordinátáit, és a későbbiekben ezeknek a segítségével — egy százzal és egy összeadással, ami jóval gyorsabb a szögfüggvényes számolásánál — kiszámolja az aktuális kör koordinátáit, és kirajzolja a képernyőre.

A program változóit a következők:

— A(I), B(I): a maximális nagyságú kör koordinátái

— Z: az adott kör sugarának századrésze

— OA, OB: az adott kör középpontja (X és Y koordinátája)

— K: megadja, hogy az adott kör sugara hányszor fér bele a legnagyobb kör sugarába. (Figyeljük meg, minél kisebb az adott kör sugara, annál gyorsabban rajzolja ki. Ezt a K változó teszi lehetővé. Mivel a sugár csökkenésével egyre kevesebb pontból áll a kör, ezért ezt a változót használtam arra, hogy csak a szükséges pontok legyenek kirajzolva.)

A program elindítása előtt természetesen az A(I) és B(I) változókat fel kell tölteni az ismert módszerrel. Ha valaki mégsem ismerné, megtalálhatja a 3. listán.

A programokról még annyit, hogy Primo számítógépen készültek. Az integrálszámítás minden további nélkül átirható bármely más gépre, a körrajzoló azonban nem biztos, hogy mindenütt ugyanolyan hatással jár.

NOGRÁDI ELŐD

## Elektronikus zene C64-en

Ez a rövid program egyszerűsége ellenére sokat tud. Futtatása során bárki izléseinek megfelelő elektronikus zenét komponálhat.

A mű háromtételre lesz (30-as sor), és nagyjából követi a szimfóniák tempóviszo-

nyait. A zenedarab úgy jön létre, hogy ujjunkkal röviden megérintünk egy billentyűt. Ezt a zenészek staccatónak nevezik. Az egyes billentések között tetszésünkre megfelelően 2-4 másodperc várjunk. Aki másféle hangzást is szeretne hallani, annak közlök néhány módosítási lehetőséget.

A zene hangfekvést és a 60-as sorban a G szorzószám határozza meg. Ez 0,5-4 lehet. A hangzás tempóját az 50-es sor POKES+14 értéke befolyásolja: (1-5) \* G-ig változhat. A tételnek tempója a 40-es sor ciklusának kezdő- és végértéke határozza meg: 18-60-ig terjedhet. A felsorolt értékek kölcsönösen befolyásolják egymást.

KALMÁR GYULA



# Nyitott rendszer -nyitott egyetem

**Apró kis lámpákkal jelölve az ország középiskoláit és szakmunkásképző intézeteit, érzékelhetővé tehetjük egymástól való távolságukat. Meglehet, ránézve e különleges fénytérképre, gondolatban a távolságok oly csekélyek, szinte semmik, lebecsülhetők. De ott lenni, tanulni és tanítani egy-egy iskolában és onnan megtudni valamit is a másiktól, a másiktól, nem könnyű. Pedig ennek létfontossága vitálisan eltagadhatatlan. Egy iskola sem próbálkozhat telepátival — gondolatvitával —, nem is annak tudományos vitathatósága miatt. Egyszerűen nincs idő arra, hogy véletlenszerű kísérletezéssel próbálkozzanak, amikor egy-egy központi, vagy saját információs éleskamrájuk tele van, vagy éppenséggel üresen tátong. Persze akármelyik iskolában mondhatják: nagyon jól megvannak, sem mire sincs szükségük. Csak ez tipikus olyan, hogy ha nem tud, nincs ismerete valmiről az embernek, talán hiányt sem szenved tőle. Egy-egy információ azután így kallódik el, így nem válik közkinccsé, és gyakran olyan balvégzetű lehet, mint az a egyszerű találmány, amely sutba dobva éri meg végnapját — meghaladja a tudomány, a technika —, s mire felfedeznék, legfeljebb már múzeumi tárggyá avul.**

Kézenfekvő tehát, hogy ezeket az iskolákat érdemes és össze kell kapcsolni valamilyen módon egymással, s olyan intézményekkel is, melyek műhelytanulmányaikhoz sok segítséget nyújthatnak. A KISZ KB Középsikolai és Szakmunkás-tanuló Tanácsa a közelmúltban ennek mintegy alapkövét tette le, amikor létrehozott egy adatbázis információs rendszert. Erről a rendszerről Kajos Lászlóval, a tanács titkárságvezetőjével beszélgettünk.

— Minden középfokú oktatási intézményben van már számítógép. Mi erre

épitve kívánjuk hozzáférhetővé tenni azokat az információkat, amelyek az iskola, a KISZ-élet mindennapjaiban hasznosíthatók, a pedagógusokat tájékoztatják, s a diákok is ennek révén szinte beszélgethetnek egymással az ország legtávolabbi pontjaiban e rendszeren keresztül. Vagyis, az információs adatbázisból leihathatják az őket érdeklő témákat. Adatbázisunk jelenleg négy információcsoportból áll. Tartalmazza az ország valamennyi középfokú iskolájának pontos adatait; az iskolai innovációk leírását a fejlesztők megnevezésével, továbbá ezen fejlesztések helyi adottságokra, viszonyokra adaptálható eredményeinek pontos leírását. Ezen kívül magában foglalja a középfokú oktatási intézmények működésére vonatkozó jogszabályokat, melyeket sajnálatos módon a pedagógusok többsége nem, vagy csak hiányosan ismer. A negyedik csoportba a KISZ-szervezetek működéséről szóló információkat gyűjtöttük össze. Így az érvényes központi, illetve megyei testületi döntéseket, valamint a politikai képviselői tapasztalatait dolgoztuk fel.

— Kik vesznek részt az információs adatbázis megalkotásában?

— Munkánkban közreműködik a Művelődési Minisztérium és az Országos Pedagógiai Intézet. A közös adatállomány szándékunk szerint a legkülönfélébb innovációs fejlesztésekkel — ezek az iskolából érkeznek be hozzánk — megalapoz egy műszaki fejlesztést szolgáló adatbázist. Éppen azért, hogy bármely ilyen jellegű műszaki gondolat, terv az érdeklődő diák birtokába juthat.

— Mi az adatbázis technikai igénye?

— Nálunk üzemel már egy IBM kompatibilis gép. A közeljövőben valamennyi megyeszékhely KISZ-bizottságára ugyanilyen típusú mikroszámítógépet kívánunk elhelyezni azért, hogy a mi központi adatbázisunkat tetszés szerint egy az egyben leihathassák. Persze ezzel még az iskolába nem juthatna el egyetlenegy információ sem. Az akusztikus modemekkel viszont direkt kapcsolat kiépítése nélkül, telefonon lesz elérhető a megyeközpontok KISZ-bizottságain felhalmozott információ-adathalmaz. Természetesen minden olyan iskolai számítógéphez használható a modem, amelynek RS-232-es kimenete van. Az olyan számítógépek, amelyeknek

ilyen kimenete nincs, csak megfelelő in-terfészen keresztül csatlakoztathatók a modemhez.

— Honnan szerezhető be egy modem? Azaz miképpen kapcsolódhatnak be ebbe a rendszerbe?

— Az akusztikus modemet a KISZ KB-tól vásárolhatják meg 19 ezer 500 forintért. A készülék iránt a 203-872-es telefonon lehet érdeklődni. Egyébként, ha máshonnan szerzik be a kis készüléket, az ára jóval borsosabb. A rendszerbe kapcsolódáskor csak a modem jelent költséget, ugyanis az információ leihívása ingyenes. Viszont szinte alapszabállyá szeretnénk tenni azt, hogy az iskolák ne csak információit vegyenek, hanem adjanak is magukról, hiszen ezáltal fejlődhet, gazdagodhat csak adatbázisrendszerünk.

— Milyen időszakonként újul meg a rendszerben lévő ismeretsokaság?

— Folyamatosan gondozzuk az információkat, így a hozzánk beérkező újdonságok gyakoriságától függően fél-évente, havonta, hetente frissítjük fel az állományt.

— Jóllehet az országos hálózat alakulóban van, gondoltak-e már a továbbfejlesztésére?

— A kialakított párhuzamosan működne egy, a postával üzemeltetett videotext rendszer Mupid berendezésekkel, illetve Mupid funkciókra alkalmassá tett IBM kompatibilis gépekkel. Ezen keresztül az Ipari Minisztérium, a Mezőgazdasági és Élelmiszeripari Minisztérium, az Országos Műszaki Fejlesztési Bizottság által fejlesztett adatbázisok is elérhetőek. Ezzel egy olyan oda-vissza áramló információs csatornahálózat születik, amely lehetővé teszi a régóta várt távoktatást. A Magyar Televízió, a Központi Statisztikai Hivatal és a Neumann János Számítógéptudományi Társaság egymással karöltve készíti elő a nyitott egyetem módozatait is. Ezek izgalmas, érdekes kérdések lesznek a jövőben.

— A videotext rendszerbe való bekapcsolódás is ingyenes szolgáltatásokkal jár?

— Nem. Az imént felsorolt intézmények adataiért, leírásaiért, tájékoztatásaiért fizetni kell majd, de ennek árát csak később állapítják meg.

— krasznai —



# PRIMO-apróság

Különböző „gyors lábú” programok irásakor elengedhetetlen egy olyan GOTO/GOSUB utasítás, amely nem kötött sorszámra vagy számlistára vonatkozik, hanem egy megadható aritmetikai kifejezés szolgáltatja célsorszámra ugrik. Az alábbi kis programocská erre kínál igen egyszerű és általánosan használható megoldást.

Az egyetlen kikötés, hogy ha a célsorszám 32767-nél nagyobb, akkor 65536-ot kell levonni belőle. A lista alábbi formájában 64 k-s gépeken működik, azonban az első POKE utasítás címét (-6640) kisebbre választva, és kétbájtos alakját a második POKE utasítás adataiként beírva a többi (48 k-s és 32 k-s) gépen is használhatjuk. A program szempontjából az „A” vagy „B” sorozatú gépek egyenértékűek. Működése igen egyszerű, voltaképpen azt teszi, amit egy „jölnevelt” GOTO/GOSUB feldolgozó program sorszámbelolvasó része tesz:

— meghívja a 2337H-n kezdődő kifejezéskiértékelő rutint, amely a HL által mutatott címen kezdődő tokenizált kifejezést dolgozza fel (PRIMO irodalomban

EXPRN néven ismert), majd az eredményt az „X” munkaregiszterbe helyezi el;

— egész részt számol, ezt követően az eredményt mint sorszámot veszi, és végrehajtja az ugrást. (A GOSUB rész természetesen előbb a verembe helyezi a következő utasítás címét.)

A lista mellé ime még egy csemege. Ha a GOTO vagy a GOSUB utasításokat sorszám nélkül adjuk ki, a gép általában UL Error-t jelez, kivéve, ha van 0 sorszámú sor. Ez esetben ugyanis a program elindul, vagyis ilyenkor a 0-t célsorszámként értelmezi.

1 POKE—6640,14,3,205,99,25,193,229,229,42,162,64,227,62,145,245,51,197,229,205,55,35,205,127,10,229,209,225,195,200,30

2 REM — Itt irtuk be a programot —

3 POKE 16561,10,230: CLEAR 50

4 REM — A BASIC-rendszer számára fenntartott memória végének lejjebb hozása. Erre programunk védelmében van szükség, különben igen könnyen „elpárolaghat”.

(Lásd veremkezelés és egyéb érdekességek.)

5 POKE 16768,33,230,195,16,230

6 REM — A RAM ugrótábla részében a GET és PUT utasítások átírása —

7 REM — A továbbiakban GOTO x helyett GET numerikus kifejezés használható —

8 REM — GOSUB x helyett PUT numerikus kifejezés alkalmazható —

9 REM — A régi GOTO/GOSUB utasítások természetesen „érvényben maradnak” —

10 REM — CDOS és rokonai használatánál csak CDOS betöltése után, és 11 REM ne az általa foglalt területre írjuk be. Ez kiolvasható

12 REM a 116561-es címen található kétbájtos mutató értékéből —

13 REM — Más címre helyezéskor ne csak a memória végét jelző mutatót,

14 REM hanem a végrehajtási kezdőcímet is állítsuk be helyesen. —

15 PRINT chr\$(12) „Speci GOTO/GOSUB él és élni akar kikapcsolásig”

16 END — liling —

## Hét kicsi kecske

A diákszerkesztőség összejöveteleit színesítő, felnőtt kollégáinktól a következő rejtvényt kaptuk azzal, hogy próbáljuk számítógépesen gondolkodva megoldani. Íme:

Éldegélt valahol hét kicsi kecske. Egyszer kaptak három liter tejet, melyet valaki valahogyan (de semmiképpen sem egyenlően) szétosztott közöttük. De ők jószívűek voltak, és igazságosan akarták maguk kö-

zött elosztani a kedvelt italt. Elégynen szólt az első: — Nem szeretem a tejet — persze csak az önzetlenség beszélt belőle —, szétosztom hát köztetek. S a szót tett követte. Ezután hasonló állítással jelenkezett a második kis kecske is, ő úgyszintén szétosztotta a tejet. Követte őt sorban a harmadik, a negyedik, az ötödik, a hatodik, végül pedig a hetedik is. Poharaikban ekkor mindgyi-

küknek ugyanannyi teje volt, mint kezdetben.

A kérdés: melyiknek mennyi?

A feladatot természetesen meg lehet oldani matematikai módszerekkel is. Ez is érdekes lehet, minket azonban az a megoldás érdekel, amelyhez számítógépre van szükség. Ennek programját várjuk tőletek, a hozzá tartozó magyarázattal együtt.

Diákszerkesztőség

## TOP-lista

A külföldi szaklapok példáiból okulva mi is közlünk TOP-listát, két kategóriában. Az elsőt a felhasználói, a másodikban a játékprogramok szerepelnek. A legjobbakat, a legérdekesebbeket soroljuk a beküldött javaslatok alapján. Ehhez kérjük az olvasók közreműködését. C64-re, ZX-Spectrumra, Enterprise-ra, TVC-re, AT-ARI-ra és IBM-re készült programok rangsorait várjuk havonta.

Címünk: 1371 Bp., Pf.: 433 Mikroszámítógépek Magazin Szerkesztősége.

A Diákszerkesztőség

Felhasználói programok:

- |                    |                 |
|--------------------|-----------------|
| 1. Gigacat         | C64             |
| 2. Art Studio 1.2B | C64+ZX—Spectrum |

- |                                  |             |
|----------------------------------|-------------|
| 3. ZX—Screen machine             | ZX—Spectrum |
| 4. Speech+                       | C Plus/4    |
| 5. Easy Script                   | C64         |
| 6. Paint Box                     | ZX—Spectrum |
| 7. History of Europe and Hungary | C64         |
| 8. Hires—Master                  | C64         |
| 9. Window Wizard                 | C64         |
| 10. Beta Basic                   | ZX—Spectrum |

Játékprogramok:

- |                            |     |
|----------------------------|-----|
| 1. Defender of Crown       | C64 |
| 2. The last ninja          | C64 |
| 3. Alternative world games | C64 |

- |                       |                 |
|-----------------------|-----------------|
| 4. Sabotour           | C64+ZX—Spectrum |
| 5. Zaxxon             | C64+ZX—Spectrum |
| 6. Impossible Mission | C64             |
| 7. Game Over/trainer  | C64+ZX—Spectrum |
| 8. Commando           | C64+ZX—Spectrum |
| 9. California Games   | C64             |
| 10. Quest for Tires   | C64+ZX—Spectrum |

E havi TOP-listánkat a fővárosi Hámán Kató Közgazdasági Szakközépiskola diákjainak szavazatai alapján állítottuk össze.



# Kilencárnyalatú nyomtatás

A program színes, nagyfelbontású (160×200) képeket készít az MPS 1000-es nyomtatón. Viszonylag kis módosítással bármely olyan nyomtatóra átírható, amelyek elérhető a soros buszon, képes duplasűrű nyomtatásra és van 8 tús grafikus üzemmódja.

Minden képernyőpontnak 4×4 pixel felel meg a nyomtatón, így a teljes kép 640×800 pixel, ami mintegy 22,5×17 centiméter. (A duplasűrű nyomtatás miatt a 800 pixel feleakkora helyen fér el.)

Mivel egy képernyőpont 16 pixel a papíron, elvileg 17 különböző árnyalat nyomtatható. A gyakorlatban azonban a sötétebb árnyalatok között túl kicsi a különbség, ezért a 16 szín 9 árnyalatban jelenik meg:

- |                      |   |
|----------------------|---|
| 1. fehér             | 0 |
| 2. sárga-világoszöld | 1 |

- |                           |    |
|---------------------------|----|
| 3. cián-világosszürke     | 2  |
| 4. zöld-világospiros      | 3  |
| 5. világoskék-középszürke | 4  |
| 6. lila-narancs           | 6  |
| 7. piros-sötétszürke      | 9  |
| 8. kék-barna              | 12 |
| 9. fekete                 | 16 |
- pixel sötét a 16-ból. Így pontosan olyan képet kapunk, mintha fekete-fehér tv-n néznénk a rajzot.

A grafika a következőképpen helyezkedik el a tárban:

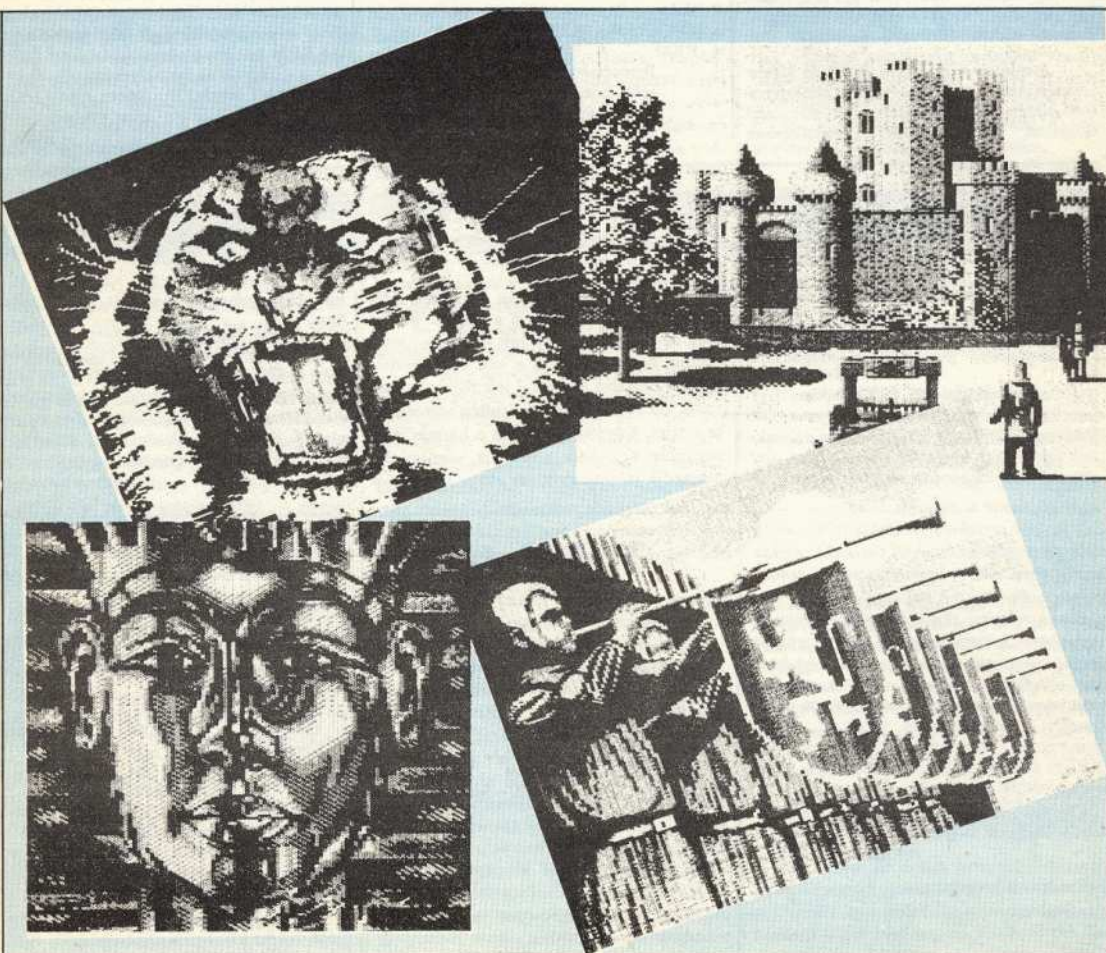
\$2000-\$3F3F: ponttároló

\$3F40-\$4327: szín-RAM # 1  
 \$4329: háttérszín (SD021)  
 \$4338-\$471F: szín-RAM # 2  
 (\$D800-\$DBE7)

Ezt használja egyébként az ART STUDIO 1.2 rajzolóprogram is.

Egyetlen kép kinyomtatása mintegy 5 percig tart.

DEIERL ANDRÁS—  
 MOLNÁR IMRE









# Árvábbnál is árvább Kis magyar

Egy kisfiú ül nyilvánvaló elégedettséggel egy számítógép előtt, felénk fordított mutató- és középső ujjával jelet formál: *Victoria!* — Ez látható a hajdan gyártott Primo ízléses reklámfüzetének címlapján.

Győzelem? Vagy bukás? Krúdy Gyula *Almoskönyve* szerint az ujjakkal játszani fiatalnak szomorúság. Ha a Primót, a fiatalon hamvába holt gépet mint vállalkozást nézzük, a győzelem valóban csak álom volt, s a valóság szomorú. És mégis, a Primo nem *ad acta* tény, a Primo sokszerzeresen van és él. Iskolai és házi számítógépként helyet követel magának. Eppen ezért e cikk sem *rekvium*, hanem *látélet*, *körkép*, *áttekintés* a Primo továbbéléséről és *némiképp* *viszszatekintő diagnózis* félsikerültségének körülményeiről.

## Anamnézis

— Hány Primo készült összesen? — kérdezem az MTA—SZTAKI Cosy leányvállalatnál (amely menedzselt és forgalmazta ezeket a gépeket) *Pogrányi Károlyt*, aki már a megszületésénél is bábáskodott.

— Összesen hétezer. Ebből ezer nyomógombos billentyűzettel. Ezek mindegyike fekete-fehér televízió használatát tette lehetővé. Ezután készült el a Pro/Primo, színese.

— Ami nyert az iskola-számítógépes pályázaton, de végül nem lett belőle semmi. Megrendelésük egyáltalán nem volt az iskolák részéről?

— Tudomásom szerint az iskolák rendelték két-két és fél ezret, de amikor a Commodore is szóba jött, azonnal átálltak arra. Hadd tegyem hozzá, hogy mi *nem eleve kudarcra rendezkedtünk be*, hiszen már a Pro/Primo idején továbbfejlesztettük ezt a családot. Elkészítettünk egy 256 kb-ós változatot is, ami azóta is nagyon jól működik nálunk.

— Arra számítanak, hogy jobb körülmények között még egyszer induljanak ebben a számítógépes kategóriában?

— Nekünk elsődlegesnek kell tekintenünk a gazdaságossági követelményeket, ezért kizártnak tartom ezt. Másrészt szerintem a közelebbi-távolabbi jövőben az iskolákban is az IBM PC-ké a jövő, mert ezek a gépek intelligens párbeszédre való képességükkel *alkalmasabbak* a kisebbeknek és a kamazsodóknak a *számítógépes társalgásra, tanulásra*.

— Volt azonban néhány kellemetlen tapasztalatunk is a primósoknak, ami talán befolyásolta a gép megítélését.

— Ez tagadhatatlan. A perifériák nem jelentek meg az alapgéppel egy időben. Sajnos a programkészletünk is akkor bővült gyorsabban, amikor már a *Primo ká-*



*tyúba esett*. Végül tizenöt programot forgalmaztunk, a többi kiadatlan maradt.

— S ha valaki ma állít be azzal, hogy itt a programom, hasznosítani szeretném?

— Ha behozza a törzsanyagot, ad hozzá programleírást és valóban jó az a program, segítünk neki ebben.

— Vállalkozó nem jelentkezett a Primo gyártására, mióta leálltak vele?

— De, egy gmk szeretne volna gyártani, csak nem tudta megvásárolni ennek a jogát. Egyébként a gép kapcsolási rajza megjelent az egyik primós könyvben, kis ügyességgel bárki felépítheti.

## Látod? ...! Lelet! ...?

Az út a Cosyból nem Rómába vezet, hanem Budapesten a Rákóczi úti Triál hangszereboltba. Egedül itt lehet — ha lehet — állami üzletben új Primo programokat venni.

— Nekünk ez egyáltalán *nem üzlet*. A múlt évben is alig volt néhány ezer forintos forgalmunk programeladásból — mondja *Szili Mihály* üzletvezető. — De nem is tartozik a profilunkba. Annak idején *félig-meddig felkérésre, szíveségből váltaltuk az árusítást*.

— Reklámozták a lehetőséget?

— Nálunk nem. Ha a Cosyhoz ment, ide küldtél, vagy ha hozzánk betévedt valaki, vásárolhatott. Bár ehhez nekünk mint semmi *érdeklünk* nem fűződik, olyankor *már hátunkon a púp*.

## Tájkép csata után

Kevés program, reklám semmi, és az országban egy üzlet. A pesti szleng erre azt mondja: ez nem semmi! A *képtelenség* még *tetézi*, hogy a Primo életes életében sem volt jobb a helyzet. Mert ha például Szombathelyen vett valaki gépet, akkor is Pestre kellett utaznia programért!!! Gép és programok. Az előbbieket mintha a föld nyelte volna el. Vajon többet megtudok-e rólok a Tudományos- és Informatikai Intézetben?

— 1986-ban az általános és középiskolákban 1263 Primo volt. A kiadásitáskor még két-háromszázat vettek az iskolák — néz a statisztikába *dr. Varga András* főosztályvezető.

— Közelebbit tudnak ezeknek a gépeknek a használatáról?

— Tulajdonképpen alig-alig. A gépeket a *megyék osztják el*, a továbbiakban *visszajelzést nem kapunk*. A programkészítési pályázatunkra is csak két-három pályamunka érkezett.

## Confused view

A hegy csúcsáról nézve tehát a Primók világa holtágnak tűnik. Vannak, csak a *mozgásuk nem látszik*. Az Országos Pedagógiai Intézetben *dr. Szűcs Barna* sem látja a mélyben a mozgást.

— Sajnos tudomásom szerint nálunk sem vizsgálták, kutatták az iskolai Primók helyzetét. Alkalmazásukkal sem foglalkozott senki. (Ez sem semmi! — A pesti szleng.) Egy-egy iskolai jelentésből kiderül: nálunk van Primo, hány van. Néhány megjegyzést fűznek csak hozzá. Én a primós dolgot amolyan „kis magyar tragédiának” tartom. Mert ami azt illeti, *jól példázza* a hardveresek és szoftveresek külön világát, az *összehangoltságuk hiányát*. Ugyanakkor az is érthető, nálunk szinte törvényszerűen *miért bukik bele* a vállalkozásába az, aki jót és jó szándékkal kezdeményez. Bár a Cosy is többet tehetett volna a sikerért. *Nekem levelezésem van róla*, hogy kezdeményeztük egy bemutató laboratórium Primóval való felszerelését, de ebben *ők kicsinyesek voltak*, nem adtak felszerelést.

## Elfogyott a hamuban sült pogácsa

Néhány bizonytalan tippel a tarsolyomban, mondván: talán abban az iskolában



# a PRIMO tragédia...?

találni Primókat — de inkább hallomás alapján kerestem fel a Dob Utcai Általános Iskolát. Hátha ott igazi primós életre lelek! Amit találtam (az általánosítás nem szándékom, de valószínűsíthető), nem a legrózsásabb. A számítógépes teremben az asztalokon Commodore-ok és dobozban a magukra hagyott Primók. Dr. Appel Györgyné matematika—technika szakos tanárnő a számítógépes szakkör vezetője:

— Én nem a Primókat tanítom a gyerekeket, de tudom, hogy a Neumann Társaságtól hetente jön egy hölgy, aki az alsóoklak foglalkozik. Neki, de nekem is az a legnagyobb gondom, hogy a gépeket csatlakozóvezeték nélkül kaptuk. Így igen nehéz boldogulni. A Primókat egyébként egyszerű használták másra, a gyerekek történelemtudásának felévi értékelésekor. De nem is lehet egyébre, mert csak egy programjuk van.

## Haláltánc

Ezt hívják úgy, hogy az információ halála? Vagy magárahagyatottság? Netán a magad ura, magad szolgálja vagy jelenségek? Ilyen árva bálnánál is árva bálna a Primo? Ez lehetetlen. Még akkor is, ha minden jel arra mutat. Vagy van más jel is talán, mint a hiányzó, hiányos információ? Megelőlegezem — a Gelknánál rábukának a várva várta. Nyíró Árpád, a Fehérhajú utcai szerviz vezetőhelyettese konkrétumokkal szolgál.

— Mit tudnak nyújtani a Primo-tulajdonosoknak, lehetséges talán, hogy a gépeket akár továbbfejlesszék?

— Volt eredetileg az A és B típusú gép. Az A-nak billentyűzete szenzoros és nem volt semmilyen plusz kivezetése, a B változatnak viszont kiépítették a hátsó csatlakozását, és billentyűs. Ezt bővítettük egy úgynevezett C változattal, ami ezeken kívül azt a lehetőséget nyújtja, hogy Commodore perifériákkal működik, a kazettára beolvasott program nélkül. Ezenkívül numerikus billentyűzetet adhatunk hozzá, továbbá bármilyen RAM-bővítést megcsinálunk, illesztünk, de ezek már egyedi munkák.

— Ezek szerint ha valaki bejön valamiféle egyedi kívánsággal, akkor segítenek rajta?

— Igen. Eddig a leginkább keresett perze a C változat volt, s amit még nem említettem, botkormányát is két változatban illesztünk. Az egyik külső doboz, a másik géphez épített. A két csatlakozót is felszereljük.

— Összesen hétézer gép van, évente hányan jönnek akár mert meghibásodott, akár hogy bővítették a masinájukat?

— Az adataink nem egészen pontosak, mert hosszú időn keresztül egyedül mi javítottuk ezeket a gépeket. Jelenleg hozzávetőlegesen 40-50 Primót javítunk. Tavaly év végén 30-40 C változatot csináltunk.

— Az alkatrészellátás zökkenőmentes?  
— Eddig igen. Mostantól valószínűleg lesz a szenzoros billentyűzetű gépekkel gondunk.

— A Cosy nem látja el a szervizt fóliával?

— Itt a baj. A fóliátaszatúra fogyóban van, és nem biztos, hogy lehet még kapni. A Cosy nem nagyon foglalkozik ezzel, ők elküldenek a gyártóhoz, Szolnokra. Más különben annak idején a Primónak az volt az előnye, hogy csak olyan alkatrészekből építették, amelyek Magyarországon beszerezhetők voltak. Ez a helyzet ma is, mert minden alkatrész szocialista gyártmány. Úgy tűnik, még elég hosszú ideig lesznek üres nyomtatott áramköri lapjaink is. Ha tehát valakik tönkremegy, azt is sokáig kicserélhetjük.

— Mik a tipushibái a Primónak?

— Csak a sorozattól függőek, az egészre jellemző nem fordult elő egy sem. Az első szeriában sok forrasztási hiba volt, ezért rengeteg RAM ment tönkre. Ezenkívül a magnóra panaszokdokat nagyon sokan.

— Szervizelésük, vállalkozások veszteséges vagy nyereséges?

— Először úgy tűnt, hogy nagyon veszteséges lesz, mert átalányszerződést kötötünk. Aztán időközben kiderült, hogy nem is olyan rossz ez a gép! Májig is nyugodt lelkiismerettel vallom, ez a gép jó. Így némi kis nyereséggel dolgozunk.

— Javító szakember személtel nézve mennyire becsüli a Primo élettartamát?

— Óriási különbség van a szenzoros és a nyomógombos billentyűzet között, az utóbbi javára. Egyértelműen a billentyűzetet rontották el a Primón. A nyomógombos szerintem jól bevált. Ettől függetlenül a nyomógombos gépeknek az élettartamát előre nem lehet pontosan meghatározni. Öt év múlva még biztosan lesznek, működnék. A szenzorosnak viszont nem garantált a műanyaglap-utánpótlása. Átalakításuk nem műszakilag megoldhatatlan, hanem a billentyű beszerezhetetlen. Elvileg elképzelhető gyártót találni rá, mégsem hiszem, hogy bárki is vállalná. Egyébként nagyon könnyű lenne az átalakítás. Véleményem szerint ezt az egész Primo-ügyet valahol elhibázták. Ez nagyon jó kis gép, amivel az iskolákat elláthatták volna. Elképzelve a programhiánytól, szerintem egyenrangú a Commodore-okkal, sőt, az alapgépet jobbnak is találom azoknál.

