Ára: 30 Ft

μ mikro számítógép magazin

1989/10

IBM PC/XT, AT kompatibilis professzionális személyi számítógépek

Scil

AJÁNLATUNKBAN

ÚJ

ALAPKONFIGURÁCIÓ P—16/AT—286

80286 8/16 MHz
2 Mbájt RAM
40 Mbájt winchester (28 ms)
Monochrom monitor
1,2 Mbájt floppy
Billentyűzet
ÁR: 220 000,— Ft+ ÁFA

80286 6/10 MHz
1 Mbájt RAM
20 Mbájt winchester
Monochrom monitor
1,2 Mbájt floppy
Billentyűzet
ÁR: 153 000,— Ft+ ÁFA

P—16/AT—386 ALAPKONFIGURÁCIÓ

80386 20/25 MHz
4 Mbájt RAM
80 Mbájt winchester
Monochrom monitor
1,2 Mbájt floppy
Billentyűzet
Torony kivitel
ÁR: 349 000,— Ft+ ÁFA

P—16/XT ALAPKONFIGURÁCIÓ

8088 4,77/10 MHz
640 kbájt RAM
20 Mbájt winchester
Monochrom monitor
360 kbájt floppy
Billentyűzet
ÁR: 105 000,— Ft+ ÁFA

GARANCIA NÉLKÜLI REKLÁMÁRAINK:

TRS aszinkron terminál	9 000,— Ft+ ÁFA
TRS mátrixnyomtató	14 500,— Ft+ ÁFA
FX—100 mátrixnyomtató	49 900,— Ft+ ÁFA
RANK XEROX 4045 lézernyomtató	290 000,— Ft+ ÁFA
UNIBOARD billentyűzet	4 900,— Ft+ ÁFA

BŐVÍTÉSI LEHETŐSÉGEINKRŐL KÉRJÉK ÁRJEGYZÉKÜNKET!

Számítástechnikai Informatikai Fejlesztő
Leányvállalat
1011 Budapest, Iskola u. 10.
Telefon: 154-065, 350-180/180, 181, 182, 184
Telefax: 35-39-15
Telex: 22-4599

μ mikro számítógép magazin

7. ÉVFOLYAM
1989/10. SZÁM

A NEUMANN JÁNOS SZÁMÍTÓGÉPTUDOMÁNYI TÁRSASÁG LAPJA

Címképünk:
Velekey József Lajos munkája

Felelős szerkesztő:
Könyves Tóth Pál

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

Levélcím:
1371 Budapest
Pf. 433

Kiadja:
MTESZ Neumann János Számítógéptudományi Társaság
1054 Budapest, Báthori u. 16.

Levélcím:
1368 Budapest 5. Pf. 240

Telefon: 329-349

Felelős kiadó:
Tóth Istvánné ügyvezető főtitkárhelyettes

Terjeszti a Magyar Posta
Előfizethető a hírlapkézbesítő hivataloknál
és a Posta Hírlap-előfizetési és Lapellátási Irodáján
(1900 Budapest XIII., Lehel u. 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
88—1135

Szikra Lapnyomda
Budapest (89—1498)
Felelős vezető:
Dr. Csöndes Zoltán vezérigazgató

INDEX: 25 629
ISSN 0236-6088

TARTALOM

Talán...

„Hiába jegyzem föl,
mi történt veletek,
hiába csikorog
ujjam között a kréta,
mindig letörlitek a táblát.
Amit képletbe sűrítő
igyekvésem elétek ád,
azt is csak elfeleditek.
Nem élhetünk másképp, csak úgy,
hogy akaratlanul
elpusztítjuk a múltat.
— Ó, nyomorult öntisztulás —
föléljük, napról napra
megesszük
memóriánk tülekvő sejtjeit."
(Fodor András: Sziszifusz az időben)

Talán sikerül, és ha igaz, akkor még ebben az évben megnyílik a magyar számítástechnika-történeti kiállítás. Talán...

Június vége van, tovább már nem húzhatom a szerkesztőségi cikk leadását, pedig még néhány hétbe beletelik, amíg biztossá válik a ma még bizonytalan remény: összegyűlik az a minimális pénz, amiből — többszöri halasztás után — végre megnyithatjuk a hazai számítástechnika elmúlt 30 évét ünneplő kiállítást.

Aki még korábbi írásaimból nem tudná, ebben az évben — január 29-én — volt 30 éve, hogy megjelent az akkori Esti Hírlapban a hír, miszerint hivatalosan is megkezdte működését az MTA Kibernetikai Kutató Csoportjánál az első magyar elektronikus számítógép, az M—3.

A hazai számítástechnika történetét ismerő olvasó joggal kérdezheti meg, hogy miért éppen az M—3 átadását kell a magyar számítástechnika születésnapjaként ünnepelni, miért nem például Nemes Tihamér, Kozma László vagy akár Kalmár László valamelyik alkotásának megszületésétől számítjuk belépésünket a számítástechnika korába.

Valóban számos „születésnapi" időpont közül választhatnánk, a sok közül én mégis az M—3 átadásának a napját érzem a legmegfelelőbbnek, mégpedig azért, mert az M—3 volt az első olyan hazai számítógép, amelyen — a mai terminológiát használva — először oldottak meg számítástechnikai módszerekkel tudományos, műszaki, vállalati, sőt népgazdasági szintű feladatokat. A mai számítástechnikai eszközöknek és módszereknek a közvetlen „felmenője" tehát minden kétséget kizáróan az M—3 volt; a Nemes- és a Kalmár-gépeket, de még Kozma László jelfogós számítógépét is nagy elődöknek, de nem a mai értelemben vett számítástechnikai eszközöknek tekintem.

Egy percig sem hiszem, hogy nem vagyok elfogult, ezt sohasem tagadtam. Ha az M—3-at támadják, akkor gondolkodás nélkül ringbe szállok a Kibernetikai Kutató Csoport eredményének a védelméért. A „vád" az szokott ugyanis lenni, hogy az M—3 semmiben sem különbözik a többi szovjet géptől, például az Ural 1—2-től, a Minsztól vagy a Razdantól. A különbség csupán annyi, hogy az M—3 itt, Budapesten, a többi pedig valahol a Szovjetunióban épült. Éppen ezért a „vádlók" általában még azt is megkérdőjelezik: egyáltalán magyar gépnek tekinthető-e az M—3.

A már hangoztatott elfogultságom miatt ebben a kérdésben biztosan nem lehetek döntőbíró, de talán nem is kell, hogy az legyek, hiszen a tények önmagukért beszélnek. Az M—3 terve, de még az alkatrészek nagy része is valóban a Szovjetunióból érkezett. Egy olyan számítógép dokumentációját kaptuk meg, amely addig még sehol sem épült meg. A szovjet, majd valamivel később a kínai változat gyakorlatilag egyszerre készült. Így érhető, hogy — becslésem szerint — a gép létrehozása során közel ezer logikai, áramköri és más tervezési hibát kellett kijavítani, amíg végül is elindult. És akkor még nem beszéltünk például a mágnesdob mechanikai konstrukciójáról, az információt hordozó krómnikkel réteg felviteléről, és még felsorolni is nehéz lenne azokat a problémákat, amelyeket végül — gyakorlatilag a szovjet partner közreműködése nélkül — a Kibernetikai Kutató Csoport munkatársai oldottak meg.

A kor szokásainak megfelelően már a gép építésével egyidejűleg megindult a rendszer továbbfejlesztése. Az I/O berendezések illesztése már eredeti konstrukció volt, kísérletek folytak egy ferritmemória készítésére, mágnesszalag illesztésére, de ez a gép korábban zenélt, mint bármelyik rokona, és ahogy a feladatok nőttek, úgy kellett a konstrukcióért felelősöknek elkészíteniük a szükséges hardver- és szoftverelemeket.

A Kibernetikai Kutató Csoport és így a hazai számítástechnika azért is érezheti különösen magyarnak a gépet, mert az M—3-ra fejlesztett szoftver- és alkalmazói környezetet a csoport munkatársai teljesen eredeti megoldásokból hozták létre. A gépi programozás mellett nagyon korán elkészült egy mnemonikus kódrendszer, elkezdődött egy fordítóprogram fejlesztése és egy sor más alkalmazási feladat.

A nyíregyházi állandó hazai számítástechnika-történeti kiállítással az elmúlt 30 évre emlékezünk. Sajnos az M—3-ból csak egyes darabok maradtak meg, de szerencsére megőriztük azokat a korai gépeket, amelyek becses értékei a számítástechnika történetének, így például a Kalmár-féle logikai gépet, a szegedi katicabogarat, a Kozma-féle MESZ 1-et, a valószínűleg már csak nálunk található Ural 2-t, a Razdant, a Minsk 22-t és a nyugatiakat, az Elliott 803-at, az ICT 1905-öt, az IBM 360-ast, a Bull-Gamma 115-öt, valamint a második és harmadik generációs magyarokat, az EMG 830-ast, a TPA-t, a Hunort, a Prepamatot, a GD 71-et és az R 10-et.

A bemutatott gépek, alkatrészek, technikai érdekességek sora körülbelül a 70-es évek elejével zárul. Sajnos a lista nem teljes, néhány gépet már nem tudunk bemutatni, azért, mert hírmondó sem maradt belőlük.

Terveink szerint a kiállítást, hacsak az érdeklődés nem lanyhul, 1991-ben zárjuk. Addig talán sikerül egy magyar számítástechnikai múzeum alapjait megteremteni, amire — véleményem szerint — föltétlenül szükség van. Már a második, a „számítógép emlőin" nevelkedett generáció is felnövőben van, ők már semmit sem tudnak az ősszámítógépekről, ezért ez a kiállítás és majd a múzeum nekik készül, és nem csak a magamfajta nosztalgiázó ötveneseknek.

A nyíregyházi tanáccsal együtt úgy tervezzük, hogy a jövő nyáron a kiállítás helyszínének közelében számítástechnika-történeti szaktábort szervezünk, ahol a táborozók megpróbálnak felélesztení néhány öreg masinát, hogy ne csak „holt gépeket", de működő öreg berendezéseket is be lehessen mutatni a közönségnek.

Nem szeretném elfelejteni, hogy júniusban egy másik örvendetes esemény is történt: Hódmezővásárhelyt, a Kossuth Zsuzsa Szakközépiskolában megnyílt az első iskolai számítástechnika-történeti kiállítás, ugyancsak a Műszaki Múzeum, az NJSZT és természetesen az iskola együttműködésének eredményeként. Remélem, az első fecskét rövidesen követni fogja a budapesti Neumann János Szakközépiskola is, és talán más szakiskolák is jelentkezni fognak, hogy számítástechnika-történeti kiállítást szeretnének rendezni. Anyagunk van bőven, ami pedig engem illet, ezeket a becses műszaki emlékeket sokkal szívesebben látom az iskolákban, érdeklődő gyerekek körében, mint poros raktárak polcain.

KOVÁCS GYŐZŐ

A PASCAL REJTELMEI

14. Grafika a Pascalban

A különféle Pascal-reprezentációk — beleértve a Turbo Pascalt is a 3.XX verzióig — viszonylag szegényes grafikai eszközökkel rendelkeznek. Bizonyos fokig javítja a kedvezőtlen összképet a Turbo Pascal által használható, a programokban az Include File direktívával meghívható grafikusrutin-gyűjtemény. Ezek a rendszerben GRAPH.P és GRAPH.BIN fájlokként „jelennek meg". Használatuk a fejlettebb verziókban is lehetséges, de ezek a változatok egyéb fegyverekkel is fel vannak szerelve a grafikát kívánó programok készítéséhez.

A Turbo Pascal 3.XX verziójában kétféle grafikus képernyőt alkalmazhatunk. Ezek a nagy felbontású (2 színű), 640 × 200 képpontos és a kis felbontású (4 színű), 320 × 200 képpontos képernyők. Természetesen az utóbbira készült programok nem színes monitorokon színek nélküli, a színek helyett a fekete-fehér különböző árnyalataiból álló képet eredményeznek a képernyőn.

14.1 Nagy felbontású képernyő

Az áttérés a nagy felbontású képernyőre a **hires** (high resolution: nagy felbontás) eljárással valósítható meg. A **hires** törli a képernyőt, és a grafikus kurzort a képernyő bal felső sarkára (ez minden felbontás esetén a 0,0 koordinátájú pont) állítja.

A képernyőn a háttér színe nem változtatható meg, mindig fekete. A rajzolás színe a **hirescolor (szín)** eljárással állítható be. Ha egy rajz közben ezt a színt megváltoztatjuk, a rajz már meglévő részei is az új színnel jelennek meg. A **szín** paraméternek integer típusúnak kell lennie, értéke a 0. . .15 tartományban értelmezett.

A színek nemcsak változókkal vagy számokkal, hanem előre definiált (a Pascal-fordító számára felismerhető) szövegkonstansokkal is megadhatók. A színek, a számok és a szövegkonstansok összefüggését tartalmazza az *1. táblázat.*

1. TÁBLÁZAT

SZÍN	SZÁM	SZÖVEG-KONSTANS
Fekete	0	Black
Kék	1	Blue
Zöld	2	Green
Cián	3	Cyan
Piros	4	Red
Lila	5	Magenta
Barna	6	Brown
Világosszürke	7	Lightgray
Sötétszürke	8	Darkgray
Világoskék	9	Lightblue
Világoszöld	10	Lightgreen
Világoscián	11	Lightcyan
Világospiros	12	Lightred
Világoslila	13	Lightmagenta
Sárga	14	Yellow
Fehér	15	White

Egyes színes monitorok nem képesek az intenzitás vezérlésére, így ezeknél a világos színek helyett is a megfelelő sötét színek jelennek meg a képernyőn.

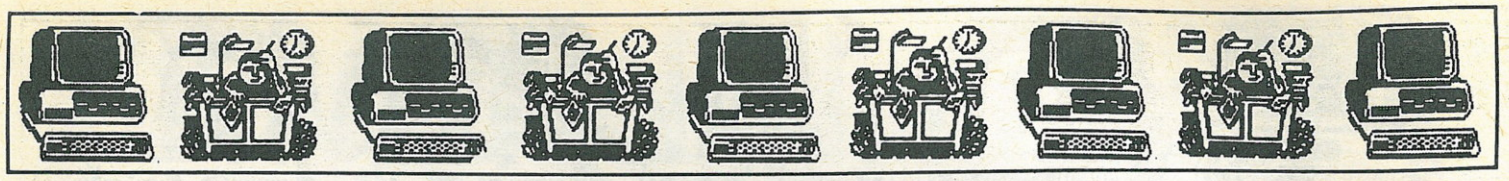
Ha a programban nem alkalmazzuk a szín kiválasztására a **hirescolor** eljárást, a rajzolás fehér színnel történik. Ha a

42. ábra

```
program teglalap;
var i,x1,y1,x2,y2:integer;
    ch:char;
begin
     repeat
          clrscr;
          writeln('téglalap rajzolása a 640x200-as képernyön');
          writeln('Bal felsö sarok koordinátái');
          write('x=');readln(x1);
          write('y=');readln(y1);
          writeln('Jobb alsó sarok koordinátái');
          write('x=');readln(x2);
          write('y=');readln(y2);
          hires;
          for i:=x1 to x2 do
          begin
               plot(i,y1,1);
               plot(i,y2,1);
          end;
          for i:=y1 to y2 do
          begin
               plot(x1,i,1);
               plot(x2,i,1);
          end;
          gotoxy(1,25);
          write('Vege: ESC, Folytatas: barmely billentyu');
          read(kbd,ch);
     until ch=#27;
     textmode;
end.
```

43. ábra

```
program teglalap;
var i,x1,y1,x2,y2:integer;
    vonalszin:integer;
    ch:char;
begin
     repeat
          clrscr;
          writeln('Téglalap rajzolása a 640x200-as képernyon');
          write('Vonalszin (0...15)=');readln(vonalszin);
          writeln('Bal felsö sarok koordinátái');
          write('x=');readln(x1);
          write('y=');readln(y1);
          writeln('Jobb alsó sarok koordinátái');
          write('x=');readln(x2);
          write('y=');readln(y2);
          hires;
          hirescolor(vonalszin);
               draw(x1,y1,x2,y1,1);
               draw(x2,y1,x2,y2,1);
               draw(x2,y2,x1,y2,1);
               draw(x1,y2,x1,y1,1);
          gotoxy(1,25);
          write('Vege: ESC, Folytatas: barmely billentyu');
          read(kbd,ch);
     until ch=#27;
     textmode;
end.
```



programban a **hirescolor** előtt nincs **hires,** csak a keret színe változik meg, és a program grafikus része nem is hajtódik végre.

A rajzolásra két eljárás, a
plot (x, y, szín) és a
draw (x1, y1, x2, y2, szín)
használható; a szín értéke 0 (háttérszín) vagy 1 (a rajzolás színe) lehet. Akik a sorok között olvasnak, talán már rá is jöttek, hogy a 0 színparaméter használata egy már megrajzolt alakzat letörlésére alkalmas.

A **plot** egy képpont rajzolására szolgál. Az x és az y a képpont koordinátái. A **draw** egy vonalat rajzol, amelynek kezdőpontját az x1 és az y1, végpontját az x2 és az y2 koordináták adják meg. Ezeknek is integer típusúaknak kell lenniük. Az x-értékek a 0. . .639, az y-értékek a 0. . .199 tartományban értelmezettek. Ha egy koordinátapáros értéke a megfelelő intervallumon kívül esik, a rajzolás fiktív módon tovább folyik (a képernyőn a tartományon kívüli rész nem jelenik meg), nem keletkezik hibaüzenet sem. Az ilyen eseteket — bár látszólag nem okoznak problémát —, lehetőleg mégis kerüljük el, mert a futási időt növelik. Célszerű tehát a programból azt is ellenőrizni, hogy a koordinátaértékek nem lépik-e túl a megadott tartományok határát.

A nagy felbontású képernyő használatát szemlélteti a *42. ábrán* látható program, amely egy téglalapot rajzol a képernyőre a felhasználó által megadott átló két végpontjának koordinátái alapján.

A program első része az adatbevitel, ezt követi a rajzolás a **for..to..do** utasítások és a **plot** eljárások alkalmazásával. A program befejező része az újraindítás, illetve leállítás megszervezését végzi. Mivel a **hirescolor** eljárást nem alkalmaztuk, a program fehér színnel rajzol.

A *43. ábrán* bemutatott program szintén egy téglalapot rajzol, de itt a vonal színét meg lehet adni, így a rajzolás a **hirescolor (vonalszín)** eljárás hívása révén az előírt színnel történik. A téglalap oldalainak rajzolása nem ciklusba szervezett **plot** eljárások, hanem a — jelen példában talán nem is egyszerűbb, de mindenképpen gyorsabb — **draw** segítségével valósul meg.

14.2 Graphcolormode

Kis felbontású képernyőre a **graphcolormode** eljárással lehet áttérni. Ez törli is a képernyőt, és a grafikus kurzort is a 0,0 koordinátájú pontra állítja. Az alkalmazni kívánt színek kiválasztására a *2. táblázatot* használhatjuk fel segítségül. Ez a paletták és a színek összefüggését és számozását adja meg.

A kívánt paletta a **palette (palettaszám)** eljárással választható ki. A palettáról színt a **plot** és a **draw** eljárásokban már ismertetett módon, a színparaméter megadásával választhatunk. Mind a palettaszám, mint a színparaméter értéke a 0. . .3 intervallumban értelmezett; típusuk integer. Például a

palette (2);
plot (10,10,2);

egy világospiros pontot rajzol a 10,10 koordinátájú helyre.

A háttérszín meghatározása a **graphbackground (háttérszín)** eljárással végezhető el. A háttérszín a 0. . .15 intervallumba eső integertípus, szám vagy előre definiált szövegkonstans lehet, úgy, ahogy azt az 1. táblázat bemutatja.

Ha egy már elkészült rajzon palettát vagy háttérszínt változtatunk, a teljes képernyő átszíneződik az új színnek megfelelően.

```
program teglalap;
var i,x1,y1,x2,y2:integer;
    paletta,vonalszin,hatterszin:integer;
    ch:char;
procedure adatbevitel;
begin
     write('Paletta (0...3)=');readln(paletta);
     write('Vonalszin (0...3)=');readln(vonalszin);
     write('Háttérszin (0...15)=');readln(hatterszin);
     writeln('Az egyi sarok koordinátái');
     write('x=');readln(x1);
     write('y=');readln(y1);
     writeln('A másik sarok koordinátái');
     write('x=');readln(x2);
     write('y=');readln(y2);
end;
procedure teglalap;
begin
     draw(x1,y1,x2,y1,vonalszin);
     draw(x2,y1,x2,y2,vonalszin);
     draw(x2,y2,x1,y2,vonalszin);
     draw(x1,y2,x1,y1,vonalszin);
end;
procedure ujpaletta;
begin
     gotoxy(1,24);
     write('Uj paletta? (i/n)');
     read(kbd,ch);
     if ch='i' then
          begin
               write('   Paletta (0...3)=');
               readln(paletta);
               palette(paletta);
          end;
end;
begin
     repeat
          clrscr;
          writeln('Téglalap rajzolása a 320x200-as képernyön');
          adatbevitel;
          graphcolormode;
          graphbackground(hatterszin);
          palette(paletta);
          teglalap;
          ujpaletta;
          gotoxy(1,25);
          write('Vege: ESC, Folytatas: barmely billentyu');
          read(kbd,ch);
     until ch=#27;
     textmode;
end.
```

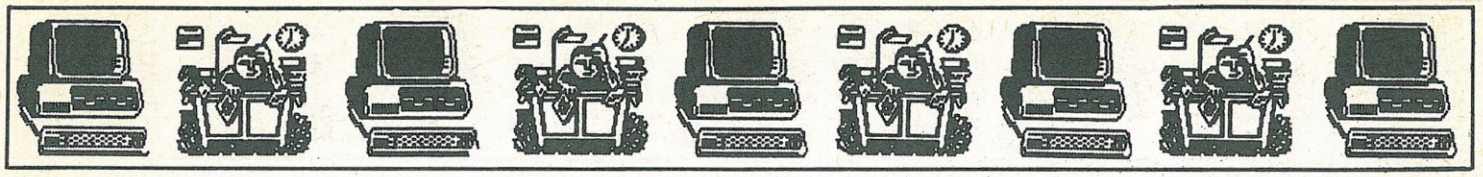
44. ábra

2. TÁBLÁZAT

A SZÍN SZÁMA	0	1	2	3
Paletta 0	Háttér	Zöld	Piros	Barna
Paletta 1	Háttér	Cián	Lila	Világos-szürke
Paletta 2	Háttér	Világoszöld	Világospiros	Sárga
Paletta 3	Háttér	Világoscián	Világoslila	Fehér

45. ábra

```
program grafikusablak;
var x1,y1,x2,y2,i:integer;
var szin:integer;
begin
clrscr;writeln('Ablakok a grafikus képernyön');
write('A rajzolás szine=');readln(szin);
hires;
hirescolor(szin);
graphwindow(0,0,101,51);
       draw(1,1,100,1,1);
       draw(100,1,100,50,1);
       draw(100,50,1,50,1);
       draw(1,50,1,1,1);
       draw(10,10,90,40,1);
graphwindow(90,10,191,61);
       draw(1,1,100,1,1);
       draw(100,1,100,50,1);
       draw(100,50,1,50,1);
       draw(1,50,1,1,1);
       draw(10,10,90,40,1);
graphwindow(180,20,281,71);
       draw(1,1,100,1,1);
       draw(100,1,100,50,1);
       draw(100,50,1,50,1);
       draw(1,50,1,1,1);
       draw(10,10,90,40,1);
gotoxy(1,20);
end.
```



Rajzolásra a kis felbontású képernyőn is csak a **plot** és a **draw** eljárások használhatók. Szintaxisuk azonos a nagy felbontású képernyőre elmondottakkal. Természetesen különbség van az értelmezett koordináták tartományában, ez most a 0. . .319, illetve a 0. . .199 intervallum.

A *44. ábrán* egy példát mutatunk be az elmondottakra. A program most is téglalapot rajzol a képernyőre, de lehetőség van a palettaváltás hatásának tanulmányozására is. Előző programjainkhoz képest e programunk kissé „elegánsabb"; az elkülönülő feladatokat egy-egy eljárás realizálja.

14.3 Ablakok

A karakteres képernyő ablakkezeléséhez hasonlóan a grafikus képernyőkön is nyithatunk ablakokat. Az ablak helyzetének megadására a **graphwindow (x1, y1, x2, y2)** eljárás használandó; a zárójelbe írt paraméterek (ezeknek is integereknek kell lenniük) a téglalap alakú ablakba „húzható" átló végpontjának koordinátái. A definiált ablak átdefiniálásig érvényes, a rajzolás csak az ablakon belül folyik. A **plot** és a **draw** eljárások alkalmazásával megadott koordináták mindig relatív koordinátákként lesznek értelmezve. Például a 0,0 koordináta nem a teljes képernyő, hanem az éppen érvényes ablak bal felső sarkát határozza meg.

Ha a definiált ablakon kívül eső koordinátákat adunk meg, a képernyőn csak az ablakon belüli rész lesz látható. Az ilyen esetekre ugyanaz vonatkozik, mint amit a nagy felbontású képernyő használatával kapcsolatban az érvénytelen koordinátákra már említettünk. A Pascal a teljes képernyőt is ablakként értelmezi, így egy ablakdefinícióból a teljes képernyőre is csak ablakdefinícióval térhetünk át; ez nagy felbontás esetén a **graphwindow (0,0,639,199)**, kis felbontás esetén a **graphwindow (0,0,319,199)** eljárással lehetséges.

A grafikus ablakok létrehozására és használatára mutat példát a *45. ábra* programja, amely három, egymást részben átfedő ablakot nyit a képernyőn, ezeket be is keretezi, majd minden ablakba egy ferde vonalat rajzol. Külön felhívjuk a figyelmet a ferde vonal rajzolására: ez szemlélteti ugyanis azt, hogy a különböző ablakokra az ugyanazon koordinátákra vonatkozó eljárás — a már elmondottaknak megfelelően — a képernyő más helyeire rajzol.

A bemutatott alapismeretek birtokában meg lehet kezdeni a rajzok készítését. Ha valaki ismeri a szabványos Pascal-függvényeket, megpróbálkozhat nemcsak pontokat és egyenes szakaszokat, hanem másfajta vonalakat (ellipszis, kör, ívek stb.) tartalmazó képek készítésével is.

A grafika készítésének megkönnyítését szolgáló Pascal-segédfájlok alkalmazásával a sorozat következő részében foglalkozunk.

NAGY IMRE

A matematika egyik érdekes, de az iskolai tanulmányokból általában kimaradt területe a véletlen (vagy véletlenszerű) események tudományos vizsgálata: a valószínűség-számítás és a matematikai statisztika. Pedig egyáltalán nem érdektelen foglalkozni a témával; a hétköznapi történések jelentős része, a műszaki és technikai élet számos folyamata csak ezeknek a tudományágaknak a segítségével írható le.