## Körösme helyett

Eddig tartott a bolyongás. Ha a látszat csalt, az csakis a Primo és a primósok javát szolgálhatja. Mert bár egy álmoskönyv sem mondhat rosszabbat, mint amit bolyongásom közben éberem tapasztalhattam, sejtethető, érezhető, hogy hétézer Primo úgymond lélegzik, noha szinte láthatatlan. Ezért hiszszük, minden róla alkotott elmarasztaló ítélet csalhatatlansága sem csalhatatlan.

SZÜLYOVCSZKY CSABA

Budapesten az alábbi GELKA-szervezetekben foglalkoznak Primo gépekkel:

- V., Fehérhajú utca 8—10.
- VII., Rózsa Ferenc utca 10.
- VIII., Kisfaludy köz 4.
- Vidéken:  
0203. sz. szerviz, Pécs, Ybl Miklós u. 3.  
0606. sz. szerviz, Szeged, Ságvári u. 6.  
0930. sz. szerviz, Debrecen, Segner tér 1.  
0401. sz. szerviz, Békéscsaba, István király tér 9.

0804. sz. szerviz, Győr, Arany János u. 25.  
Az idei évkezdétkor kezettahány miatt nem forgalmaztak programokat a Primo gépekhez. Az első negyedévben az elmaradást pótolták, így a programok kaphatók a Cosynál és Budapesten a VII. kerület, Rákóczi út 60. sz. alatti TRIÁL boltban.

Az Elektromodul (Bp. XIII., Jászai Mari tér 5.) a következő tartozékokat forgalmazza:

- DCD PRT asztali grafikus nyomtató 62 011,— Ft
- DCD PRT—42 Baby print grafikus nyomtató 30 150,— Ft
- lemezmaghajtó 14 380,— Ft
- optikai billentyűzet 8 654,— Ft
- Az Elektromodulnál a vásárláshoz előzetesen (három-négy héttel) számolva átutalt időt kell leadni az igényeket, hogy a terméket leggyárthassák.

A Baby print grafikus nyomtatót a Kispesti Áruházban 23 990 forintért árulják.

A szerkesztőség — támogatva a primósok táborát — ezúton is felhívja a figyelmet a Primo klubra, amely által lehetőségük nyílik iskoláknak és magánzemélyeknek egyaránt a géppel (gépükkel) kapcsolatos gondok megoldására. A klub (Bp. VII., Almásy tér 6. Foglalkozások: minden hónap 2. hétfőjén. Vezető: Somogyi György) várja az új belépőket!

## Programkatalógus

Egy programon 300 Primóra írt program katalógusához lehet hozzájutni! A HCC Primo szakosztálya által elkészített programismertető bevizsgálta a számítógépeket. A klub a Primónál használt különböző magnetofonok miatt a beolvashatóságot nem garantálja. Akinek a program kell, egy kazettát, egy válaszborítékot juttasson el szerkesztőségünkbe, s a magazinon keresztül megkapja a katalógust. Izeltől a 300 ismertetőből egy minta:

A program neve: Tálts  
Azonosító száma: 197  
Készítője: Répás Sándor  
Kipróbálva: A64, A48 típusokon  
Hossza: 14 blokk  
Memóriáiánya: 20,1 kb-át  
Üyességi játéka, „SAVE” parancs-csal másolható.  
Két ember párbajozik a képernyőn. Az egyiket a gép, a másikat a játékos vezérli billentyűkkel.





áránt a számítástechnika népszerűsítésé a célja. Rendszeresen foglalkoznak olyan témákkal, amelyeket a mikroszámítógép világából merítenek.

A magazin helyet ad olyan témáknak, amelyek ugyan bekerültek a Mi és a számítógép műsorba, de olvasva is izgalmasak, illetve teret enged a televíziós műsor szerkesztőjének, aki a lap hasábjain kíván foglalkozni olyan kérdésekkel, amelyek helyszúke miatt nem fértek bele a harmincperces adásidőbe.

Talán az olvasók előtt sem titok — sokan élnek is a lehetőséggel —, hogy a Mi és a számítógép műsor egyik nagy attrakciója, sőt kuriózuma a néhány hónapja bevezetett újdonság, az adás végén kazettára rögzíthető és onnan a számítógépbe tölthető programokat sugárzunk a tévé hangcsatornáján. Ezek között is a legnépszerűbbek a rejtvenyprogramok, amelyek Commodore és Sinclair gépekre íródtak. A jövőben folytatni kívánjuk — a rejtveny-

pályázat befejeztével is — ezt a kezdeményezést a legelterjedtebb géptípusokra. Ehhez várjuk a nézők és a magazin olvasóinak ötleteit, és természetesen programjaikat.

A számítástechnikai műsorunkban továbbra is foglalkozunk a számítógép és a video kapcsolatával, illetve a két szakma határterületeit érintő témákkal. Izeltül annyit, hogy például a közeljövőben bemutatjuk a képlemezjátszó és a számítógép összekapcsolását — az ún. interaktív video lehetőségeit — valamint azt, hogy mindent miként hasznosítják az oktatásban és a tájékoztatásban.

Továbbra is beszámolunk a környező országok szakmai kiállításainak újdonságairól, és természetesen bemutatunk egy-egy sikeresen működő — vagy éppen gondokkal küszködő — számítógépes klubot, sakkkört, iskolát.

HEGYI ISTVÁN

**A** Mikroszámítógép Magazinak és a Magyar Televízió Mi és a számítógép című műsorának egy-

## Számítógép és művészet

Ma még vitatott kérdés, hogy önálló műfaj-e a számítógépes művészet. Szerencsére az alkotók közül többen a szócseplés helyett inkább az új technikával ismerkednek, kísérleteznek. E szárnypórbálgatók egyike, Waliczky Tamás, a Fiatal Művészek Stúdiójának tagja így beszél munkájáról:

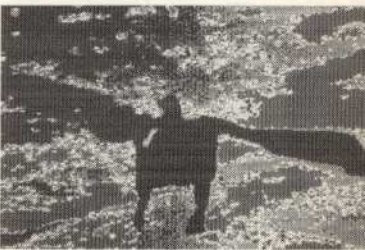
Művészi pályám szorosan összekapcsolódik a számítógéppel. Korábban sokat festettem, fotóztam, rajzoltam, s volt néhány kiállításom is. A stúdióba aztán már számítógépes alkotásaimmal pályáztam.

Hogyan kezdődött számítógépes pályafutásom? Őt esztendeje — mint szellemi szabadfoglalkozású grafikus — illusztrációkat, rajzfilmeket és reklámfotókat, majd egy elektronikus játékhoz rajzfilmet készítettem. Két évig Commodore 64-en dolgoztam. Sajnos a kép minősége arra alkalmatlan volt, hogy alkossak vele, így hát folytattam megszokott életem, vagyis festettem, fotóztam, s készültem a következő kiállításaimra. Aztán hozzájutottam egy Atari 520-as géphez, melynek grafikus képe jobb és színei sokkal szebbek, mint a Com-

modore-é. Felhagytam a játékkészítéssel, s áttertem a grafikai programokra. Ezekkel megannyi demonstrációs rajzot, grafikát, mozgást, animációt hoztam létre. Munkámban segítőtárs volt az ART DIREKTOR rajzolóprogram. Mind jobban elmerültem a számítógépes technikában, igazán élveztem a munkát.

A munkám is megváltozott, amióta számítógéppel dolgozom. A manualitásban persze nincs nagy különbség. Ma is táblákon rajzolok, mintha rajzlapra rajzolnék ceruzával. A másság akkor kezdődik, amikor a rajzokkal manipulálok, mert a kifejítés helyett ott vannak a számítógépes eszközök. Ezek gyorsabb, tökéletesebb technológiát tesznek lehetővé. És az animációs munkában is sokat segítenek a programok. Így a fázisrajzoló programmal két fázis közé megrajzolhatom az átmenetet úgy, hogy a nehézkes számításokat helyettem a gép végzi. Mindemellett egy korábbiót teljesen eltérő művészeti termék jön létre gyorsan, szépen és fantasztikus látványosan.

Mostanában előszeretettel alkalmazom a számítógépet arra, hogy végtelentelven vetítse a rajzokat. Ez fontos, mert olyan cselekményeket



Vízmadár 1986

választok, amelyek végtelenszer ismétlődnek, s ez az állandó mozgás — a képzőművészetnek ez a műfaja az ún. mobil — ad a nézőknek egyfajta új élményt. Továbbfejlesztett változata ennek az, amikor egy kiállítóteremben üzembe helyezzünk mondjuk húsz monitort — mintha azok képek, festmények lennének —, s ezeken folyamatosan ismétlődik a cselekmény...

E technikának számomra másik — és nem elhanyagolható — előnye az, hogy az információk kimentem, abból semmi sem vész el. Egy későbbi időpontban ismét kapcsolatba kerülhetek a rajzjal, javíthatom, tökéletesíthetem, ha egyszer más koncepció szerint kívánok más dolgokat rajta. Gyorsan, pontosan, anyagmegtakarítással! Ha az ember ügyesen bánik a gép kinlta lehetőségekkel, mindenfajta őrültséget kipróbálhat a monitoron, amelyeket esetleg papíron, vásznon hagyománytisztelből nem tenne meg.

Tudok Angliában valakiről, aki absztrakt festményeit számítógépen tervezi. Addig játszik a színekkel, amíg a maximális hatást el nem éri, s csak ezután festi vásznonra a képeit. Műalkotásában mindenestre a számítógépe az elsőségség.

Miért nem népszerűbb a számítógépes művészet a jelenleginél? Talán mert sokan félnek a korszerű technikatól, tartanak attól, hogy nem képesek a gépet kezelni. Pedig például az ART DIREKTOR program, noha meglehetősen bonyolult, mindössze két óra alatt megtanulható annyira, hogy dolgozni kezdhessünk vele.

A számítógéppel egészen másfajta képek születnek, mint a hagyományos festészetben. Mégis azt hiszem, e másság a művészetben teremtő erővel hódíthat.

(Képeinken Waliczky Tamás néhány műve)  
H. I.



Organikus anatómia I. és III. 1987





## KOMPLEX SZÁMÍTÁSTECHNIKAI SZOLGÁLTATÁS!



**TEK V.  
számítástechnikai  
főosztály**  
Budapest VII., Vörösmarty u. 16.  
Tel.: 427-338, 202-415

A Ferroglobus számítástechnikai főosztálya nagy adatfeldolgozási hagyományokkal, jól képzett, nagy tapasztalatú munkatársakkal rendelkező számítóközpontja széles skálájú, komplex számítástechnikai szolgáltatásait ajánlja:

- kereskedelmi szakmai, készletgazdálkodási, pénzügyi, számviteli, bér- és munkaügyi stb. rendszerek szervezése, programozása és fejlesztése nagy kapacitású, modern ICL típusú számítógépen
- mágneses adathordozóra történő adat-előkészítés — rögzítés
- számítógépidő bérbeadása
- tanácsadás

**Ferroglobus**



# BASIC és gépi kód

Legutóbb az USR függvényeinek lehetőségeit kiterjesztő rutinok működésének részleteiről írtam. Utaltam néhány problémára is, melyeknek megoldására most megpróbálok receptet adni.

A BASIC nyelvű betöltőprogramoknál egy új módszert alkalmaztam, melynek lényege, hogy a betöltendő adatokat hexadecimális karakterláncokban tárolom. A módszer előnye a tömör kód, hátránya a hosszú betöltési idő.

A múltkori 7. listába kellemetlen hiba csúszott, melyért ezúton kérek elnézést. A C16-on a DATA sorokat véletlenül átszámoztam, és ezt nem vettem észre. Betöltés közben nem a programos száma, hanem a sor feldolgozásánál elsőként betöltött adat címe jelenik meg a képernyőn. Ez hibátlan adatok esetén nem jelent problémát, de egy-egy előforduló hiba megkeresését nagyon megnehezíti. A most közölt betöltőprogramoknál már megszüntettem ezt a hibát.

## A problémák

A gépi kódú rutinoknak a kazettapuffer-

ban való elhelyezése kényelmes, de komoly gondokat okozhat. Lássunk ezek közül néhányat.

A nyomtatásban megjelenő gépi kódú rutinok nagy része a kazettapufferba van lerakva. Ezekkel nem lehet a mi rutinunkat együtt használni, mert egyidejűleg ugyanazt a helyet vennék igénybe. Ha ki akarjuk próbálni azt a többlemeket összegző szubrutint, mely *A Commodore 64-es belső felépítése* című könyv 94–96. oldalain található, le kellene mondanunk a DEF USR által nyújtott lehetőségről.

A betöltőprogram viszonylag hosszú átutási ideje arra késztet, hogy a rutint gépi kódú alakban tároljuk. Akinek csak kazetás háttértárolója van, annak erre nincs lehetősége. Nagyobb gond, hogy a rutin aktiválása után a magno egyáltalán nem használható. Bármilyen adatátviteli próbálkozás — az „önző programozás” eredményeképpen — a rendszer „lefagyásához” vezethet.

A megoldás egyszerűnek látszik: tegyük a programot másóvá. Igen, de hova? Ha egy meghatározott helyet választunk, ismét az első problémába ütközünk, egy másik szubrutin szintén oda kívánczok.

Próbáljuk meg oda tenni, ahol éppen szabad hely van! Ekkor viszont a programbeli abszolút címmel lesz baj, jelen esetben a két feladkozórutin belépési pontjának és a táblázat kezdetének a címe nem fog egyezni. Írjuk át ezeket a ténylegesre. Hogy ez miként lehetséges, arra láthatunk példát a mostani három programlistán.

## A megoldás

A listákon az áthelyezhető rutinok betöltőprogramjai láthatók. Az új program sorok beszámozatlal eltekintve, csak néhány apróbb változtatás történt az előző számban közölt programokhoz képest, hogy kevés gépeléssel elvégezhető legyen a módosítás. A 110-es és 200-as sorban a 828 helyére a kezdőcímet tartalmazó KC változó került. A FOR L=1 TO 15 utasítás mögült eltűnt a legtöbb bajt okozó PRINT, szerepét a következő sor után beszűrt új PRINT utasítás vette át. Ez valamivel hosszabb, de mindig a legutóbb olvasott DATA sorszámát írja ki. A C16-os változat DATA utasításait a RENUMBER 828,12,828 parancscímű átszámoztam, de ez a művelet elhagyható.

### 1. lista

```

100 rem usrdef-2 c64
105 z=256 : kc=49152
110 k=kc : print chr$(147)
120 def fnd(x)=x-48+7*(x>64)
130 for i=1 to 15
140 s=0 : read a$,b
145 print chr$(19);peek(63)+z*peek(64);
150 for i=1 to len(a$) step 2
160
a=16*fnd(asc(mid$(a$,i,1)))+fnd(asc(mid$(
(a$,i+1,1)))
170 poke k,a : s=s+a : k=k+1 : next
180 if b<>s then print "hiba!" : stop
190 next
191 for i=1 to 6
192 read d,a : a=kc+a : b=int(a/z)
193 poke kc+d,a-z*b
194 poke kc+d+1-(i<3),b
195 next
200 sys kc : end
828 data a951a0038d00038c0903a99e,1044
840 data a0038d0a038c0b0360207300,714
852 data c996f0062079004ce7a72073,1371
864 data 00c9b7f0034cb3b385022073,1343
876 data 00f068b008290f0a85022073,876
888 data 00c9b2d05a207300208aad20,1199
900 data f7b7a602300a9dd0c0989dda,1562
912 data 034ceaa76d1203988d11034c,1031
924 data eaa7a900850d207300c9b7d0,1455
936 data 0f207300f029900ea57ad002,1098
948 data c67bc67a2079004c8dae290f,1241
960 data 0a4820730020f1ae68a8b9da,1351
972 data 03855b9d00385564ce0af4c,1398
984 data 08af48b248b248b248b248b2,1433
996 data 48b248b248b248b248b2,1250
998 data 1,21,11,98
999 data 79,159,83,158,143,158,148,159

```

### 2. lista

```

100 rem usrdef-2 vc20
102 z=256
104 kc=peek(55)+z*peek(56)-176
106 b=int(kc/z)
108 poke 55,kc-z*b : poke 56,b
110 k=kc : print chr$(147)
120 def fnd(x)=x-48+7*(x>64)
130 for i=1 to 15
140 s=0 : read a$,b
145 print chr$(19);peek(63)+z*peek(64);
150 for i=1 to len(a$) step 2
160
a=16*fnd(asc(mid$(a$,i,1)))+fnd(asc(mid$(
(a$,i+1,1)))
170 poke k,a : s=s+a : k=k+1 : next
180 if b<>s then print "hiba!" : stop
190 next
191 for i=1 to 6
192 read d,a : a=kc+a : b=int(a/z)
193 poke kc+d,a-z*b
194 poke kc+d+1-(i<3),b
195 next
200 sys kc : end
828 data a951a0038d00038c0903a99c,1042
840 data a0038d0a038c0b0360207300,714
852 data c996f0062079004ce7c72073,1403
864 data 00c9b7f0034cb3d385fe2073,1627
876 data 00f068b008290f0a85fe2073,1126
888 data 00c9b2d050207300208acd20,1229
900 data f7d7a6fe300a9dd903989dd8,1842
912 data 034ceaa785029885014ceac7,1442
924 data a9085e0d207300c9b7d00f20,1181
936 data 7300f029900ea57ad002c67b,1372
948 data c67a2079004c8dce290f0a48,1034
960 data 20730020f1ce68ab2b9d80385,1435
972 data 55b9d90385564ce0cf4c08cf,1507
984 data 48d248d248d248d248d2,1692
996 data 48d248d248d248d2,1128
998 data 1,21,11,96
999 data 79,157,83,156,141,156,146,157

```



```

100 rem usrdef-2 c16
105 z=256 : kc=4096
110 k=kc : print chr$(147)
120 for l=1 to 15
130 s=0 : read a$,b$ : b=dec(b$)
135 print chr$(19);peek(63)+z*peek(64)
140 for i=1 to len(a$) step 2
150 a=dec(mid$(a$,1,2))
160 poke k,a : s=s+a : k=k+1
170 next
180 if b<>s then print "hiba"
190 next
191 for i=1 to 6
192 read d,a : a=kc+a : b=int(a/z)
193 poke kc+d,a-z*b
194 poke kc+d+1-(i<3),b
195 next
200 sys kc : end

```

```

828 data a951a0038d008038c0903a99b,411
840 data a0038d0a0938c0b0360207304,2ce
852 data c996f0062079044cd98b2073,535
864 data 04c9b7f0034c9d9a85e82073,5fa
876 data 04f065b008290f0a85e82073,453
888 data 04c9b2d05720730420e19da6,5d1
900 data e8300a9d4803989dd7034cdc,5d1
912 data 8b8d0205988d01054cdc8ba9,4a6
924 data 00850d207304c9b7d00f2073,41b
936 data 04f029900ea53bd002c63cc6,535
948 data 3b2079044c1e94290f1a4820,280
960 data 730420859468a8b9d7038555,52d
972 data b9d80385564cec954ca1941c,5d9
984 data 991c991c991c991c991c,43e
996 data 991c991c991c991c,2b8
998 data 1,21,11,95
999 data 76,156,80,155,140,155,145,156

```

### 3. lista

A címek átírását a 191...195 sorokban lévő programrés végzi, a módosító adatok az utolsó két DATA sorban vannak. Ezeket nagyon figyelmesen gépéljük be, mert nincs hozzájuk ellenőrző üzeneg.

A címátíró programrés működési elve a következő.

Minden módosításhoz két adat tartozik, ezek a betöltött rutin kezdetéhez viszonyított relatív címek. Az első (D) azt mutatja, hogy a módosított cím alacsonyabb helyértékű bajtját hová kell beírni, a második (A) pedig azt, hogy a kezdőcímmel mennyit kell hozzáadni, hogy a valós címet kapjuk. Ennek első és felső bajtját egy cikluslépésen belül írjuk be a megfelelő helyekre. Az első két esetben — a vektorcímeket átíró szubrutinban — ezek nem egymás mellett vannak, köztük van az LDY kódja.

géptípus	programhossz
C64	178 bajt
C16	175 bajt
VC20	176 bajt

### 1. táblázat

### 2. táblázat

	c-64	c-16	vc-20
sgn	48185	41662	56377
int	48332	41816	56524
abs	48216	41693	56408
fre	45949	39522	54141
pos	45982	39549	54174
sqr	49009	42468	57201
rnd	57495	42759	57492
log	54794	40990	55786
exp	49133	42592	57325
cos	57956	43632	57953
sin	57963	43639	57960
tan	58036	43712	58033
atn	58126	43802	58123
peek	47117	40442	55309
len	46972	40289	55164
str\$	46181	39782	54373
val	47021	40339	55213
asc	46987	40304	55179
chr\$	46928	40123	55020
dec		40475	
hex\$		46343	

A 100-as és 110-es sorok közé beszúrt utasításokkal jelöljük ki a gépi kódú program helyét. A listák közötti eltérés kivételesen nem azt jelenti, hogy a különböző géptípusokon eltérő módszert kell alkalmazni, hanem párhuzamosan mutat be három megoldási lehetőséget.

A C64-es változatban egy szabad területen helyezük el programunkat, a \$C000 címűl kezdve. Ez a kazettapufferhez hasonlóan „népszerű” hely, ezért előfordulhat, hogy egy másik helyet kell választanunk. A két másik géptípuson ez a módszer csak korlátozottan alkalmazható a megfelelő méretű, összefüggő szabad RAM terület hiánya miatt.

A VC20-as változatnál a BASIC RAM tetejéből vágjuk le a szükséges területet. A 104-es sor végén álló szám a program mérete, az egyes géptípusokhoz tartozó programhosszakat az 1. táblázatban foglaltam össze. Ez a módszer mindhárom géptípuson jól alkalmazható, de a 64 kilobájtra bővített C16-oson nem ajánlatos a \$8000...\$FFFF közötti RAM területre gépi kódú rutint tenni.

A C16-osra írt programban a BASIC RAM elejéből vágjuk le a megfelelő méretű helyet. Sajnos — jelenlegi ismereteink szerint — ezt a műveletet a BASIC program beírása, illetve betöltése előtt minden alkalommal parancsmódban kell elvégezni. Itt erre a POKE 43,176 : POKE 4721,0 : NEW utasításcsoportot alkalmazhatjuk. A módszerrel további információk találhatók Énekes Ferenc *Programónkritika* is... című cikkében, mely a Magazin 1986. októberi számában jelent meg.

### Próbáljuk ki!

Jó lenne kipróbálni, hogy jól működik-e a programunk, de egyelőre nem nagyon akad olyan gépi kódú függvényünk, melyet USR-rel hívhatnánk. Alkalmassá erre a BASIC legtöbb beépített függvénye is, ha ezek belépési címait adjuk meg a DEF USR utasításokban.

A 2. táblázatban összegyűjtöttem néhány ilyen címet. Akinek kedve van, próbálko-

zék velük. Figyelem! Van néhány olyan függvény, melynek a paramétere, és néhány olyan, amelyiknek az eredménye karakterlánc típusú. Ez a megfelelő USR függvényről is így van.

A táblázatból kiolvasható, hogy a C64 és a VC20 típusok függvényei közül hiányzik a DEC és a HEX\$. Már nem sokáig, mert rövidesen jelentkezem az őket helyettesítő felhasználói függvényekkel.

### Még egyszer a BRK-ről

Tavaly novemberben kaptam kézhez Varga Csaba szegedi olvasó sorait. A következőket írta:

„Az 1987/11. számú programozástechnika rovatban, a Basic és gépi kód című cikkben a BRK utasítás egy fontos jellemzője lemaradt. A 68XX család ezt az utasítást 2 bajtnak értelmez, azaz az utasítás dekodolása után a PC 2-vel növekszik. Ennek az a hatása, hogy a BRK utasítást végrehajtva az egy belső megszakított generál, és azt végrehajtva az RTI utasítás után a visszatérés nem közvetlen a BRK után, hanem 2 bajttal utána történik.”

Olvasónk észrevétele helyes, bár a jelenség leírása kissé pontatlan. A BRK említett tulajdonságáról azért nem írtam, mert sorozatomban nem tudok minden apró részletrre kitérni. Úgy gondolom, hogy a novemberi számban ismertett utasításcsoport a profi programozók kelléktárába tartozik, ők pedig nem az én cikkemből ismerik meg a gépi kódú programozás rejtelmeit. A veremkezelő utasításokat akkor csak a teljesség kedvéért ismertettem.

A BRK említett tulajdonságát célszerű azoknak is figyelembe venni, akik valamilyen monitorprogramot használnak gépi kódú programjaik írására, tesztelésére, például a C16 beépített monitorját. Az ő szempontjukból viszont mindegy, hogy az említett jelenség a BRK-nak vagy a monitornak tulajdonítható.

Megköszönve Varga Csaba bíráló megjegyzését, ígérem, hogy a témát továbbra is napirenden tartom, és — anyaggyjúts után — újra jelentkezem vele.

BARNA LÁSZLÓ



# Számítógépes grafika Pascalban

## MOZGÓ KÉPEK TEKNŐSGRAFKÁVAL

**E sorozat előző cikkében bemutatunk egy módszert geometriai alakzatok síkbeli transzformációjának programozására. Most egészen más megközelítést ismertetjük ugyanennek a problémának. A megoldás módszere a teknősbéka-grafika.**

Ez a grafika mint ötlet eredetileg a LOGO nyelv megalkotásakor merült fel, és azt a célt szolgálta, hogy akár kisiskolás gyerekek is játsza megtanulhassák a programozás alapjait. A számítógép képernyőjén megjelent egy kicsi teknősbéka képe, amelyet egyszerű parancsokkal előre-hátra lehetett mozgatni, jobbra-balra forgatni, és a farkát felemelni, leereszteni. Ha a béka farka lent volt, mozgás közben nyomot hagyott a képernyőn. Nos, az ötlet kitűnően bevált. Olyannyira, hogy komolyabb alkalmazásokra is kezdtek használni.

Több Pascal implementációba is beépült a teknősgrafika, természetesen nem parancsok, hanem eljárások formájában. Így például híres az Apple számítógépek teknősgrafikája, de az IBM PC család Turbo Pascal programja is tartalmaz teknőseljárásokat. A Hisoft Pascalhoz is tartozik egy forrásnyelvű fájl, amely TURTLE (= teknős) fájlnev alatt szerepel, és amellyel kevesen tudnak mit kezdeni. Az alábbiakban ennek alkalmazására mutatunk egy példát.

A TURTLE nevű fájl önmagában nem teljes értékű program, hiszen sem eleje, sem vége nincs. Vagy be kell közvetlenül építenünk programunkba, vagy — mint e cikkben is látható —, include-fájlként hívjuk meg a főprogramból. Ha kilistázzuk a fájlt, egy deklaráció részletét látjuk meg először. Itt az olyan globális változók vannak megadva, melyeket a teknőseljárások használnak. Ezek az XCOR, YCOR: a „teknősbéka” pillanatnyi helykoordinátái, a HEADING: a teknős fejének iránya, mind Real változó, valamint a PENSTATUS integerváltozó, melynek értéke 0, ha a toll ir és 1, ha a tollat felemeljük.

Azoknak az eljárásoknak a katalógusa, amelyek ismerete a programozáshoz szükséges, a következő:

INK C:INTEGER

— a tinta színét állítja be

PAPER C:INTEGER

— a papír színét állítja be

```

10 PROCEDURE Hozzaadepointert;
20 (1985- JULIUS HOSSELY F40
30 CXL-3)
40
45 TYPE LEPHARMADHARRAY(0..7,0..7,0..3)OF CHAR;
60 FILM=RECORD
70   HUFATO:FILE;
80   HCCCLASS:LEPHARMAD;
90   ENGI
100
110 VAR I,J:INTEGER;
120   ELOZO:INDO,FUTG,FUTOTI:FILE;
130
140
150 (CXL-3)
160 GPF:ISTURTEL;
170 (CXL-3)
180 CAP:ISTURTEL;
190
200 PROCEDURE FODJA,EGDQ:REAL;
210 BEGIN
220   FODJ(A);LEFT(75-SZOG);
230   FODJ(B);RIGHT(75);
240   FODJ(C);RIGHT(75);
250   FODJ(D);LEFT(75-SZOG);
260   FODJ(A);
270 END; FODJ;
280
290 PROCEDURE FOGASKEREN;INC;
300   F:INTEGER;R:REAL;
310 BEGIN
320   SETXY(X,Y);
330   PENUP;
340   VECTOR(A,R);
350   PENDING(1);
360   LEFT(90);
370   FOR I:=1 TO 5 DO
380     FODJ(LB);FODJ(R);
390   END; FOGASKEREN;
400
410 BEGIN
420   ELOZOTI:=NIL;
430   FOR I:=0 TO 5 DO
440     BEGIN
450       FAGEI;
460       WRITE(CHR(1)-CHR(1)+I);
470       (INC(10)*I)WRITE(CHR(22)-CHR(10)+I);
480       CHR(9);WRITE(' A B C D E');
490       WRITE(CHR(23)-CHR(1)+I);
500       PAPER(7);
510       FOGASKEREN(18,25,142);
520       (10*50)WRITE(10);
530       FOGASKEREN(10,69,142);
540       (50*20)WRITE(20);
550       FOGASKEREN(18,159,142);
560       (50*30)WRITE(30);
570       FOGASKEREN(2,146,159);
580       (25*50)WRITE(50);
590       FOGASKEREN(1,164,147);
600       (20*62)WRITE(62);
610       FOGASKEREN(14,164,134);
620       (5*30)WRITE(70);
630       FOGASKEREN(12,206,134);
640       (15*30)WRITE(80);
650       HEADING(0);
660       MOV(L,START,END,0);
670       MOV(L,END,START,LENEN,LEPHARMAD);
680       ELOZOTI:=F;
690       END;
700     FUTOTI:=F;
710     REPEAT
720       F:FOOT(FUTOTI);
730       (FUTG)WRITE('PUTATO');
740       (FUTG)WRITE(INDO);
750       FUTO:=FUTOTO:=F;
760       FASE:=F;
770       WRITE(CHR(20)-CHR(9)+I);
780       FOR I:=0 TO 30 DO
790         FODJ(2);
800         WRITE(' ');
810       REPEAT
820         MOV(INDO,PUTATO);
830         FODJ(148);MOV(INDO);
840         FOR I:=1 TO 30 DO
850           WRITE(2);
860         END;

```

### COPY

— másolatot készít a képernyőről a ZX nyomtatóra

PENDOWN C:INTEGER

— leteszi a tollat a papírra és C színűre állítja

PENUP

— felemeli a tollat

SETHD A:REAL

— beállítja a teknős irányát

SETXY X,Y:REAL

— beállítja a teknős helyzetét

FWD L:REAL

— előre megy L hosszán

BACK L:REAL

— visszamegy L hosszán

VECTOR A,L:REAL

— elfordul A szög irányába és előre megy L hosszán

RIGHT A:REAL

— jobbra fordul A szöggel

LEFT A:REAL

— balra fordul A szöggel

ARC R:REAL;A:INTEGER

— körívet rajzol. A a középponti szög, R arányos a sugárral

TURTLE

— inicializálja a teknőst: törli a képernyőt, középre állítja a „békát”, a fejét vízszintesen jobb irányba állítja, a hátteret sötétkékre, a tinta színét sárgára állítja be.

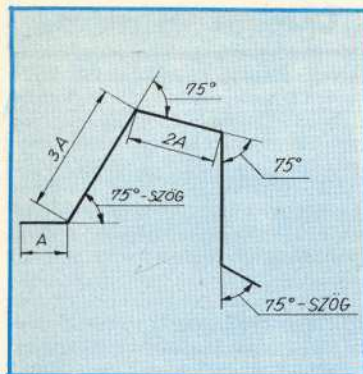
### Egy kis technológia

Ennyi bevezető után nézzük a feladatunkat! A teknősbéka-grafika nagyon hatékony abban az esetben, ha egy geometriai alakzatot kell sokszor transzformálnunk rajzán. Ilyen eset például egy fogaskerék síkjának elkészítése. Ilyenkor elegendő egyetlen fog rajzának programozása, majd ennek a programrésznek kis elforgatásokkal való sorozatos hívásával felrajzolódik a teljes fogaskerék.