Szerzőnk egyszerű, de az említett problémákat jól bemutató programgyűjteményét éppen ezért közöljük, remélve, hogy sok olvasónknak támad kedve behatóan foglalkozni a témával. Különösen lapunk — egy számítástechnikai lap — olvasóinak volna érdemes (vagy ildomos?) ezzel a szélesebb körben nem művelt tudományággal alaposabban megismerkedniük, hiszen az élet szinte minden területét behálózó alkalmazását éppen a számítógép- és számítástechnika mai színvonala tette lehetővé.

A rovatvezető

Játsszunk a véletlennel!

Mivel véletlenszerűen alakuló dolgokat szeretnénk a számítógéppel vizsgálni, először elő is kell állítani őket. Erre, ha nem is a tökéletes eszköz, de jobb híján megfelel a BASIC közismert RND függvénye, amely vagy hasonló funkciójú függvény más programnyelvekben is megtalálható.

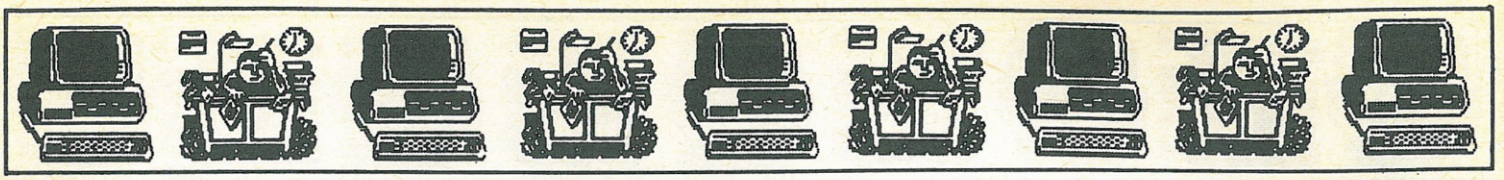
Mielőtt azonban a véletlent az RND-vel szimulálnánk, ne bízzuk magunkat a véletlenre, és vizsgáljuk meg az RND-vel előállított számokat. Ebből a célból futtassuk le többször az *1. listán* látható „programkezdeményt"!

Az eredmény az RND-t nem ismerők számára meglepő: a program mindig ugyanazt a számsorozatot szolgáltatja. A BASIC RND-je tehát nem igazi véletlen számokat állít elő. Akik az RND működésével tisztában vannak, azok tudják, hogy ennek magyarázata igen egyszerű: az RND — egyszerűen fogalmazva — egy kifejezéssel dolgozik, amelyben az „ismeretlen" kezdetben mindig ugyanazt az értéket kapja. A következő szám kiszámításához a kifejezésbe az előző értéket helyettesíti be és így tovább.

Különböző sorozatok előállítása újabb utasítás alkalmazása nélkül úgy oldható meg, hogy az RND-vel előállított sorozat elejét egy erre szolgáló programrészlettel elfogyasztjuk. Erre mutat példát a *2. listán* látható program. A program 130-150-es soraiban lévő rész a sorozat első N számú elemét „megeszi", csak ezután kezdi meg az elemek kiírását a képernyőre.

Ezt a megoldást általában nem szükséges alkalmaznunk, mert a legtöbb BASIC-reprezentációban megtalálható a RANDOMIZE utasítás. Ez gyakorlatilag azt a feladatot oldja meg, amelyre az előbbi példában nekünk kellett programot írnunk: ti. az RND által használt kifejezésbe más-más kezdőértékeket helyettesít be. (Csak a teljesség kedvéért jegyezzük meg, hogy egyes géptípusoknál közvetlenül is lehetőség van más RND-sorozatok előállítására, így egy program újraindításánál egymástól eltérő sorozatokhoz juthatunk.)

Most, miután a számítógéppel — pontosabban a BASIC-kel — előállított véletlenek korlátaival megismerkedtünk, nekiláthatunk a véletlenek elemzésének.

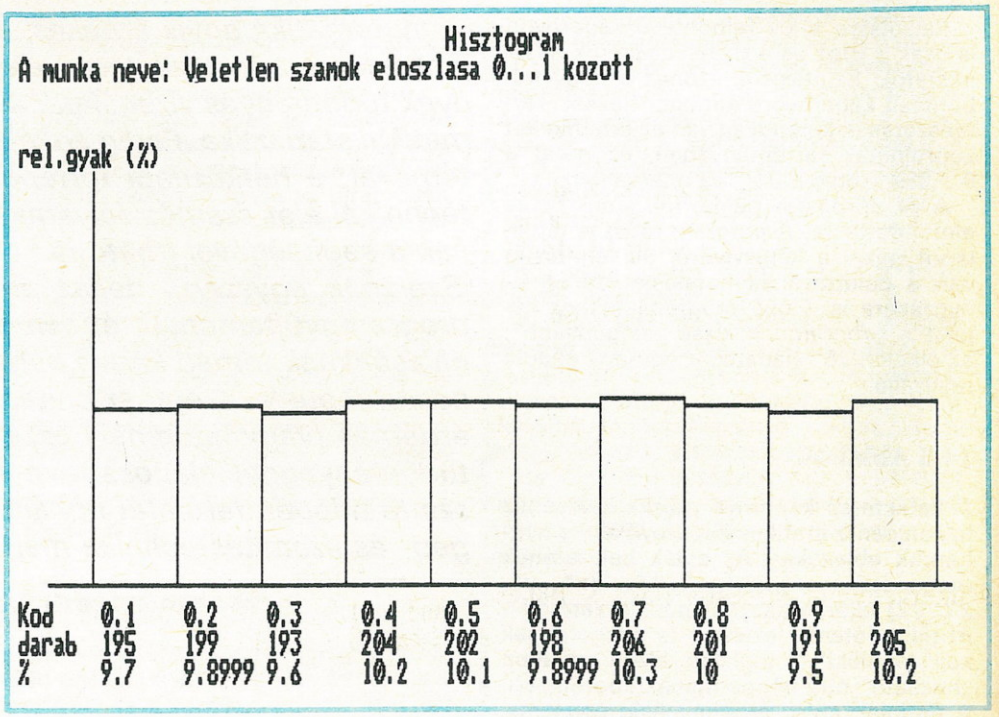


Véletlen eloszlás

Osztályozzuk a véletlen számokat a 0 . . . 1 tartományban nagyság szerint. Az első osztályba a 0 . . .0,1 közötti, a második osztályba a 0,1 . . .0,2 közötti számok kerüljenek és így tovább! Írassuk ki az egyes osztályokba kerülő véletlen számok számát. Hogy a program a többszöri futtatás során ne ugyanazokkal a számokkal dolgozzon, használjuk fel a 2. listán is alkalmazott megoldást! Próbálkozzunk a program többszöri futtatásával úgy is, hogy különböző mennyiségű számot (N) osztályozunk (például 100, 1000, 10 000). Az osztályozást végző program a *3. listán* látható.

Ha kellően nagy mennyiségű számot választunk (például 1000 felett), azt tapasztaljuk, hogy kisebb-nagyobb eltérésekkel ugyan, de az egyes osztályokba kerülő számok száma nagyjából N/10 lesz. Ezt matematikailag szakszerűen úgy fejezhetjük ki, hogy a relatív gyakoriság (az egyes osztályokba kerülő számok száma osztva az osztályok számával) 0,1 vagy másképpen 10%. Minél nagyobb N-értéket választunk, a relatív eltérés az N/10-től annál kisebb lesz. Igen nagy N-érték esetén (és ha az RND valóban véletlen számokat állítana elő) az egyes osztályokba azonos mennyiségű szám kerülne, azaz véletlen számaink a 0 . . . 1 intervallumban egyenletes eloszlásúak lennének.

Az egyes osztályokba kerülő számok számát oszlopdiagramban is ábrázolhatjuk (az *1.ábrán* látható diagramon a Kod a tizedenként beosztott 0 . . . 1 intervallum felső határát jelzi; a program N=2000 értékkel futott le). Az ilyen diagram neve hisztogram. Alkalmazása sokkal szemléletesebb képet ad az eloszlásról, mint egy számsor. Nem véletlen, hogy a matematikai statisztika tudományának egyik leggyakrabban használt eszköze. (Akik számítógépük grafikai lehetőségeit csak kicsit is ismerik, megpróbálkozhatnak egy hiszto-



1. ábra

```
100 CLS
110 PRINT "Véletlen számok RND-vel"
120 FOR I=1 TO 10
130 PRINT RND
140 NEXT
150 END
```

1. lista

```
100 CLS
110 INPUT "Hányadik számtól kezdve: ";N
120 FOR I=1 TO N
130 W=RND
140 NEXT I
150 FOR I=1 TO 10
160 PRINT RND
170 NEXT I
180 END
```

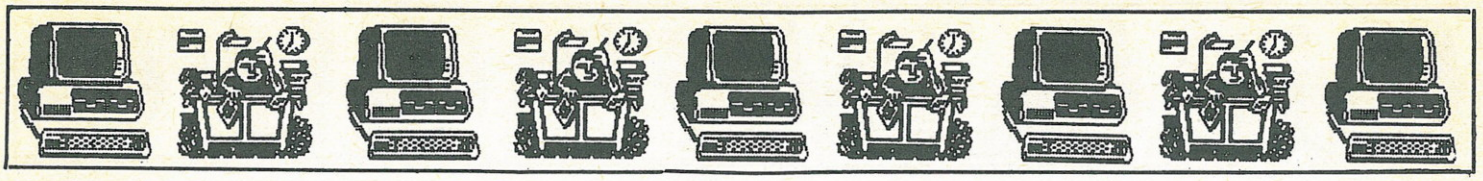
2. lista

```
100 CLS
110 DIM O(10)
120 PRINT "Véletlen számok eloszlása"
130 INPUT "Hányadik számtól kezdve: ";M
140 INPUT "Hány számot vizsgálsz:   ";N
150 FOR I=1 TO M:W=RND:NEXT I
160 FOR I=1 TO N
170 W=RND
180 RN=W-INT(W)
190 J=INT(10*RN)+1
200 O(J)=O(J)+1
210 NEXT I
220 FOR I=1 TO 10:PRINT O(I):NEXT I
230 END
```

3. lista

4. lista

```
100 CLS
110 DIM O(11)
120 PRINT "Sorosan kapcsolt ";
130 PRINT "ellenállások eredője"
140 RANDOMIZE
150 FOR I=1 TO 1000
160 R1=1+.2*(RND-.5)
170 R2=1+.2*(RND-.5)
180 RE=R1+R2
190 J=INT((RE-1.8)*25+.5)+1
200 O(J)=O(J)+1
210 NEXT I
220 FOR I=1 TO 11:PRINT O(I):NEXT I
230 END
```



gramot rajzoló program elkészítésével!)

Soros ellenállások eredője

1 Mohmos, 10% tűrésű ellenállásaink vannak. Tételezzük fel, hogy az ellenállásokat beméréskor úgy válogatták szét, hogy azok értéke 0,9 és 1,1 Mohm között egyenletes eloszlású.

Ilyen ellenállásokból kettőt sorba kapcsolva állítsunk elő 2 Mohmos eredőket. Vizsgáljuk meg, hogy az eredő ellenállásoknak milyen az eloszlásuk! Ennek a vizsgálatnak a programja a *4.listán* látható. A program az eredők legkisebb és legnagyobb értékének megfelelően csak az 1,8 . . . 2,2 Mohm intervallumot vizsgálja (lásd a 190-es sort).

Az eloszlást a *2. ábrán* látható hisztogram szemlélteti. Futtassuk le a programot többször (ha gépünk BASIC-je ezt igényli, a RANDOMIZE utasításban különböző értékek megadásával)! Az eloszlás mindig a 2. ábrán láthatóhoz hasonló — nem matematikai precizitással: háromszög jellegű — lesz.

Normál eloszlás

Vizsgáljuk meg annak a véletlenszám-sorozatnak az eloszlását, amelynek minden eleme tizenkét egyenletes eloszlású véletlen szám összege mínusz hat!

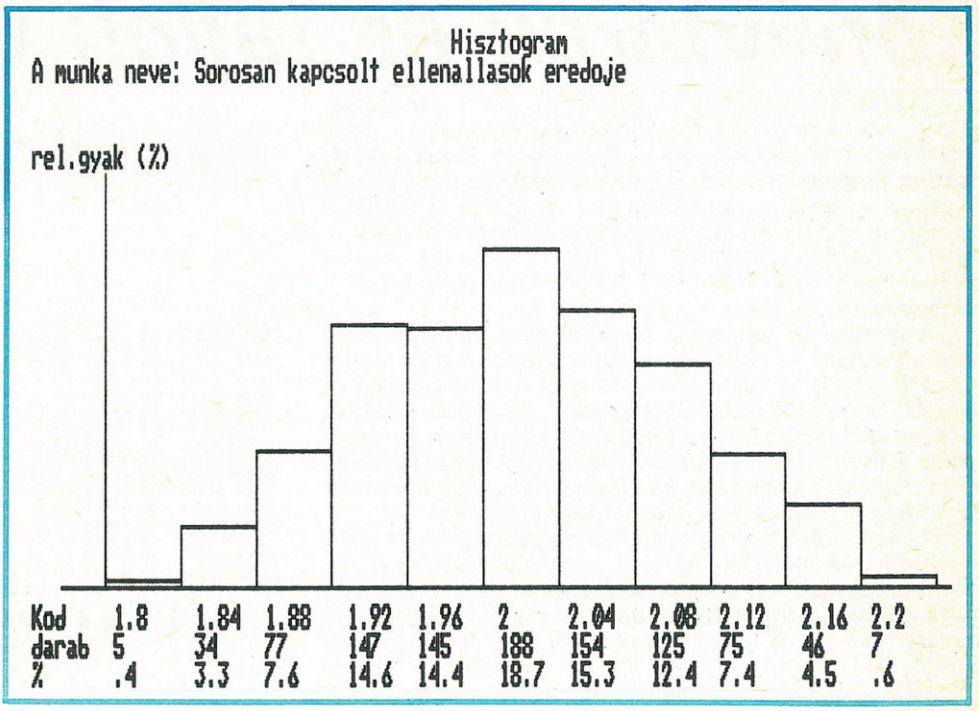
A sorozatot előállító és az egyes osztályokba (0 . . . 1,1 . . . 2 stb.) sorolást elvégző program az *5. listán* látható. Az eredményeket a *3. ábra* hisztogramja szemlélteti, az egyes intervallumok felső határát a Kod sorba írt számérték jelzi.