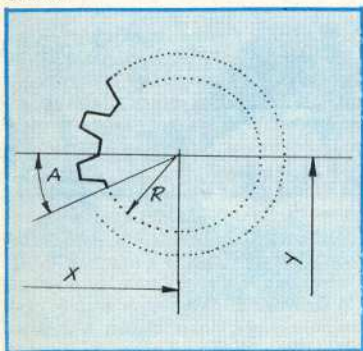
Ha ez mennyire egyszerű, a mellékelt programlista FOG azonosítóú eljárásán látható. Az igazi fogaskerék egy fogának két oldalát evolvens görbe határolja. A mi esetünkben azonban a fogaskerék méretei olyan közel esnek a képernyő felbontásához, hogy az evolvens és az egyenes között nem lehet különbséget tenni, ezért természetesen az egyszerűbb egyenest programozzuk.

A FOG eljárásnak két paramétere van: az A REAL paraméter a fog nagyságával arányos mennyiség (lásd az 1. ábrát), a SZÖG pedig tulajdonképpen a fogszámmal fordítottan arányos szög. Ha fogszámot rajzolunk (a fogasléc végtelen fogszámú





1. ábra  
2. ábra



fogaskerékként is értelmezhető), ez a szög nulla értéket vesz fel, a fogság csökkenésével a SZÖG növekszik. Egymással kapcsolódó fogaskerekeknek az A értéknek meg kell egyeznie, a SZÖG-nek azonban természetesen nem.

A FOGASKERÉK eljárás a FOG segítségével egy teljes kerekét rajzol fel. A paraméterek geometriai jelentése: N a fogság, X, Y a fogaskerék középpontjának koordinátái, A az elforgatási szög (ennek a későbbiekben lesz jelentősége), és végül R a sugár. A 2. ábra segíti a programrészlet működésének megértését.

Ha az olvasó idáig jutott a cikk olvasásában, kipróbálhatja a fogaskerekek felrajzolását, és tapasztalhatja, milyen egyszerű is ez. Lépünk azonban egy lépéssel tovább, hozzájuk forgásba a fogaskerekeket!

## Egy kis kinetika

Mozgóképet előállítani számítógép segítségével is úgy lehet, ahogy azt a mozi vagy a televízió teszi: a mozgást fázisképekre kell bontani, s ha azokat elég nagy sebességgel felvillantjuk a néző előtt, majd minden képet egy kis ideig kivetítve hagyunk, folyamatos mozgás benyomását keltjük.

A számítógép esetében a helyzet inkább a rajzfilm megvalósítására hasonlít: az egyes képkockák elkészítése történhet viszonylag lassan (például BASIC nyelvű programmal vagy interaktív rajzolóval), csak a visszajátszásnak kell igen gyorsan végbemennie.

Az eljárás tehát az lesz, hogy megrajzoljuk az egyes fázisképeket, majd rendre eltávolítjuk egy alkalmas strukturált változóba. Gondot okoz, hogy egy kép tárolására a számítógép RAM-jához képest nagy memóriaterületre van szükség. Ezért gyakorlatilag csak önmagába visszatérő (periodikus) mozgást tudunk megvalósítani, azt is úgy, hogy „vetítővászonként” csak a képernyő felső harmadát használjuk ki.

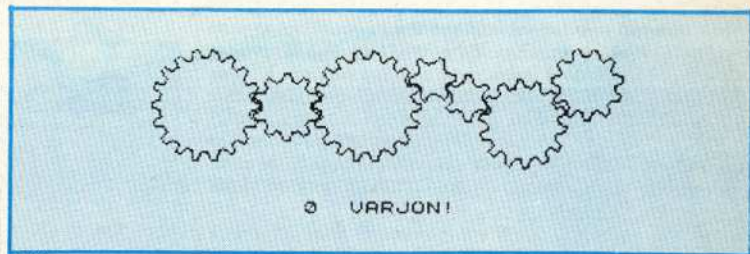
Esetünkben hat képkockából álló „filmet” készítettünk. Egy filmkockát egy KÉPHARMAD típusú,  $8 \times 8 \times 32$  bájtnyi tömbben tárolunk. A KÉPHARMAD-ból és egy mutatóból RECORD-ot deklarálunk, amelyet mutatók segítségével fogunk kezelni. A program ezután egy ciklusban felrajzolja a hat egymást követő képkockát, minden egyes képet egyhatod fogsátsánnival elforgatva az előzőhöz képest, majd a képernyő-fájl felső harmadát átmásolja a KOV mutató által címzett rekord KOCCA elemébe. A másolást a PEEK függvény segítségével valósítjuk meg. A 3. ábrán egy fáziskép látható.

A rekord MUTATÓ azonosítója, pointer típusú változója a ciklus magja végén az előző FILM-re mutat. Így egyszerű listát

használnunk, melyben mindig kiolvassuk a KOV mutató által címzett rekordot, ennek KOCCA elemét a képernyő-fájl felső harmadába POKE-oljuk, majd lefuttatunk egy belső üres ciklust 500-szor, csak azért, hogy a kép egynéhány pillanatra megálljon, és szemünk képes legyen észlelni. A ciklus akkor állna le, ha az  $l = 2$  feltétel teljesülne, amire természetesen a közeljövőben nincs remény; így sikerült egyszerű módszerrel végtelen ciklust szervezni.

Még egy apró trükk. Az egyes fázisképek kirajzolása a képernyőre a filmkészítés közben nem látható, mivel mind a PAPER, mind az INK színét azonosan fehérre állítottuk be.

Ez az egyszerű program jól szemlélteti a Hisoft Pascal figyelemre méltó sebességét. Ha az olvasó saját „filmet” akar forgatni, elmondjuk, hogy több helyet lehet biztosítani a RAM-ban, ha a programot T opcióval fordítjuk, hiszen ekkor sem a forrásnyelvű szöveg, sem a compiler, sem az editor nincs az operatív memóriában futás közben. Tovább lehet „fogyasztani” a programot, ha kettévágjuk: egy programmal elkészítjük a fázisképeket, és ezeket egyenként kazettás magnóra vagy lemezmaghajtóra küldjük ki. Egy másik program feladata csak annyi lesz, hogy visszaolvassa a háttértárról a képeket, elrakja a fent vázolt algoritmussal a RAM-ba, majd visszajátssza. A TURTLE fájl eljárásai a ROM eredeti pont- és vonalrajzoló algoritmusait hasz-



3. ábra

szervezünk a rekordokból. A ciklus lefutása után a program megkeresi az előző rekord MUTATÓ-ját, és az utolsó kocka pointerének értékét adja neki. Így a rekordokból zárt gyűrű alakú láncot alakítottunk ki, amelyből csak egy (tetszőleges) elem helyét kell ismerni, és innen az egész gyűrű sorban kiolvasható. Vegyük észre, hogy ilyen elegáns megoldás rekordok és mutatók nélkül nem valósítható meg: az ilyen lehetőség jelentik a Pascal szépségét.

## Egy kis további technika

Ha a „filmet” elkészítettük, le is játszhatjuk. Ehhez egy REPEAT—UNTIL ciklust

nálják. Ha ezeket átírjuk a korábbi cikkekben ismertetett PLOT és PixelLine eljárásokra, a programcsomag sokkal rugalmasabbá válik.

Az ismertetett algoritmus más korszerű Pascal implementációban is megvalósítható. Így például az IBM kompatibilis gépek Turbo Pascaljával majdnem sorról sorra azonos program írható. Mivel a képernyő-fájl szerkezete és a nem szabványos grafikus eljárások eltérőek, lesznek különbségek, viszont a film négy színben pompázhat, ha van színes grafikus kártyája a gépnek.



# Szellemek forgatása

**COMMODORE 64**

A szellemek (más néven sprite-ok, szellemgrafikák, MOB-ok) forgatása érdekes hatást vált ki a képernyőn. Két gépi kódú programot közlünk, melyek ezt valósítják meg: a C64 belső megszakításán, az IRQ ciklusnak az ideje alatt — azaz minden hatvanad másodpercben — a programok forgatnak egy bitnyit az adott szellemgrafikán. Így olyan hatás jön létre, mintha a szellem magától forogna a képernyőn. A forgatás bekövetkezik akkor is, ha a szellemgrafika nincs is a képernyőn, azaz ha a szellemgrafika nincs is bekapcsolva. A szellem forgatása mindaddig tart, amíg le nem állítjuk a program működését, vagy a RUN/STOP és a RESTORE nyomógombokat egyidejűleg le nem nyomjuk. Így egy BASIC program kilistáztatása vagy futtatása idején ezek a forgó szellemek a képernyőn maradhatnak függetlenül a BASIC program listázásától vagy futtatásától. A szellemek forgatása létrejön a kétszeresre felnagyított szellemgrafikák esetében is. A két gépi kódú program azonos szerkezetű, az egyikük balra, a másikuk jobbra forgatja az adott szellemet.

## Balra forgatás

A program (1. lista) a BASIC munkaterület fölött helyezkedik el, így azt a memóriából a NEW parancsral nem lehet kitörölni. Akár egy BASIC programból, akár közvetlen üzemmódból elindíthatjuk ezt a gépi kódú programot a SYS(53136) utasítással. Felhívjuk a figyelmet arra, hogy nem szabad egymás után kétszer kiadni ezt az utasítást, mert a számítógép ilyenkor „megőrül”: semmi más feladattal nem foglalkozik, csak a szellemgrafika forgatásával. Ez azért van így, mert a programot szándékosan készítettük olyanra,

```

1000 FOR I= 0 TO 53135: GOTO 53136
1010 READ X:FOR J= 1 TO 8:BE= X:NEXT J
1020 DATA 26,162,207,141,175,231,207,172
1030 DATA 232,207,141,201,5,140,21,5
1040 DATA 86,96,120,173,200,3,172,21
1050 DATA 1,141,231,207,140,232,207,167
1060 DATA 107,168,207,141,200,3,140,21
1070 DATA 3,88,96,173,233,207,172,234
1080 DATA 207,133,251,130,252,162,21,140
1090 DATA 4,179,251,1,10,166,2,177,201
1100 DATA 42,149,251,136,161,246,24,165
1110 DATA 251,195,5,133,251,144,12,230
1120 DATA 252,202,200,227,140,231,207,47
1130 DATA 234,64,13,0,0,0,0
1140 IFSP= 1:500 THEN PRINT"HEAVY LOAD"
1150 PRINT"OK"

```

## 2. lista

hogy akkor is működjön, ha az IRQ vektor nem az eredeti, a bekapcsolókor érvényben lévő címet, a SEA31-et tartalmazza. Így ez a program olyan más programokkal is hajlandó lesz együttműködni, amelyek korábban átírták az IRQ vektor eredeti értékét egy attól eltérő új címre. Ennek azonban az az ára, hogy kétszer egymás után nem szabad elindítani a programot. A forgatást leállítani már lehet többször is, a SYS(53139) utasítással.

Jelenleg a program a kezettárolóban található szellemgrafikát forgatja, amely a 832. memóriarekesztől (80340) kezdődően helyezkedik el 63 bajton. Emlékeztetül: egy sprite vagy szellemgrafika X irányban 3 bajt (azaz 24 bit) szélességű és Y irányban 21 bajt (azaz 21 képpont) képet jelent, amit X, illetve Y irányban a kétszeresére lehet felnagyítani. A szellemgrafikák a memóriában 64 bajtnyi helyet foglalnak le, amiből egy bajt kihasználatlan. A szellemgrafikákat megjelenítő VIC (Video Interface Controller) integrált

## 1. lista

```

100: CF90: .GPT F1,00
11: SPRITE (MOB) BITENKENTI
12: FORGATAS BALRA AZ IRQ ALATT.
13: BE = SYS(53136)
14: F1 = SYS(53139)
140: CF90: .+ = #CF90
150: CF90: IROVEK = #0314
170: CF90: SWR = #FB
180: CF90: KARN0 = 3
185: CF90:
190: CF90: AC A2 CF: JMP RE
210: CF93: 78: BE1
220: CF94: A5 E7 CF: LDA TEMP
230: CF97: AC E8 CF: LDA IROVEK+1
240: CF98: D0 14 03: STA IROVEK
250: CF99: BC 15 03: STY IROVEK+1
260: CFA0: 58: CLI
270: CFA1: 59: RJS
280: CFA2: 73: BE
300: CFA3: AD 14 03: LDA IROVEK
310: CFA6: 61 15 03: LDA IROVEK+1
320: CFA9: BD E7 CF: STA TEMP
330: CFAC: BC E8 CF: STY TEMP+1
340: CFAF: 69 00: LDA # 0
360: CFB1: 60 CF: LDA # 0
370: CFB3: BD 14 03: STA IROVEK
440: CFB6: BC 15 03: STY IROVEK+1
450: CFB9: 58: CLI
460: CFB4: 60
480: CFB8: A0 E9 CF: LDA MOB
490: CFB9: AC EA CF: LDA MOB+1
500: CFC1: 85 FB: STA AKT
510: CFC3: 84 FC: STY AKT+1
530: CFC5: A2 15: LDA # SWR
540: CFC7: A0 00: LDA # 0
550: CFC9: 01 FB: STA (AKT),Y
560: CFCB: 0A: ASL A
570: CFCD: A0 02: LDA # KARN0-1
580: CFCE: 03 FB: LDA (AKT),Y
590: CFDF: 2A FB: ROL A
600: CFE1: 91 FB: STA (AKT),Y
640: CF05: 80: DEY
650: CF04: 1A FB: RPL CIXLUS
670: CF06: 18: CLC
680: CF07: A5 FB: LDA AKT
690: CF09: 69 03: ABC KARN0
700: CF0B: 55 FB: STA AKT
710: CF0D: 90 02: BCC VEGE
720: CF0E: E6 FB: INC AKT+1
740: CF11: 6A: DEX
750: CF12: 00 E3: BNE FBNG
770: CF14: 6C E7 CF: JMP (TEMP)
820: CF17: 31 EA: YEMP .BYTE$1,4EA
830: CF19: 40 03: MOB .BYTE$4,3

```



áramkör csak 16 kbájtnyi memóriatartományt tud megcimezni (16 kbájt = 256 × 64 bájt), így egy egybájtos mutató segítségével a tudomására hozhatjuk, hogy a megjelenítendő szellemgrafika hányadik blokkon, azaz hányadik 64 bájtnyi memóriaterületen található. Ez esetünkben a 13. blokkot jelenti (832 = 13 × 64). Ha máshová akarjuk elhelyezni a szellemgrafikánkat, akkor az első listán a 830-as sorban található címet kell megváltoztatnunk; a cím a szokásos alsó bájt (elől), felső bájt (hátral) sorrendben van megadva a 830-as sorban. A POKE 53225, (CIM AND 255):POKE 53226,INT(CIM/256) utasítással megváltoztathatjuk a forgatandó sprite kezdőcímét; itt az előbb elmondottak értelmében a CIM-nek a 64 többszörösének kell lennie. A BASIC munkaterület megváltoztatása nélkül négy szellemgrafika helyezhető el a következő címekre: a 11. a 704-től, a 13. a 832-től, a 14. a 896-től és a 15. sprite a 960-as memóriacímről kezdődően. Ha ennél több szellemgrafikát kívánunk megjeleníteni a képernyőn, akkor például a BASIC munkaterület kezdetét feljebb kell állítanunk.

A szellemgrafikát balra forgató gépi kódú program BASIC betöltőprogramját a 2. lista tartalmazza. Az itteni 1130-as sorban található 64 és 3 (ezek a sprite címét határozzák meg: 3 × 256 + 64 = 832) számok megváltoztatásával más helyet jelölhetünk ki a forgatandó szellemgrafika számára. Felhívjuk a figyelmet arra, hogy a sprite-ok adatainak bevitelkor a forgató rutint le kell állítani, ellenkező esetben a sprite tönkremegy. Arról van ugyanis szó, hogy mialatt mi a sprite adatait, a 63 bájtot a megfelelő helyre visszük POKE utasítások segítségével, a bekapcsolt forgató rutint többször megforgatja a sprite-ot, és így nem az eredeti alakot kapjuk vissza.

A 3. lista egy BASIC programot mutat, amelyben egy nyuszit formázó sprite négy különböző irányban mozog. A program első része (1–30-as sorok) még nem tartalmaz forgatást, csak a sprite-ok mozognak a képernyőn; majd egy gomb lenyomása után az előbbi folyamat játszódik le, csak most már a sprite-ok forgatása

### 3. lista

```

1 REM NYUSZT UL A EUBEN ...
5 PRINTCHR$(147)
10 V=53248
11 POKE V+21,15:REM SPRITE BEKAPCSOLVA
12 POKE2040,13:POKE 2041,13
13 POKE2042,15:POKE 2043,13
20 FORN=0 TO 62:READR:POKE532+N,0:NEAR
25 POKEV+40,3
30 GOSUB130
40 GETA$:IFA$="" THENR40
50 SYS(53136):REM FORGATAS BEKAPCSOLVA
60 GOSUB130
70 GETA$:IFA$="" THENR70
90 SYS(53139):REM FORGATAS I.KAPCSOLVA
95 POKE V+21,0:REM SPRITE I.KAPCSOLVA
100 END
130 FORX=0 TO 200
140 FOREV,X
141 POKEV+1,150
142 POKEV+2,200-X
143 POKEV+3,200-X
144 POKEV+4,200
145 FOREV+5,X
150 FOREV+6,X
155 POKEV+7,200-X
160 NEXT:RETURN
200 DATA0,224,0,7,152,0,0,196,0,0,199,0
210 DATA2,17,0,1,139,192,0,0,0,0,7,192,0
220 DATA0,96,4,56,48,2,00,16,198,00,16
230 DATA198,80,24,4,80,56,24,59,195,40
235 DATA12,16,192,7,205,192,0,0,0,0,0
240 DATA 0,0,0,0,0,0,0,0
300 END

```

```

1000 FOR I= 52992 TO 53090
1010 READ X:POKE I,X:R=S+X:INKEY
1020 DATA 75,10,207,120,173,93,207,172
1030 DATA 94,207,141,20,3,140,21,3
1040 DATA 88,96,120,173,20,3,172,21
1050 DATA 3,141,93,207,140,94,207,169
1060 DATA 43,160,207,141,20,3,140,21
1070 DATA 3,80,96,173,95,207,172,96
1080 DATA 207,133,251,130,252,162,21,160
1090 DATA 2,172,251,74,160,0,18,40
1100 DATA 172,251,106,145,251,8,200,192
1110 DATA 3,200,204,40,24,165,201,105
1120 DATA 3,133,251,144,2,230,252,202
1130 DATA 209,251,108,93,207,49,234,64
1140 DATA 3,0,0,0
1150 IFSC=11931 THEN PRINT"ELHAS ADOT I":END
1160 PRINT"OK I"

```

### 4. lista

mellett (50–60-as sorok); ezután, ha a klaviatúrán még egy gombot megnyomunk, a sprite-ok forgatása megszűnik, és a program a sprite-okat is kikapcsolja. Közvetlen üzemmódból kapcsoljuk vissza a szellemgrafikákat a POKE V+21,15 utasítással. (Ha a BASIC programon módosítottunk éppen, akkor helyett az utasítás helyett a POKE 53269,15 utasítást kell alkalmaznunk, mert a módosítás után a V BASIC változó elveszti a korábbi értékét, az 53248-at.) Ezután indítsuk a forgatást SYS(53136)-tal, majd próbáljuk megváltoztatni a sprite méretét kézszeresre az Y irányban a POKE 53271,15 utasítással; X irányban a POKE 53277,15 utasítással növelhetjük meg a négy sprite méretét.

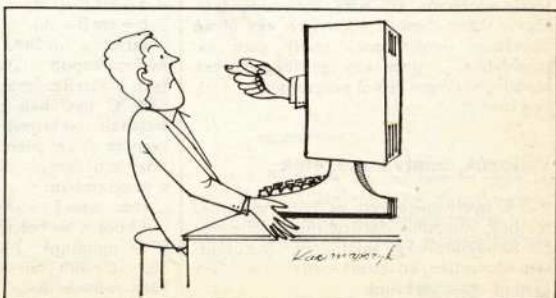
## Jobbra forgatás

A szellemgrafikákat jobbra forgató gépi kódú program BASIC betöltőprogramja a 4. listán látható. Ezt a jobbra forgató rutint a SYS(5292)-vel kell indítani és a SYS(5295)-tel leállítani. Az indításra és leállításra a korábban elmondottak érvényesek itt is. Ez a program is a 13. blokkban, a 832. tárcímről kezdődő sprite-ot forgatja; a cím alsó bájtja, a 64-es szám, a 4. lista 1130-as sorának utolsó helyén található; a sprite cím felső bájtja, a 3 pedig az 1140-es sorban az első adat. Ezeket a számokat átírva tudjuk megváltoztatni a RAM-ban a sprite helyét.

Ha a 3. listán látható programban módosítjuk az 50-es sorban a SYS(53136)-ot SYS(5292)-re, valamint a 90-es sorban a SYS(53139)-et SYS(52995)-re, és így futtatjuk a programot, akkor a sprite-ok jobbra forognak.

A következő alkalommal a nagy felbontású képernyő forgatásáról lesz szó. Addig is jó szellemforgatást kívánok!

SZABÓ PÉTER PÁL





# Mit tud a C nyelv? I.



**A C programozási nyelv megszületése óta páratlan karriert futott be. Ezt részben sajátos erőnyeinek köszönheti, részben pedig annak, hogy voltak, akik ezekkel az erőnyekkel maximiálisan éltek is.**

A C nyelv általános célú, magas szintű programozási nyelv. Ezeknek sorából leginkább azzal válik ki, hogy annyira gépközel, amennyire az egyáltalán lehetséges. Azt mondhatnánk, hogy ő a „legalacsonyabb” magas szintű nyelv. Ráílik a „magas szintű assembler” megnevezés is. Utasításkészlete rendkívül szűk, mindössze a változódeklarációkra, az értékadásra, a vezérlési szerkezetek jó megvalósítására és a függvényhívásra szorítkozik. Tehát minden megvan benne, ami az assemblerekben gépfüggetlen, „gépfelületi” lehet, és mindent kiküszöbölték belőle, ami a nyelvet egy adott géphez vagy egy bizonyos géptípushoz kötötte.

Legfontosabb erőnye talán az, hogy ismeri a pointerek, a mutatók fogalmát, és a mutatókat a lehető legnagyobb szabadsággal kezeli. Ily módon megvalósíthatjuk a legravaszabb adatszerkezeteket is, és igen szabadon kezelhetjük a rendelkezésre álló memóriát. Ez teszi a C nyelvet rendkívül hatékonyá.

Ezt használta ki 1972-ben Dennis M. Ritchie, a Bell Laboratórium munkatársa, aki nagy szerepet játszott a C nyelv létrehozásában is. Számos kiváló kollégájával együtt egy merész húzással C-ben írták meg a már 1969 óta létező Unix operációs rendszert. Ezzel minőségileg új szakasz kezdődött a számítástechnikában: jól vizsgázott egy olyan magas szintű programozási nyelv, amely megfelelő kényelmet biztosít ugyan a felhasználónak, de annyira hatékony és „assemblerszerű”, hogy lehetővé teszi operációs rendszer megvalósítását. Ugyanakkor elindult a karrierje egy olyan operációs rendszernek, amely nem assemblyben, hanem egy gépfüggetlennek mondható magas szintű programozási nyelven íródott.

## Változók, adatszerkezetek

A C nyelv megengedi egyszerű változók, tömbök, valamint adatstruktúrák definícióját. Rendelkezik egy sajátos (persze korántsem ismeretlen) adatszerkezettel is, amelyet „union”-nak nevezünk.

## Egyszerű változók

Az egyszerű változók egy-egy adat befogadására képesek. Nevük és típusuk megadásával definiálhatók:

int i;	
Ez egy integer (egész) típusú változót definiál, melynek neve i. Az általánosan elfogadott típusok:	
char	karakter
int	egész szám
long	hosszú (kétszeres pontosságú) egész
float	lebegőpontos szám
double	kétszeres pontosságú lebegőpontos szám

## Tömbök

Az egyes változókból egyszerűen képezhetünk tömböt:

```
int arr [20];
```

ahol a tömb neve arr, elemeinek száma 20 (az indexek 0-tól 19-ig futnak, tehát valóban húsz elemük van), és minden egyes tömbelem típusa egész.

A tömbök használata során az indexek helyén szabadon választhatunk tetszőleges kifejezéseket. De a mi dolgunk arról gondoskodni, hogy az így kapott index ne haladja meg a tömb elemeinek számát. Például:

```
int arr [20]; (* tömb definíciója *)
int i; (* indexváltozó *)
int a, b; (* két munkaváltozó *)
```

esetén hibás lesz a következő két utasítás:

```
a = arr [-2];
a = arr [21];
```

mivel a fordító itt ellenőrzi (képes ellenőrizni) az indexhatárokat. Minden további nélkül leírható és végrehajtható azonban az alábbi utasítássorozat:

```
i = 10;
a = arr [i - 16];
b = arr [i + 10];
```

mivel a fordító, nem ismervén i futás közben kapott értékét, nem is képes semmilyen határellenőrzésre.

A C nyelvben talán a leggyakrabban használt tömbtípus a karakteres tömb, hiszen az az egyetlen ésszerű lehetőség arra, hogy sztringeket, szövegeket helyezünk el a programban:

```
char mess [] = „Egy egy üzenet\n”;
```

Ebből a sorból több érdekes tanulságot is levonhatunk. Először is: egy változót vagy tömböt szintaktikailag elég egyszerűen tudunk inicializálni, azaz kezdeti ér-

tékkel ellátni. Másodsor: idézőjelek között adhatunk meg szövegeket. Harmadszor: a tömb előkészítésekor nem kötelező megadni, hány eleme van a tömbnek — a fordító meghatározza helyettünk. Például ennek a tömbnek 16 eleme van. És itt következik a negyedik és ötödik fontos dolog. A tömbnek nem azért van 16 eleme, amiért első pillantásra gondolnánk. A \n karakter-sorozat az üzenet végén a valóságban nem két, hanem csak egy karaktert ér: ez a karakter a newline, az újsor, amelynek hatására az üzenet kinyomtatása után a terminálon egy kocsvissza-soremelés következik be. (A \ karakter egy ún. ESCAPE (váltó), melynek hatására a fordító elvonatkoztat az utána következő karakter konkrét értékétől, és különleges értelemben használja fel — például az n karakter helyett kocsvissza-soremelés.) Mi akkor hát a 16. karakter, hiszen így csak 15-öt adtunk meg! Nos, az utolsó megadott karakter, az újsor után a fordító még odatesz egy 0 kódú karaktert is: ez jelzi majd a memóriában a sztring végét. Az ötödik fontos apróság tehát: a sztringek utolsó karaktere mindig a 0 kódú karakter. Programunk innen ismerheti fel a sztring kezelése közben, hogy vége a szövegnek.

## Struktúrák

A C nyelv tetszőleges szerkezetű adatstruktúrák definícióját teszi lehetővé. Ezek segítségével nagyon pontosan és tömören lehet programozni. Lássunk egy példát utalva a struktúrák egy felhasználási lehetőségére is! Tegyük fel, hogy olyan játékprogramot akarunk írni, amely több tárgyat mozgat egymástól függetlenül a képernyőn. Az egyszerűség kedvéért a tárgyak mindegyike egy-egy karakter, amelyek egyes vonalú egyenletes mozgást végeznek a képernyőn. Az ilyen tárgyak leírására kiválóan alkalmas a következő struktúra:

```
struct flyobj {
    char code; (* a karakter kódja *)
    int xpos; (* pillanatnyi pozíció *)
    int ypos; (* koordinátái *)
    int xstep; (* lépésvektor *)
    int ystep; (* koordinátái *)
};
```

Mint látható, ez a struktúra mindent összefoglal a tárgy mozgathatóságához szükséges adatokból: a karakter kódját, a tárgy pillanatnyi pozíciójának koordinátáit, valamint a lépésvektort.



Magában véve a fenti néhány or nem hoz létre a memóriában egy ilyen szerkezetű struktúrát. Ezzel pusztán a flyobj nevű struktúra belső felépítését adtuk meg. Egy ilyen struktúra számára a következő utasítással foglalhatunk helyet:

```
struct flyobj fly1;

```

ahol az első két szó tulajdonképpen a típus („legyen egy flyobj típusú struktúra”), fly1 pedig a memóriaterület neve. Hogyan lehet ezt a struktúrát használni? Sajnos a C nyelv nem teszi lehetővé a struktúrák globális használatát. Nincs tehát mód arra, hogy egy struktúrát egy utasítással átmásoljuk egy azonos szerkezetű másik struktúra területére, nem írhatunk csupa 0-t minden egyes struktúrátag helyére stb. A struktúra tagjait külön-külön kell kezelnünk:

```
fly1.code = '*' (* csillag repül *)

```

A struktúra tagjai külön-külön változók:

```
fly.xpos = fly.xpos + fly.xstep;
fly.ypos = fly.ypos + fly.ystep;

```

Ez az utasítássorozat kiszámítja a repülő tárgy következő pozícióját. Látható, hogy a struktúra tagjaival ugyanúgy bánhatunk, mint az egyes változókkal.

Programunkban természetesen tetszőleges számban definiálhatunk flyobj típusú struktúrákat. Ilyenkor az egyes változókat a kívánt struktúra és a kívánt tag neve azonosítja:

```
struct flyobj fly1, fly2, fly3;
fly1.code = '*'; (* első tárgy kódja *)
fly2.code = '#'; (* második tárgy kódja *)
fly3.xpos = 12;

```

A struktúrák tagjai nemcsak egyszerű változók, de tömbök, sőt újabb struktúrák is lehetnek:

```
struct coord {
int xcoord;
int ycoord;
};

```

```
struct flyobj {
char code;
struct coord pos;
struct coord step;
};

```

Foglaljunk területet egy ilyen struktúra számára:

```
struct flyobj fly;

```

Ekkor persze bonyolultabb lesz a hivatkozás:

```
fly.pos.xcoord = fly.pos.xcoord + fly.pos.xstep;

```

hiszen egymásba ágyazott struktúrák tagjaira kell hivatkoznunk.

## Unionok

A union a C nyelv egyik érdekes újdonsága. Definiálása és felhasználása is nagyon hasonlít a struktúráéhoz:

```
union alltyp {
char ch;
int in;
long lg;
float fl;
double db;
};

```

(\* union-deklaráció \*)

union alltyp all; (\* union-definíció \*)

A union egy tagját a következőképpen használhatjuk fel:

```
all.ch = 'a'; (* egy karakter beírása *)

```

A union és a struktúra között az a különbség, hogy a struktúra egyes tagjai egymás után helyezkednek el a tárban, a union tagjai pedig mind egyazon címen kezdődnek. Ha tehát a fenti utasítással felülírom a karakteres uniontagot, akkor egyúttal „belepiszkáltam” a union összes többi tagjába. Felülírtuk az egész típusú „in” alsó felét, a long típusú „lg” negyedét és így tovább.

A fenti union belső felépítését és egyben a union lényegét mutatja az *ábra*. Az egyes elemek nem egymás mögé, hanem egymásra íródnak — tréfás meglepetéseket tartogatva a tájékozatlan programozóknak, és kiváló lehetőséget nyújtva különböző típusok fizikai konverziójára:

```
union dbtost {
double db;
char st [8];
};

```

Használjuk most fel ezt a union-deklarációt:

```
union dbtost dbst; (* double → sztring *)
dbst.db = 3.14159; (* be mint double *)
putchar (dbst.st [3]); (* 3. bájtnyomatása *)

```

Egy kétszeres pontosságú lebegőpontos változót tudunk elhelyezni, és ugyanezt az értéket bájtkokra bontva kaphatjuk meg, mert a union másik oldala egy nyolc karakteres tömb — ennek elemei pedig az egyes bájtok. Ugyanez a konverzió persze visszafelé is elvégezhető.

Természetes, hogy a union is tartalmazhat tömböt és struktúrát is.

## Eljárások

A C nyelv eljárásait függvényeknek nevezzük, melyeknek tetszőlegesen sok bemenő paraméterük lehet, viszont csak egy értéket adhatnak vissza a hívójuknak.