Az ábrázolt hisztogramot egy ún. haranggörbével lehet közelíteni. Ezt az eloszlást normál (Gauss) eloszlásnak nevezik. A műszaki életben különösen nagy jelentősége van: ilyen eloszlást mutatnak például egy meghatározott értékre gyártott alkatrészek valóságos értékei. (Bár ezt akkor nem említettük, az ellenállások gyártásánál a 0,9 . . . 1,1 Mohm intervallumban így oszlanak el a névlegesen 1 Mohmos ellenállások értékei is.)

BAKONYI GÁBOR

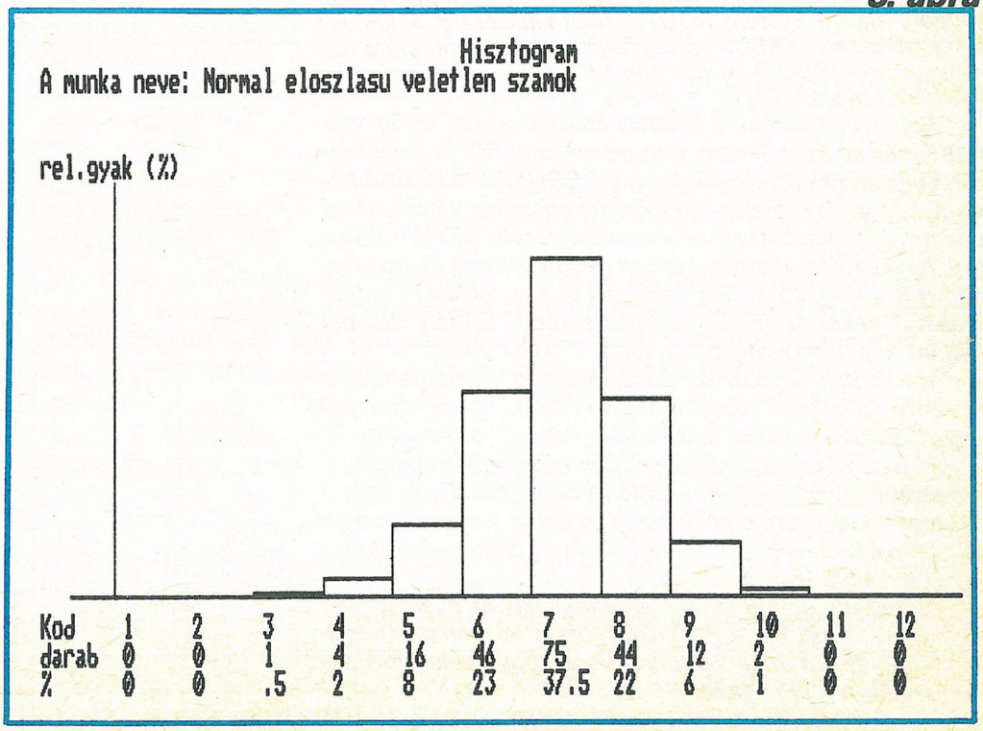
5. lista

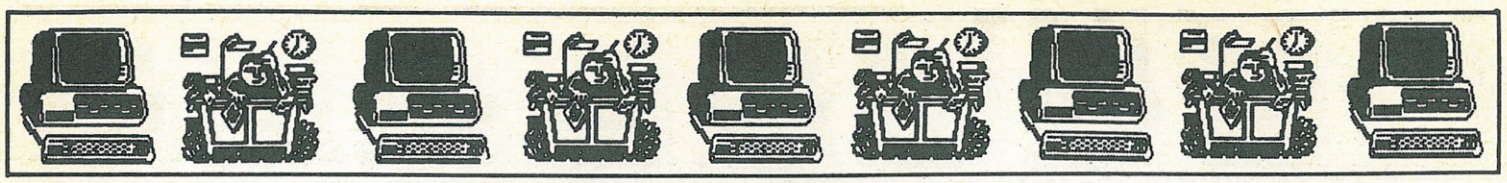
```
100 CLS
110 DIM O(12)
120 PRINT "Normál eloszlású ";
130 PRINT "véletlen számok"
140 FOR I=1 TO 200
150 RANDOMIZE
160 RG=0
170 FOR J=1 TO 12
180 W=RND:RN=W-INT(W)
190 RG=RG+RN
200 NEXT J
210 RG=RG-6:K=INT(RG+7.5)
220 O(K)=O(K)+1
230 NEXT I
240 FOR I=1 TO 12:PRINT O(I):NEXT I
250 END
```



2. ábra

3. ábra





Könyvtárak és fájlok védelme

Gyakran előfordulhat — különösen olyan gépkörnyezetben, ahol egy gépen sok felhasználó dolgozik —, hogy értékes szoftver lesz véletlen törlés áldozata. Ebből a szempontból kiemelten veszélyes helyek például az iskolák, ahol a gépet a tanulók vagy hallgatók tanulásra-gyakorlásra használják; most nem is beszélve azokról, akik már némi ismeret birtokában erejüket kipróbálva, assembly nyelvű programokkal vagy PCTOOLS programmal „túrják fel" a merevlemezt.

A szoftver — ha nem is feltörhetetlen, de egyszerűen megvalósítható — védelme céljából közlöm a már régóta használt és jól bevált módszeremet, amellyel könyvtárak vagy fájlok tehetők hozzáférhetetlenné. Az eljárás alkalmazása természetesen fokozott figyelmet kíván, mert a merevlemez nyilvántartásába piszkál bele; így egy bájt „jól irányzott átírásával", különösen, ha elfelejtettük, hogy mit is tettünk, komoly problémákat okozhatunk magunknak.

A megoldáshoz a merevlemez nyilvántartásának szerkezetéről csak annyit kell tudnunk, hogy a könyvtárak és fájlok neveinek utolsó értékes karaktere után space (hexadecimális kódban: 20) karakterek vannak. Ha az első hexa 20 karaktert a hexa FF kódra átírjuk, a könyvtár vagy fájl nem érhető el. Könyvtárváltásnál Invalid directory, a fájlnév megadásakor Bad command or file name rendszer-hibaüzenet keletkezik. Ha a könyvtárat vagy fájlt ismét használni akarjuk, az előzőben említett eljárás fordítottját kell elvégeznünk, vagyis az FF kódot visszaírjuk 20-ra.

Nézzük ezek után a megoldás egyes lépéseit! Indítsuk el a PCTOOLS programot! Bejelentkezés után nyomjuk le az F3 billentyűt; ezzel a lemez- és speciális funkciókba lépünk be. A menü megjelenésekor az E billentyűvel válasszuk ki a view/Edit funkciót! A program ekkor lehetőséget ad a lemezazonosító megadására — ez jelen esetben természetesen a C. A lemezazonosító megadása után a képernyőn az **1.ábrán** *látható kép jelenik meg; ez a merevlemez fizikai kezdetétől mutatja annak tartalmát.*

A kép alján látható menüből F2-vel választhatjuk a szektorszám közvetlen megadását is, ha tudjuk, hogy az elrejteni kívánt szoftver (legalábbis körülbelül) hányadik szektoron van. Egyszerűbb, különösen azoknak, akik e műveleteket még nem gyakorolták be, ha a PgDn segítségével a lapozást választják a kereséshez, így is eljutnak néhány másodperc alatt a System ROOT terület kijelzéséig. A képernyő jobb oldalán (ASCII value) folyamatosan láthatók a főkönyvtárban (ROOT) lévő alkönyvtárak és fájlok nevei is (lásd a **2.ábrát**).

Rejtsük el például a DOSGYAK elnevezésű alkönyvtárat! Ehhez válasszuk ki az Edit parancsot (F3)! A kurzorvezérlő billentyűkkel keressük meg a DOSGYAK utáni első hexa 20 kódot, és ezt írjuk át FF-re! Az átíráshoz válasszuk az update (F5) funkciót! Mivel a rendszerterület átírásáról van szó, figyelmeztető jelzést kapunk (lásd a **3.ábra** *alulról számított 3. sorát). Az átírást végezzük el az U billentyű lenyomásával, ekkor az FF kód a lemezre íródik. Ezután ESC-pel lépjünk ki a szerkesztésből, majd a PCTOOLS-ból.*

Ha ezután a DOS dir parancsával a nyilvántartást a képernyőre kérjük, abban a DOSGYAK könyvtár szerepel ugyan, de belépni nem lehet a könyvtárba.

A főkönyvtáron kívül levő fájlok védelméhez az eljárás két kisebb különbséggel hasonló. A fájlok nevei már nem a System ROOT területen vannak, ezért a keresés tovább tart, elmarad továbbá a felszólítás is az átírás igazolására (az U billentyű lenyomására).

Végezetül egy jó tanács azoknak, akik az eljárást még nem gyakorolták be: a PCTOOLS futása közben az átalakítás előtti, illetve utáni screenről készítsünk nyomtatott dokumentumot a Print Screen funkcióval!

NAGYNÉ TATAY KLÁRA

1. ábra

```
PC Tools Deluxe R4.22
--------------------------------Disk View/Edit Service--------------------------------
Path=C:
                        Absolute sector 0000000, System BOOT

Displacement ------------------ Hex codes--------------------     ASCII value
 0000(0000)  EB 34 90 4D 53 44 4F 53 33 2E 33 00 02 08 01 00     4 MSDOS3.3  .
 0016(0010)  02 00 02 9F 4F F8 08 00 11 00 04 00 11 00 00 00       O . <   <
 0032(0020)  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 12
 0048(0030)  00 00 00 00 01 00 FA 33 C0 BE D0 BC 00 7C 16 07           3+  = !
 0064(0040)  BB 78 00 36 C5 37 1E 56 16 53 BF 2B 7C B9 0B 00    =x 6 7^V S++!
 0080(0050)  FC AC 26 80 3D 00 74 03 26 8A 05 AA 8A C4 E2 F1      & = t &     -
 0096(0060)  06 1F 89 47 02 C7 07 2B 7C FB CD 13 72 67 A0 10     v G   +! = rg >
 0112(0070)  7C 98 F7 26 16 7C 03 06 1C 7C 03 06 0E 7C A3 3F    !  & !   !   ! ?
 0128(00B0)  7C A3 37 7C B8 20 00 F7 26 11 7C BB 1E 0B 7C 03    ! 7!    &<! ^ !
 0144(0090)  C3 48 F7 F3 01 06 37 7C BB 00 05 A1 3F 7C E8 9F    +H    7!=    ?!
 0160(00A0)  00 B8 01 02 E8 B3 00 72 19 8B FB B9 0B 00 BE D6          ! rv
 0176(00B0)  7D F3 A6 75 0D 8D 7F 20 BE E1 7D B9 0B 00 F3 A6    } u  .   }
 0192(00C0)  74 18 BE 77 7D E8 6A 00 32 E4 CD 16 5E 1F 8F 04    t^ w} j 2 = .v
 0208(00D0)  8F 44 02 CD 19 BE C0 7D EB EB A1 1C 05 33 D2 F7     D =v +}     3
 0224(00E0)  36 0B 7C FE C0 A2 3C 7C A1 37 7C A3 3D 7C BB 00    6 ! + <! 7! =!=
 0240(00F0)  07 A1 37 7C E8 49 00 A1 18 7C 2A 06 3B 7C 40 38      7! I   ^!* ;!@8

Home=beg of file/disk  End=end of file/disk
ESC=Exit  PgDn=forward  PgUp=back  F2=chg sector num  F3=edit  F4=get name
```