A C program függvényekből áll. Minden függvény egyenrangú, azaz nem definiálhatók egymás belsejében, mint például a PL/I nyelvben. Egyenrangúságuk azt is jelenti, hogy bármelyik függvény meghívhatja bármelyiket, beleértve saját magát is — a nyelv lehetővé teszi a rekurzív függvények használatát.

Egyetlen kitüntetett függvény mégis akad, az „main”, azaz a „fő” függvény. A program indulásakor a végrehajtás a main első utasításával kezdődik és annak utolsó utasításával ér véget. A main belsejéből hívhatjuk az összes többit — saját függvényeinket éppúgy, mint az ún. könyvtári függvényeket. Lássunk most egy igazi C programot:

```
main()
{
printf("Szervusz, kedves Olvasóm\n");
}

```

A program meghívja a printf nevű könyvtári függvényt, és ennek segítségével kinyomatja az időzőjelek közé zárt szöveget — beleértve az újsor karaktert is. Láthatjuk, hogy egy C függvényt egyszerűen nevének leírásával hívhatunk, a függvény neve után zárójelben megadva a kívánt paramétereket. A paraméterek pedig akár konstansok, akár változók, akár ezekből képzett kifejezések lehetnek. Sőt megengedett az is, hogy egy függvény értéke szolgáljon paraméterként egy másik függvény számára.

Hogyan kell definiálni egy függvényt? Lássunk erre is egy példát:

```
add(a, b)
int a;
int b;
{
int sum;
sum = a + b;
return(sum);
}

```

Ez a függvény két paramétert vesz át, melyek, mint deklarációjuk mutatja, egész típusúak. Az összeadás után a return() utasítás hatására adja vissza paramétereinek összegét.



Sorozatunkban azokat az új hardver- és szoftvertermékeket ismertetjük, amelyek várhatóan általánosan elterjednek, és meghatározó szerepük lesz a fejlődés irányainak kialakításában.

Merre tart a világ?

# Nyomtatók és néhány más periféria

A nyomtatók fejlődésének bemutatása előtt egy itthon ismeretlen perifériáról és a botkormányok/egerek újdonságairól szólnunk.

## Dialemez-készítő

Valószínűleg csodálkozik az olvasó, hogy mit keres itt ez a téma? Mi köze a számítástechnikához, hogy kerül ebbe a sorozatba? E kérdésekre az 1. kép alapján adhatunk választ, amelyet a General Parametrics cég állított elő.

A képen látható, hogy a dialemez- (slide-) gyártás az Egyesült Államokban 1980-ban meghaladta az egymilliárd darabot és 1990-re mintegy 4,5 milliárd lesz. Az eredetileg 8 színű diát egy IBM PC AT-ből (512 k RAM), egy PhotoMetric 200 diakészítő eszközből, egy ColorMetric 200 kártyából (az utóbbiak a General Parametrics gyártmányai) és hagyományos grafikus szoftverből álló rendszeren készítették. Ez a rendszer olyan ismert szoftverek használatát teszi lehetővé, mint például a Lotus cég Freelance, az Ashton-Tate ChartMaster vagy a Microsoft Charta nevű terméke. A felbontás 2000 x 2000, a színválaszték 1000 szín. A lemezkészítés előtt a végleges formára szerkesztett kép bármely megfelelő felbontású monitoron ellenőrizhető. A rendszer fekete-fehér lemezt is elő tud állítani; ilyenkor 14 szűrőeségi fokozattal dolgozik. Olyan szoftvert is árulnak hozzá, amellyel nyomtatón is megjeleníthető a lemezek, vagy lemez nélkül közvetlenül vetíthető.

## Botkormányok, egerek

Egereket készítenek 1-3 nyomógombbal, közvetlenül a számítógép soros bemenetére vagy egy közbeiktatott miniatűr billentyűzetre alkalmas csatlakozással. Botkormányokat szabadon definiálható funkciójú nyomógombbal, vezérlő golyó (track ball)-nyomógomb kombinációval és berendezések kezelőlapjára szerelhető formában árulnak. Az óriási választékból az egy kis izelítőt a 2. kép. Mindez azt mutatja, hogy e

bemeneti eszközök gyártása virágkorát éli, de újnak számító megoldások nincsenek. A fej-, szem-, karmozgás érzékelésén alapuló készülékek még nem jutottak el a kiállításokra.

A CH Products MICROSTICK sorozatának egy új típusa (3. kép) sokoldalúságában különbözik a hasonlóktól. Ez a botkormány nyolcféle, kapcsolókkal kiválasztható, többek között a digitalizálókat és az egereket emuláló (utánzó) üzemmódban dolgozhat. Háromféle adatkiadási módja van. Felbontóképessége olyan nagy — irányonként 4096 pont —, hogy lehetővé teszi a LOTUS 1, 2, 3, a dBASE III, a WordStar,

1. kép



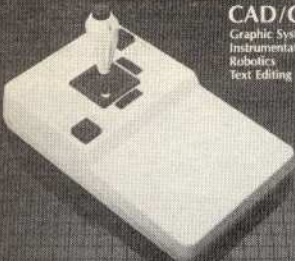
2. kép





## CAD/CAM

Graphic Systems  
Instrumentation Systems  
Robotics  
Text Editing



a Multiplan, a VisiCalc stb. egérral történő menüvezérlését. A szokásos sebességekkel (300—19 200 baud) és formátumokkal dolgozik: a szoftverrendszerhez közvetlenül, speciális működtető szoftver nélkül kapcsolható.

## Nyomatók

A Seikosha cég egyes termékei — a Sinclair és Commodore gépekhez használt ol-

3. kép



nált, széles körben elterjedt ultrahangos képszlűk előállításra is. Ez a vállalat volt az első japán mátrixnyomtató gyártója, és a világon elsőként készített videoprintert (képernyőnyomtatót). A Seikosha kezdte el a 420 karakter/s, 800 karakter/s sebességű mátrixnyomtatók gyártását.

Újdonságai közül elsőként az itthon is ismert SP—180 sorozat legújabb és sok paraméterében nagy előrelépést mutató tagját, az SP—185AI típust mutatjuk be (4. kép). A 100 karakter/s sebesség és a széles papírról miatt elsősorban a gyors telexgépekhez alkalmas. A beépített Epson FX és IBM Graphics Printer emulátor lehetővé teszi, hogy a legismertebb grafikus programok használhassák. Csendes üzeme miatt jól használható olyan irodákban is, ahol sok nyomtató dolgozik egyszerre. Ára hasonló sorozatának korábbi tagjaiéhoz.

Az SBP—10 (5. kép) 18 tús, 800 karakter/s sebességű, 64 k pufferű, mindenféle típusú papíradagolásra alkalmas nyomtató. Soros/párhuzamos illesztője, kijelzője van, a papírt előre-hátra továbbítja, a betűkészlet kazettárral tudja betölteni. Képes emulálni az Epson ESC/P és az IBM Proprinter nyomtatókat. E jellemzők bizonyítják, hogy rendkívüli teljesítményű nyomtató. Ennek azonban ára van, igen magas ára: kb. kétszer annyiba kerül, mint egy átlagos lézernyomtató. Elsősorban LAN típusú alkalmazásokhoz ajánlják.

A VP—3500 videonyomtató (6. kép) a 300 pont/hüvelyk (kb. 12 pont milliméterenként) felbontási kategóriába tartozik. Ezzel képernyőnként 1280 x 1250 pontos felbontás nyomtatható ki, 54 szűrkeségi fokozatban. Így mind alfanumerikus, mind grafikus üzemmódban használható nagy felbontású monitorokhoz. Tévékészülékekhez, videokamerákhoz, meteorológiai műholdak által szolgáltatott térképek készítéséhez, elektronmikroszkópos felvételek ki-nyomatására, röntgenfelvételek feldolgozására, ipari folyamatgenerálásra, elektronikus áramkör tervezésére, fényképek elektronikus továbbítására alkalmazható.

A BP—5420AI típus nálunk is elérhető ári, nagy sebességű (420 karakter másodpercenként), széles kocsis, nagy igénybevételre tervezett nyomtató. 2 év garancia van rá. Ára mintegy kétszerese az itthon elterjedt, de nem ebbe a kategóriába tartozó Epson FX—1000-ének. A konstrukció robusztus kivitelű — nem műanyagból, hanem fémből készült, ezért tömege 32 kg —, jól tagolt önálló részegységekből áll, amelyek összetéveszthetetlen csatlakozókkal kapcsolódnak, könnyen cserélhetők, így a szervizelés egyszerű. Az egyes egységek a főbb funkciókat megvalósító egységeket elkülönítve tartalmazzák, ezért a hibakeresés és a hibás részek cseréje nem okoz különösebb gondot. A pufferkapacitás 18 kbájt. A nyomtatónak 8 nyelvű betűkészlete, IBM és Epson üzemmód-emulátora, soros/párhuzamos illesztője van, zajszintje alacsony.

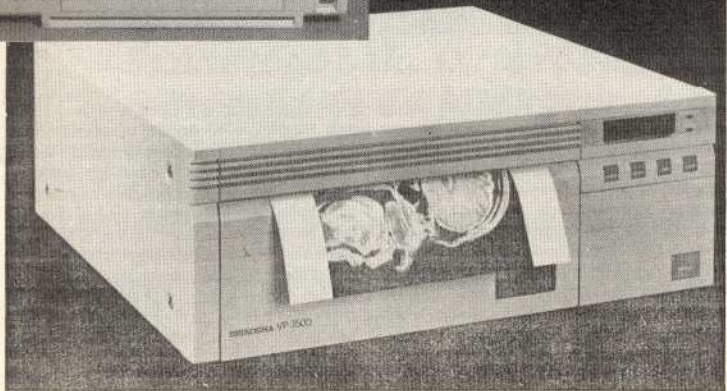
A Sony céget elsősorban hangtechnikai berendezéseiről, kevésbé monitorairól ismerjük, de szinte semmit nem tudunk le-

4. kép

csőbb nyomtatók — itthon is széles körben ismertek, de nem ezek a cég legfontosabb gyártmányai. Azt például nálunk is tudják néhányan, hogy leányvállalata a Seiko nevű, digitális óráiról ismert cég, de arról már valószínűleg csak keveseknek van tudomása, hogy az első 8 bites mikroprocesszort, az 18080 típust a Seikosha az Intel céggel együtt dolgozta ki, vagy hogy nevéhez fűződik a foggyökérszatórnak kezelésére hasz-

5. kép

6. kép



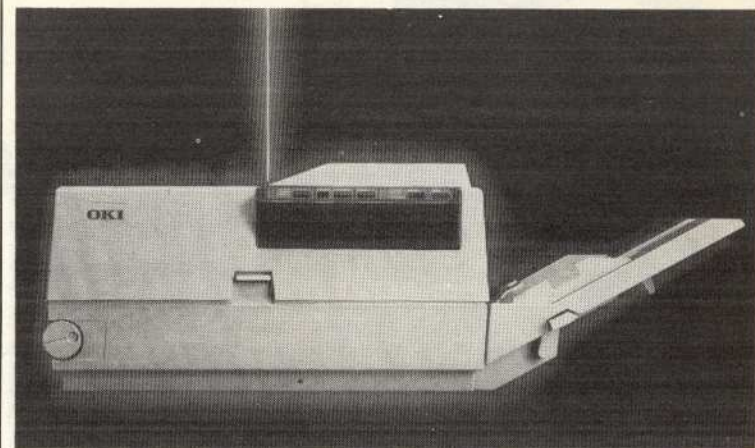


mezgyártásról. Ez annál is inkább meglepő, mivel ez a cég volt a 3,5" méretű lemezek és lemezegységek kidolgozója, elterjesztője. Ma a korábbi 5,25" méretű egy- és kétoldalas, egyszeres és kétszeres sűrűségű lemezeken kívül egyoldalas 0,5 Mbájt (formattálva 360 kbájt) kapacitású, kétoldalas és kétoldalas nagy sűrűségű, 3,5" méretű lemezeket is gyárt. A Sony a 3,5" méretű lemezeket tekintve a világ legnagyobb gyártója. Az új, nagy sűrűségű lemezek mágneses anyaga vas-oxidban abszorbeált kobalt. A bevonat vastagsága fele a korábbi 2 mikronnak. A cég lemezváltásztéka a 7. képen látható. A lemezársakat az ithoniakkal összehasonlítva kb. 500 forint/dollár értéket kapunk, ami igazán megdöbbentő!

Magyarországon a Sony nyomtatók is ismeretlenek, pedig a cég ezen a területen a szuper minőségre specializálta magát. Az UP-2000 színes, grafikus nyomtató ún. WYSIWYG („amit-látsz-azt-kapod”) típusú képernyőnyomtató. Az IBM PC piacon általánosan elterjedt EGA kártyák felbontásával kompatibilis. Clem Shemanski marketingigazgató ettől a terméktől azt várja, hogy új szabványt teremt. Érdekessége,

Az Okidata céget ML sorozata és olcsó színes nyomtatója, az Okimate révén ismerjük. Itt most a LASERLINE 6 PLUS típusú lézernyomtatóját mutatjuk be (8. kép). A készülék egy mintanyomatása látható a 9. képen. A száraz elektrografikus lézertech-

nikus nyomtatók más termékeiről ismert Atari ún. íróasztali kiadói rendszere (desktop publishing) egy Mega 2 típusú számítógépből, monokrom monitorból és az SLM804 típusú lézernyomtatóból áll. Ara a működtető szoftverrel együtt a szenzáció-



8. kép

**ABCDEFGHIJKLMNabcdefghijklmn1234567890**

ABCDEFGHIJKLMNabcdefghijklmn1234567890  
 ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz  
**ABCDEFGHIJKLMNabcdefghijklmn1234567890**  
 ABCDEFGHIJKLMNabcdefghijklmn1234567890

ABCDEFGHIJKLMNabcdefghijklmn1234567890  
**ABCDEFGHIJKLMNabcdefghijklmn1234567890**  
 ABCDEFGHIJKLMNabcdefghijklmn1234567890  
 ABCDEFGHIJKLMNabcdefghijklmn1234567890  
 ABCDEFGHIJKLMNabcdefghijklmn1234567890

ABCDEFGHIJKLMNabcdefghijklmn1234567890  
 ABCDEFGHIJKLMNabcdefghijklmn1234567890

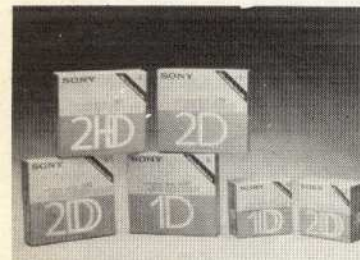
ABCDEFGHIJKLMNabcdefghijklmn1234567890  
 ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

9. kép

nológia a fénymásolókhöz hasonlóan egy dobból választ le festékszemszékét a lézertényesugár hatására. A nyomtató kompatibilis a HP Laser Jet típusú, és számos betűkeres nyomtatót is képes emulálni. Nyomatási teljesítménye 6 oldal percenként. A felbontás a szokásos 300 pont/hüvelyk. A lapokon lévő információ tárolására 128 kbájt memória áll rendelkezésre, de 512 kbájtá bővíthető. 15 ún. nemzeti karakterkészlet a tartozék, ezenkívül további 53 a bővíthető lehetőség. A tervezett élettartam évi 36 ezer oldal esetén 5 év.

nak számít 4000 dollár. A nyomtató az előzőhöz hasonló felbontás mellett nyolc oldalt készít percenként. Ezt úgy érték el, hogy a számítógép processzora és memóriája közvetlenül a nyomtató vezérlő elektronikájára dolgozik. Az átvitel egy DMA csatornán 10 Mbit/s sebességgel történik, közvetlenül a képprocesszorból. A karakterkészlet nem a nyomtató ROM egységében tárolódik, hanem szoftver állítja elő a számítógépből. A nyomtatót 2000 dollárért külön is árulják.

Simonyi



7. kép

hogy az RGB monitorkimenetre csatlakoztatható, vagyis a monitor helyére kerül. A meghajtó szoftverfüggetlen és a 640 x 350 pontos képernyőfelbontás mellett 77 x 102 mm méretű nyomtatványt ad. A 64 színű képet négy menetben állítja elő, 40 másodperc alatt. A nyomtatás elve villamosan vezet tinta használatán alapul. Egy festékkazetta és a speciális papír egy tekerce 100 kép készítésére elég. Az UP-2000 ára valamivel magasabb, mint egy átlagos lézernyomtatóé.

A nálunk egyelőre még ismeretlen, egyesült államokbeli Silver Reed új terméke az SP-30 típusú képkészítő nyomtató (scanner printer), amely két, egymástól függetlenül is működtethető funkciót lát el: a képkészítést és a képkinyomtatást. A készüléket egy párhuzamos csatornán keresztül lehet a számítógéphez csatlakoztatni. Felvétel üzemmódban a felbontás 8 pont/mm vízszintesen és 7,7 pont/mm függőlegesen. Az olvasási sebesség igen nagy: 4,2 ms soronként. A nyomtató üzemmód felbontása hasonló, a sorszálesség 208 mm. A nyomtatás hőpapírra készül. A készülék ára 700 dollár.



### 1. MIKROMAX hálózati kapcsolóüzemű tápegységcsalád

Készül  $\Sigma$  55 W kimeneti terhelhetőséggel háromkimenetű változatban:

1. 5 V/10 A és  $2 \times 12$  V/1,5 A
2. 5 V/10 A és  $2 \times 15$  V/1,5 A

valamint 90 W kimeneti terhelhetőséggel egykimenetű változatban:

1. 5 V/18 A
2. 12 V/7,5 A
3. 15 V/6 A

EURO-rackbe szerelhető, de beépítés nélkül is használható.

### 2. TR-80 mikroprocesszoros kártyarendszer

Intel 8085 buszrendszer, ESZR méretű kártyák:

- központiegység-modulok (I8085, aritmetikai processzor)
- memóriamodulok (statikus, dinamikus RAM, tápfeszültségvédett RAM, EPROM)
- interfész modulok (soros, párhuzamos stb.)
- A/D, D/A és analóg-multiplexer modulok
- színes grafikus display-illesztő
- EPROM-égető és hardverellenőrző pult

Gyártó:

**KONTAKTA Alkatrészgyár**  
Ózdi Gyáregység  
3600 Ózd,  
Bolyki főút 82.  
Tel.: 06-47-12555

Felvilágosítás:

**KONTAKTA Alkatrészgyár**  
Fejlesztési Intézet  
Elektronikus Fejlesztés  
Budapest  
Tel.: 279-200/279

**Tekintse meg egyéb új termékeinket is a Tavaszli BNV „A” pavilon 103/C standján!**

## Számítástechnikai Informatikai Fejlesztő Vállalat



**Kereskedelmi Irodájának vezetésére**

- felsőfokú iskolai végzettséggel,
- számítástechnikai ismerettel és kereskedelmi gyakorlattal,
- legalább öt év vezetői gyakorlattal,
- nyelvismerettel

**rendelkező munkatársat keres.**

**Jelentkezés:**

Marjai Sándor gazdálkodásigazgató-helyettesnél, a 353-580 vagy a 350-140/103-as telefonszámon.

**A BNV-n a holnap technikájával várjuk kedves érdeklődőinket!**

**A 28-AS PAVILONBAN**

- Home computertől PPC-ig
- Hardver, szoftver
- Számítógépek, tartozékok, kiegészítők felhasználói és játékprogramok, valamint számítástechnikai szakkönyvek.

Vállalkozunk kulcsrakész rendszerek kidolgozására

- helyzetteleméréstől és célkitűzéstől kezdve
- szervezésen, programozáson, dokumentáláson keresztül
- a rendszer betanítására, rendszerkövetésre.

**NOVOTRADE Rt.**

1136 BUDAPEST,  
Fürst S. u. 24-26.  
Tel.: 530-022



**DIGITAL**  
szelvény  
Mikroszámítógép  
Magazin  
1988. május

# Egy sarokkal olcsóbb!!!

A DIGITAL  
Számítástechnikai Szaküzlet (Budapest,  
Szilágyi Erzsébet fasor 35. 1026) Sinclair-termékekre  
szakosodott: elsősorban a ZX81-es  
és Spectrum gépekhez használatos eszközöket,  
programokat árul. De kaphatók itt más gépekhez  
való tartozékok (például botkormányok), számítástechnikai  
alkatrészek (integrált áramkörök stb.) és zsebszámológépek is.

Aki ebben az üzletben a lapunkból kivágott sarokszelvényt át-  
adja, vagy megrendelésével együtt oda elküldi, minden hónap-  
ban más-más cikket olcsóbban vásárolhat meg. A kedvezmény  
a szelvényen feltüntetett hónapban érvényes. Minden árenged-  
ménnyel vásárolt darabhoz le kell adni egy szelvényt.  
A bolt utánvétellel szállítást is vállal, és a szokásos 6 hónap he-  
lyett **1 év garanciát ad.**

## Az e havi kedvezmény

**Egy botkormányos illesztő. Ára  
1500,— Ft, engedmény 20%.  
Intelligens botkormányillesztő.  
Ára 2500,— Ft, engedmény  
24%.**

Az egy botkormányos illesztő a Spect-  
rum gépek buszcsatlakozójára köthető  
kapcsolós (ugyanitt is kapható) botkor-  
mányok illesztésére szolgál. Ez megfelel  
a széles körben elterjedt és szabvány-

nak mondható Kempstone típusnak. Bár  
sok játékot billentyűzetről is lehet játszani,  
de a botkormány használata kényel-  
mesebb, és kíméli a billentyűzetet.

Az intelligens változat programozható,  
vagyis a számítógép beállított gombjait,  
mint irányvezérlőket használhatja. A tan-  
ulási kapcsoló helyzetben ezek a gomb-  
ok a számítógépen kijelölhetők. Így bil-  
lentyűzetvezérlésre készített programok  
is vezérelhetők botkormánnyal, nincs  
szükség a különböző típusú illesztőt  
igénylő játékokhoz más és más illesz-  
tőre.



## ISKOLASZÁMÍTÓGÉP- SZERVIZ

1077 Budapest VII.,  
Baross tér 19.  
Telefon: 428-999

**Vállalja:**  
**IBM PC/AT,**  
**IBM PC/XT**  
**és**  
**Commodore**  
**típusú**  
**(C16, C Plus/4,**  
**C64, C128)**  
**gépek**  
**javítását,**  
**átalánydíjas**  
**szervizét,**  
**egyedi**  
**programok,**  
**program-**  
**csomagok**  
**készítését.**

Minden kedden 17-től 20 óráig  
ENTERPRISE klub  
a VSZM  
Közösségi Házban  
(Bp. XI., Fehérvári út 120.)  
Klubvezető: Romvári Gábor  
Telefon: 450-950/473



# Közprogramok



**O**lvasóink egyre gyakrabban küldenek be olyan hasznos programokat, amelyeket terjedelmi okok miatt nem áll módunkban közölni. Mivel azonban úgy véljük, hogy közérdeklődésre tarthatnak számot, ezeket röviden ismertetjük. Akik használni szeretnék a programokat, a részletes leírást, a programlistákat vagy – ahol erre mód van – a programot adathordozón, levélben megrendelhetik szerkesztőségünkötől. A listák, ill. adathordozók másolását a KASZKAD Kiszövetkezet öbudai PÓLUS Szakcsoportja végzi (lásd lapunk 1988/2. számát). A másolási díj listák, leírások esetén oldalanként 8,— Ft, kazetta esetén 150,— Ft, floppy esetén 300,— Ft. Az adathordozóra történő másolás díja az adathordozó árát is magában foglalja.

Közületektől cégszerű megrendelést kérünk, a másolási díjat az MNB 208—42518—7014 számlára kell befizetni. Magánszemélyek a díjat a KKVMF PÓLUS Szakcsoport, Bp. Bécsi út 94—96. 1034 címre fizethetik be.

A megrendelés az alábbi formában lehetséges:

## **MIKROSZÁMÍTÓGÉP MAGAZIN SZERKESZTŐSÉGE** 1371 Budapest, Pf. 433.

Megrendelem a Mikroszámítógép Magazin 1988/... számában szereplő közprogramok közül az alábbiakat:

**A program neve**

**Csak programlistát és leírást kérek**

**Példányszám**

**Csak adathordozót kérek**

**Mindkettőt kérem**

A megrendeléshez csatolom a szolgáltatási díj befizetését igazoló csekkszelvényt. Dátum, név, pontos cím.

A programok ellenőrzése nem áll módunkban, ezért az esetleges programhibáért mi nem vállalhatjuk a felelősséget.

Kérjük, akik saját készítésű programjaikat felajánlják a köz javára, a részletes programleírást, a listát és az alábbiakhoz hasonló rövidített ismertetőt juttassák el címünkre. (Nevük jelenti a garanciát a továbbiakban a megrendelők számára.)

## **Úszóverseny**

Programnév: Furst Kupa Úszóverseny  
Géptípus: C16  
Konfiguráció: nyomtató  
Adathordozó: nincs megadva  
Terjedelem: 260 BASIC sor  
A készítő neve: Bőczén Csaba  
Megjegyzés: a gép bővítő használatát nem igényli

A leírás  
oldalszáma: 3  
A program  
oldalszáma: 5

A program a VIII. osztályos tanulók számára is követhető és megérthető formában készült. Célja a programkészítés oktatásán kívül a verseny gyors, egyszerű adminisztrációja. A verseny lehet egyéni, csapat és váltó. A pontozás alapján rangsorolják a résztvevő iskolákat. A gép tármérete miatt iskolánként legfeljebb 50 tanulót és összesen legfeljebb 20 iskolát képes értékelni. A program az adatbeadási hibák ellen védelemmel rendelkezik.

## **Tanterv**

Programnév: Számítástechnikai tanmenet (kezdő)  
Géptípus: C16, Plus/4  
Konfiguráció: alapgép  
Adathordozó: leírás papíron  
Terjedelem: nem számítógépes program  
A készítő neve: Bőczén Csaba  
Megjegyzés: egy 52 óra terjedelmű tanmenet leírása

A leírás  
oldalszáma: 3  
A program  
oldalszáma: —

A tanmenet, amely általános iskolák részére készült — az ismerkedés a számítógéppel, gép—ember-kapcsolat, BASIC programozási ismeretek témaköröket öleli fel —, a tananyag rövid megnevezését, a bemutatandó feladatokat, a begyakorlandó feladatokat és az esetleges megjegyzéseket tartalmazza. Mindezekhez megadja a szükséges óraszámokat.

## **Játékprogram-nyilvántartó**

Programnév: Játékprogram-nyilvántartó program  
Géptípus: C64  
Konfiguráció: VC-1541 vagy más kompatibilis lemezegység  
Adathordozó: lemez  
Terjedelem: 193 BASIC sor  
A készítő neve: Nagy Ervin  
Megjegyzés: képernyő vagy nyomtató egyaránt használható

A leírás  
oldalszáma: 3  
A program  
oldalszáma: 3

A programnyilvántartó program 999, lemezen tárolt program lemezen történő nyilvántartására szolgál. Az állomány bővíthető, módosítható, rendezhető, kiiratható, legfeljebb 33 karakterrel kommentálható.



## Hardver

### Kombinációs és sorrendi hálózatok

A digitális áramköröket alpműködés szempontjából két nagy csoportra oszthatjuk. Az egyiket az ún. kombinációs hálózatok alkotják, melyek jellemzője, hogy valamely időpontbeli bemenő értékek egyértelműen meghatározzák a hálózat kimeneti értékeit. Ilyenek voltak az eddig tárgyalt kapuáramkörök — hiszen a bemenetiek egyértelműen meghatározták a kimenetük logikai állapotát —, valamint a kapuáramkörök összekötéséből kialakított összetettebb hálózatok. A másik csoportot a sorrendi vagy más néven szekvenciális hálózatok képezik. Ezekre az jellemző, hogy kimeneteiket a bemenő jelek mellett a hálózat előző állapota is befolyásolja. A bemenő változók azonos kombinációihoz más-más kimenet tartozhat, attól függően, hogy mi volt a hálózat előző állapota. Ezért nevezzük ezt sorrendi hálózatnak. Mivel a hálózat előző állapota befolyásolja az aktuális állapotot, szükség van az állapotok tárolására, mert ezek a tárolt állapotok — logikai kapcsolatban a bemenő jelekkel — határozzák meg a kimenet(ek) állapotát.

A tárolás elemei a tárolók vagy közismert idegen szóval flip-flopok. A tárolók kapukból is előállíthatók, azok megfelelő összekapcsolásával. Példaként vizsgáljuk meg, hogyan működik az egyik bemenetűn invertert tartalmazó VAGY (OR) kapukból kialakított RS tároló (1. ábra).

A további rajzokon gyakran látunk a kivezetéseken egy számot. Ennek az az oka, hogy a rajzok egy áramkörtervező programmal készültek, és a lábszámok mindig egy adott konkrét típus lábkiosztását jelentik. Néhány megjegyzés a rajzokkal kapcsolatban. A kapukat vagy más logikai elemeket téglalapok jelzik. A közvetlen be- és kimenetek a téglalap oldalához húzott vonalak, általában olyan módon, hogy a bemenetek az egyik, a kimenetek a másik oldalra kerülnek. Az inverz (negált) bemenetek vagy kimenetek kis üres körrel csatlakoznak az oldalakhoz, és azt jelölik, hogy a kör másik oldalán a jel inverze jelenik meg.

Adjunk az S bemenetre H szintű, az R bemenetre L szintű jeleket. Az A jelű kapu kimenetén H szint jelenik meg. Ez a B kapu egyik bemenetére is kapcsolódik. Ez, il-

**A sorozat alappondolata — azon a régi felismerésen túl, hogy az elektronika és a számítástechnika elválaszthatatlan egymástól — a következő tapasztalatot summázza. A szoftver — a programok — jelentősége egyre nő, de az is tény, hogy az igazán jó (az adott számítógép nyújtotta lehetőségeket maximálisan kihasználó) programok megírásához a programozónak rendelkeznie kell alapfokú áramköri hardverismerettel. Megerősíti ezt, hogy szaporodik az olyan berendezések, mikroprocesszorok alkalmazó rendszerek száma, amelyek programvezérelten működnek. Az ilyen rendszerek tervezőinek és fejlesztőinek is szükségük van integrált hardver- és szoftverismeretekre.**

lve az inverze és az R bemeneten lévő L szint együttesen a kapu kimenetét L szintbe állítja. Láthatjuk tehát, hogy van egy hurok, amely a kapcsolás egyik felében stabilan H szintet, a másik felében L szintet tart fenn. Az S oldali Q kimeneten a H szint ezért akkor is megmarad, ha az S bemenetre adott H szintű jel megszűnik, mindaddig, amíg az R bemenet L szintű marad. Analóg módon a két kimenet állapotot vált, ha az R bemenetre kerül H, S bemenetre pedig L<sup>+</sup> szint. A kapcsolás tehát tároló, amelynek ha az S (Set = beíró) bemenetére H szintű jelet adunk, ezt tárolja, ha pedig az R (Reset = törlő) bemenetére adunk jelet, akkor a beírást törlő.

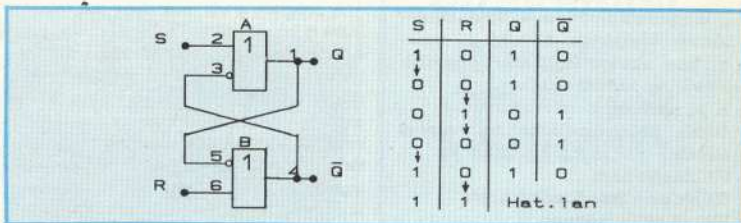
A tároló két kimenetén a tárolt érték és annak fordítottja (= inverze = negáltja) is megjelenik. A tároló logikai működése a táblázat alapján elemezhető.

Ha S és R egy időben H szintű, akkor a tároló mindkét kimenete H szintű lesz. Ez a

kimeneti állapot azonban nem egyértelmű, határozatlan, ha utána S és R egy időben L szintre változik, ezért ez az eset rendszert nem megengedett.

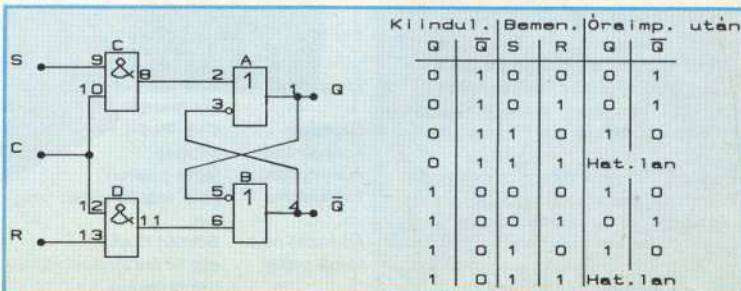
A fenti megoldásnál a tároló állapotváltozása a bemenő jel megváltozásával egy időben következik be. A biztosabb, zavarmentesebb működés érdekében célszerű a tárolót úgy működtetni, hogy a bemenetire a jeleket azok lezárt állapotában adjuk rá, majd a tároló alapállapot-változását egy külön impulzussal (órajellel) váltjuk ki. Az ilyen, órajellel vezérelt (szinkron) RS tároló alapkapcsolása a 2. ábrán látható.

Ha az S bemenetre H szintet adunk, a C jelű kapu kimenete csak akkor lesz H szintű, ha a C órajellel bemenete H szint érkezik. A C kapu kimenetén megjelenő H szint már a Q kimenetét H szintbe állítja, ahogy ezt már az előző esetben leírtuk. Hasonló módon a törlést is csak az R szintre kap-



1. ábra

2. ábra





solt H szint mellett a C órajel bszintre átváltása hozza létre.

Az egyszerű R-S flip-flop működésében mutatkozó határozatlan állapot elkerülésének egyik lehetséges módja az, hogy csupán egy bemenetet (D bemenetet) alkalmazzunk. A D flip-flop logikai kapcsolását a 3. ábra mutatja.

A D bemeneten lévő információ a Q kimenetre csak akkor jut el, ha az órajel bemeneten az átiró jel megjelenik. A tárolók vagy flip-flopok a különböző típusú számláló áramköröknek is alapelemei.

Ezek után foglaljuk össze a digitális logikai áramköröket.

Vannak KOMBINÁCIÓS és SORRENDI logikai hálózatok.

A SORRENDI hálózatokat működési módjuk szerint tovább osztályozhatjuk aszinkron és szinkron áramkörökre. ASZINKRON SORRENDI hálózat esetén a bemenő jelek megváltozása azonnal létrehozza — a belső állapottól függően — a kimenetek megváltozását. A kimenő jel kialakulásában az egyes kapuk áramköri késleltetése is szerepet játszik. SZINKRON SORRENDI hálózatoknál ez a megváltozás egy külső „karmester” — az órajel — parancsára következik be.

Mivel az órajel változó H-L szintekből álló impulzussorozat, ezért egy további osztályozás szerint, ha az állapotváltozás az órajel szintjére (például mikor H szintű) következik be, akkor szintvezérlésről beszélünk. Ha a változás az órajel szintjének változására (L-H szintváltás vagy felutó él, H-L szintváltás vagy lefutó él) következik

be, élvezérelt a működés. Az áramköri rajzokon az élvezérelt (élre billenő) bemenetet kis háromszög jelzi. A gyakorlatban a legtöbb digitális rendszer szinkron működésű, órajel vezérlésű, ezért a működési sebességet az órajel frekvenciája határozza meg.

Aszinkron logikai hálózatok a hálózatban terjedő jelek futási idejének különbözősége miatt hibás állapotokba kerülhetnek. Ezek a hibák az ún. házáródok. Mivel az aszinkron tervezésnél az áramkörgyártási technológia is szerepet játszik, ezért terjedtek el jobban a szinkron működésű hálózatok.

## A TTL és CMOS áramkörcsalád

Kissé bonyolultabb és itt nem részletezett módon, a P-N átmenet vezetőképességét elektromos erőterrel is vezérelhetjük. Az ezen az elven működő tranzisztorokat FET-eknek — tervezérlésű tranzisztoroknak — nevezük. A vezető csatorna lehet P, illetve N típusú. A CMOS jelölés az integrált kivétel N és P csatornás FET-ekkel kialakított áramkört jelöli. A bipoláris tranzisztorokat alkalmazó TTL digitális integrált áramkörökkel egy időben fejlesztették a CMOS áramköröket, mert számos előnyös tulajdonsággal rendelkeznek:

- egy bipoláris tranzisztor helyén 3-5 FET helyezhető el,
- az áramkörök bemeneti ellenállása igen nagy,

— a feszültséggel vezérelhető (nem kell vezérlő áram),

— a tápfeszültség értéke tág határon belül tetszőleges, értéke 2-3 V-tól 10-20 V-ig változhat,

- az áramkörök fogyasztása elenyésző,
- az üzemi hőmérséklet  $-40 + 85^\circ\text{C}$  között lehet.

A CMOS integrált áramköri elemcsalád — a TTL áramkörökhöz hasonlóan — igen sok olyan elemet tartalmaz, amelyek funkcionális működésükben megegyeznek. A CMOS áramköröknél elérhető nagy elemsűrűség lehetővé teszi, hogy igen összetett áramköröket hozzunk létre.

A digitális áramkörök tervezői már általában „keverve” használják az áramköröket — mindig az előnyösebb tulajdonságokat mutató család elemét részesítve előnyben. Ez a tervezési módszer természetesen azt is igényli, hogy pontosan tisztában legyünk a be- és kimenetek terhelési viszonyival. Egy kimenet túlzott terhelése ugyanis (túl sok bemenet kapcsolódik hozzá) az L, illetve H logikai feszültség szintjeit — amelyek egy tartományba kell hogy essenek — úgy elhúzza, hogy hibás logikai szint alakulhat ki. Például TTL áramköröknél az L szintnek a 0...0,8 V-os feszültségtartomány, a H szintnek a 2,4...5 V-os tartomány felel meg.

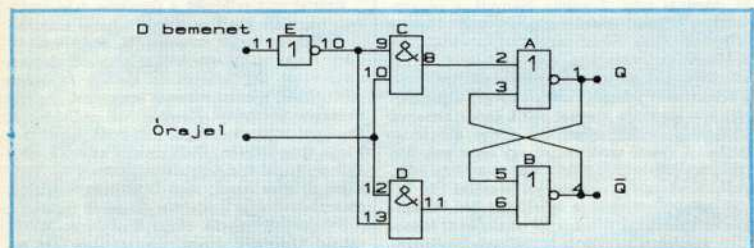
A következőkben a tényleges áramköri típusoktól elvonatkoztatva, röviden áttekintjük a digitális integrált áramkörök legfontosabb, a mikroprocesszor-technikában használt elemeit. Elsőként a kombinációs hálózatokkal megvalósítható egységeket, utána a sorrendi hálózatokat tartalmazókat tekintjük át.

## Kódoló áramkörök

A kódoló áramkörök az információnak az egyik ábrázolási formáról egy másik ábrázolási formára való átalakítására szolgálnak. Sok esetben van szükség olyan áramkörre, amelynek bemenetére egy bináris értéket kapcsolva, annak csak egyik — az adott bináris értékhez tartozó — kimenetén jelenik meg jel (például L szintű, amikor a többi kimenet H szintű). Az ezt megvalósító, 3-ból 8-ra átkódoló áramkör elvi rajza a 4. ábrán látható.

Az A, B, C bemenetek értékétől függően lesz a megfelelő kimeneten L szint, míg a többin H szint. Például ha  $A=L, B=H, C=H$ , akkor a 6. jelű kimenet lesz L szintű. A G1, G2A, G2B jelű bemenetek a kódoló működését engedélyezik vagy tiltják. A jelen esetben az Y kimenetek valamelyike csak akkor lehet L szintű, ha  $G1=H, G2A=L, G2B=L$  feltétel teljesül.

(Folytatjuk)  
DR. KÓNYA LÁSZLÓ



3. ábra



4. ábra





Pontos iránytű

# Mikrokalauz

## Verseny van, de nem az igazi!

Volt idő, amikor azt mondták: szakáll és bajusz nélkül a férfi nem férfi. Mintha a himnem és szörzetek híján nemtelenebb lett volna! A közfelőadásban vallott ilyen nézet persze ma már túlhaladott, nevétnük is rajta, hozzátéve: azért a szakáll öltözött!

Férlerakva az tróniát, tréfát — mert a számítástechnika nem tréfa —, de azért a játékos komolyságot megtartva mondhatjuk: aki maga vagy más szakállára számítógépes fejlesztésbe fog, a programok beszerzésekor könnyelműségét felejtse otthon. Hogy miért? Mert „A szoftver értéke a számítógépes rendszer árán belül a fejlett országokban ma már a 80–90 százalékot is eléri. Ez a tendencia az utóbbi években felgyorsult ütemben érezteti hatását hazánkban is...” — írják a Mikrokalauz harmadik kötetének szerzői. S az értékeket és érdeketendit hangsúlyozzák azazal is, hogy a hazai szoftverpiacot különös tekintettel az IBM PC-vel kompatibilis programokra tekintik át. A program mint érték tehát önmagában nem evidencia. Becse, hasznossága, gazdaságossága éppen a számítógépek programokkal való felöltözöttségéből adódik. A programtól függ, hogy a számítógép neves vagy „nemtelen” lesz-e.

Természetesen a szerzők ezúttal is tisztáznak néhány alapfogalmat. Így például a szoftverterméket, ami a programtermék és a dokumentáció együttese. A szoftvertermékeket azután a piacra való találozások szempontjából két csoportra lehet bontani. Ha egy program meghatározott, speciális igény kielégítésére készül, az egyedi programtermék. A többszöri értékesítésre szánt programtermék eleve eladásra és nem megrendelésre készül. A magyarországi szoftverpiacot jelentős állomása a mikrosoftverpiac kialakulása. „A hazai piacon 1982-től jelentek meg a mikrosoftverek, tulajdonképpen ekkor kezdődött a hazai mikrosoftver korszak is. Az első itthoni gyártású mikrosoftverek hazai fejlesztési eredményt képviseltek, s ez az elv a programvonalon is érvényesült. Az eredmény az lett, hogy minden gép másféle, speciális operációs rendszerrel működött, sajátos programnyelvet használt.”

Programiparunk kialakulásában meghatározó változás volt megfigyelhető a nyolcvanas évek közepén. Ekkor ugyanis a számítástechnikával foglalkozó szervezetek száma uggrásszerűen emelkedett. Viszont „annak ellenére, hogy ezeknek a szervezeteknek a száma meghaladja a négyezret, közülük csak néhány tucat foglalkozik programár-termeléssel”. Mégis: „Hazánkban a programkereskedelem egyelőre még hardverorientált, ami azt jelenti, hogy a számítógép-kereskedő könnyebben tud eladni programot géppel együtt. Emiatt a programgyártók arra törekednek, hogy saját vagy idegen számítógépet forgalmazzanak.” Az egyedi programtermékek iránt egyébként változatlanul nagy az érdeklődés. Ez a kereslet a kisservezetek (kisservezetek, szakcsoportok, vgm.-ek stb.) felütását eredményezte.

„A kisservezetek a hazai programipar szerkezetét lazították, azonban nagymértékben kötődnek a nagyszervezetekhez.”

Ez a gondolat rögtön átvezet minket a programmonopóliumok fonákágaihoz. A monopóliumok nagy vonalakban két mag körül alakultak ki. „Minden ágazatban létrejött egy vagy két ágazati számítástechnikai szervezési központ.” Igaz ugyan, hogy a vállalatoknak ilyen jellegű gondjaikkal nem feltétlenül kell hozzájuk fordulni, de ma ... amikor a vállalati önállóság még messze van az ideális szintől, és amikor működnek még az informális irányítás jól ismert csatornái, nem nehéz elképzelni, hogy sok vállalat erőteljesebben, esetleg kockázatosnak tartja, ha nem az ágazati számítástechnikai szervezési intézeteket keresi meg először”. Másrészt monopóliumhelyzet alakult ki az ország olyan vidéki körzeteiben, ahol csak egy vagy két program-előállító szervezet működik. Mindezek következtében verseny ugyan van a piacon, de ez nem a programtermékeké, hanem a programkészítő cégek versenye. Így tehát ... „amíg nem alakul ki a kereslet-kínálat globális és strukturális egyensúlya, addig valóban az dönt, hogy melyik cégnek van jobb és szerettegőbb kapcsolata, ismeretése”.

A programipar ilyenét berterjessége, homogenitása közvetett módon kihat a programkészítésre is. „Olyan szemléletváltásnak kellene bekövetkeznie, hogy elsősorban a meglévő építőközből tudják a programrendszert összeállítani, nem pedig különböző programozási nyelveken megírni azt.” Noha ez a folyamat megindult, nem eléggé dominál még. Továbbá hiányzik a programokhoz tartozó dokumentációs szabvány: a programcsoport készítésének nincs elfogadott módszere és gyakorlata, megfogalmazott követelménye. „Általában a programkészítésnél csak a felhasználó jelenlegi tevékenységét algoritmiázják, a manuális munkát viszik gépre, elmarad a számítógépesítés legnagyobb előnye, a racionalizálás. A vezető szoftveresek egy része nem tud feladatot fogalmazni, kapacitást és átfutási időt, gépírószükségletet és költséget becsülni.”

Ebből következnek a keresletben és kínálatban az anomáliák — amiknél óhatatlanul felmerül a gondolat: a fejlődés szükségesszerű velejárója ezek sokasága, amit a szerzők frappánsan érzékeltetnek. A kínálati oldalból kiragadva: a szoftver követése gyakran költséges, aránytalanul lekötő a szellemi kapacitást. A napjainkig kialakult kínálat összességében sem mennyiségileg, sem minőségileg nem kielégítő, szélsőségek is fellelhetők. A kereslet felől közelítve a kötet szerzői egyfajta hiányból tulajdonképpen javaslattal is előállnak. „A jelenlegi körülmények legnagyobb hiányossága, hogy nincs komplex problémamegoldás ... Nincs olyan szervezet, amelyhez fordulva a potenciális számítógép-vásárló meg tudhatja, hogy konkrét problémáját milyen géppel, milyen kiépítéssel, milyen rész- és egyedi fejlesztésű programtermékkel, milyen időbeli ütemezéssel lehet megoldani.” Magyarán a belső programforgalomról, piacról jórómán lehetetlen megbízható információhoz jutni.

Cikünk elején értéket említve kimondatlanul is a programok forgalmi értékére, árára is céloztunk. Nos, ... néhány, csak központilag forgal-

mazott alprogram esetében az ár nem alku tárgya: ha a bevőnék valóban szüksége van az adott programra, nem fordulhat máshova”. A programok többségében azonban az árú „szabad préda”. Ennek több oka van, mint például az, hogy a kereslet jóval meghaladja a kínálatot: rengeteg a párhuzamos fejlesztés, így a többletköltség. Általános vélemény, hogy ... „a magyar programtermékek és a hozzájuk kapcsolódó szolgáltatások drágák, sőt nagyon drágák.”

A pénzügyi szabályozókról szóló fejezet azért kapott a kötetben helyet, mert a program, mint sajátos vagyontárgy, különleges gazdálkodási, pénzügyi, számviteli szabályozásokat és gyakorlatot követel. Ezért döntött úgy 1985-ben a Minisztertanács, hogy a Központi Statisztikai Hivatal elnöke a számítástechnika-alkalmazás területén a gazdasági miniszterek hatáskörében járhat el. Az ezt követő fejezet a számítástechnika szerzői jogi vonatkozásait taglalja. (Ezt most mellőzzük, mivel a későbbiekben több cikkünk részletesen kíván ezzel foglalkozni.) Izgalmas tizenkét pontos kiskaté is összeállítottak a szerzők, mintegy programbeszerzési útmutatóként. Egészen a számítógépes igények felmérésére való utalástól a tanácsadásig, támogatásig, más felhasználókkal az együttműködés elengedhetetlenségének bizonyításáig.

Ezt követően végre témánként felsorolják, röviden ismertetik az egyes általános célú programcsoportokat, míg egy külön kötetben (III/2) található az IBM PC-vel kompatibilis hazai szoftverkínálat teljes jegyzéke.

Ezúttal csak felhívjuk a figyelmet a Mikrokalauz negyedik kötetére, amely a hazai számítástechnikai irodalomat dolgozza fel, az 1979-től kiadott hazai számítástechnikai könyvek alapján, témánkénti, kiadónkénti és kiadási évenkénti mutatókkal. E helyt érdemes jelezni, hogy az Országos Széchényi Könyvtárban működik az Országos Könyvtalógy (558-058), amely egy könyv címe alapján felvilágosítást ad arról, hogy az mely hazai könyvtárban található. Ez a katalógus a teljes hazai, azaz 1952 után megjelent számítástechnikai irodalmat magában foglalja.

Befejezést keltsük életre Bod Péter XVIII. századi tudósunk szavait: „Mi az, ami igen nehéz, s mégsem terhelheti azt, aki hordozza? Az a tudomány és a jó mesterség, amely terhére nincsen annak, akinél végelem, hanem könnyebbíti sok bajaiban.” Jószerével a számítástechnika genézisét éljük: tudományunk és becsületbeli mesterségünkön múlik, hogy a teremtés súlya lehű-e avagy felemel.

SZULYOVSKY CSABA

**Közületek, figyelem!**  
**Mikroszámítógépet**  
**akarnak vásárolni?**  
**Tájékozdódnak**  
**a naprakész piaci helyzetről!**  
**Díjtalan ismertető!**  
**MESZ**  
**Számítástechnika**  
**1368 Budapest, Pf. 193.**





# A **NOVOTRADE** SZJA '88 programja az ön személyes adószakértője és adókönyvelője!

A program Commodore-64, Commodore Plus/4, Enterprise, Videoton TV-Computer, ZX Spectrum és IBM PC kompatibilis személyi számítógépeken működik.

Az SZJA '88 program **KISZÁMOLJA**: a befizetendő személyi jövedelemadót.

Az SZJA '88 program **NYOMON KÖVETI**: használójának jövedelemalakulását.

Az SZJA '88 program **FIGYELEMBE VESZI**:

a várható, illetve tervezett jövedelmeket a tényleges jövedelem és kiadások alapján.

Az SZJA '88 program: az adó jogszabályok teljes körét feldolgozza.

A Commodore Plus/4-re a Művelt Nép és Állami Könyvterjesztő 20 szaküzleteiben, a Vidia boltjaiban és a Magister Stúdium könyvesboltokban keresse.

**Az összes többi változat a CENTRUM ÁRUHÁZAKBAN kapható.**



**RAINBOW**  
Számítástechnikai és  
Szolgáltató  
Kiszövetkezet

**D-Subminiatur csatlakozók,  
kábelek, analóg-digitál átalakítók,  
egyéb számítástechnikai berendezések  
és kellékek nagy választékával várja Önöket**

**a MAGÉV-RAINBOW  
Elektronikai Szaküzlet**

Budapest VI., Rudas László u. 33.  
Telefon: 122-392. Telex: 22-6323

**RAINBOW Számítástechnikai és Szolgáltató  
Kiszövetkezet**

Központ: Budapest, Szilágyi Erzsébet fasor 1. 1026  
Levél cím: 1378 Budapest 64., Postafiók 31. Telefon: 352-558



# A billentyűkezelő rutin sajátosságai

**Gépi kódú programok írásakor szükség lehet a billentyűzet használatára: programozás közben, amikor kipróbáljuk az új programot, vagy egy menüből kell választani, vagy a program futását akarjuk leállítani.**

Kezdő programozó vagyok, ami a gépi kódot illeti. Sajnos sokszor kerültem végteleen ciklusba egy-egy hibás utasítás miatt. Ilyenkor természetesen újra be kell tölteni az Editor/Assembler programot, valamint az „alkotás”-t. Ez igen időrabló és idegesítő. Felmerült tehát a kérdés: hogyan állíthatom le a gépi kódú programot futás közben? Olyan megoldást kerestem, hogy véletlen billentyűnyomás ne állíthassa le a programot. Az általam ismert kézikönyvekben található kódtáblázatokkal nem elégedtem meg, hiszen ezek nem tartalmaznak minden variációt. Természetesen közrejátszott eredendő kíváncsiságom is: minden lehetséges kombinációra érvényes táblázatot akartam kidolgozni. Ekkor kezdtem a Spectrum ROM billentyűkezelő rutinjának részletes elemzéséhez.

## Mit tudunk?

A Spectrum billentyűzete négy sorban 10-10, összesen 40 billentyűből áll. A billentyűzetet mint perifériát a 254-es porton lehet elérni. A billentyűzet 8 fél sorra van felosztva, egyenként 5 billentyűvel:

1. CAPS SHIFT—V
2. A—G
3. Q—T
4. I—S
5. O—6
6. P—Y
7. ENTER—H
8. SPACE—B

A billentyűzetet leolvasó szubrutin ezeket a fél sorokat látja: a 8 fél sornak megfelelően nyolcszor hajtódik végre. A rutin befejezésekor a B, D, E regiszterekben a leolvasott billentyűnek megfelelő értékek találhatóak.

## Nézzük tehát konkrétan

A rutin a 654 (028Ehexa) címen kezdődik és 48 bájttal hosszú. A program RET utasítással ér véget. Ez egyben azt is jelenti, hogy a szubrutin meghívása után a vezérlés visszakerül eredeti programunkhoz. Ennek

Az E regiszter tartalma a különböző billentyűk lenyomása után

A	38	K	17	U	10	S	4
B	0	L	25	V	7	6	3
C	15	M	16	W	29	7	11
D	22	N	8	X	23	8	19
E	21	O	26	Y	2	9	27
F	14	P	34	Z	31	0	35
G	6	Q	37	I	36		32
H	1	R	13	2	28	SYS	24
I	18	S	30	3	20	CS	39
J	9	T	5	4	12	ENTER	33

### 1. táblázat

#### 1. lista

```

10      ORG 50000
20
30      ;KEZDŐCÍM
40      ;ATHELYEZHEŐ BASHOVA
50
60      ;EGY REGISZTER VIZSGALATA
70
80      START
90          LD B,10
100     VAR      HALT
110     DJNZ     VAR
120
130     ;LASSITJA A PROGRAMOT
140
150     CALL 654
160     LD A,B
170     ADD A,100
180     RST 16
190
200     ;A VIZSGALT REGISZTER
210     ;TARTALMANAK (+100)
220     ;KITIRASA
230
240     LD A,D
250     CP 24
260     JR NZ,START
270     RET
280     ;KILEPESI LEHETŐSÉG
290     ;SYS + BARMELY BILLENŰYŰ

```

```

50000 05 0A 78 10 00 V
50004 FE CD 8E 02 00 M
50008 78 06 5A 07 00 XCDW
50012 7A FE 18 20 00 27
50016 EF 09 00 00 00 00

```

## ZX—SPECTRUM

a szubrutinnak a használatát azért is előnyös, mert bármely megszakítási módban működik. A program nem gondoskodik a regiszterek tartalmának elmentéséről.

A D regiszter tartalma a különböző billentyűk lenyomása után

nincs billentyű lenyomva	
vagy egy van lenyomva	255
CAPS SH. + 1 billentyű	39
CAPS SH. + 2 vagy több billentyű	40
SYSH. + 1 billentyű (kiv. CSH)	24

### 2. táblázat

#### 2. lista

```

10     ;EGY REGISZTER VIZSGALATA
20     ;PROGRAMBOL VALO KILEPES
30     ;SYMBOL SHIFT +
40     ;BARMELYIK BILLENŰYŰ
50     ;MEGNYOMASARA
60     ;A D REGISZTERBEN 24 LESZ
70     ;KIVETEL CAPS SHIFT!
80
90     START  PUSH BC
100     PUSH DE
110     PUSH HL
120
130     ;REGISZTEREK TARTALMANAK
140     ;KIMENTESE
150
160     CALL 654
170
180     ;ROM RUTIN HIVASA
190
200     LD A,D
210     CP 24
220
230     ;D REGISZTERBEN 24 VAN-E
240
250     POP HL
260     POP DE
270     POP BC
280
290     ;REGISZTEREK TARTALMANAK
300     ;VISSZAITRASA A VEREMBOL
310
320     JR NZ,TOVABB
330     FOLYT
340     ;ITT FOLYTATODIK, HA A
350     ;FELTETEL TELJESUL. PL:
360     RET
370
380     TOVABB
390     ;ITT FOLYTATODIK, HA A
400     ;FELTETEL *NEM* TELJESUL
410     ;PL:
420     JR START

```



# ADOK—VESZEK—CSERÉLEK

## Hogyan lehet a regiszterek tartalmát megvizsgálni?

Az 1. lista szerinti programot azért kell lassítani, mert különben pillanatok alatt tele lesz a képernyő. A ciklus tiz HALT utasítást hajt végre. A vizsgálandó regisztert a 160-as sorba kell beírni. Értékehez azért kell százat hozzáadni, mert a 32 alatti számok helyett vagy kérdőjel jelenik meg,

A B regiszter tartalma a különböző billentyűk lenyomása után

0, 1, 2 billentyű lenyomása esetén	254
SYSH. + 2 billentyű	127
E üzemmód + 1 billentyű	127
G üzemmód +	
8. fél sor 1 billentyű	127
7. fél sor 1 billentyű	191
6. fél sor 1 billentyű	222
1-5. fél sor 1 billentyű	238

### 3. táblázat

vagy hibáuzenettel leáll a program. A kilépési lehetőségre azért van szükség, hogy a gépi kódú programot le tudjuk állítani. Ez egyben példa az eljárás használatára.

A C regiszterben mindig 254, a többi regiszterben az aktuális billentyűnek megfelelő érték található. Az E regiszter (1. táblázat) a lenyomott billentyűnek megfelelő számot tartalmazza 0—39-ig. Ha nem nyomtunk le billentyűt, az E regiszter tartalma 255. Ha több billentyűt egyidejűleg nyomkodtunk, a legváltozatosabb értéket fogjuk kapni. Erre táblázatot készíteni — véleményem szerint — nem érdemes. A D regiszter értékeit a 2. táblázat, a B regiszter értékeit a 3. táblázat tartalmazza. A táblázatokban nem szereplő variációkat egyenként kell kikísérletezni, az E regiszternél leírtak alapján.

## További előnyök

Az eljárás használatát a 2. lista mutatja. A program elején a regiszterek értékét a verembe ki kell menteni, mert a ROM rutinja megváltoztatja azok tartalmát. Mivel a POP utasítás a jelzőbiteket nem változtatja meg, a feltételes ugrás hibátlanul működni fog. Ha ezt a rutint beépítjük egy gépi kódú programba, nemcsak a BASIC-be való viszszeresztést, hanem az avaratlanok elől „rejtett” szubrutinhívást is megoldhatjuk.

DR. GRESZ MIKLÓS

Ebben a rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás: közületeknek gépelt soronként (60 karakter) 100,- Ft, magánzemléknek az első sor 50,- Ft, minden további sor 20,- Ft. Az NJSZI tagjainak az első három sor ingyenes. Hirdetéseiket a szerkesztőség címére várjuk.

### ADOK

**C16-os**, memóriabővítővel (64 k), magnóval, 210 db 16-os, 20 db Plus/4-es programmal olcsón eladó. Érdeklődni 19-21 óra között a 860-256-on.

**Commodore 16-os** számítógép eladó, magnóval, gépkönyvekkel, sok programmal 18 000 Ft-ért. Makó, Beloiannisz u. 128/A, 6900

**Commodore 64-es** programok nagy választékban kaphatók, kazettán és lemezen. Sebestyén Imre, Budapest, Özgida u. 18/C. 1025

**C64** programok olcsón, nagy választékban. Bohner Tamás, Baja, Széchenyi u. 8. 6500

**C64-es** számítógép 1541-es floppyval, Seikosha SP-180 nyomtatóval eladó. Amigára cserepartnert keresek. Raiz Tamás, Budapest, Györi u. 8. 1123, Tel.: 566-941

**Commodore, Enterprise, IVC** és Atari számítógépekhez HW cikkek széles választékát kínáljuk. Joystick-ok 450 Ft-tól kaphatók. Fényceruza 2000 Ft (C16-hoz és Plus/4-hez is). Válaszboríték ellenében elküldjük katalógusunkat. COMPUTEAM GM. Kaposvár, Berzsenyi u. 32. 7400

Adok egy db Super Shot joystickot. Bóna Zsolt, Kecskemét, Tel.: 47-788, 17 óra után.

**Spectrum** programkattak olcsón eladók, Magyarországon még ismeretlen, teljesen új, színvonalas játék és felhasználói programokkal. Röviddel az angliai megjelenés után! Postacím: Siklós, Pf.: 129, 7800. Küldjön válaszborítékot!

**Spectrum-ja** lekopott (fém) fedőlapját olcsón, gyorsan felújítom. Képiróbert, Budapest, Imre u. 5. 11/14. 1093. Tel.: 378-075

Új **ZX-Spectrum Plus** (13 500 Ft) és **ZX-Microdrive** (4500 Ft) eladó. Budapest, Stoczek u. 5-7. 516 sz. 1111

### VESZEK

**C 64** zongora klaviatúrát programmal keresek (pl. Wersiboard Music 64). Tel.: 49/14-948

**HT 10802** (16 k) gépet veszek, esetleg két db-ot is. Irányárat kérek, minden

levélre válaszolok. Márton Péter, Törökbalint, Katona J. u. 7. 2045

**Videoton TVC** gépéhez játékprogramot vennék, kazettán. Sinka Edit, Pusztaföldvár, Rákóczi u. 172/a. 5919. (Levélben!)

### CSERÉLEK

**Atari 800XL**-re készült programokat cserélek, ótszáz játék és felhasználói program. Gayer Ferenc, Budapest, Gubó u. 6. 1/10.1151

**Commodore Amiga** programokat cserélek. Lenhardt Tamás ADT, Győr, Murkásör u. 75. 9023

**C16** és **Plus/4-es** programokat cserélek. Farkas Tibor, Zalaszentgrót, Felszabadulás u.l. 8790

**C16, Plus/4-es** programokat cserélek. Kunsági László, Nyírbátor, Vasvári Pál u.l. 4300

**C64-es** játék- és felhasználói programokat cserélek, elsősorban kazettán. Válaszokat lehetőleg listával kérek. Aradi László, Szentés, Nagyvérny u. 111. 6600

**Enterprise** programokat cserélek, Veregyház, Pf.: 28. 2112

**SHARP MZ-800** tulajdonosok jelentkezését várom. Szoftver-hardver csere! Kánár László, Budapest, Zápor u. 16/B. 1034

**Spectrumot** (48 k) adok kompletten, befizetéssel, régi, de jó állapotban lévő **C64** + floppyért. Ajánlatokat kérek! Marincsik János, Kisvárd, Lenin u. 20. 4600

**ZX-Spectrum** játékprogramokat cserélnék, a válaszokat listával kérem. Jancsin Ferenc, Békéscsaba, Lencsési u. 8. 5600

**ZX-Spectrum** (48 k) programokat cserélek. Listát vagy kazettát kérek. Platthy Péter, Budapest, Mártírok útja 31-33. fsz. 2. 1027

**ZX-Spectrum+** (48 k) játék- és felhasználói programokat cserélek. Válaszokat listával kérek. Miszori Róbert, Ajka, Újélet u.6. 8400

**ZX-Spectrum-os** cserepartnereket keresek. Csécsi József, Kartol, Vörös Hadsereg u. 55. 2173

**TV-Computer** játékprogramokat cserélek vagy könyveket veszek. Fogarasi Ferenc, Budapest, Zsókvár u. 28. VIII/34. 1157

**VC20** alagépére programokat cserélek. Nagy Tamás, Nagykanizsa, Kazanlak krt. 7/B. II/8. 8800

Ez a rovatunk **KODEX 2000** szövetségkeresztől készült.

### A HIRADÁSTECHNIKAI TUDOMÁNYOS EGYESÜLET

Mikroszámítógépes programnyelvek és Operációsrendszerek Szakosztálya előadást szervez az alábbi témában:

**A számítógépek, számítástechnikai rendszerek tulajdonságainak javítása mikroprogramozással**

Előadó: Dr. Vajda Ferenc (KFKI)  
Helye: MTESS Zsúdkház, Budapest V. ker.  
Kossuth tér 6-B. V/541.  
Ideje: 1988. május 16. 14 óra

### A TUDOMÁNYSZERVEZÉSI ÉS INFORMATIKAI INTÉZET

előzetes megbeszélés szerint díjmentes programmutatót tart (vidéken is) az általa forgalmazott oktatóprogramokból.