2. ábra

```
PC Tools Deluxe R4.22
--------------------------------Sector Edit Service--------------------------------
Path=C:
                        Absolute sector 0000017, System ROOT

Displacement ------------------ Hex codes--------------------     ASCII value
 0000(0000)  49 4F 20 20 20 20 20 20 53 59 53 27 00 00 00 00    IO      SYS'
 0016(0010)  00 00 00 00 00 00 01 00 21 10 02 00 87 5B 00 00            !>    [
 0032(0020)  4D 53 44 4F 53 20 20 20 53 59 53 27 00 00 00 00    MSDOS   SYS'
 0048(0030)  00 00 00 00 00 00 01 00 21 10 08 00 B0 75 00 00            !>.  *u
 0064(0040)  41 43 41 44 20 20 20 20 20 20 20 10 00 00 00 00    ACAD        >
 0080(0050)  00 00 00 00 00 00 81 52 49 13 18 00 00 00 00 00          RI  ^
 0096(0060)  44 4F 53 33 33 20 20 20 20 20 20 10 00 00 00 00    DOS33       >
 0112(0070)  00 00 00 00 00 00 91 51 49 13 17 00 00 00 00 00          QI
 0128(00B0)  44 4F 53 47 59 41 4B FF 20 20 20 10 00 00 00 00    DOSGYAK     >
 0144(0090)  00 00 00 00 00 00 50 7E 8D 12 A2 06 00 00 00 00          P~
 0160(00A0)  50 41 53 43 41 4C 20 20 20 20 20 10 00 00 00 00    PASCAL      >
 0176(00B0)  00 00 00 00 00 00 39 5C 49 13 43 04 00 00 00 00          9\I C
 0192(00C0)  41 55 54 4F 45 58 45 43 42 41 54 20 00 00 00 00    AUTOEXECBAT
 0208(00D0)  00 00 00 00 00 00 D3 81 D7 12 76 06 CC 01 00 00            v
 0224(00E0)  54 59 50 45 53 45 54 20 20 20 20 10 00 00 00 00    TYPESET     >
 0240(00F0)  00 00 00 00 00 00 4B 7F 4E 13 5C 01 00 00 00 00          K.N \
Sector is in the system area - confirm update by pressing "U" ("Esc" to cancel)
 ^ v -> <- = cursor  F1=swap entry area  F5=update  F6=cancel update  ESC=exit
Home=first pos  End=last pos  PgUp=1st half  PgDn=2nd half
```

3. ábra

```
PC Tools Deluxe R4.22
--------------------------------Sector Edit Service--------------------------------
Path=C:
                        Absolute sector 0000017, System ROOT

Displacement ------------------ Hex codes--------------------     ASCII value
 0000(0000)  49 4F 20 20 20 20 20 20 53 59 53 27 00 00 00 00    IO      SYS'
 0016(0010)  00 00 00 00 00 00 01 00 21 10 02 00 87 5B 00 00            !>    [
 0032(0020)  4D 53 44 4F 53 20 20 20 53 59 53 27 00 00 00 00    MSDOS   SYS'
 0048(0030)  00 00 00 00 00 00 01 00 21 10 08 00 B0 75 00 00            !>.  *u
 0064(0040)  41 43 41 44 20 20 20 20 20 20 20 10 00 00 00 00    ACAD        >
 0080(0050)  00 00 00 00 00 00 81 52 49 13 18 00 00 00 00 00          RI  ^
 0096(0060)  44 4F 53 33 33 20 20 20 20 20 20 10 00 00 00 00    DOS33       >
 0112(0070)  00 00 00 00 00 00 91 51 49 13 17 00 00 00 00 00          QI
 0128(00B0)  44 4F 53 47 59 41 4B 20 20 20 20 10 00 00 00 00    DOSGYAK     >
 0144(0090)  00 00 00 00 00 00 50 7E 8D 12 A2 06 00 00 00 00          P~
 0160(00A0)  50 41 53 43 41 4C 20 20 20 20 20 10 00 00 00 00    PASCAL      >
 0176(00B0)  00 00 00 00 00 00 39 5C 49 13 43 04 00 00 00 00          9\I C
 0192(00C0)  41 55 54 4F 45 58 45 43 42 41 54 20 00 00 00 00    AUTOEXECBAT
 0208(00D0)  00 00 00 00 00 00 D3 81 D7 12 76 06 CC 01 00 00            v
 0224(00E0)  54 59 50 45 53 45 54 20 20 20 20 10 00 00 00 00    TYPESET     >
 0240(00F0)  00 00 00 00 00 00 4B 7F 4E 13 5C 01 00 00 00 00          K.N \

 ^ v -> <- = cursor  F1=swap entry area  F5=update  F6=cancel update  ESC=exit
Home=first pos  End=last pos  PgUp=1st half  PgDn=2nd half
```

A VC-1541-es lemezkezelése

A C64-hez a legelterjedtebb lemezegység a 1541-es floppymeghajtó. Erről szeretnék egy kicsit részletesebb információval szolgálni.

A lemezegységnek egy író-olvasó feje van, ezért a lemezeknek csak az egyik oldalát tudja adattárolásra felhasználni. Sokan ismerik azt a lehetőséget, hogy ha a lemezt megfordítva tesszük be a meghajtóba, és az írásvédelem céljára szolgáló rést kivágjuk, akkor lemezeink tárolókapacitását a kétszeresére növelhetjük. A lemezre a lemezkezelő rendszer (DOS) 35 sávban írja fel az információt. A léptető motor nemcsak 35, hanem legalább 80 különböző pozícióba képes a fejet a lemezen pozicionálni. Az eltérés oka az adatbiztonság. Csak minden második pozíció tartalmaz adatokat. Lehetséges azonban a 35. sáv feletti és a sávok közötti terület használata is. Sávonként a lemez 256 bájt hosszú blokkokra van bontva. A blokkok száma a következőképpen változik:

Sáv	Blokk
1 - 17	21
18 - 24	19
25 - 30	18
31 - 35	17

Összesen 683 blokkot tartalmaz a lemez. A blokkszámozás nullától kezdődik. Azt, hogy a lemezen milyen adatokat tárolunk, és ezek hol találhatók, a 18. sávon tartja nyilván a rendszer. Adattárolásra így 664 blokk marad. Az adatblokkok első két bájtja a következő blokkra mutat. Az utolsó adatblokk első bájtja 0, a második pedig a blokkban található — még az adathoz tartozó — bájtok száma. A 18. sáv felépítése a következő:

0. blokk

Byte	Felhasználása	Leggyakoribb tartalma
0 - 1	A tartalomjegyzék első blokkja	18,1
2	Lemezformátum azonosító	'A'
3	Nem használt	0
4 - 143	BAM	
144 - 161	A lemez neve	
162 - 163	Lemezazonosító	
164	SHIFT SPACE	160
165 - 166	Formátum azonosító	'2A'
167 - 170	SHIFT SPACE	160
171 - 255	Nem használt	0,0,...

A blokkok foglaltságát nyilvántartó terület (BAM) felépítése (egy sávhoz négy bájt tartozik):

Byte	Felhasználása
0	A sáv szabad blokkjainek száma
1	0 - 7 blokk bittérképe
2	8 - 15 blokk bittérképe
3	16 - 23 blokk bittérképe

A bittérkép tartalma: foglalt blokk esetén 0, szabad blokk esetén 1.
A lemez tartalomjegyzéke az egyes blokkban kezdődik, és a következőképpen épül fel:

Byte	Felhasználás
0 - 1	A következő blokk cime
2 - 255	file bejegyzések

A file bejegyzések 30 byte -ból állnak.

Byte	Felhasználás
0	Az adat tipusa
1 - 2	Az első adatblokk cime
3 - 18	File.név
19 - 20	Relativ file -oknál az első segédszektor cime
21	Rekordhoszz relativ file -nál
22 - 25	Nem használt
26 - 27	Felülirásnál az első adatblokk cime
28 - 29	Az adatok blokkszáma

A legfontosabb adattípusok:
128 — törölt
129 — SEQ
130 — PRG
131 — USR
132 — REL

Egy blokkba 8 bejegyzés fér el, így összesen 144 bejegyzés tehető a tartalomjegyzékbe. Ha egy programot kitörlünk („SO: név"), akkor ez csak a tartalomjegyzék adott bejegyzését és a BAM-ot érinti. Az *1. program* ezt használja ki arra, hogy egy véletlenül törölt adatot visszaírjon a tartalomjegyzékbe. A programot futtatva végignézi a tartalomjegyzéket, törölt adatokat keresve. Ha talál ilyet, akkor — mivel az adat típusát automatikusan nem lehet megállapítani — a megadott típusként visszaírja.

Ha az egész lemezt töröljük („NO: név"), akkor a BAM és a tartalomjegyzék első blokkja törlődik. A *2. program* megpróbálja a törölt (nem formattált!) lemezt helyreállítani. Ez nem mindig lehetséges automatikusan.

Ha van a tartalomjegyzékről olyan listánk, amely nemcsak az adatok nevét, típusát tünteti fel, hanem az adat első sáv- és blokkszámát is, akkor ezek segítségével és egy tartalomjegyzék-manipuláló programmal (például EXDOCTOR, DISC WIZZARD stb.) tökéletes biztonsággal helyreállíthatjuk lemezünket. Mindkét programnál feltétel, hogy a helyreállításnak közvetlenül a törlés után kell megtörténnie. Ha törlés után kiírunk valamit a lemezre, akkor már csökkennek az esélyeink a helyreállításra.

Bakos Imre—Kelemen Róbert

1. program

```
100 REM "■" - CLR
110 REM "■" - RVS ON
120 REM "■" - RVS OFF
130 REM "■" - CRSR LE
140 POKE 53280,11:POKE 53281,11:POKE 646,15
150 Z$=CHR$(0)
160 PRINT"■"TAB(172)"KEREM A LEMEZT."
170 PRINTTAB(16)"<RETURN>"
180 GET A$:ON -(A$<>CHR$(13)) GOTO 180
190 PRINT"■ VALASSZA KI AZ ADATTIPUST."
200 PRINT"■ ■PRG,■SEQ,■USR,■REL,■NEM KELL VISSZAIRNI.■"
210 OPEN 15,8,15,"I":GOSUB 520
220 OPEN 2,8,2,"#"
230 S=1:T=S
240 IF P<>0 THEN PRINT#15,"U2 2 0 18";SS
250 IF T=0 THEN 490
260 PRINT#15,"U1 2 0 18";S:GOSUB 520
270 GET#2,A$:T=ASC(A$+Z$)
280 GET#2,A$:SS=S:S=ASC(A$+Z$)
290 P=2
300 PRINT#15,"B-P 2";P
310 GET#2,A$:F=ASC(A$+Z$)
320 IF F>127 THEN P=P+32:GOTO 470
330 N$=""
340 FOR I=0 TO 17:GET#2,A$:IF A$="" THEN A$=Z$
341 N$=N$+A$:NEXT
350 N$=MID$(N$,3,16)
360 IF ASC(N$)=0 THEN T=0:GOTO 240
370 PRINT "■"N$"■"TAB(17)"(P/S/U/R/N)";T$=""
380 GET A$:ON -(A$="N") GOTO 460
390 IF A$="P" THEN T$=CHR$(130)
400 IF A$="S" THEN T$=CHR$(129)
410 IF A$="U" THEN T$=CHR$(131)
420 IF A$="R" THEN T$=CHR$(132)
430 ON -(T$="") GOTO 380
440 PRINT#15,"B-P 2";P
450 PRINT#2,T$;
460 P=P+32
470 ON -(P>255) GOTO 240
480 GOTO 300
490 CLOSE 2:PRINT#15,"V":GOSUB 520
500 CLOSE 15:END
510 REM ***  HIBAKEZELES ***
520 INPUT#15,EN,EN$,ET,ES
530 IF EN=0 THEN RETURN
540 PRINT "■"EN;EN$;ET;ES
550 CLOSE 2:CLOSE 15:END
```