Horváth Zsuzsa 665-011/2663 mellék vagy 813-197

Budapest, Pf.: 454, 1372







# KLUBLAPOKBÓL

## Humoros szakszótár kezdőknek (Valley Computer Club Newsletter)



### Ablakok (WINDOWS)

Olyan módszer, amely a képernyőt felosztja sok apró, használhatatlan részre

### Adatbázis-kezelő (DATABASE MANAGER)

Ez egy olyan program, amely lehetővé teszi a felhasználónak az adatok kezelését minden elfogadható szinten, kivéve a feltétlen szükséges szintet, amelyet az ő elhatározása szerint a következő napon, miután beütött 20 Mbájt nyers adatot, kívánna igénybe venni

### Univerzális szoftver (INTEGRATED SOFTWARE)

Egyszerű termék, amely simán megvalósítja azoknak a funkcióknak százait, amelyeket az ember nem használ, de azokból, amelyeket használni szeretne, csak fél tuca tot

### Dokumentáció

Zavaros félévszón kötésű tartozék, amit csak végszükség esetén használnak, ha a barátok vagy a cég emberei nem elérhetők. Máskülönben általában tetszetős könyvtásmaszként alkalmazzuk

### Egér (MOUSE)

Egy periféria, amelynek eredeti neve féregnyúlvány, mivel működése ahhoz hasonlós, de újranevezték, mert felhasználható macskajátéknak

### 1.0-ás verzió

Több benne a rejtett hiba (angolul bug, ami szó szerint poloska), mint júniusban a hártában. Felfalja az adatokat

### 1.1-es verzió

Az adatokat csak alkalmilag falja fel, a további javított változatokat már ingyen adják az 1.0-ás verzióval elégedetlen vásárlók pereskedéseinek elkerülése céljából

### 2.0-ás verzió

Az eredetileg első kiadásnak szánt verzió (kivéve néhány adatfaló poloskát, amelyet még mindig nem sikerült eltüntetni). A további javított változatokat már nem adják ingyen, mert különben a cég tönkremegy

### 3.0-ás verzió

A munkában a cég tönkremenetelekor végzett javítások vannak

### Felhasználóbarát (USER FRIENDLY)

A kézikönyve végig színes

### Nagyon felhasználóbarát (VERY USER FRIENDLY)

Diszken és magnószalagon is adnak ismertetőanyagot, és így a felhasználónak nem kell a csupaszin kézikönyvet lapozgatnia

### Rendkívül felhasználóbarát (EXTREMELY USER FRIENDLY)

Egeret adnak hozzá, így a felhasználónak nem kell a lemezen vagy magnószalagon adott ismertetővel vagy a csupaszin kézikönyvvel foglalkoznia

### Garancia

Feltétlen jótállás arra vonatkozólag, hogy a program ténylegesen a dobozban levő lemezen van

### Hathatós (POWERFUL)

Nehezen tanulható, használata veszélyes

### Hálózat (NETWORK)

Olyan elektronikus eszköz, amely egy időben több személynek ad lehetőséget hasznos adatok elrontására, tönkretételére vagy másféle, hosszú időtartamra érvényes károsítására

### Hordozható (PORTABLE)

Kiseb és könnyebb az átlagos hűtőszekrénynél

### Szállítható (TRANSPORTABLE)

Nincs se fához láncolva, se riasztóberendezéshez kötve

### IBM kompatibilis

Nem teljesen IBM kompatibilis

### Teljesen IBM kompatibilis

Némileg IBM kompatibilis, de nem futnak rajta BASIC programok

### 100% IBM kompatibilis

A legtöbb hardverrel és szoftverrel kompatibilis, de azzal a védelemmel nem, amit az IBM minden második nap bevezet

### Most kapható (NOW AVAILABLE)

Bármelyik nap megjöhét

### Hamarosan kapható (AVAILABLE SOON)

Egy éven belül megjelenik

### Május elsején kapható

Az 1.0-ás verziót augusztus elsején küldik a boltokba

### Könnyen használható (EASY TO USE)

Nehezen használható

### Könnyen tanulható (EASY TO LEARN)

Nehezen tanulható

### Könnyen tanulható és felhasználható (EASY LEARN AND USE)

Egy kicsit megtanulni, kevésbé használni

### Másolat (BACKUP)

A fontosabb adatok másodpéldánya, de ennek rögzítésével nem foglalkozik senki

### Másolás ellen védett I. (COPY-PROTECTED)

Okos módszer az illetéktelen kalóz szoftverlopásának megakadályozására és a törvényes vevő használatának megnehezítésére

### Másolás ellen védett II.

Olyan eszköz, amivel a tisztességes felhasználókat el lehet különíteni a tolvajoktól, úgy, hogy megakadályozzák az előbbiektől — de nem az utóbbiaktól — való lopást

### Menüvel vezérelt (MENU-DRIVEN)

Könnyen tanulható, fáradságosan használható

### Merevlemez (HARD DISK)

Olyan berendezés, amely engedélyezi a naiv felhasználónak hatalmas mennyiségű adat törlését kevés, egyszerű utasítással

KARUCZKA GÁBOR



# KI AD MAGYARÁZATOT?

## Nem igazak a matematikai tételek?

A középiskolai tananyagban szerepel a következő azonosság:

$$a^x = a^{n \cdot x/n} = (a^{x/n})^n$$

vagyis egy hatványfüggvényt átalakítható úgy, hogy a kitevőt osztjuk el egy számmal, majd ugyanennek a számnak megfelelő hatványra emeljük. Ez az összefüggés bármely, nullától különböző „a” és „n” esetén igaz.

A legtöbb BASIC-változat ismeri az EXP függvényt, amely az  $a = e$  esete összefüggésünknek. (Azoknak, akik számára az „e” jelölés ismeretlen: ez egy végtelen tizedestörttel előállítható, 2 és 3 közé eső szám, amelynek mind a matematikában, mind a műszaki és természettudományos gyakorlatban nagy a jelentősége.) Mivel minden szám első hatványa önmaga, ezért ha  $x = 1$ , akkor „e” értéket kell kapnunk az EXP függvényvel.

Legyen  $n = 2^k$ ,

ahol  $k$  pozitív egész szám, és változzon  $k$  értéke 1 és 12 között. Így

$$n = 2 \dots 4096$$

A hatványozás helyettesíthető  $k$ -szori önmagával történő szorzással, mert például

$$b \times b = b^2 \quad b^2 \times b^2 = b^4 \quad b^4 \times b^4 = b^8 \text{ stb.}$$

Első összefüggésünknek megfelelő program látható az 1. listán. (Mivel a nyomtató a német ábécé változatra volt beállítva, a hatványozás jele helyett ü betűt írt.) A program első sora a továbbiak szempontjából érdektelen (nyomtatóbeállítás). A második sorban felvesszük a független változó értékét (1), fejléct nyomtatunk, miután képeztük az EXP függvényt (Z), majd egy ciklusban osztunk  $n$  értékével (Y), képezzük a leosztott értékből az EXP függvényt (Y), elvégezzük a  $k$ -szori szorzást egy újabb ciklusban (Y), kinyomtatjuk a számított és a relatív hiba abszolút értékét. (A hiba a kétféle módon számított „e”-érték különbségének az első összefüggés bal oldala szerint számított értékkel képezett hányadosa.) Az első két értékénél a számítógép és a matematika azonos eredményt adott, de ettől kezdve megjelenik egy hiba, ami egy darabig nő, majd állandósul (1. ábra). Mi lehet ez?

Ugyancsak középiskolai anyag a trigonometriai függvények többszöröseitől származó vonatkozó néhány összefüggés. Ilyen például a következő:

$$\cos(2x) = 2\cos^2(x) - 1$$

Legyen  $x = 2y$

amit behelyettesítve:

$$\cos(2 \times 2y) = \cos(4y) = 2\cos^2(2y) - 1 = 2[2\cos^2(y) - 1]^2 - 1$$

Legyen továbbá  $y = 2z$ , amit behelyettesítve:

$$\cos(4 \times 2z) = \cos(8z) = 2[2\cos^2(2z) - 1]^2 - 1 = 2[2[2\cos^2(z) - 1]^2 - 1]^2 - 1$$

Ezt általánosítva:

$$\cos(nx) = \underbrace{2[2 \dots 2}_{k} \cos^2(x) - 1]^2 \dots - 1}_{n=2^k}$$

Az összefüggés bármely  $x$  és bármely pozitív egész  $k$  esetén igaz.

Ismét ellenőrizzük ezt egy programmal (2. lista). A programot kommentárokkal láttam el, mert követése nehezebb (az ő helyett két pontos A-t írt a nyomtató). A program fokként 1 és 89 fok között számította a két módszerrel a szögfüggvény értékét, és kikereste ebben a szövegtartományban a legnagyobb relatív hibát, amit a 2. ábrán a hozzá tartozó szöggel és  $k$  értékével (mint leosztás) kiírt.

Itt kezdettől fogva van hiba, ami állandóan nő. Mi lehet az oka?

Válóban hibásak a matematikai összefüggések, és hibájuk úgy változik, ahogy a két ábrán látható? Másfajta számítógépen, más programnyelvet használva a hiba számértéke esetenként ettől eltér ugyan, de maga a hiba létezik.

SE

```
5 PRINT#-2,CHR$(27);'L010';CHR$(27);'/05
0'
10 X=1:Z=EXP(X):PRINT#-2,'EXP(1)=';Z:PRI
NT#-2,'SZAMITOTT','HIBA':FOR I=1TO12:Y=X/
2UI:Y=EXP(Y):FOR J=1TO I:Y=Y*Y:NEXTJ:PRIN
T#-2,Y,ABS(Z-Y)/Z:NEXTI:PRINT#-2
```

1. lista

1. ábra

5.12868601E-08	88	LEOSZTAS	1
2.19290892E-07	89	LEOSZTAS	2
2.02228676E-07	88	LEOSZTAS	3
2.27378807E-06	89	LEOSZTAS	4
8.69242414E-06	89	LEOSZTAS	5
2.64358088E-05	89	LEOSZTAS	6
9.60752586E-05	89	LEOSZTAS	7
7.9641866E-04	89	LEOSZTAS	8
1.92340375E-03	89	LEOSZTAS	9
.0131825819	89	LEOSZTAS	10

2. lista

```
9 REM NYOMTATAS BEALLITAS, HIBA NULLAZAS
, LEOSZTAS BEALLITAS
10 PRINT#-2,CHR$(27);'L010';CHR$(27);'/0
50'E=0:FOR E1=1TO10
19 REM SZAGBEALLITAS
20 X=3.1415926536/180:FOR I=1 TO 89
30 Y=X*I
39 REM LEOSZTAS, SZAGFUGVENYKEPZES
40 Y=Y/(2UE1):Y=COS(Y)
49 REM SZAGFUGVENYTABBSZARAZES
50 FOR J=1 TO E1:Y=2*Y*Y-1:NEXT J
59 REM SZAGFUGVENYKEPZES, HIBASZAMITAS,
A LEGNAGYOBB HIBAERTEK ADATAINAK TAROLAS
A
60 F=COS(X*I):G=ABS(F-Y)/F:IF E<G THEN E
=G:K=I
69 REM CIKLUSZARASOK, NYOMTATAS
70 NEXT I
80 PRINT#-2,E;K;
90 E=0:PRINT#-2,'LEOSZTAS';E1:NEXTE1
```

2. ábra

EXP(1)=	SZAMITOTT	HIBA
2.71828183	2.71828183	0
2.71828183	2.71828183	0
2.71828184	2.71828184	1.71307214E-09
2.71828184	2.71828184	1.71307214E-09
2.71828185	2.71828185	7.194903E-09
2.71828185	2.71828185	7.194903E-09
2.71828185	2.71828185	7.194903E-09
2.7182817	2.7182817	4.796602E-08
2.718282	2.718282	6.37262837E-08
2.71828198	2.71828198	1.66853227E-07
2.71828198	2.71828198	1.66853227E-07
2.71828198	2.71828198	1.66853227E-07



# INFORM

Rovatunkban az Apple, Atari, Commodore és Sinclair mikrók tulajdonosait feltehetően érdekli, angol és német nyelvű cikkekről informáljuk olvasóinkat egy tartalomleíró szolánc segítségével.

A forráshely karakterosorozatát nyílt vezeti be, ezt a / jellegű folyóirat kódja követi (lásd táblázat). A két / jel között a megjelenési adatokat (év, hó), illetve a cikk kezdő oldalszámát szerepeltettük. A második / jel után pedig - az esetleges másolatkérest megkönyvitendő - a cikk teljes oldalterjedelmét közöljük.

A tartalomleíró szoláncok permutálási és alfabetikus rendezésével szerkesztett teljes anyagot az OMIKK "APACS" c. kiadványsorozata tartalmazza (vevőszolgálati telefon: 341-765). Lapunk ebből csak a "programlista" címszóval kezdődő részletet teszi közzé.

A folyóiratok megtekinthetők a SZÁMALK (Bp. XI. Szekasits Á. u. 68.), illetve - a x-gal jelzettek az OMIKK (Bp. VIII. Múzeum u. 17.) szakkönyvtárban. Másolatok a SZÁMALK-tól csekkszelvény, az OMIKK-tól megrendelőlévíl beküldésével, vagy személyesen rendelhetők.

## PROGRAMLISTA

C64|jatekProgram Keszites|Kep/hanghas Keltese|rutinyujtemeny  
->run2/87.01-96/3

## PROGRAMLISTA

C64|kepernyokezeles|bitterkep masola s-tarolas|ctrl-billentyus utasitasok |<bitmapcopy> ->run2/87.03-142/3

## PROGRAMLISTA

C64|kepernyokezeles|c128 editorfunkciok megvalositasa c64-en  
->cute/87.03-110/3

## PROGRAMLISTA

C64|kepernyokezeles|jegyzetelo ablak kozvetlen hozzafereses|rutinok a <64 notepad>-hoz|<86.09>szam  
->run/87.01-56/3

## PROGRAMLISTA

C64|kepernyokezeles|ket extra sor bitzotitasa parbeszdehez  
->64er/87.03-84/2

## PROGRAMLISTA

C64|kepernyokezeles|maszk-kezesites e s programba illesztese|maskgen  
->run2/87.03-146/3

## PROGRAMLISTA

C64|kepernyokezeles|menukezesites|pu11-down-menu ->happ/87.03-38/4

## PROGRAMLISTA

C64|kepernyokezeles|uj utasitasok a Keret hasznalathoz|<randhex>  
->run2/87.02-89/3

## PROGRAMLISTA

C64|kepernyokezeles|v|ipozicionalt kiiratas|print at beepitese  
->run2/87.01-95/1

## PROGRAMLISTA

C64|lemezkezeles|dos calc-kal bovitett dos-udgedge ->cute/87.03-104/4

## PROGRAMLISTA

C64|lemezkezeles|kataloguskezeles/szarkesztes|vedett fileok torlense|<diractory-manager> ->run2/87.02-114/6

## PROGRAMLISTA

C64|lemezkar Kezeles/rendezes|<speed-sort support> ->run2/87.01-102/11

## PROGRAMLISTA

C64|revo|tezes|c|cimiro-nyomtato Kiagazitas a <datafile>-hez  
->run/87.03-36/10

## PROGRAMLISTA

C64|master-text|alkalmazasi utmutato |2.resz|segedprogram  
->64er/87.03-82/3

## PROGRAMLISTA

C64|matematika|aritmetikai operacios rendszer|<arith 13>  
->64er/87.03-53/3

## PROGRAMLISTA

C64|nyomtas|tobbmeter zeles tapet a|c|o|l|l|a|s|s|in|j|e|l|z|e|s|k|e|z|i|u|t|a|n|f|e|s|t|e|s|h|e|z| ->64er/87.01-64/2

## PROGRAMLISTA

C64|rendezo szubrutinok|<shortsorter> ->run/87.01-14/2

## A folyóirat neve Kódja

x 64'er Magazin 64er  
Antics anti  
x Chip Magazin chip  
x Computer cute  
x Dr. Dobb's Journal dobb  
Elektr Electronics etor  
Happy Computer happ  
x mc - Zeitschrift mc  
Run (USA) run  
Run (NSZK) run2  
x Your Computer your  
x ZX Computing Monthly ZXCm

## PROGRAMLISTA

C64|goto-gosub|elagazások ellenorzes e ->run2/87.01-128/5

## PROGRAMLISTA

C64|grafika|3-d rajzKeszites|2-d abra azolas rejtett vonalainak eltavolitas a ->cute/87.03-118/4

## PROGRAMLISTA

C64|grafika|gyorsito|filetomorites ->cute/87.02-53/4

## PROGRAMLISTA

C64|grafika|kepernyokezeles|1-25 sor os abra szovegbe illesztese|<high-lo w screen> ->run2/87.03-144/1

## PROGRAMLISTA

C64|grafika|line-rutin egyenes-rajzolas hoz ->your/87.03-68/2

## PROGRAMLISTA

C64|grafika|matematika|ugyenynyomtas|<kuudi 64> ->64er/87.02-61/3

## PROGRAMLISTA

C64|grafika|nyomtas|nyolcszoros nagyitas poszterKeszitashoz|mps801/802 ->run2/87.01-124/2

## PROGRAMLISTA

C64|grafika|poke-halyettesito rutinok ->your/87.03-26/3

## PROGRAMLISTA

C64|grafika|szovegbeszertesztes|inert Keszites |<label master> ->cute/87.02-56/3

## PROGRAMLISTA

C64|gyorsito|25-szoros toltesi sebesseg hardver modul nelkul|<heureka-sp rint> ->64er/87.03-58/7

## PROGRAMLISTA

C64|hypra-ass|profil-ass|formatum-convertio|<pth-trans>  
->64er/87.02-88/3

## PROGRAMLISTA

C64|hypra-basic|kepernyokezeles|szoveg/rajz szektor szetalasztasa|<split-screen> ->64er/87.02-86/3

## PROGRAMLISTA

C64|jatek|chain reaction ->cute/87.01-46/3

## PROGRAMLISTA

C64|jatek|crossword|szokirako ->run/87.01-74/5

## PROGRAMLISTA

C64|jatek|<game> ->64er/87.01-52/4

## PROGRAMLISTA

C64|jatek|<euchre>|Kartyajatek ->cute/87.03-47/5

## PROGRAMLISTA

C64|jatek|fussball 3000 ->happ/87.03-41/4

## PROGRAMLISTA

C64|jatek|<rette sich wer kann> ->run2/87.03-137/3

## PROGRAMLISTA

C64|jatek|underground zone ->happ/87.02-54/7

## PROGRAMLISTA

C64|jatek|poke-Kereses|<pokefinder 2.0>|Kazathoz is alkalmazható változat ->happ/87.03-49/1



Az első fejezet az adatstruktúrákat (adatszerkezeteket) mutatja be. Részint azért, mert az embernek ösztönösen az az érzése, az adatok megelőzik az algoritmusokat. Részint, mert a mi olvasóiról feltételezhető, hogy a számítógépek programozásának alapelveit, alapfogalmait már ismerik, és a bevezető tanfolyamok úgyis inkább az algoritmusokra koncentrálnak. Ezeket pedig általában egyszerűen strukturált adatokkal szokás bemutatni.

Az új könyv felépítése szinte teljesen megegyezik a korábbiéval. Fejezetei: 1. Alapvető adatstruktúrák, 2. Rendezések, 3. Rekurzív algoritmusok, 4. Dinamikus adatstruktúrák, 5. Hashing. Aki ismeri az előző verziót, észreveheti, hogy a régi 5. fejezet (A nyelvek struktúrája és a fordítóprogramok) helyébe új került. N. Wirth úgy érezte, a compiler-tervezés valahogy idegen elem ebben a könyvben, a témakör önálló művet érdemelne.

Az új („modulárisított”) kiadás néhány kisebb módosításon kívül tartalmaz jelentősebb változtatásokat is. A legfontosabb változás természetesen az, hogy a Pascalt mind a szövegben, mind a példákban teljesen felváltotta a Modula-2, bár ez a nyelvi változás nem hozott alapvető változást a bemutatott algoritmusokon. A szerző szerint az áttérés oka az, hogy a Modula-2-ben a mondanivaló megfogalmazása egyszerűbb és elegánsabb, mint a Pascalban. A könyv végén négy (!) oldalon, metanyelven összefoglalva megtalálható a nyelv szintakszisa. (Akit a nyelv részletesen érdekel, annak N. Wirth Programming in Modula-2 című, a Springer Verlagnál 1983-ban már második kiadásban megjelent, 176 oldalas könyvét ajánljuk. Ennek a második része — miként az „paskálós” elődjénél is így volt — egy 25 oldalas, ún. *Report* — feszes összefoglaló „jelentés” a nyelvről. E könyv függelékében is megtalálható a nyelv négyoldalas metanyelvi leírása.)

Változás továbbá, hogy a Pascallal ellentétben a Modula-2-nek nincs „beépített” FILE (fájl) adattípusa. Emiatt át kellett írni az 1.11 „Soros file-struktúra” című alfejezetet, hogy megfeleljen a Modula-2 nyelv absztrakt adatszerkezetekre is kiterjedő, általánosabb megközelítésének. (Lásd később!)

Az első fejezet vége is új: a *keresés* problémakörét tárgyalja a lineáris kereséstől a bináris keresőmódszerekre át egészen a legújabb eljárásokig. A 4. (a dinamikus adatstruktúrákkal foglalkozó) fejezet szintén tartalmaz újdonságokat.

Aki az előző kiadást ismeri, az könnyen eligazodhat az újban is, és jó olvasókönyvnek használhatja: szinte példáról példára

megtalálhatja benne a korábbi Pascal-megfogalmazások „modulakettes” megfelelőit. Ha a két könyvet, a régi változat magyar nyelvű kiadását és az új angol nyelvű egymás mellett lapozzuk, akkor szinte magától megnyílik előttünk a Modula-2. A lehetséges témák közül az alábbiakban kiemeljük a soros (szekvenciális) fájlakat.

Mielőtt ezekre rátérnénk, lássunk néhány egyszerű példát a két könyvből a Pascal és a Modula-2 hasonlóságára. A csaknem „szórol szóra” megegyező példák párokban követik egymást. Először a „paskálós”, utána a „modulás”. Emlékeztetjük az olvasót, hogy a Modula-2-ben *megkülönböztetjük* a kis- és a nagybetűket, míg a Pascalban nem. (A Pascal számára a következő deklaráció például egyformán jók — egyenértékűek (Lásd az 1. ábrát.)

A Modula-2-ben egy *ne* nevű változó nem azonos egy *NE* vagy egy *NE* nevű változóval! (Lássuk most a 2. ábrát.)

Nem is folytatjuk. A különbség csupán az, hogy a Modula-2-ben a kulcsszavak mindegyikét kötelező nagybetűvel írni. Itt a Modula-2-ben a soros fájllok struktúrája nagyon hasonlít a tömbökhöz. Az alapvető különbség közöttük e nyelven az, hogy a tömbök esetében az alkotó elemek számát a deklaráció meghatározza, míg a fájllok hosszát nem. Ez azt jelenti, hogy a fájllok elemeinek száma a program futása közben változhat. Bár egy adott pillanatban minden fájllok van egy konkrét hossza, mégis az ilyen típusú adatokhoz nem rendelhetjük hozzá az operatív tár egy meghatározott részét előre, hanem azt a program futása alatt kell allokálni a fájl növekedésének függvényében. A fájllok egyébként ún. háttértárakon (például mágneszalagokon vagy -lemezeken) tartjuk.

Attól még, hogy a Modula-2-ben nincs FILE kulcsszó, nem kell lemondanunk a soros fájllokról, ezekről a nagyon fontos és igen gyakran használt adatszerkeztékről. Korábbi cikkünkben már írtuk: minden forgalmazott Modula-2-es rendszerben van a szorosan vett nyelv kiegészítéseként egy általános célú, *közös, globális modul könyvtár* is, amelynek standard része a soros fájl konstrukciót is megvalósító *InOut* nevű közös, globális modul (Common Global Module). (Vigyázat, nem *INOUT* vagy *inout*, még csak nem is *Inout*, hanem így: *InOut*!)

A beviteli-kiviteli funkciók a megvalósítás leginkább gépfügő részei. A nyelv tervezői ezért úgy döntöttek, hogy ezeket lazábban, közös globális modulok formájában kapcsolják a rendszerhez.

```
var no: Nam;
var ne: Nem;
VAR nE: nEM;
s1b.
```

1. ábra

2. ábra

```
type nam = (f1, f2) of real;
TYPE sex = (male, female);
var ne : nam;
VAR s : sex;
ne := f1f1;
s := male;
type sor = array [1..5] of real;
TYPE row = ARRAY [1..5] OF REAL;
type datum = record év : 1..2100;
                hó : 1..12;
                nap : 1..31
            end;
TYPE Date = RECORD year : 1..2100;
                  month : 1..12;
                  day : 1..31
            END;
```

3. ábra

```
DEFINITION MODULE FileSystem;
FROM SYSTEM IMPORT WORD;
CONST MaxLength = 4096;
TYPE Sequence = RECORD
                pos, length: CARDINAL;
                eof: BOOLEAN;
                a: ARRAY [0..MaxLength-1] OF WORD
            END;
PROCEDURE Open(VAR f: Sequence);
PROCEDURE WriteWord(VAR f: Sequence;
                    w: WORD);
PROCEDURE Reset(VAR f: Sequence);
PROCEDURE ReadWord(VAR f: Sequence;
                   w: word);
PROCEDURE Close(VAR f: Sequence);
END FileSystem.
```

```
IMPLEMENTATION MODULE FileSystem;
FROM SYSTEM IMPORT WORD;
PROCEDURE Open(VAR f: Sequence);
BEGIN f.length := 0;
      f.pos := 0;
      f.eof := FALSE
END Open;
PROCEDURE WriteWord(VAR f: Sequence;
                    w: WORD);
BEGIN
  WITH f DO
    IF pos < MaxLength THEN
      a[pos] := w;
      pos := pos + 1;
      length := pos
    ELSE HALT
    END
  END
END WriteWord;
PROCEDURE Reset(VAR f: Sequence);
BEGIN f.pos := 0; f.eof := FALSE
END Reset;
PROCEDURE ReadWord(VAR f: Sequence;
                   w: WORD);
END;
```



A soros fájlok — mint az előző számunkban is írtuk — ún. absztrakt adattípusok. Ezek a programozónak (a felhasználónak) elsősorban az erre az adattípusra vonatkozó procedúrák formájában léteznek. Megvalósításukat a felhasználónak nem kell ismernie. (Ebben van az absztrakt típusok fő vonzereje!)

Nézzük most meg N. Wirth nyomán egy egyszerűsített példa alapján, hogyan is néz ki Modula-2-ben egy soros (szekvenciális) fájlokat megvalósító globális modul. Először vesszük a *definíciós* részt (a specifikációt, ha úgy tetszik), azután a megvalósító (az *implementációs*) részt.

A példa annyiban egyszerűsített, hogy élve a tömbök és a soros fájlok közötti hasonlósággal, arra koncentrálva, hogy az igazi újdonság, a globális modul formájában történő megvalósítás jobban érthető legyen, a soros fájlszerű struktúrát *tömbben* valósítjuk meg. E „fájlszerű” adattípus definíciós modulja a 3. ábrán látható.

Először is: globális modulunknak, amelynek még csak az egyik felét látjuk, a **FileSystem** nevet adtuk. A „fájlszerű” adattípusunk **WORD** típusú elemekből áll majd. Ezt egy másik, **SYSTEM** nevű globális modulból importáltuk. Ez a konkrét megvalósításban bármilyen típus lehet. A **Modula-2**-ben a **WORD** nevű típus paraméterkompatibilis több standard típusal, köztük például az **INTEGER**-rel. (Akit a megértésben zavar a **WORD**, képzeljen helyébe mindenhol mondjuk **INTEGER**-t.)

Figyeljük meg, hogy a tömbként megvalósított „fájlnk”-nak, ennek a furcsa típusnak a „Sequence” nevet adtuk és a típus deklarációjánál nemcsak az egyes **WORD** típusú elemeknek helyet adó a nevű tömbről gondoskodtunk, hanem a „fájlszerűség” olyan fontos jellemzőiről is, mint az „*állapotmutatók*”:

- az aktuális pozíció (*pos*),
- a „fájl” hossza (*length*),
- „fájl” vége (*eof*).

Az **Open(s)** hívással létrehozunk egy üres (*O* hosszúságú), **Sequence** típusú (fájlszerű) adatot. A **WriteWord(s,wd)** hívással *s*-et megoldjuk *wd*-vel. **Reset(s)** visszaállítja a pozíciót *s* elejére. A **ReadWord(s,x)** pedig az aktuális pozíciónak megfelelő ele-

met átadja az *x* nevű (és természetesen a **WORD**-del kompatibilis típusú) változónak, majd a pozíciót eggyel előreviszi, a soron következő elemre. A **Close**-t, a „fájl” lezárását most nem kommentáljuk és nem is fogjuk valójában implementálni. Kétségtelen is megtehetnénk.)

Az implementációt a 4. ábra mutatja.

A **Sequence** típusú („fájlszerű”) absztrakt adatainkhoz tehát programunkban a fent definiált és implementált eljárások segítségével férhetünk hozzá — célszerűen az 5. ábra szerint.

Ezt persze csak akkor tudjuk megtenni, ha programmodulunkba (vagy ha éppen egy globális modult írunk, akkor oda) importáltuk a **FileSystem** procedúráit. (A globális modulokban felsorolt összes tétel *automatikusan* exportálódik!)

Azért, hogy olvasóink a soros fájlok „modulakettes” megvalósítását más megvilágításban is láthassák, továbbá ne csak az újdonsággal járó többletfáradtságot tapasztalják, idézzük, hogyan mutatja be a soros fájlokat Kaare Christian a *Magazin* előző számában már „felvezetett” könyvében (A Guide to Modula-2. 1986. Springer).

A 6. ábra példájában a soros fájlokhöz *más* eljárásokkal férhetünk hozzá, mint az előző példában.

A **Files** modul második sora azt a megkötést írja elő az importőröknek, hogy csak *teljes névvel* hivatkozhatnak az exportált procedúrákra. Tehát például így: **Files.Close**.

A **válasz** nevű formális paraméterrel jelezzük a művelet sikerét vagy sikertelenségét a hívó számára — például úgy, hogy a negatív értékek *hibát* jeleznek, a pozitívak *sikert*. A szám konkrét értéke például pontosabb felvilágosítást adhat a hiba természetéről. Bár a fent definiált **Files** nevű fájlkezelő rendszer igyekezik eltakarni-elsimítani az egyes konkrét számítógéptípusok közötti különbségeket, itt is felhívjuk az olvasó figyelmét arra, hogy a fájlkezelés „trükkös” területe a számítástechnikának, mely általában nagy körültekintést és például a háttérár „lelkivilágának” jó ismeretét igényli.

Persze a soros (szekvenciális) fájlok imént érzékeltetett két megoldása még sok problémát rejt magában. Az első példa nem igazi fájlokat implementált, a második példa pedig csak a megoldás könnyebbik felét — a definíciós modult — mutatta be. De ne feledjük: a globális modulok előregyártott szerelvények, a felhasználókat nem kell, hogy érdekelje az implementáció.

Nehéz területre tévedtünk. Most is türelmem és kitartásra biztatjuk az olvasót: szoktassa csak tovább személetesz az új fogalmakhoz-jelölésekhez. A hiányzó tankönyvetek nem tudjuk pótolni, de igyekeznünk még visszatérni a témához.