2. program

```
1000 REM "■" - CLR
1010 REM "■" - CRSR LE
1020 REM "■" - CRSR BALRA
1030 REM "■" - CRSR FEL
1040 REM "■" - HOME
1050 REM "■" - RVS ON
1060 POKE 53280,11:POKE 53281,11:POKE 646,15
1070 MAX=8:Z$=CHR$(0)
1080 FOR C=1 TO 15:SP$=SP$+CHR$(160):NEXT C
1090 DIM T(35,20),BT(35,20),BS(35,20),NT(MAX),NS(MAX)
1100 PRINT "■"TAB(170)"KEREM A TOROLT LEMEZT."
1110 PRINT TAB(16)"<RETURN>"
1120 GET A$:IF A$<>CHR$(13) THEN 1120
1130 OPEN 15,8,15,"I":GOSUB 2170
1140 OPEN 2,8,2,"#"
1150 PRINT#15,"U1 2 0 18 1":GOSUB 2170
1160 FOR C=0 TO 255:GET#2,A$:A=ASC(A$+Z$)
1170 S=S+A:NEXT C
1180 IF S<>255 THEN 2210
1190 PRINT#15,"U1 2 0 18 4":GOSUB 2170
1200 GET#2,A$:IF A$=Z$ OR A$=CHR$(18) GOTO 1290
1210 PRINT "■■ KIS TURELMET, DOLGOZOM."
1220 PRINT#15,"U1 2 0 18 1":GOSUB 2170
1230 PRINT#2,CHR$(18)CHR$(4);
1240 PRINT#15,"U2 2 0 18 1":GOSUB 2170
1250 CLOSE 2:PRINT#15,"V":GOSUB 2170
1260 PRINT#15,"UI"
1270 REM ***  BAM BEOLVASASA  ***
1280 OPEN 2,8,2,"#"
1290 PRINT#15,"U1 2 0 18 0":GOSUB 2170
1300 PRINT#15,"B-P 2 4"
1310 PRINT "■"TAB(4):FOR T=1 TO 35
1320 TT=INT(T/10)
1330 TT$=CHR$(TT+48):TE$=CHR$(T-TT*10+48)
1340 PRINT TT$"■■"TE$"■";
1350 GET#2,B$,B0$,B1$,B2$
1360 B(0)=ASC(B0$+Z$)
1370 B(1)=ASC(B1$+Z$)
1380 B(2)=ASC(B2$+Z$)
1390 FOR S=0 TO 20
1400 B=INT(S/8)
1410 T(T,S)=(B(B) AND 2↑(S-B*8))=0
1420 NEXT S
1430 NEXT T
1440 PRINT "■■"
1450 FOR S=0 TO 20:PRINT STAB(4);
1460 FOR T=1 TO 35
1470 IF T(T,S) THEN PRINT"*";:GOTO 1490
1480 PRINT".";
1490 NEXT T:PRINT
1500 NEXT S
1510 REM ***  BLOKK FOLYTATAS  ***
1520 FOR T=1 TO 35
1530 IF T=18 THEN 1680
1540 FOR S=0 TO 20
1550 IF T(T,S) THEN 1670
1560 P=1024+80+3+T+S*40
1570 POKE P,PEEK(P) OR 128
1580 PRINT#15,"U1 2 0";T;S:GOSUB 2170
1590 GET#2,T$,S$
1600 BT(T,S)=ASC(T$+Z$)
1610 BS(T,S)=ASC(S$+Z$)
1620 IF BT(T,S)>0 AND BT(T,S)<36 THEN 1670
1630 IF BS(T,S)>1 AND BS(T,S)<21 THEN 1670
1640 IF BT(T,S)=0 AND BS(T,S)>1 THEN 1670
1650 POKE P,PEEK(P) AND 127
1660 T(T,S)=-1
1670 NEXT S
1680 NEXT T
1690 REM ***  ANALIZALAS  ***
1700 FOR T=1 TO 35
1710 IF T=18 THEN 2020
1720 FOR S=0 TO 20
1730 IF T(T,S) THEN 2010
1740 P=1024+80+3+T+S*40
1750 IF BT(T,S)=0 THEN FT=T:FS=S:GOTO 1820
1760 REM ***  VEGIGKERESNI  ***
1770 TT=T:SS=S
1780 FT=BT(TT,SS):FS=BS(TT,SS)
1790 IF T(FT,FS) THEN 1660
1800 IF BT(FT,FS)<>0 THEN TT=FT:SS=FS:GOTO 1780
1810 REM ***  UTOLSO BLOKK  ***
1820 DB=DB+1:IF DB>MAX THEN DB=MAX:GOTO 2040
1830 NT(DB)=T:NS(DB)=S
1840 PR=DB:GOTO 1910
1850 REM ***  BELELANCOLODOTT  ***
1860 FOR C=1 TO DB
1870 IF NT(C)=FT AND NS(C)=FS THEN NT(C)=T:NS(C)=S:GOTO 1890
1880 NEXT C:GOTO 2010
1890 PR=C
1900 REM ***  FELFUZES  ***
1910 TT=T:SS=S
1920 P=1024+80+3+TT+SS*40
1930 POKE P,PR:T(TT,SS)=-1
1940 FT=BT(TT,SS):FS=BS(TT,SS)
1950 IF FT=0 THEN 1990
1960 IF T(FT,FS) THEN 2010
1970 TT=FT:SS=FS
1980 IF BT(FT,FS)<>0 THEN 1920
1990 P=1024+80+3+TT+SS*40
2000 POKE P,PR:T(TT,SS)=-1
2010 NEXT S
2020 NEXT T
2030 REM ***  BEIRAS A TARTALOMJEGYZEKBE  ***
2040 PRINT#15,"U1 2 0 18 1":GOSUB 2170
2050 FOR C=1 TO DB
2060 PRINT#15,"B-P 2";C*32-30
2070 PRINT#2,CHR$(130);CHR$(NT(C));CHR$(NS(C));
2080 PRINT#2,CHR$(64+C)+SP$;
2090 NEXT C
2100 PRINT#15,"U2 2 0 18 1":GOSUB 2170
2110 CLOSE 2:PRINT#15,"UI"
2120 PRINT#15,"V":GOSUB 2170
2130 CLOSE 15
2140 PRINTDB,"FILE VISSZAALLITVA.";
2150 END
2160 REM ***  HIBA KEZELES  ***
2170 INPUT#15,EN,EN$,ET,ES
2180 IF EN=0 THEN RETURN
2190 PRINT "■";EN;EN$;ET;ES
2200 CLOSE 2:CLOSE 15:END
2210 PRINT"EZ NEM EGY TOROLT LEMEZ!";
2220 GOTO 2200
```

FELADATOK — MEGOLDÁSOK

Sorozatunkat elsősorban középiskolásoknak szánjuk, de reméljük, hogy minden olvasónknak tanulási lehetőséget és szórakozást nyújt.
A feladatok a Nemes Tihamér országos számítástechnikai verseny színvonalának felelnek meg. Minden esetben olyat választunk, amely röviden, gyorsan megoldható, de a megoldáshoz ötletre van szükség. A megoldást mindig a következő számban közöljük.
Mivel változatosságra törekszünk, különböző programozási nyelveket használunk. Az is előfordul, hogy egy feladatra több programnyelven is közlünk megoldást, ezzel is elősegítve az ismeretszerzést.
A szerkesztőség várja az olvasók, a versenyzők leveleit. A legötletesebb program beküldőjét könyvutalvánnyal jutalmazzuk. Ne feledjenek azonban a programhoz leírást is mellékelni!

16. feladat: Sorba rendezés

Írjon programot, amely nagyság szerint növekvő sorba rendez egy N egész számból álló egydimenziós tömböt. Törekedjen arra, hogy a program nagy N értékekre is gyors legyen!

Megoldás

A sorba rendezés a számítástechnikában gyakran felmerülő probléma. Így aztán nem is csodálkozhatunk, hogy számos megoldás látott napvilágot. Itt most hármat ismertetünk.

Kezdjük talán a legegyszerűbbel. Válasszuk ki a vektorból mindig a legkisebb elemet, és tegyük azt legelőre! Ezután a megmaradók közül keressük meg a nagyság szerint következőt, és azt tegyük utána! Folytassuk ezt mindaddig, míg vannak elemek!

Ez a megoldás, bár kétségtelenül igen egyszerűnek tűnik, számítógépre nehezen adaptálható. Amikor kiválasztunk egy elemet a vektorból, azt a sor legelejére kell raknunk. Ehhez az összes addigi elemet meg kell mozgatni, ami az algoritmus jelentős lassulását eredményezi. Ez a megoldás számítógépes megvalósításra olyannyira alkalmatlan, hogy programot sem közlünk rá.

A másik szintén igen egyszerű megoldás a következő. Menjünk végig a vektoron. Ha találunk két egymás melletti számot, ahol az előbbi nagyobb, mint az utóbbi, akkor cseréljük fel őket. Ismételjük meg ezt mindaddig, amíg a vektorban a csere előfordul.

Az így működő program megírása nem túl bonyolult, és kevés elem esetén elegendően hatékony is.

Mielőtt a mellékelt Turbo Pascal 5.0 nyelven IBM PC-re megírt programban az ezt megvalósító szubrutint áttekintenénk, nézzük a kiszolgáló rutinokat! *(1. lista)*

A RandomVektor a vektort tölti fel véletlenül választott értékekkel. Az értéktartomány 0-tól 2000-ig terjed. Ilyen értékek mellett a feladatot megoldó szubrutinok működése jól követhető.

A másik szubrutin a Kiír, a rendezett vektor kiírását végzi. A képernyőn sorokra tördelve jelenik meg a vektor.

Mindkét rutin igen egyszerű. Különösebb magyarázatot nem igényelnek. Az előző számban megjelent program is tartalmazott ezekhez nagyon hasonló részeket.

1. lista

```
program BuborékRendező;

uses
  Crt;

const
  Elemszám = 100;

type
  vekt = array[1..Elemszám] of real;

var
{ Ezt a vektort fogjuk rendezni. }
  vektor : vekt;

procedure RandomVektor(
                 var vektor : vekt);
{ Feltölti vektort véletlenszerűen
  választott értékekkel }

var
  i : integer;

begin {RandomVektor}
  Randomize;
  for i:=1 to Elemszám
  do vektor[i]:=Random(2000);
end; {Randomvektor}


procedure Kiir(var vektor : vekt);
{ Kiírja a vektort }

var
  i : integer:
  c : char;

begin {Kiír}
  for i:=1 to Elemszám
  do begin
    if i mod 10 = 1
      then Writeln;
    Write(vektor[i]:7:0);
  end;
  Writeln; c:=ReadKey;
end; {Kiír}


procedure Rendez(var a: vekt);
{ Buborékrendező algoritmussal nagyság
  szerint rendezi a vektor elemeit. }

var
  i     : integer;
  munka : real;
  vége  : boolean;

begin {Rendez};
  repeat
    vége:=true;
    for i:=1 to Elemszám-1
      do begin
{ Ha egy nagyobb elem megelőz
  egy kisebbet, akkor fel kell
  őket cserélni. }
        if a[i]>a[i+1]
          then begin
            munka:=a[i];
            a[i]:=a[i+1];
            a[i+1]:=munka;
        { Csere történt jelzés: }
            vége:=false;
          end;
      end;
  until vége;
end; {Rendez}

begin {BuborékRendező}
  Randomize;
  repeat
    RandomVektor(vektor);
    Rendez(vektor);
    Kiir(vektor);
  until false;
end. {BuborékRendező}
```

2. lista

```
program GyorsRendező;

uses
  Crt;

const
  Elemszám = 1000;

type
  vekt = array[1..Elemszám] of real;

var
{ Ezt a vektort fogjuk rendezni. }
  vektor : vekt;


procedure RandomVektor(
                var vektor : vekt);
{ Feltölti vektort véletlenszerűen
  választott értékekkel }

var
  i : integer;

begin {RandomVektor}
  Randomize;
  for i:=1 to Elemszám
  do vektor[i]:=Random(2000);
end; {Randomvektor}


procedure Kiir(var vektor : vekt);
{ Kiírja a vektort }

var
  i : integer;
  c : char;

begin {Kiír}
  for i:=1 to Elemszám
  do begin
    if i mod 10 = 1
      then Writeln;
    Write(vektor[i]:7:0);
  end;
  Writeln; c:=ReadKey;
end; {Kiír}


procedure Rendez(var a: vekt);
{ Gyorsrendező algoritmussal nagyság
  szerint rendezi a vektor elemeit. }


procedure sort(bal, jobb: integer);
{ Önmagat rekurzívan hívja es így
  rendezi a bal és a jobb
  indexek közt a vektort. }

var
  balindex, jobbindex   : integer;
  középső, munka  : real;

begin {sort}
  balindex:=bal; jobbindex:=jobb;
{ középső a feltetelezett középső elem }
  középső:=a[(bal+jobb) div 2];
  repeat
{ Szétválogatas: }
    while a[balindex]<középső
      do balindex:=balindex+1;
    while középső<a[jobbindex]
      do jobbindex:=jobbindex-1;
    if balindex<=jobbindex then
    begin
      munka:=a[balindex];
      a[balindex]:=a[jobbindex];
      a[jobbindex]:=munka;
      balindex:=balindex+1;
      jobbindex:=jobbindex-1;
    end;
  until balindex>jobbindex;
{ Ha a baloldalt kell még rendezni: }
  if bal<jobbindex
    then sort(bal,jobbindex);
{ Ha a jobboldalt kell még rendezni: }
  if balindex<jobb
    then sort(balindex,jobb);
end; {sort}

begin {Rendez};
  sort(1,Elemszám);
end; {Rendez}

begin {GyorsRendező}
  Randomize;
  repeat
    RandomVektor(vektor);
    Rendez(vektor);
    Kiir(vektor);
  until false;
end. {GyorsRendező}
```

A rendezést a Rendez szubrutin végzi. A belső „for" ciklus halad végig a vektoron, és keres olyan számpárokat, ahol az első nagyobb, mint a második. Ha talált ilyet, akkor felcseréli őket, és „vége" „false" értékű lesz, ezzel jelezve, hogy csere történt. A rendezésnek akkor van vége, amikor csere nem történt, azaz „vége" „true" értékű maradt.

Az algoritmus két egymásba skatulyázott ciklust tartalmaz, ezért futásideje az elemszám négyzetével arányos.

Lehet-e ennél hatékonyabb algoritmust konstruálni? A válasz: igen, bár egy kissé bonyolult.

A gyorsrendező (más néven quicksort) algoritmus C. A. R. Hoare-tól származik, és N log N lépésben rendez egy N elemű vektort. Az eljárás alapgondolata a következő.

Válasszunk ki egy közbülső elemet! Ezután válogassuk szét a vektor elemeit az ennél az elemnél kisebbeket és a nála nagyobbakat tartalmazó két részvektorba! A két részvektor nagyjából az eredeti vektor fele.

30	40	51	20	10	40	10	40	89	98	11
					↑					

E szerint válogatunk

Szétválogatás után:

10	10	11	20	30	40	40	40	51	89	98
						↑				

Ha ezeket a részvektorokat is ugyanezzel a módszerrel kettéosztjuk és így haladunk tovább, egy idő után az egész vektor rendezetté válik.

A második Turbo Pascal 5.0 nyelvű program ennek az algoritmusnak a megvalósítására mutat példát (*2. lista*). Random Vektor és Kiír eljárások az előző programban bemutatott módon működnek.

A vektor rendezése itt is „Rendez" feladata. „Rendez" azonban csak a keretet biztosítja a tényleges rendezést elvégző „sort" rekurzív eljárás számára. „sort" a „bal" és a „jobb" indexek közötti elemek rendezését végzi az előbb ismertetett módon. A szétválogatás felváltva a jobb és a bal oldal felől folyik. „jobbindex"- és „balindex"-változók használata biztosítja, hogy az elemeket sohase kelljen eltolni, mindig csak csere történik.

A szétválogatás után a szubrutin rekurzívan hívja önmagát a keletkezett részvektorok rendezéséhez mindaddig, amíg a részvektor hossza egy nem lesz.

Még ennek az algoritmusnak a továbbfejlesztésére is van lehetőség, bár jelentős eredményt elérni már nem lehet. Matematikailag bizonyítható, hogy bármely rendező algoritmus legalább N log N összehasonlítást igényel N elem esetén.

A további javítást a középső elem kiválasztásának finomítása jelentheti: a rekurzív program iteratívvá átírása és a rövid részvektorok más módon — például buborékrendezéssel — való rendezése. Az olvasó gyakorlásképpen bármelyikkel próbálkozhat.

17. feladat: Anagramma

Írjon programot, amely megadott betűkből a betűk felcserélésével előállítható összes szót kilistázza, és a listában minden szó csak egyszer fordul elő. A program a betűket csak egy példányban tárolhatja!

PINTÉR GÁBOR

ZAK McKRACKEN and the alien mindbenders II.

Legutóbb ott hagytuk abba, hogy az első kristályt megszereztük. Használatával többfajta alakváltozásra is képesek leszünk.

Menjünk vissza San Franciscóba, és a buszról leszállva menjünk fel a 14. utcán a bolt utáni ajtóhoz, ott dobjuk be a levélládába a kristályt. Pár perc múlva a tévében is látott lány, Annie jön ki, és elhalmoz minket kérdésekkel. Annietól bent megkapjuk a sárga kristály egyik felét és egy új parancsot a menübe: Switch. E paranccsal átkapcsolhatunk a játék többi szereplőjére, többek között Annie-ra is. Kapcsoljunk is át rá, és vegyük fel a mappa alól a hitelkártyáját. Ezek után kapcsoljunk át Leslie-re. Leslie és társa, Melissa a Marson végez tudományos kutatásokat. Leslie-vel menjünk be az űrjárműbe, nyissuk ki a kesztyűtartót, és vegyük ki a hitelkártyákat és a biztosítékot. Menjünk vissza Melissához, adjuk oda neki a hitelkártyáját (csak a sajátját fogadja el).Ezután menjünk balra a Monolith-hoz. Tegyük be a hitelkártyát a Monolith-on lévő nyílásba, egy tokent kapunk. Sétáljunk be a jobbra lévő hotelbe. Nyissuk ki a tokennel a jobbra található panelt, és cseréljük ki a biztosítékot. Csukjuk be a nagy ajtót a mellette lévő gombbal, és nyissuk ki a csukott kicsit, szintén az amellett található gombbal (így zsilipelünk!). Lépjünk be. Nyissuk ki a szekrényeket, és vegyük fel az elemlámpát és a szalagot (vinyl tape). Húzzuk félre a takarót, vegyük fel a seprűszerű idegent (ez nem megy könnyen, de legyünk kitartóak!), és a szoba végéből a létrát. Zsilipeljünk vissza a hotel elé, és söpörjük el a homokot (a homok alatt napelemtáblák vannak), hogy később használni tudjuk a villamost.

Térjünk vissza Melissához, és adjuk oda neki a szalagot. Ezután menjünk jobbra a hatalmas archoz (huge face), és támasszuk a létrát a gombok után az ajtóhoz. Kapcsoljunk vissza Zakra. Zakkal menjünk a lakás utáni kirakathoz és szedjük fel a jól megtermett csipeszt. Használjuk a drótvágót (wire cutter). Azután Annieval együtt menjünk a reptérre. A busznál mindkettőjük jegyét Zak fizesse a hitelkártyájáról. A reptéren Annie vegyen egy jegyet Zaknak Miamiba, őt pedig utaztassuk Londonba. Zakkal Miamiban adjuk oda a könyvet a reptéren kódorgó csavargónak, mire ő egy üveg whiskyt ad. Innen repüljünk Londonba, majd onnan Nepálba (Katmanduba).

Nepálban a yak-taxizás után menjünk jobbra az őrhöz, és adjuk oda neki a könyvet. Erre, mivel ezek szerint mi a nagy Guru követői vagyunk, beenged hozzá. A Guru elmondja, hogyan használjuk a kék kristályt. Jöjjünk ki, és menjünk jobbra. Gyújtsuk fel az ott található szalmarakást. A füstöt látva az őr szól a rendőrnek, aki iderohan a yaktól balra lévő őrszobáról. Ballagjunk az őrszobához, vegyük le a tetejéről a zászlót, és bentről a börtönkulcsot. Most menjünk Zairébe. Itt némi dzsungelbolyongás után adjuk oda a golfütőt a sámánnak, amiért cserébe ő megtanít az ajtónyitási kódra. A táncoló négerek leguggolásának sorrendjében kell a gombokat a Marson Leslie-nek megnyomnia az ajtónyitáshoz. Kapcsoljunk Leslie-re. Nyissuk ki az ajtót, vegyük fel a járműből a magnót (boom box) és a lézerkazettát (digital audio tape). Menjünk be az ajtón Leslie után. A hatalmas teremben menjünk mindkét szereplővel az első ajtóhoz (massive door). Leslie támassza a létrát a kristályt tartó oszlophoz, majd Melissa helyezze a szalagot a lézerkazettába, a kazettát a magnóba, és állítsa a magnót felvételre. Ezután Leslie fogja meg a kristálygömböt. A hangot, melyre kinyílt az ajtó, így sikerült rögzíteni. Nyissuk ki a második és a harmadik ajtót is a hang segítségével (playre állítjuk a magnót az ajtók előtt).

Az ajtónyitások után menjünk a terembeli második szoborhoz, és olvassuk el az idegen jeleket. (Az ábrát tanácsos felírni egy darab papírra.) Kapcsoljunk Zakra és repüljünk a legrövidebb úton Mexikóba. Ott a dzsungelben való bóklászás után találunk egy ősi piramist. Menjünk be az egyik kapun (három van), és másszuk addig, míg egy olyan szobához nem érünk, amelyben egy hatalmas szobor áll. (Minden sötét teremben találunk fáklyát, ezt az öngyújtóval meggyújtjuk. Ez a tájékozódás szempontjából sem rossz, hiszen ahol világos van, ott már jártunk.) A szobron szintén találunk helyet a marsbéli jelnek. Rajzoljuk rá a Marson látott jelet, és ekkor már felvehetjük a sárga kristály második darabját a szoborról. Mexikóból repüljünk Peruba, és keressük meg a dzsungelben a madáretetőt. Helyezzük a kenyérmorzsát az etetőbe. Erre megjelenik egy madár, és leszáll táplálkozni. Amíg ezzel van elfoglalva, bűvöljük meg a kék kristállyal (use blue crystal on bird). Madárrá változásunk után repüljünk jobbra, és a folyón túl a gigantikus szobor bal szemében landoljunk. Itt vegyük fel a papírtekercset, repüljünk vissza a testünkhöz, és adjuk oda a papiruszt Zaknak. Térjünk vissza Zak testébe, és utazzunk Londonba. (A madáretetőnél gyorsan kell cselekedni, mert egy idő után megjelenik egy idegen, és visszaszállít minket a lakásunk alatti szobába, ahol elvesz minden fontos tárgyunkat.)

Londonban adjuk oda Annie-nak a papiruszt és a whiskyt. Annie itassa le az őrt whiskyvel, és így már le tudjuk kapcsolni a kerítésben lévő nagyfeszültségű áramot. Zakot és Annie-t vigyük be a Stonhenge-hez. Ehhez Zaknak el kell vágnia a kerítést drótvágóval. A Stonhenge-nél Zakkal állítsuk be az oltáron lévő nyílásba a zsákmányolt zászlórudat, és tegyük mellé a sárga kristály két darabját. Az előkészületek után Annie olvassa fel a papírtekercs szövegét. A villámcsapás hatására összeforr a két darab, most már csak a kezelését kell megtanulnunk. Ennek érdekében repüljünk ismét Zakkal Zairébe, és keressük fel újra a sámánt. Miután a sámán megtanított a kristállyal teleportálni (use yellow crystal), rajzoljuk le a látott térképet a nálunk lévő tapétára a sárga ceruzával. A térképet nézegetve teleportáljunk a három kérdőjellel jelölt helyekre (kivéve a Bermuda-háromszöget).

Az így feltérképezett helyekre most már bármikor teleportálhatjuk magunkat. Térképésztevékenységünk után teleportáljunk Peruba, és vegyük fel a háromágú gyertyatartót. Peruból teleportáljunk Seattle-be, kedvenc barlangunkba, és onnan repülőre szállva menjünk el Miamiba. Itt vegyünk egy jegyet a Bermuda-háromszögbe. Üljünk fel a repülőre, és hallgassuk végig a pilóta idétlen vicceit. Miután elvesztünk, egy hatalmas űrhajón találjuk magunkat. A pilóta már otthonosan mozog itt, mi pedig írjuk le az ajtónyitáshoz a gombok megnyomásának sorrendjét. Ezután benézhetünk az idegenek királyához, és csalhatunk a lottón, de ez nem olyan fontos. (Ne tartózkodjunk bent sokat, mert visszavisznek a fent írt helyre.) Üssük be az ajtónyitó gombokat, és lép-

jünk le. Zuhanás közben használjuk a repülőn kapott ejtőernyőt. A loccsanás után hívjunk egy delfint a szájharmonikával. Használjuk a delfinen a kék kristályt. Delfin képében merüljünk le, és ússzunk jobbra, ahol egy hínárcsomó alatt villog valami. Vegyük fel, ússzunk fel Zakhoz, és térjünk vissza a testébe. A tengerből teleportáljunk Egyiptomba, és a reptéren szabaduljunk meg az akváriumi haltól és a víztől. Kapcsoljunk Leslie-re, és keressük meg az első ajtó mögötti labirintusban a szellőzőberendezést. Használjuk az elemlámpát! A szellőző bekapcsolása után levehetjük a sisakokat.

ÖRÖKÉLET

Ezúttal Vass Tamás küldte el C Plus/4-re készült örökélet poke-jait.

DIABOLO	5689,226
ENIGMA	8931,100
GUZZLER	9026,205
	9027, 7
INTO THE DEEP	RUN/RESET
LEAPER I.	10273, 67
	SYS 10112
SMALL JONES	12321,220
SPACE SWEEP	9078,234
	9079,234
ZONE CONTROL	8323,220
RAIDER	6346,226
WHO DARES WINS	10386,255
VARMIT	6655,205
	6656, 7
	6658,205
	6659, 7
BEBY BERK'S	9364, 0
BERK'S	9759, 0
BERK'S III.	9846, 0
FIRE AMT	7199,226
	7201,226
POD—16	9467,220
ZAGAN WARRIOR	5642,255
YIE AR KUNG—FU II.	11181, 0

Négydimenziós kocka

Zűrzavaros és mégis logikus

A sokdimenziós terek alkalmazása a matematikában nem újdonság. Már rengeteg matematikai és fizikai tételben használják a sokdimenziós tér fogalmát. Például Einstein is a négydimenziós térben alkalmazta tételeit. Természetesen elképzelni is képtelenség, hogy hogyan nézhet ki a négydimenziós tér és abban egy test.

Viszont ha van egy C64-es gépünk, egy „profi assembler" nevű fordítóprogramunk és elegendő türelmünk, akkor szinte kézzelfoghatóvá tehetjük a negyedik dimenziót.

Mikor először indítottam el ezt a programot, barátaimmal együtt negyedórán keresztül bámultuk a képernyőt. Úgy éreztük, mintha minden pillanatban becsapnák a szemünket. A hiperkocka élei a semmiből jöttek elő, majd tűntek el ismét. A képek olyan zűrzavarosak, mégis oly ördögien logikusak voltak, mint amire az ember joggal számíthat, ha már négy dimenzióról van szó.

A kísérletezgetés máig is tart, hiszen a csúcsok elforgatását végző BASIC program átírható, számtalan formában variálható.

Ejtsünk néhány szót a négydimenziós, más néven hiperkockáról. Induljunk ki a kisöccséből, a (háromdimenziós) kockából.