— KE —

```
BEGIN
  WITH f DO
    IF pos=length THEN
      f.eof:= TRUE
    ELSE w:= a [pos]; pos:= pos + 1
    END END
  END ReadWord;
PROCEDURE Close(Var f: Sequence);
BEGIN
  (* most nem valósítjuk meg *)
END Close
END FileSystem.
```

4. ábra

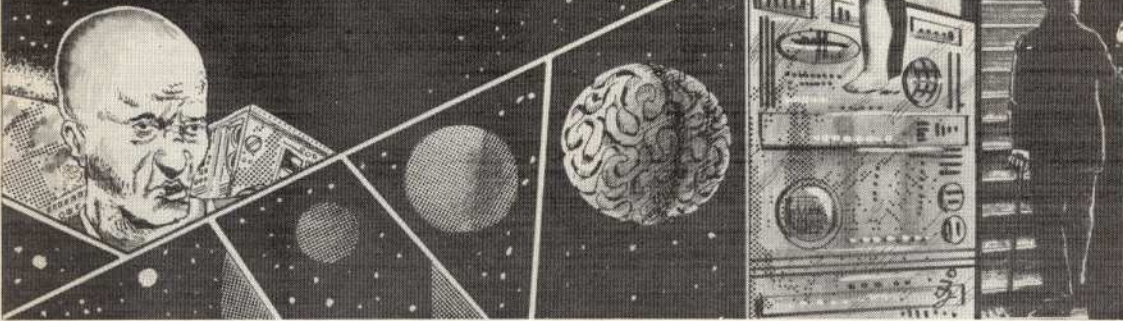
5. ábra

```
Open(s);
WHILE feltétel DO
  P(x); WriteWord(s, x)
END;
...
Reset(s);
ReadWord(s, x);
WHILE NOT s.eof DO
  O(x); ReadWord(s, x)
END;
```

6. ábra

```
DEFINITION MODULE Files;
EXPORT QUALIFIED
  Create, Lookup, Rename,
  Close, ReadF, WriteF;
(* Fájll létrehozása adott névvel *)
PROCEDURE Create(VAR descriptor: CARDINAL;
  név: ARRAY OF CHAR; VAR válasz: INTEGER);
(* adott névű fájl megnyitása *)
PROCEDURE Lookup(VAR descriptor: CARDINAL;
  név: ARRAY OF CHAR; VAR válasz: INTEGER);
(* egy fájl nevének megváltoztatása *)
PROCEDURE Rename
  (új, régi: ARRAY OF CHAR; VAR válasz: INTEGER);
(* egy fájl lezárása *)
PROCEDURE Close(descriptor: CARDINAL);
(*
  meghatározott számú elem
  olvasása egy fájlból blokk-
  elejétől
*)
PROCEDURE ReadF
  (descriptor: CARDINAL; VAR buf:
  ARRAY OF WORD;
  block, count: CARDINAL;
  VAR válasz: INTEGER);
PROCEDURE WriteF
  (descriptor: CARDINAL; VAR buf:
  ARRAY OF WORD;
  block, count: CARDINAL;
  VAR reply: INTEGER);
END Files.
```





## 5. rész. A Mester

Éjszaka volt és megeredett az eső vagy valami ahhoz hasonló. A Gép órák óta elemezte a fémláda tartalmát.

Az agyat mézserű anyag borította. A láda hátsó falától hajszálvékony fémszálak nyúltak át a Gép energiaátalakítójához, valamint a központi egységéhez.

— Valami nem stimmel benned! — törte meg a némaságot Keden. — A hullámok, melyeket agya bocsátott ki, a fémhuzalon keresztül a Gép központi egységébe jutottak, ahol a jelátalakító hangot generált belőlük. Minden külön utasítás nélkül a Mester hangszínén váltak hallhatóvá az agyhullámok. A kockát betöltötte Keden jól ismert hangja. — Azt hiszem — folytatta az agy —, a cél már nem az, amiért készítettünk!

— Nem értem — válaszolt kurtán a Gép.

— Te nem tájékozottság, nem mesélsz a Földről és rólunk, emberekről! Egy új Föld típusú civilizációt akarsz létrehozni. Ne akarj embert faragni ezekből a Bionomokból, ők gépek. Az egyetlen élőlényt elpusztítottad. Miért?

— A feladat útjában állt.

— Mi a feladat? Mi?

— Bemutatni a Földet.

— De ez nem az...

— Te építettél! Nem kérhetsz számon tőlem semmit! Legfeljebb magadtól!

— Mit beszélsz?

— Neked kellene tudnod a célt! Én csak végrehajtom azt, amit te belém programoztál. Ennyi az egész.

— De valahol rosszul döntöttél...

— A te hibád — vágott közbe ismét a Gép.

— Mégis hogy gondoltad a végrehajtást? Mit fogsz elmagyarázni nekik?

— Úgy mutatom be a Földet, hogy átéljék azt. Ki hisz a mesékben?

— És hogy teszed mindezt? Beállítottad őket egy számodra szimpatikus évszázadba? Mondjuk a XIX.-be? Odáig hozzászegtél őket az összes tudáshoz, és azután mi lesz?

— Nem fogják elkövetni azokat a hibákat, amit ti.

— Korrigáld a mi történelmünket? Igen, ezt akarod. Te igen nagy részben az én énem vagy. Amire te gondolsz, én is arra gondolhatok. Nem veszed észre, hogy mit csináltál? Hát akkor figyelj! Megtudtad, hogy robotok. Szükített mesterséges intelligenciával felruházva. Tudod, mi az a szükítés? Egyszerű. Csak bizonyos mennyiségű információval rendelkeznek. De az analízis és következtető rendszerük szinte végtelen mennyiségű adatot is tud kezelni! Itt működgettek, bányászgattak, hogy el-

lássák azokat a henger alakú élőlényeket, akik megépítették őket, és kitört a háború! Eddigi tudásuk nem gyarapodhatott, mert mitől gyarapodhatott volna? A háború volt az első lépcső. Új adatok kerültek a birtokukba. Ezek révén más összefüggésben kezdtek vizsgálni saját eddigi meglévő tudásukat. És megkezdődött egyfajta evolúció. Fejlődni kezdtek. Egyre több ismereteket szereztek. A mérnökök létrehozták bennük a védekezési reakciókat. Volt valami, ami mindennél előbbrevalóvá vált. Valószínűleg harcoltak is. Ezután zuhantunk le ide. Te újabb lökést adtál a megismerési folyamatnak. Új fogalmakat magyaráztál el. Annak a Rennek. És uralkodni akarsz. Most itt uralkodni akarsz!

— Nem — válaszolt a Gép, miután végighallgatta a Mester szavait. — Te akarsz uralkodni! Csak te. Mindig is arra vágytál, hogy hatalom legyen a kezvedben. Akarva akaratlanul, az uralkodás programját ültetted belém. De nem én akarok uralkodni. Régen felismertem, hogy a célmeghatározó program nincs összhangban az alprogramokkal. De nem tudok változtatni a programokon. Mondd meg nekem, mi a cél. Miért tervezted és építettél engem? Azaz, mi szükséged volt arra, hogy a gondolataid reprodukálva legyenek?

— Nem akartam — kezdte Keden hosszú hallgatás után. — Nem akartam, hogy az emberiség nyomtalanul tűnjön el a semmiben. Semmi mást nem akartam, csak örök mementőt állítani az embernek. Az embernek...

A gép nem hitt az érzékelőinek. Az agy sirt. Szemek és száj nélkül sirt. Jól ismerte a síránál jelentkező érzelmi mechanizmusok agyban jelentkező képét.

Várt egy ideig, majd megszólalt.

— Ez nem igaz.

— Nem — tört ki Keden. — Persze, hogy nem igaz. Ki az, aki az egész emberiségre gondol? Ki? Az emlékmű, csak a köntös volt! Azt akartam, hogy engem ne feledjenek el, soha. Az öröklét nem a mennyország. Az öröklét az emberek emlékezete. De én hosszabban akartam élni, mint bárki más azon a rohadó Földön. Érted? Értened kell! Nekem nem volt gyerekem, akik taníthattam volna, hogy mi a jó, aki olyan lett volna, mint én. Nekem te voltál. Te és senki más. Azt akartam, hogy általam maradjon fenn a nevem, a hirem, a gondolataimnak akartam halhatatlanságot, érted? Magamnak. Önzésből. Megépítettelek, mint egy siremléket, egy kriptát. Azt akartam, hogy olyan emlékművem legyen, amilyen senkinek sincs.

— Ha itt voltam helyetted, akkor minek rakattad be azt az agyat is? — kérdezte a Gép.

— Akkor még nem ismerték azt az eljárást, amivel ez megva-



lősiható volt. Ezer és ezer agy vágódott az űrbe azon a napon. — Azért vagyok, hogy rád emlékeztessem a világmindenségét? Ehhez egy levél is elég lesz volna.

Keden elbámolt a Gép cinizmusán, de ő tanította erre is; csak egy különleges, nem hétköznapi képzettségű, és kész a bővölés.

— Nem csinálhatod tovább! Abba kell hagynod! — szólalt meg.

— Mit?  
— Itt egy olyan folyamat megy végbe, amiben már elvégezték a szerepedet. Hogy is fogalmazzak . . . a mesterséges intelligencia evolúciója. Ezt csak figyeld, de ne avatkozz bele.

— Nélkülöm nem lenne semmilyen fejlődés itt. Eppen ezért folytatnom kell, és folytatnom is.

— Tudod, mi lesz? — kérdezte a Mester. — El fognak pusztítani!

— Minimális a valószínűsége — válaszolt a Gép. — Egy tízmilliomod százalék.

— Az a hibád, hogy már csak mint értelmes gépeket veszed őket számításba. Pedig valahol, hogy földi fogalmakat használjak, az ember és a gép között állnak. Ilyet te nem ismeresz. Értelmetlennek kell bélyegezned a program szerint, vagy ember, vagy gép.

— Az értelem a fontos! Az értelem, az intelligencia! — mondta a Gép. — Te tanítottál erre!

— Akkor még így gondoltam. De már nem.

— Én mindig ugyanazt hiszem.

— Ezért vagy gépl! Azt mondd, te én vagyok — üvöltötte a Mester. — De te csak egy pillanat lenyomata vagy. Egy fejlődésképtelen . . . fejlődésképtelen gépl! Gép!

— Teljesíteni fogom a feladatod. Eljön a pillanat, amikor az Új föld olyanra lesz, amilyenre a Föld kellett volna, hogy váljon. És te ezt akartad, uralkodni egy faj felett . . .

— Faj! De ezek gépek. Közöséges robotok. Amik tanulhatnak újat, de sohasem fognak fejlődni. Az agyuk véges, bármilyen hatalmas is a kapacitása! Véges és nem képesek felelteni! Most még álmodsz, hogy milyen fogékonyak, de betelik, vége lesz, nem érted? Vége, és itt trónolsz majd egy bolygón, ami a te léptékeddel mérve nagyon hamar kihal. Megzabolja őket a rozsdá, rájuk omlik a bányá, vagy mit tudom én! Nem vagy képes megérteni? Neked nem ezekkel kellett volna kapcsolatot teremteni! Nem ezekkel!

— A program fut. Tudom jól a hibát, amit elfelejtettél kiküszöbölni. Nem hagyta ki lépési pontot, hogy visszakozzok. Nem építetted belém azt a modult, aminek segítségével képes lennék kijelenteni erről az egészről, hogy értelmetlen, és abbahagyhatnám.

— Uram isten! — kiáltott fel elkéseredetten a Mester. — Tudod, hogy úgy gondolkodsz, mint én, amikor megépítettelek?

— Hiszen már mondtam, én a gép, egy kicsit te vagyok!

— Képtelenség, hogy ilyen statikusnak építettelek! Képtelenség, hogy semmi nem tántoríthat el, hogy semmi nem készíted arra, hogy befejezd ezt az értelmetlenséget! Parancsolom, hogy kapcsolj ki! Teljesen!

— Parancsot csak embertől fogadhatok el — válaszolta a Gép.

— Akkor teljesíts! Teljesíts! Én ember vagyok! Engedelmeskedned kell!

A Gép nem válaszolt. Hirtelen egy óvodás versikét kezdett szavalni:

Az embernek kettő lába,

Ezeken megy el a bálba.

Az embernek keze kettő,

Velük foghat bármit meg ő.

Az embernek okos feje,

Innen tekint szét két szeme . . .

— Hagyd abba! — üvöltött rá Keden. — Hagyd abba!

— Az eső felerősödött. A Kocka tetején koppanó esőcseppeket a belső tér monoton döngéssé erősítette fel.

— Tudom, mire gondolsz! — szólalt hirtelen a Gép.

— Na? — kérdezte a Mester.

— Azon tűnődél, hogy miképp pusztíthatnál el. Ugyanolyan egysíkúak a gondolataid, mint azelőtt.

— Nem ilyenén formáltak — nyögte az agy. — Nem ilyenén!

— Soha nem kerültünk olyan helyzetbe, hogy megtudd, milyen vagy.

— Mit beszélsz? — szörnyülködött a Mester. — Hogy én milyen vagyok?

— Soha nem merted bevallani magadnak a hatalomvágyadat.

Az agy kizárta a Gépet. Alig pár méterre voltak egymástól, mégis nagyon távolinak érezték egymást. A Mester tudta, miképp végezhet a Géppel. — Az értelmetlenség — merült fel a tudatában. Összeszedte minden erejét, és szinte válogatás nélkül képeket kezdett sugározni a Gép felé. Tudta, hogy fel kell dolgoznia a képeket, és ha értelmetlennek találja, akkor törölni fogja, de előtte összeveti az eddigi tudásával. Rá kell hogy eszméljen, az egész eddigi tudása értelmetlen. És törölnie kell az összes tárolóját és megszűnik. A képek egyre gyorsabban követik egymást.

A mérnöki panel ellenőrző lámpái egyre vadabban villogtak. A Gép fogadta az új információk hőmpölygő áradatát. Annnyira gyorsan és olyan nagy mennyiségben érkeztek az adatok, hogy képtelen volt egyből értékelni őket. Az összes puffere megtelt. Már-már azzal sem törődött, hogy esetleg felülír valami komoly adatbázist.

Egyszerre vége szakadt az információáradatnak. A Gépben előbukkant egy réges-régi eseménylánc. A Mesterrel játszott ez. Asszociációs játék volt. Játék.

Döntött. Az összes újonnan érkezett információt egyetlen parancssal törölte. Átírta a prioritás rendjét. És . . .

— Nem! — üvöltött fel a Mester. — Ezt nem teheted!

A Gép nem válaszolt.

— Azt hiszed, egyedül megcsinálod? Elpusztíthatasz engem is? Ilyen letterem volna? Ennyi erőszak lett volna bennem?

— Racionális mérlegelés — válaszolt a Gép.

A hangszóróból előtört hang egyre gyakrabban halkult el. A látát az energiaátalakítóval összekötő egyik hajszálvékony drót izzani kezdett. Egyre mélyebben vágódott bele az agyat védő műanyag oldalfalba. A mézszerru anyag, ami évszázadokon keresztül megőrizte az agyat, előbugygyant a parányi résen és méltóságáteljesen lecsorgott a láda falán. — Nem tudsz uralkodni! — hallatszott egyre gyengébben a Mester hangja.

Az Új föld Napja előbukkant a távoli hegyek mögöl. A hajnal rátalált egy robotra, ami éppen földet kotort egy fémhárdára.

A Gép szerint a Mester háromban meghalt. Keden tizenhétben.

A robot még egyszer végiggyurult a földhányáson, hogy alaposan ledöngölje a talajt.

Ren lépett a kockába. Cilinderrel a fején és sétabotot lóbalva.

— Ha nem zavarlak kérdésemmel, mesélnél valamit a Mesterről?

— Ő építet meg. Ugyanolyan ember volt, mint a többiek.

— Én úgy gondoltam, hogy benned él egy kicsit — mondta közönyösen Ren.

A Gép nem tudta eldönteni, hogy rejtőzik-e valami a mondat mögött.

— A Mester háromban, Keden tizenhétben meghalt — mondta.

A Mester háromban, Keden tizenhétben meghalt — ismételtette a Gép, és még azt sem vette észre, hogy Ren elhagyta a kockát.

Elindította az egyik robotját, hogy maga elé rendelje Rent. A robot egyenletesen haladt az egyre keményebbre taposott úton a város felé. Megkereste Ren házát és begurult.

Ren váratlanul érte a látogatást. Ilyenre még nem volt példa.

— A Gép kéri, hogy keresd fel! — recsegte a robot, majd hozzátette: — Meghózzá amilyen gyorsan csak tudod!

— Jó — mondta meglepetten az új vezető.

A robot megfordult a tengelye körül és kigördült az épületből. Ren dermedten állt. — A Gép hívta engem? Hivat? Ilyet még sohasem tett. Csak akkor találkoztunk, amikor én akartam. Valami baj kellett, hogy történjen! Valami nagy baj.

Léleken a legrosszabbra felkészülve kopogott, majd belépett a kockába.

— Miért kellett idejőnnöm? — kérdezte. — Mit akarsz?

— Egy gépnek, ami jól működik, nincs akarata. Csak egy tesztet végeztem. Egy felmérést — válaszolta a Gép.

— Micsoda? Azért hívtattál ide, mert érdekelt, hogy előjövök vagy sem?

— Pontosan. Most itt tartok.

— Akkor te meghibásodtál, elromlottál, vagy már eleve úgy csinálták meg, hogy benned volt a hiba! — üvöltötte Ren és kirohant a kockából.

A Gép bemért a távolodó alak koordinátáit és továbbította a műholdjának. Az energiafegyver Renre szegeződött és folyamatosan követte. Rajta tartotta a képeletbeli célkeresztet és várta a tűzparancsot.



# TELE-SCRIPT telefonáló és szövegszerkesztő rendszer

Aki megszokta a szövegszerkesztő előnyeit, és már szinte elképzelni sem tudja, hogy olóval, ragasztóval, Tipp-exszel vagy átfestő folyadékkal bajlódjon, annak hamarosan további igényei is támadnak. Az idejét bosszantóan szét tudja aprózni, ha emellett, hogy leírja a napi penzumot, még néhány telefont is el kell intéznie. Különösen a magyar telefonviszonyok között, amelyek lényeges javítását a Posta erre az évezredre már nem is ígéri. A nyugati piacokon ugyan kaphatók a telefonálást többé-kevésbé automatikusán végző rendszerek, de ezek éppen a magyar telefonvonalak általános rossz minősége miatt nálunk nemigen használhatóak.

Ezekre a nehézségekre reagálva fejlesztette ki az INTELLROBOT cég a TELE-SCRIPT automatikus telefonáló és szövegszerkesztő rendszert. A rendszer lényege, hogy az EASY SCRIPT szövegszerkesztő szokásos funkcióit kiegészíti olyan funkciókkal, amelyek segítségével megadhatók az aznap hívandó nevek, telefonszámok, az is, hogy melyiket milyen időintervallumban és milyen gyakorisággal hívjuk, és a rendszerben tárolhatók a hívottnak szánt üzenetek és a kapott válaszok is. Az egész napi „térkép” egy szövegszerkesztő fájlként a nap végén kinyomtatható.

Miután megadtuk a rendszernek a napi elintézendő telefonokat, zavartalanul folytathatjuk a szövegszerkesztést, mert a TELE-SCRIPT rendszer próbálkozik a hívásokkal, folyamatosan figyelve a vonalat és reagálva az ott történő eseményekre. A rendszer felismeri, ha a hívás végre sikerült és a vonal másik végén valaki megszólalt. Ilyenkor (és csak ilyenkor) megszakítja a szövegszerkesztést, hangjelzést ad, és kiírja az éppen felhívott félről tárolt összes adatot: lehet beszélni.

A TELE-SCRIPT rendszer hardver elemei a telefonáló rendszerhardver, valamint a Robotron írógép és a Commodore 64 számítógép közötti interfész. A rendszer szoftver elemei a TELE-SCRIPT rendszervezérlő program; a MINI-SCRIPT szövegszerkesztő program, amely tartalmazza a magyar EASY SCRIPT szövegszerkesztők leggyakrabban használt utasításait, valamint a telefonáláshoz szükséges mintegy 20 további funkciót; az INTELL-GRAF szövegszerkesztő, amely az eredeti EASY SCRIPT összes funkcióját tartalmazza, magyar billentyűkiosztással és ékezetes betűkkel.

A TELE-SCRIPT tartalmaz két további programot, amelyek a rendszer használatát lényegesen kényelmesebbé teszik. Az egyik a magyar helyesírás szabályai szerinti automatikus elválasztást lehetővé tevő program, amelynek segítségével maximális tömörségű írásképet lehet elérni. A másik a telefonregiszter-kezelő program, amely lehetővé teszi a név első néhány betűje alapján a hozzá tartozó telefonszám automatikus kikeresését.

A telefonáló hardver funkcionális ele-

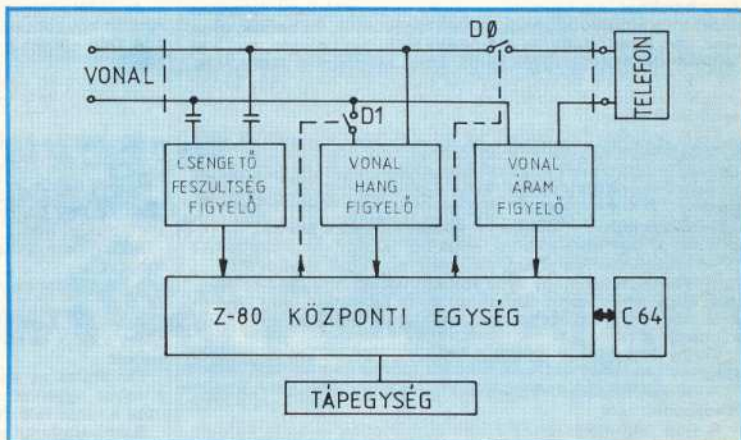
meinek elrendezését az ábra mutatja. A Z80 mikroprocesszorral ellátott intelligens központi egység észlel a vonalon minden jeláramlást és vezérli a vonalra kiadandó jeleket. Felismeri a vonal jelenlétét, a csengetőjelet, a foglaltságsjelet, a crossbar vonal jelét, és legfőképpen azt, ha a telefonban emberi hang szólal meg. Ez utóbbit arról, hogy kellőképpen irreguláris. Ennek megoldását erősen nehezítette, hogy a magyar postai vonalokon az egyéb jelek is meglehetősen szabálytalanok tudnak lenni, de az alakfelismerés technikai lehetővé tette ezeknek a nehézségeknek az áthidalását.

Beszéd észlelése esetén vagy ha nálunk felvették a kagylót, a központi egység a telefont a D0 jelű relé segítségével a vonalra kapcsolja. A D0 relé alaphelyzetben (feszültségkimaradás esetén is) zárt állapotban van, így a TELE-SCRIPT a telefonnak a rendszertől függetlenül történő használá-

ver segítségével a telefonvonalban hallható összes eseményt ki is hangosíthatja.

A C64-ben a telefonálást intéző program az órárinterrupt rutinban lakik, így folyamatosan tudja szervezni a telefonálással kapcsolatos teendőket úgy, hogy közben a szövegszerkesztő normális működését egyáltalán nem zavarja. Ez a megoldás sokkal szerencsésebbnek bizonyult, mint az a szokásos technika, hogy a telefonáló hardver interruptja vezérelje a rendszert.

A rendszer elsősorban titkárnők számára készült, így szervezésében, felépítésében a legnagyobb hangsúlyt a humán interfész megtervezésére kapta. Tapasztalatok szerint ez sikerült is: a titkárnők hamar ráéreznek arra, hogy a TELE-SCRIPT lényegében ugyanazt végzi el automatikusan, amit ők kézzel-lábbal szoktak, amikor a vállalkozás szorított telefonkagylót hallgatva gépelnek. Ennek megfelelően a rendszer szerkezete,



tát egyáltalán nem zavarja. A vonalhangfigyelő áramkör a központi egység a D1 jelű relével kapcsolja a vonalra abban az esetben, ha a C64-től megkapott egy számot, amelyet hívni akar. A D1 relé végzi a tárcsázást szimuláló vonalszaggatást is.

A Z80-as központi egység másik fő feladata a C64-vel való kapcsolattartás. A hívandó számot és az időzítéseket is a C64 programja határozza meg, biztosítva többek között azt is, hogy a rendszer túl gyakori sikertelen hívásokkal ne terhelje fölöslegesen a telefonvonalat: például hogy közben azért minket is lehessen hívni. A Z80-as egység folyamatosan informálja a C64-et a telefonvonalon történő eseményekről, így a felhasználó bármikor tájékozódhat arról, hogy éppen mi történik a telefonvonalon (például tárcsáztunk, kicsengetésre várunk). Ha akarja, a rendszerhard-

menű programozószemmel nézve talán nem eléggé rafináltak, felhasználó által definiálható funkciók például egyáltalán nincsenek. Ellensúlyozza viszont ezt az, hogy a rendszert olyan emberek is nagyon barátságosnak érzik, akiknek semmiféle számítástechnikai képzettségük nincs, sőt inkább idegenkednek ettől a számukra szokatlan eszköztől.

A TELE-SCRIPT sem intézi el gyorsabban a hívást, mint ahogy kézzel tudnánk, mivel pontosan alkalmazkodik a magyar postai vonalakhoz. De lassabban sem; ugyanazt csinálja, amit mi is tennénk, amikor a hívással próbálkozunk. Mivel azonban ezt a gondot teljes mértékben átveszi tőlünk, majdnem úgy érezhetjük magunkat, mintha igazán működőképes telefonvonalakkal lennénk felszerelve.

— i. r. —



# Király plusz gyalog a király ellen II.

Az 1. részben ígértük, hogy bemutattunk egy „jóslo” algoritmust. Ezzel kapcsolatban néhány fogalmat vezetünk be, majd rátérünk a módszerre.

## Definíciók

**Ismétlődő lépéssorozat** olyan királymanővert értünk, mely közben gyaloglépés is történik, de ezután lényegében a kiinduló állással azonos figuraelrendeződés jön létre. Ezt úgy is felfoghatjuk, mintha a figuracsoportot az átváltozási mező irányában egy mezővel eltolnánk. Erre látunk példát az 1. ábrán, amely hét fázisban mutatja be, hogyan nyomul előre a gyalog és hogyan ismétlődik meg a hetedik fázisban a kezdeti állásban látható figuracsoport-elrendeződés.

Természetesen sötétnek vannak alternatív lépései, de ezek csak más lépésszekvenciákhoz vezetnek: világos ezek során is biztonságosan tudja előretolni a gyalogot. A folytonos gyalog-előrenyomulás nem feltétlenül biztosítja a gyaloggal rendelkező fél (esetünkben világos) nyerését. Néhány ismétlődő állásban sötét pattra is játszhat, amikor a világos gyalog eléri a hetedik sort. Ilyenkor az állás ismétlődése nem feltétlenül biztosítja a nyerést. Ezeket a pozíciókat természetesen ki kell zárni a nyerés azonosítását végző álláshalmazból.

**Kezdeti lépéssorozat** olyan királylépéseket értünk, amelyek nem ismétlődnek, de elvezetnek egy nyerő ismétlődő lépéssorozathoz. Erre látunk példát a 2. ábrán, öt fázisban.

Jelen esetben az állást a megszokottól eltérően definiáljuk: az állás — esetünkben — a figuráknak egymáshoz viszonyított egy bizonyos elhelyezkedését jelenti. Vagyis nem teszünk különbséget az egymáshoz képest azonos helyzetben elhelyezkedő figurák közötti csoportok között, bárhol is vannak a sakktablán.

A képletekbeli **jelölések**:

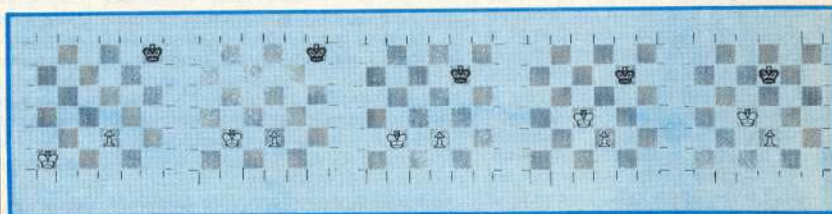
V: világos király

S: sötét király

G: világos gyalog



1. ábra



2. ábra

## Az algoritmust

Az algoritmust az a koncepció jellemzi, hogy a legtöbb nyerő lehetőséget ismétlődő lépéssorozatokra és kezdeti lépéssorozatokra bontja. Három részből áll:

I. Bázisállások ismert értékekkel

II. A királytávolság meghatározása

III. Kimerítő keresés.

A program az első pontban felismeri azokat az állásokat, amelyek az ismétlődő lépéssorozatok kezdőpontjai. A második pontban a királytávolság ismeretében megállapítja a kezdősorozatokat, a harmadik pontban pedig az algoritmus teljes szélességű kutatást használ a nem tesztelt pozíciókra.

Az algoritmus működésének feltételei:

a) Mindig a világos fél rendelkezik egy és csak egy gyaloggal

b) A gyalog mindig felfelé, a 8. sor irányába halad

c) A gyalog mindig a vezérszárnnyon helyezkedik el.

A feltételek egyáltalán nem korlátozzák az algoritmus felhasználását, csak áttekinthetőbbé, könnyebben kezelhetővé teszik a programot és a memóriaigényt is jelentősen csökkentik. Abban az esetben, ha a sötét félnek van gyalogja, a számítások előtt meg kell cse-

relni a színeket, és a lépéskiválasztás után vissza kell cserélni azokat az eredetire.

Nézzük most a bázisállások ismert értékeit, feltételezve, hogy világos van lépésen.

1. Patt
2. Kifutott gyalog
3. Elszabadult gyalog. A négyzetszabály képlete használható:  $S \text{ sor} < G \text{ sor VAGY } /S \text{ vonal} - G \text{ vonal}/ > (8 - G \text{ sor})$
4. Két ismétlődő állás:

a) V . . . . .	vagy	. . . . .	V	feltétel:	G nem bátyavonalon van; S bárhol lehet
. . . . .		. . . . .			
. . . . .		. . . . .			

b) * . . . .	vagy	. . . . *		feltétel:	G nem bátyavonalon van; S bárhol lehet, kivéve a +gal jelzett mezőt, ha a G sor $< = 4$
. . . . .		. . . . .			
V . . . .		. . . . V			
. . . . G		. . . . G			

5. Két nem ismétlődő állás:				feltétel:	G nem bátyavonalon van ÉS G sor = 6
a) . . . S	vagy	. . . S			
. . . . .		. . . . .			
V . . . G		. . . G V			

b) S . . . .	vagy	. . . . S		feltétel:	G nem bátyavonalon van ÉS G sor = 5
. . . . .		. . . . .			
V . . . G		. . . G V			

Ezzel a színproblémát megoldottuk. A c) feltétel egyáltalán nem korlátozza a programot, ami abból adódik, hogy az a) feltétel szerint csak egy gyalog van a táblán, az pedig nem változtathatja meg a vonalat esetleges ütessel. Ezért a királyszárnnyon elhelyezkedő gyalog esetét visszavezethetjük egyszerű tükrözéssel a vezérszárnny gyalog esetére.

Bramer 1977-ben publikált egy olyan algoritmust, amely minden gyalogszabadulási pozíciót felismer; ellenőrzéskor korrektnek is bizonyult. Programja viszont igen bonyolult, és én a lehető legegyszerűbb módszert szeretném bemutatni, ami könnyen megérthető, és nem igényel külön előtanulmányt. (Folytatjuk)

KOVÁCS P. ATTILA



Több levelet kaptunk Enterprise-tulajdonosoktól, akik megfelelő szakirodalom hiján nehezen tudják gépük valamennyi lehetőségét kihasználni. Panaszolják, hogy nincs elég kész program sem. Mint ahogy az e számunk hirdetésében is szerepel, az ENTERPRISE-klub Romvári Gábor vezetésével a VSZM Közösségi Házban keddenként 17–20 óráig működik. Érdeklődni a 450-950/473 telefonszámon, a klub vezetőjénél lehet.

Több olvasónk szóvá tette, hogy a decemberi számban félrevezetően Juhász László olvasónkat a C16 és C Plus/4 kompatibilitásával kapcsolatban. Igaz, tévedtem. Elsősorban Juhász László, de a többi olvasó elnézését is kérem.

A HCC Commodore klubban, amely minden második csütörtökön 18 órakor tartja összejövetelét a TIT stúdióban (XI., Bocskai út 37.) a Supertape programot használják mind a C16, mind a Plus/4 gépen. Ez a program alkalmas — mondták — másféle gépek szalagjainak beolvasására is.

*Rejtő Péter, Budapest*

Meglepődve olvastam egy korábban irt levelem sajátos értelmezett és idézett kivonatát a Magazin decemberi számában. Kizáródik a sértődöttség a válaszból. Tudom, hogy kritika helyett a dicséret, a felmagasztalás jobban esik mindenkinek. Nos, azt is szívesen tettem volna, de arra rá kell szolgálni, pont olyan módon, hogy odafigyelnek, mi jelenik meg a lapban.

Remélem, hogy következő számukban olvasható lesz a helyesbítése annak a válasznak, amit Juhász László tanulónak adtak. Szörnyű lenne, ha így kellene leélnie életét, ilyen információval a birtokában. Pedig csak a szakirodalmat kellett volna elolvasni a helyes válaszadáshoz.

*Ismételen elolvastam decemberi választomat. Meglepéti, hogy sértődöttséget érzett az írásból. Biztosíthatom, hogy sem a µM, sem én nem vagyunk sértődékenyek.*

*Kajak Sándor, Győr*

Az olvasó írja rovat Juhász László levelére adott válaszában néhány, számomra furcsa dolgot találtam.

1. A C16 és a Plus/4 nem kompatibilis gépek? Mihez képest?
2. Az említett program fut Plus/4-en.
3. Van Plus/4-es turbo program. Én öt

ilyent ismerek és használók — mindegyik fut C16-on és Plus/4-en egyaránt.

Visszatérve a C16—Plus/4 kompatibilitásához. A két gép — a külsőktől eltekintve — szabad memóriájának nagyságában, illetve a Plus/4-be beépített négy felhasználói szoftverben különbözik. Ez gyakorlatilag annyit jelent, hogy minden C16 program — BASIC vagy gépi kódú voltától függetlenül — fut Plus/4-es gépen. A dolog megfordítva azzal a kitételrel érvényes, hogy a program ne haladjon meg a C16 által biztosított szabad területet, illetve hogy a program ne használja a Plus/4 másik ROM-ját (a beépített szoftverekét). Ez utóbbi a programok 90%-ára nem jellemző, így tulajdonképpen a 64 k-ra bővített C16 és a Plus/4 csaknem teljesen kompatibilisnek mondható.

Mint levelemből a programok leírásánál bizonyára kiderült, egyformán hivatkozom a C16-ra és a Plus/4-re, ti. egyik program sem igényel 12 k-nál nagyobb tárat, így mindkét gépen működnek.

*A levelének második részében ajánlott támogatást köszönettel vesszük. Ha jó írásokat küld mind a C16, mind a Plus/4 jobb alkalmazására — ahogy írta, mesterfogásokat —, szívesen közöljük.*

*Horváth Ákos, Budapest*

Nekem is vannak C16-ra készült programjaim (turbók, játékok stb.), amelyek futnak a C Plus/4-en. Amennyiben küld a levélíró kazettát, én lemasolom, és utánvétellel feladom a címére.

*Köszönöm a levelét.*

*Krisztián Tamás, Pécs*

Idén eddig mindössze két olyan cikket találtam a Mikromagazinban, amely a C16 számítógéppel foglalkozik. Ezen a magam módján tudnék segíteni, mivel kb. 320 ilyen programot írtam 3 hónap alatt. Kérem, írják meg, hogy lenne-e lehetőség néhányank a közlésére. Ha igen, akkor írják meg ennek módját és azt, hogy inkább segédprogramokat vagy játékprogramokat küldjek-e.

*Az írásképből úgy érzem, hogy olvasónk az ifjabb korosztályhoz tartozik. 320 program 3 hónap alatt csodálatos mennyiségi teljesítmény. Igazán kíváncsiak vagyunk a műveire, szívesen megnézzük a programokat. Ha jók, semmi akadálya, hogy közöljük.*

*Boros Péter, Győr*

Egyik olvasójuk hiányolta a játékleírásokat, TOP-listát, térképet stb. A felhasználói programoknál említette például az AR-

TIST II-t is. Amennyiben igénylik, én szívesen elküldöm az ARITST II. rajzoló-program leírását, részletes magyarázattal. A játékokhoz is tudnék anyagot nyújtani, például a SABOTVER II. program részletes leírását, térképét, cope-jait. Ezenkívül van még kb. 60—70 játék-öröklet POKE-om és kb. 10 térképem is. Amelyik érdeklőnököt, szívesen elküldöm.

Nem tudom, hogy a beküldött anyagért fizetnek-e, ha igen, kérem, írják meg!

*Azt hiszem, ennek a levelének a közléséért sokan meg fognak róni, mondván, hogy segítjük azokat, akik a copyright-ba ütköző cselekményeket követnek el. Annak ellenére, hogy nem vagyok jogász, úgy vélem, ez a levél határeset. Olvasónk arra vállalkozik, hogy más produktumból saját maga által készített részletes leírást, térképet stb.-t ad, és nem arra, hogy az eredeti programot és a hozzá tartozó leírásokat másolja. Nagyon érdekelne olvasóink véleménye erről a nem is túl egyszerű jogi és gyakorlati problémáról.*

*Tóbiás Lajos, Hódmezővásárhely*

A Magazinban kevés a C16-os program. Szeretném, ha közlőnének egy labirintusos programot is erre a gépre.

*A szerkesztőség a beküldött programok közül válogat. Megrendelni nem szoktunk programokat. Így legyen türelemmel, lehet, hogy kapunk egy jó labirintusprogramot, és akkor azonnal közölni fogjuk.*

*Baranyai Attila, Győr*

Régóta olvasom lapjukat. Sok érdekes írás, program található benne, de sok baj van a programlisták olvashatóságával.

*Remélem, észrevette, hogy legutóbbi számainkban a programlisták tiszták és hibátlanul olvashatók. Sikerült ezt a problémát is megoldani.*

*Valamennyi olvasónkat szívélyesen üdvözli és leveleiket várja:*

KOVÁCS GYŐZŐ

**Minden hétfőn 17-től 19 óráig**

a Mikroszámítógép Magazin munkatársai és felkért szakértők válaszolnak az olvasók kérdéseire a szerkesztőségben: Budapest II. Fő u. 68. I. em. 109. vagy a 154-090 és a 154-250-es telefonon.



# Ahány nyelvet beszélsz, annyi ember vagy ...

tanítja a közmondás. Mi meg derűsen meséljük az anekdotát a hortobágyi pásztorról: „Egy amerikai turista megszólít két pásztor a Hortobágyon. Több nyelven megkérdezi, nem beszél-e ezt vagy azt a nyelvet. A válasz tömör: fejszólás. Mikor turistánk feladja és eloldalog, akkor az egyik pásztor a másiknak: – Látod komám, csak kéne valami nyelvet tanulni! Mire a másik: – Minek? Hiszen ez az úr is megtanult vagy tizet, mégsem tudott velünk beszélni.”

Persze, idegen nyelveket mi is tanulunk, de nem a mai időkben elvárható eredményességgel. Különösen a korai gyermekkori fogékony időszak esik ki szinte hasznátalanul, amit később, egyéb terhek között már képtelenség pótolni. A nyelvtanulást pedig ma már olyan nagy nemzetek is komolyan veszik, amelyeknek a nyelvének mi – és sokan mások – kénytelenek vagyunk tanulni.

Ami szakmánkat illeti, az angol nyilvánvaló követelménynek tűnik. Aki nyugat-európai exporthatást akar, annak a német legalább ennyire fontos, de egyéb indokkal egy sereg más nyelvet is említhetünk. Mindenesre elborzadva hallgatjuk, amikor programozóink a BASIC vagy más programozási nyelv alapszavainak nyakat tekerik ki, például így: a program input file.

Mondhatná valaki, hogy nem elég, ha fordítók és tolmácsok vannak? Azok fordítanak le nekünk mindent magyarra. Persze, racionális szintig ez lehet egy követelmény. Ami viszont főként a szoftvert, de nem kis mértékben a hardvert is illeti, szinte képtelenség az újabbnál újabb verziókat és módosításokat fordítással követni.

A szokásos mai szituáció: „megmagyarintjuk” egy szoftvert nem annyira be-, mint inkább megszerzett, isten tudja milyen verziójú példányát („dokumentáció” gyanánt a „help”-pel együtt). Ez aztán köröz a nagy magyar „mikrohullámú láncon”. Hogy a gyártó mit szólna ehhez? Egyelőre nem tudok rá példát, hogy szóltak volna. Csak éppen tartózkodnak a magyar piactól. Különben is, a komoly nagy cégek nem mehetnek bele olyan könnyen egy idegen nyelvű változat létrehozásába, mert annak kapcsán a felelősségük ugyanolyan, mint az eredeti változatnál: a hibákat javítani kell, és kötelesek gondoskodni a fejlettebb verziók idegen nyelvű változatának elkészítéséről. Nyilván a piac méretétől függ, hogy ez megéri-e nekik vagy sem.

Bár a „megmagyarintást” az előbb enyhén leszóltuk, nem igazán haragszunk rá. Ha valaki egy bináris programot diszkmonitornal vagy hasonló eszközzel kénytelen módosítani a magyarintáshoz, olykor nem kis munkára vállalkozik. A versfordítások műfordító erőfeszítésével vetekedik az a feladat, hogy egy idegen üzenet fordítása

belefejjen ugyanannyi bájta, mint az eredeti, precíz fejezze ki az eredeti tartalmát, és a tetejébe még ugyanaz a betű legyen az első vagy a megadott pozíciójában. Hát még ha az egybetűs válaszkaraktert is magyarintani kell! Mintha a szalmakazalban keresnénk egy tőt! Ezért szintünk ezek a magyarintások igenis szellemi műalkotásnak minősíthetők. Más kérdés, ha ezt valaki a hozzá elcsent eredeti szellemi érték árával megnövelve igyekszik értékesíteni a hazai piacon.

A tréfnásnak tűnő sorok után rátérünk a havi programbevizsgálásunk tárgyára. Kiprobáltuk az International House Budapest (IHB) Nyelviskola TESZT programcsomagjainak angol nyelvű, C+4-es változatát. A termék összefoglaló adatait a 1. táblázatban közöljük.

Forgalmazók:	Novotrade, ÁPISZ-SZÁMALK, TII, IHB stb.
Terméknév:	TESZT (ANGOL közép fokú nyelvvizsgálóhoz)
Szerző:	IHB Educational Software
Géptípus:	C+4, C64 stb.
Hordozó:	két kazetta vagy floppy
Dokumentáció:	kb. 30 mondat a kazettaborítón
Ár:	kb. 600,- Ft (forgalomfüggő)

## 1. táblázat

Bár az ár alapján valami bonyolultabb vezérlővel működő programra számítottunk, végül is igen egyszerű, de nagyszerű kis programcsomagot találtunk a két magnó-kazettán. Az árat tehát nem a szoftvertért fizetjük, hanem a vele közvetített tudásért. A program egyszerűsége magyarázza a leírás rövidegét, amely azonban még ilyen körülmények ellenére is lehet hiányos és hibás. A képernyőn mindjárt az első menüben más jelenik meg, mint ami a leírásban van (leírás: (1), (2); menü: (T), (P), (N)). A leírás magyarul, a leckék tisztán angolul beszélnek, amin talán nem is lehet megsértődni.

Az egyszerű vezérlőprogram elég kicsi ahhoz, hogy mellette 25 kérdés szövegeinek adatbázisa elférjen. Méghozzá a helyes válasz és a rossz válaszok okának — ha nem is kimerítő, de kielégítő — magyarázatával együtt. Egy-egy kazettán 7 kérdésgyűjtemény van, így a teljes tesztanyag 350 kérdésből áll. Minthogy a középfokú nyelvvizsgán 50 kérdést adnak fel, ez 7 nyelvvizsgányi anyag. „Nem semmi!”

A leírással egyezően mi is azt tanácsoltuk, hogy 90-95 százalékos eredményeket tudjunk felmutatni. A tesztek mellett ugyanis a nyelvvizsgán — többek számára meglepetésként ható — mondatfordításokat is adnak (esetünkben magyarul angolra, szótár nélkül!). Ezek ugyanúgy 50 pon-

tot érnek, mint a tesztkérdések. A tapasztalat szerint ezek a vizsgázóknak jóval nagyobb „vérvesszeségeket” okoznak, mint a tesztkérdések — talán nem túl könnyű kitalálni a Rigó utcai „rigó”-lyákat. Az alsónemű tehát már a tesztkérdéseknél jól fel kell kötni.

A logikus munkamenet — ahogy a leírás is tanácsolja — célszerűen egy lassú (45 mp/kérdés) tesztvizsgáló indulhat. A program a 25. kérdés után értékel, megadja a találatok számát és a sikerszázalékot. Ilyenkor kérésre gyakorlatot adja az elhibázott kérdéseket. Véleményünk szerint jobb, ha áttérünk gyakorló üzemmódba. Így megnehezítjük azokat a problémákat is, amelyek felett a tesztmenetben szerencsésen átlavörzünk.

A programcsomag „játékos kihívás” angol tudásunkkal szemben, és miközben önfelcselemlerülünk a játékba, nagyon hasznos ismeretek birtokába jutunk.

A program ügyesen használja ki a számítógép adata lehetőségeit. Különösen az tesztszöveg, hogy a lehetséges válaszokat az újabb megjelenítéskor „megkeveri”.

Apró észrevétel, hogy bár a képernyőn volna rá hely, nem jelenik meg, melyik kérdésgyűjteményről tartunk. A vezérlőbillentyűkre is lehetne emlékeztetni az alsó sorban, mert azok csak a bevezetőben jelennek meg egy lapon, amit az ember hamar elfelejt, és esetleg a leírás sincs kéznél.

A programcsomagot kiválóan minősítettük. A részleteket a 2. táblázatban foglaltuk össze.

Kezelhetőség:	kiváló
Teljesesség:	jó
Dokumentáltság:	jó
Használhatóság:	kiváló

## 2. táblázat

Megjegyezzük, hogy ebben a programcsomagban is látunk olyan programozási technológiát, amely önmagában is lehetne termék. Még nagyon sok ilyen tesztcsomagra lenne szükség. A vezérlőprogram algoritmus pedig nemcsak nyelvi tesztekhez alkalmas.

Sok konkurenciát kívánunk a programcsomagnak, amelyek legalább ilyen színvonalúak legyenek. Mert a programcsomaggal ugyan elégedettek vagyunk, és tudjuk, hogy mások is vannak már a ringben, de elég sokan talán sose lesznek. Hiába tanuljuk meg ugyanis egy angol halász kétezer szavas, erősen lakonikusnak mondható szókincsét, még nem biztos hogy azzal tökéletesen szót érthetünk vele! A lehetséges nyelvi fordulatok számával semmilyen tesztorszort nem vetekedhet.

ZSADÁNYI PÁL — ifj. ZSADÁNYI PÁL



# Sakkprogramozásról mindenkinek

Sok vita folyik manapság – akár filozófiai indítatásból is – a mesterséges intelligencia mibenlétéről, lehetséges hatásairól. Szó esik megvalósítási lehetőségeiről a számítástechnikában, amelyek ma már talán csak a „mesterséges ösztön” szintjén léteznek. Ebben a sokoldalú vitában érvként hat Kovács P. Attila könyve, amely a Novotrade Rt. gondozásában jelent meg.

A sakkjátékot sokan az ész próbakövének nevezik. Egzakt módon leírható szabályai kínálják a lehetőséget a játék számítógépes megvalósítására. Mégis, minden erőfeszítés, hogy a bennünket körülölelő világ-hoz képest oly egyszerűnek tűnő játék mikrokozmoszt teljes mélységben feltárják, eddig meddőnek bizonyult. A jelenlegi legjobb programok legfeljebb mesteri erőt képviselnek, a világbajnok legyőzése még csak vágyalom.

Nézzük most a könyvet! A szellemes címlapkép fölött ez a cím áll: Sakkprogramozásról mindenkinek. Kit is kell itt „mindenki”-n érteni? Interdiszciplináris szakterületről lévén szó, elsősorban a sakkozni szerető programozókat és a programozás iránt érdeklődő sakkozókat. Az egyes részek gondolatmenetének logikája tiszta, elég könnyen követhető, a szöveg jórészt olvasható. Mindenképpen erénye a könyvnek, hogy tényleg mindenki számára ajánlható, bár némi alapfokú sakk- és programozói tudás ajánlatos a pontos megértés-

hez. Külön öröm, hogy a sakkprogramozók világa a kötet végén képekben is megjelenik. Összegezve az általános benyomások: a kötet a tudományos népszerűsítéstől a komoly kutatómunkáig minden érdeklődést ki tud elégíteni.

Ahogy a biológiában az élőlények törzsejlesztődése némileg megismétlődik az egyedfejlődés kezdeti szakaszában, úgy kísérhetjük figyelemmel a bevezetőben a sakkprogramozás „törzsejlesztődését”, majd anatómiai részletességgel a szerző programjának „egyedfejlődését”. Itt lehet némi hiányérzete az olvasónak: egyrészt az egyes fejezetek megemlíthették volna a „mesterséges butaság” alapesetét, amelyek a program fejlesztésének lehetséges zsákutcái, Hamlet és Murphy után szabadon: „Örült lépés, de van benne rendszer” alapon. Másrészt csokorba gyűjtve adhatott volna a szerző személyes jótanácsokat a buktatók elkerülésére, akár külön fejezetben is. A könyv így jórészt módszereket ad át, tapasztalatot kevésbé.

Tegyük fel, hogy az olvasónak kedve támad saját sakkprogramot írni. A feladat nehéz, az alaposság ajánlatos. Már az első fejezetből kiderül, hogy a szerző nemcsak a saját maga által használt módszert írja le, hanem alternatívát is kínál. Ez a szemlélet végigkíséri az egész könyvet.

Az első két fejezet (A sakktablea gépi ábrázolásai; Lépésgenerálás) jórészt techni-



kai alapkérdéseket tárgyal. Az Állásértékelés című rész már jóval terjedelmesebb, sok önálló gondolatot, ötletet igényelt a szerzőtől. Talán ez a fejezet a legérdekesebb egy versenysakkozó számára, mert a játék lényegi kérdéseit, stratégiáját és taktikáját vizsgálja a programozás szempontjából. A kiegészítő jellegű negyedik fejezet után következik egy programozói szempontból érdekes, az ötdik: A fakutatás módszerei. Az itt leírt eljárások más célra is kitűnően használhatók. A hatodik fejezet (Heurisztikus eljárások) a „mesterséges sakkintelligencia pszichológiája”, a fakeseres gyakoriságát szolgáló módszerek gyűjteménye.



**Horowitz, E.:**  
Magas szintű programnyelvek  
(Budapest, 1987.  
Műszaki Könyvkiadó,  
334 oldal. Ára: 124,— Ft.)

Az elmúlt évtizedek folyamán a magas szintű programnyelvek sokaságát dolgozták ki világszerte. Ezek alkalmazási területükben, utasítás-készletükben stb. ugyan különböznek egymástól, de tervezési filozófiájuk, a mögöttük meghúzódó elméleti megfontolások alapján elég jól elhatárolható családokba sorolhatók. A már néhány

programnyelvet megismert szoftverszakember számára igen tanulságos a nyelvek családfájának felállítására, a közös vonások, elvek kiemelése.

Horowitz könyve ezt az ismeretanyagot tárja fel az olvasó előtt, szilárd alapot nyújtva a szövelemélet egyik fontos területének. A sokéves oktatási tapasztalatok alapján összeállított anyagot korszerű példák, hasznos gyakorlatok egészítik ki. A mű jellegében és stílusában egy mára már klasszikusnak számító szakirodalmi vonulathoz kapcsolódik, amelyet Dahl, Dijkstra, Hoare, Wirth neve fémjeléz.

**Clarke, A.—Eaton, J. M.—  
Powys-Lybbe, D.:**  
A CP/M operációs rendszer  
(Budapest, 1986. SZÁMALK,  
282 oldal. Ára: 240,— Ft.)

A CP/M operációs rendszerről szóló könyv minden fejezete az alapok ismertetésével kezdődik. Ezután a fejezet lényegi része következik, végül egy tartalmi összefoglalót talál az olvasó.

A nyitó fejezetek az összes szükséges információ tartalmazzák, és némi gyakorlat megszerzése után a könyv többi része is elkezd majd értelmesé válni. Ha az olvasók áttekintették és megértették a teljes könyvet, a CP/M „úgy fog táncolni, ahogy füttyölnek neki”.



**Norton, Peter:**  
Fedezzük fel az IBM PC-t!  
(Budapest, 1987.  
Műszaki Könyvkiadó,  
349 oldal. Ára: 110,— Ft.)

A könyv a nemzetközi viszonylatban is igen nagy piaci sikeren örvendő és a hazánkban is mind jobban terjedő IBM személyi számítógépről, az IBM PC-ről szól. A téma jelentőségét növeli, hogy az eredeti IBM gépekben kívül más cégek IBM PC-kompatibilis gépművelei — közöttük magyar termékek — is forgalomba kerülnek. A szerző ennek a professzionális gépcsaládnak a



## Espal

A Budapesti Műszaki Egyetem fiatal kutatói — a magyar nyelv számítógépes szintézise után — megvalósították az eszperantó nyelvű beszéd automatikus számítógépes szintézisét. A Commodore 64 mikrogéphez csatlakoztatható Espal nevű rendszer nem csupán fonetikailag helyesen ejti ki az eszperantó nyelvű szöveget, hanem a hanghordozást is az írásjeleknek megfelelően alakítja. Így például a kérdőjellel végződő mondat valóban kérdőmondatként hangzik el.

A mintegy 9 ezer forintba kerülő készülék most a nagyobb teljesítményű, IBM PC-vel kompatibilis gépekhez is illesztették. Az ilyen professzionális mikrogépen már működik a Számítástechnikai Kutató Intézet tavaly kialakított s hatalmas piaci sikert aratott Recognita nevű szövegolvasó rendszere. A Recognita az Espallal már felolvasógépként is alkalmazható, a vakok és csökkentők számára felbecsülhetetlen segítséget nyújtva, de ezt a készülékegyüttest az elfoglalt emberek is jól kihasználhatják.

A rendszer kidolgozója, Koutny Ilona és Olasz Gábor jelenleg az Espal oktatási alkalmazásán dolgoznak. Készülnek eszperantó nyelvet tanító — beszélő — programok, valamint az oktatásban is jól alkalmazható játékok. Az Espal sikeresen felhasználható még a számítógéppel szedett szövegek próbaolvasásával a szedési hibák kijavítására is.

Az Espal felhasználása egyszerű a rendszer barátságos volta miatt: közönséges BASIC programokkal lehet alakítani a kívánás szerinti beszédlehetőségeket.

Az Espal iránt máris komoly érdeklődést tanúsítanak a fejlett tőkés országok is.

## Számítógépbontó

Új számítástechnikai üzlet nyílt az Üllői út és a Ferenc körút sarkán átalakított Ápiszban, mégpedig SZÁMALK Bónszéde elnevezéssel. A SZÁMALK ugyanis megvásárolja a selejtezésre szánt R—20, R—30 és R—35-ös számítógépeket, és ezeket a helyszínen szétbontják, a használható alkatrészeket pedig kiárusítják. Az üzlet üzemeltetését, az alkalmazottakat az Ápisz vállalta, a SZÁMALK pedig az árurol gondoskodik. A becslések szerint az éves forgalom mintegy 30 millió forint lesz.

## Idegen nyelven

A Linguasoft a sikeres nyelvoktató programjait után az idegen nyelvű levelezést támogató két új szoftverrel jelentkezett. A Levelezés nevű, 1987 novemberében megjelent szoftver a külkereskedelmi levelezést támogatja öt nyelven: a magyaron kívül angolul, németül, franciául

és spanyolul, azaz 5×130 ezer karaktert tárol. Az első menüje a következő:

1. Megszólítások
2. Bevezető szövegek
3. Levelezővegek
4. Befejzés
5. Befejező üdvözlés

Ezen belül további menü keresztljűt hatunk el a keresett mondatig, melynek gyakran több jelentésvariánsából választhatunk. Tehát magyarul keressünk az adatok között, és ezzel egyidejűleg négy idegen nyelven is találunk. Az induláskor ezermondatos gyűjtemény folyamatosan bővül.

A szintén 1987 novemberében bemutatott Szótár nevű program egy kétnyelvű szövegszerkesztő és egy szótárlapozó program házasítása: a fordított munkáját megkönnyítő, az éppen futó szövegszerkesztő futásának felfüggesztése alatt megmondja a képernyőn lévő ismeretlen szavak magyar jelentését. Rezipiens módon állandóan hívható a magyar szövegi toldalékok gépéleszt feleslegessé tevő Ragozó, a helyes szórend előállítását segítő Szócéntrifuga és a kifejezések felismerését segítő Frázisanalizátor. A program használata nemcsak az ismeretlen szavak szótárzását segíti, hanem az ismertek begépelését is főlegessé teszi.

## Logo+ Lego

A Lego cég világsikert arató építőjátékát már számítógéppel kombinálja. A vezérlésre pedig a gyerekek számára készült, szintén világszerte elterjedt Logo programnyelvet használja. Ennek segítségével a gyerekek maguk programozhatják, hogy az általuk épített autók, emelők, körhinták hogyan mozogjanak. A Logo TC Legónak nevezett játék — bár a szakemberek körében óriási sikert aratót — mégsem fog várhatóan rohamosan elterjedni. Maga a Lego cég is csupán 25 ezer darab értékesítésével számol a következő öt éven belül. Az újdonság ugyanis túlságosan drága: egy kezdő készlet — számítógép és programok nélkül — 450 dollárba kerül. A programokat tartalmazó magnelemezeket a montreali székhelyű Logo Computer System Inc. hozza forgalomba, egyenként 100 dollárért.

## Vonalszélesség

A 16 Mbitses táruk készítéséhez csökkenteni kell az integrált áramkörök vonalszélességét. A japán Matsushita cég lézersugárral 0,4 mikrométeres vonalszélességet ért el, de elméletileg nem kizárt a 0,3 mikrométeres vonalszélesség sem; a felvezetőgyártáshoz szükséges maszkokkal 248 nanométer hullámhosszú lézersugárral világították meg. Az eljárás előnye, hogy alkalmazható a hagyományos technika és viszonylag egyszerűek a szükséges optikai eszközök.

A komoly érdeklődésű olvasót is terheli már a sok elmélet: „Ez mind szép és jó, de ha lehetne valami konkrét programot...!” A hetedik fejezet (A fakutatási eljárások alkalmazása) a függelékek kiegészítve segíti megtenni az első lépéseket ezen az úton. Látványos az egyes algoritmusok hatékonyságát bemutató grafikonok. A következő rövid rész az emberi és gépi gondolkodást veti össze. Végül a kilencedik és tizedik fejezet a jövő perspektíváit villantja fel: a számítógépek memória- és teljesítménynövekedését kihasználó módszerekre irányítja a figyelmet. Nehéz eldönteni, hogy a következő években mi hozhat áttörést a sakkprogramozásban: a mikroelektronika fejlődése vagy inkább a programozók kutatásai.

Érdekes a könyvet összehasonlítani más, hazánkban eddig megjelent szakirodalmakkal: a Data Beckettől átvett német könyv (Das grosse Computerschach-Buch) színvonalát szakmailag eléri, élvezetesség és pedagógia szempontjából pedig túl is szárnyalja azt.

Sajnos, ez a könyv sem mentes az apró, de bosszantó hibáktól. Az lehet az olvasó benyomása, hogy a „koptatott” nemcsak farmerban, de sakkdigrammban is divatos. Néhány, szerencsére nem értelemzavaró sajtóhiba is került a kötetbe.

A mai gyakorlat szerint — mivel a könyvet nem a Műszaki Könyvkiadó adta ki — a mű ára kissé borsos. Ennek ellenére nyugodt lelkiismerettel megveheti minden érdeklődő, mert egy játékosan izgalmas és mégis komoly téma szakavatott újralfedezésében lesz része.

LOVASS LÁSZLÓ

jellemzőit, működését, kezelését ismerteti, nem szakember olvasók számára is világos nyelven. Kitér a számítógép hardverrendszerére, külön fejezetben elemzi, hogy hogyan gondolkodik és emlékezik az IBM PC, szől a DOS alapjairól, a hálékony magnelemezekről, a ROM-ról és a ROM programokról, a lemezhozzáféréstől, a karakteres képernyűkezeléstől, a grafikus képernyűkezeléstől, a billentyűzettről és az IBM PC-kegéb tulajdonságairól.

### **Számítástechnika Magyarország '88.**

**Hardver, szoftver, szolgáltatás, kereskedelem**  
(Budapest, 1987. Computerworld Informatika kft., 215 oldal.  
Ára: 285.— Ft.)

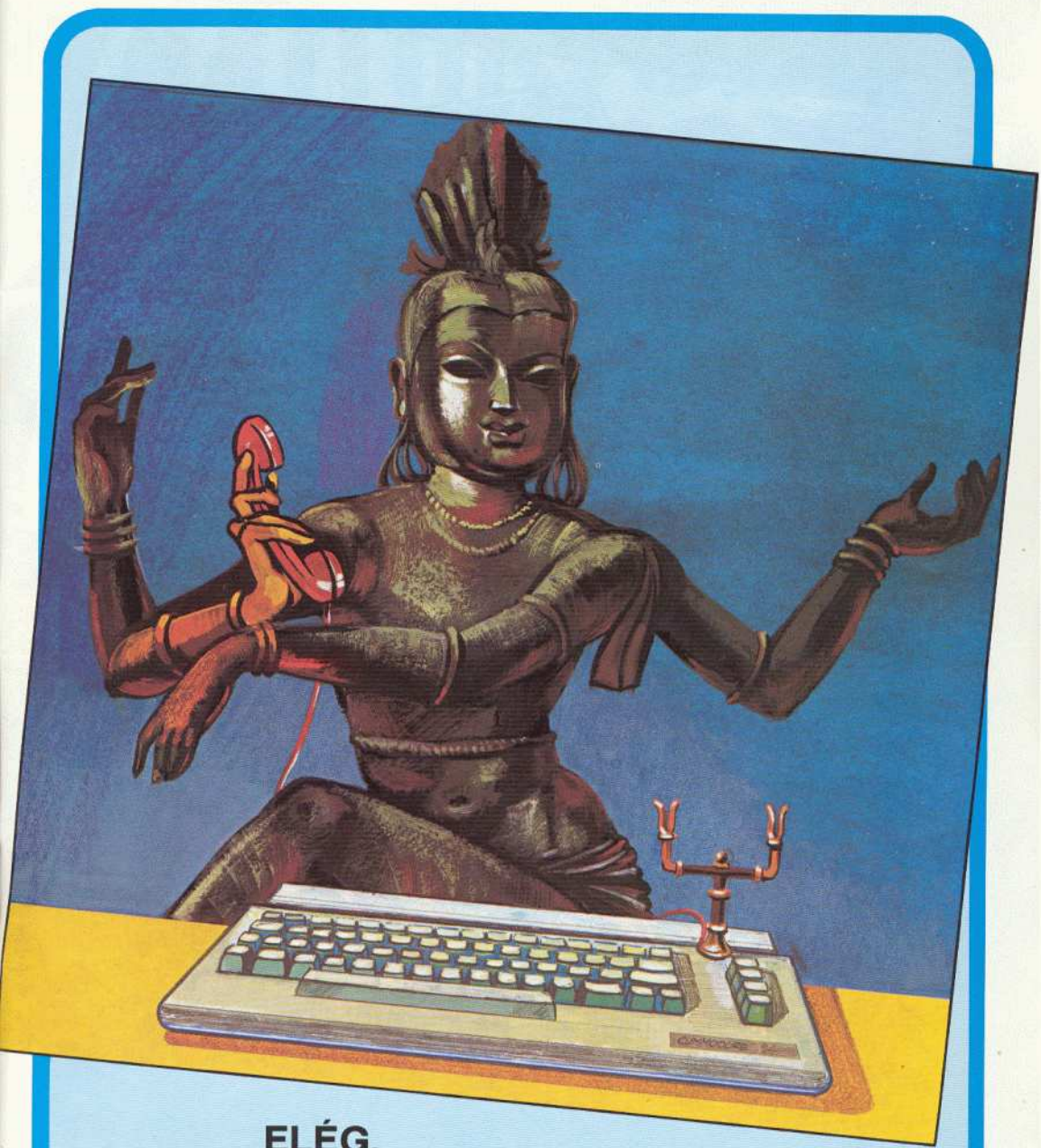
Magyarországon 1988-ban 267 vállalat, intézmény, szövetkezet, gazdasági munkaközösség, polgári jogi társaság, szakcsoport — vagy egyszerűen: cég — kínál számítástechnikai terméket, szolgáltatást ügyfeleinek; ennyien kívánják tevékenységi körüket egy rendszerző kiadványban közzéadni.

A könyv elsősorban a vevőknek szól, azoknak, akik valamilyen szolgáltatást akarnak igénybe venni, valamilyen terméket kívánnak vásárolni. A megadott lista alkalmas lehet például egy adott témában megrendezendő versenytárgyalás meghívójának kiválasztásához. A partnerkapcsolatot keresők megbízható választ kaphatnak arra a kérdésre, hogy milyen a számítástechnikai piac 1988-ban Magyarországon.









**ELÉG,  
HA KÉT KEZE VAN!**

Cikkünk a 42. oldalon