A kockának nyolc csúcsa van. Jelöljük kettes számrendszerben a csúcsokat. Így kapjuk az első csúcsra: ØØØ, a másodikra: ØØ1 és így tovább, 111-ig. Ha „r" betűvel jelöljük az origó középpontú, 2r élhosszúságú kockát, akkor úgy kaphatjuk meg a csúcsainak a koordinátáit, ha a csúcsok bináris kódját használva, a nullák helyébe „−r", az egyesek helyé-

1. lista

```
207 ;---------------------------------
208 ;- DRAWP-BEN X1                  -
209 ;- DRAWP+1-BEN Y1                -
210 ;- DRAWP+2-BEN X2                -
211 ;- DRAWP+3-BEN Y2                -
212 ;---------------------------------
213 DRAW    LDA DRAWP
214 :       CMP DRAWP+2
215 :       BCC CIMX
216 :
217 :       SEC:SBC DRAWP+2
218 :       STA DIFX
219 :       JSR CSERE
220 :       JMP UJ
221 :
222 ::CIMX LDA DRAWP+2
223 :       SEC:SBC DRAWP
224 :       STA DIFX
225 :
226 :   :UJ LDA DRAWP+1
227 :       CMP DRAWP+3
228 :       BCC CIMY
229 :       SEC:SBC DRAWP+3
230 :       STA DIFY
231 :       LDA #1:STA RAJZY
232 :       JMP UJ1
233 :
234 ::CIMY LDA DRAWP+3
235 :       SEC:SBC DRAWP+1
236 :       STA DIFY
237 :       LDA #0:STA RAJZY
238 :
239 :  :UJ1 LDA DIFX
240 :       CMP DIFY
241 :       BCC CIMXY
242 :
243 :       LDA #0:STA RAJZXY
244 :       JMP UJXY
245 :
246 :CIMXY LDA #1:STA RAJZXY
247 :
248 ::UJXY LDA RAJZY:BNE VH
249 :
250 :       JSR DRAWE
251 :          RTS
252 :
253 :   :VH JSR DRAWS
254 :   RTS
255 ;
```

```
365 ;---------------------------------
366 ;-  MNK1-BEN AZ OSZTANDO         -
367 ;-  MNK2-BEN AZ OSZTO            -
368 ;-  MNK3-BAN A HANYADOS          -
369 ;---------------------------------
370 OSZTAS LDA #0
371 :       STA MNK3:STA MNK4
372 :       STA MNK5:LDY #8
373 :       LSR MNK2:ROR MNK4
374 :
375 :    :ELEJE LDA MNK1
376 :           CMP MNK2:BMI END
377 :    :VONAS LDA MNK5
378 :           SEC:SBC MNK4
379 :           STA MNK5:LDA MNK1
380 :           SBC MNK2:STA MNK1
381 :           SEC:JMP FORG
382 :      :END CLC
383 :     :FORG ROL MNK3
384 :           LSR MNK2:ROR MNK4
385 :           DEY
386 :        BNE ELEJE
387 : RTS
```

2. lista

```
1 REM HIPERKOCKA
5 R=30
10 CIM=4*4096
20 PRINT"ELSO FORG":INPUT F:F=F/180*π
30 PRINT"MASODIK  FORG":INPUTG:G=G/180*π
35 PRINT"HARMADIK FORG":INPUTH:H=H/180*π
40 FOR CS=0 TO 15:PRINT CS
50 X=2*R*(1 AND CS)-R
60 Y=R*(2 AND CS)-R
70 Z=R*(4 AND CS)/2-R
80 V=R*(8 AND CS)/4-R
90 FORI=0 TO 127
91 SZ=I*2*π/128
92 XF=X*COS(SZ+F)-SIN(SZ+F)*Y
93 YF=X*SIN(SZ+F)+COS(SZ+F)*Y
94 ZF=Z*COS(SZ+G)-SIN(SZ+G)*V
95 VF=Z*SIN(SZ+G)+COS(SZ+G)*V
100 X1=XF*COS(SZ)-SIN(SZ)*VF
110 V1=XF*SIN(SZ)+COS(SZ)*VF
140 Y1=YF*COS(SZ+H)-SIN(SZ+H)*ZF
150 Z1=YF*SIN(SZ+H)+COS(SZ+H)*ZF
155 TAV=200/(Z1+200)
160 POKEI+CIM+CS*256,Y1*TAV+64
170 POKEI+CIM+CS*256+128,X1*TAV+64
175 NEXT
177 NEXT
```

4. lista

be „+r" értékeket írunk. Így az első csúcs koordinátái: (−r, −r, −r), a másodiké: (−r, −r, +r), és így tovább, végül a nyolcadik csúcsra kapjuk az (+r, +r, +r) koordinátákat.

Csak azokat a csúcsokat kell összekötnünk, melyek bináris kódjai csak egy jegyben különböznek. Például az első csúcsot, vagyis a (ØØØ) kódút a második, (ØØ1) kódú, a harmadik, (Ø1Ø) kódú és az ötödik, (1ØØ) kódú csúcsokkal kötjük össze.

Na most ugyanez a helyzet a hiperkockával is, csak ott mindenből több van. A négydimenziós térben négy független tengely van, minden csúcsot négy koordinátával adunk meg. A csúcsok bináris kódja négyjegyű (ØØØØ, ØØØ1, ØØ1Ø, ... 1111). Így 16 csúcsot kapunk, és az előzőekben leírtak alapján 32 élt.

Az x, y, z, v tengelyekkel (v-vel jelöltem a negyedik, független tengelyt) forgatási síkokat határozhatunk meg. Kiválasztunk két tengelyt, például az x-et és az y-t, és ebben a síkban forgatunk, vagyis minden csúcs x- és y-értéke változik, viszont a z és a v marad.

Forgatás α-szöggel, origó középponttal:

$x' = x\cos\alpha - y\sin\alpha$

$y' = y\sin\alpha + \cos\alpha$

Ha figyelmesen átnézzük a BASIC programot, felfedezhetjük benne a forgatás képletét. Annál is inkább, mert négy is van benne. Ha az olvasó többet szeretne megtudni a hiperkockáról, olvassa el a *Tudomány* 1986/5. számának Észjáték című rovatát.

A BASIC program kiszámolja a 16 csúcs forgatásának 128 fázisát, és ezt tárolja a $4000–$5000 területen. Ezt használja a gépi kódú program.

Először az INIT nevű rutin megszerkeszti a képernyőt, és beállítja a mutatókat. Ahhoz, hogy élvezhető sebességgel tudjon a gép egy-egy képet megjeleníteni (ez kb. 1/12 másodpercet jelent), gyorsabban kezelhető grafikus területre van szükségünk, mint a bittérképes (320 × 200) grafika. Ezért a karaktertábla átdefiniálásával jobb eredményt érhetünk el. Először a karakteres képernyőt töltjük fel a nulla kódú karakterekkel, majd egy 16 × 16 karakterből álló négyzetet hozunk létre úgy, hogy az egymást követő karakterek — egy oszlopon belül — egymás alá kerüljenek. Ezután kapcsoljuk át a karaktertábla kezdőcímét $3800-ra a VIC chip 24 regiszterének segítségével. Szükségünk van egy munkaterületre is, ahol megrajzoljuk a képet, és a már kész képet kapcsoljuk be a képernyőre a MUTS rutin segítségével. A munkaterület kezdőcíme $3000, legalábbis kezdetben, mert a MUTS felcserélgeti a címeket.

A DRAWK rutin *(1. lista)* elkészíti a 32 él kezdő- és végpontjainak koordinátáit, majd 32-szer hívja a vonalrajzoló — DRAW — rutint a 32 él megjelenítéséhez. A vonalrajzoló rutin — ha szükséges —, a csererutin meghívásával felcseréli a két végpontot, hogy a kisebb x koordinátájú legyen a kezdőpont. Majd megvizsgálja, hogy a vonal emelkedő-e vagy ereszkedő (ez a képernyőn fordítva látszik), és azt is eldönti, hogy az x- vagy y-koordinátát léptesse-e egyesével. (Az így kiválasztott koordináta párját egységnél kisebb meredekségi hányadossal léptetjük, hogy szép folyamatos vonalat kapjunk.)

Az ereszkedő és emelkedő vonalakat rajzoló rutinok lényegükben azonosak. Mindkettő az osztásrutin segítségével állapítja meg difx és dify-ból a meredekségi hányadost, majd lépteti a szükséges értékkel a megfelelő koordinátákat.

Az OSZTAS nevű rutinnal *(2. lista)* egy 1 bájtos osztandót osztunk egy nála nem kisebb, szintén 1 bájtos osztóval. Az eredmény alsó, egynél kisebb helyiértékeket tartalmazó regisztere jelenik meg.

Az osztót két bájton csúsztatjuk, ha kisebb, mint az osztandó, kivonjuk az osztót az osztandóból, és a hányados regiszterbe egy magas bitet csúsztatunk, különben csak csúsztatjuk a hányadost és az osztót.

Ezt az eljárást nyolcszor végezzük el. Nagyon hasonlít a papíron végzett osztáshoz. Az osztandó — helyesebben az, ami maradt belőle — és az osztó összehasonlítása csak a magas bájtokon történik. Így nem lesz minden esetben halálpontos a hányados. Lehet, hogy ezt a Szilícium-völgyben másképp csinálják, de nekünk így is nagyon megfelel, hiszen a képernyőn ezek a pici számolási hibák észrevehetetlenek.

Eredeti terveink szerint a teljes assembler nyelvű programot szerettük volna közölni, de hely hiányában ettől el kell tekintenünk. Így csak néhány részletet mutatunk be. Természetesen aki kíváncsi a teljes assembler-listára, az a 16. oldalon található BASIC-betöltő *(3. lista)* futtatása után a monitorból disassemblálva megnézheti. Ennek előfeltétele a paraméterek *4. lista* szerinti előállítása.

A BASIC program (2. lista) — mint már említettem — átírható. A közölt program perspektivikusan számolja ki a csúcspontok képernyő-koordinátáit. Ettől könnyen megszabadulhatunk, ha kitöröljük a TÁV kiszámító sort, és nem szoroztatjuk meg ezzel az értékkel az x- és y-koordinátákat.

Ha van az olvasóban kellő elszántság, megjelenítheti akár a hipertetraédert is. Ennek a hipertestnek az a sajátossága, hogy minden csúcsa egyenlő távolságra van egymástól, éppúgy, mint az egyenlő oldalú háromszög vagy a tetraéder esetében.

Próbáljuk meg kiszámítani az 5 csúcs négy-négy koordinátáját, és módosított BASIC programmal forgassuk el azokat.

A gépi kódú program alkalmas még egyszerű animációs grafika megjelenítésére is. Ha kedvünk van, át is írhatjuk, használhatunk több szakaszt, több végpontot. Ha elég ügyesek vagyunk.

KELEMEN CSABA

```
10 DATA32,21,192,32,120,192,32,141,192,32,164,192,32,111,193,32,63,195,76
20 DATA3,192,169,5,32,210,255,169,147,32,210,255,169,4,133,253,169,0,133
30 DATA252,160,0,145,252,200,208,251,230,253,166,253,224,8,208,243,169,4
40 DATA133,252,169,172,133,251,160,0,132,253,152,145,251,24,105,16,200,192
50 DATA16,208,246,24,165,251,105,40,133,251,165,252,105,0,133,252,230,253
60 DATA165,253,160,0,201,16,208,223,169,29,141,24,208,169,0,133,251,133
70 DATA253,169,48,133,252,169,56,133,254,96,165,252,133,97,165,254,133,252
80 DATA165,97,133,254,173,24,208,73,2,141,24,208,96,165,254,133,98,165,253
90 DATA133,97,162,8,160,0,145,97,200,208,251,230,98,202,208,246,96,166,255
100 DATA189,0,64,141,136,195,189,128,64,141,152,195,189,0,65,141,137,195
110 DATA189,128,65,141,153,195,189,0,66,141,138,195,189,128,66,141,154,195
120 DATA189,0,67,141,139,195,189,128,67,141,155,195,189,0,68,141,140,195
130 DATA189,128,68,141,156,195,189,0,69,141,141,195,189,128,69,141,157,195
140 DATA189,0,70,141,142,195,189,128,70,141,158,195,189,0,71,141,143,195
150 DATA189,128,71,141,159,195,189,0,72,141,144,195,189,128,72,141,160,195
160 DATA189,0,73,141,145,195,189,128,73,141,161,195,189,0,74,141,146,195
170 DATA189,128,74,141,162,195,189,0,75,141,147,195,189,128,75,141,163,195
180 DATA189,0,76,141,148,195,189,128,76,141,164,195,189,0,77,141,149,195
190 DATA189,128,77,141,165,195,189,0,78,141,150,195,189,128,78,141,166,195
200 DATA189,0,79,141,151,195,189,128,79,141,167,195,198,255,16,4,169,127
210 DATA133,255,96,160,0,185,70,195,170,189,136,195,153,0,80,185,70,195,170
220 DATA189,152,195,153,32,80,185,102,195,170,189,136,195,153,64,80,185,102
230 DATA195,170,189,152,195,153,96,80,200,192,32,208,211,160,31,185,0,80
240 DATA133,63,185,32,80,133,64,185,64,80,133,65,185,96,80,133,66,132,113
250 DATA32,191,193,164,113,136,16,226,96,165,63,197,65,144,12,56,229,65,141
260 DATA134,195,32,202,194,76,217,193,165,65,56,229,63,141,134,195,165,64
270 DATA197,66,144,13,56,229,66,141,135,195,169,1,133,139,76,248,193,165
280 DATA66,56,229,64,141,135,195,169,0,133,139,173,134,195,205,135,195,144
290 DATA7,169,0,133,140,76,11,194,169,1,133,140,165,139,208,4,32,23,194,96
300 DATA32,111,194,96,165,140,208,42,173,135,195,133,163,173,134,195,133
310 DATA83,32,15,195,169,0,133,164,32,235,194,230,63,24,165,141,101,164,133
320 DATA164,165,64,105,0,133,64,165,63,197,65,144,232,96,173,134,195,133
330 DATA163,173,135,195,133,83,32,15,195,169,0,133,164,32,235,194,230,64
340 DATA24,165,141,101,164,133,164,165,63,105,0,133,63,165,64,197,66,144
350 DATA232,96,165,140,208,45,173,135,195,133,163,173,134,195,133,83,32,15
360 DATA195,169,0,133,164,32,202,194,32,235,194,198,63,24,165,141,101,164
370 DATA133,164,165,64,105,0,133,64,165,63,197,65,176,232,96,173,134,195
380 DATA133,163,173,135,195,133,83,32,15,195,169,0,133,164,32,235,194,198
390 DATA64,24,165,141,101,164,133,164,165,63,105,0,133,63,165,64,197,66,176
400 DATA232,96,165,63,133,140,165,65,133,63,165,140,133,65,165,64,133,140
410 DATA165,66,133,64,165,140,133,66,96,128,64,32,16,8,4,2,1,164,64,165,253
420 DATA133,81,165,254,133,82,165,63,41,7,170,165,63,74,74,74,74,102,81,24
430 DATA101,82,133,82,189,227,194,17,81,145,81,96,169,0,133,141,133,164,133
440 DATA140,160,8,70,83,102,164,165,163,197,83,48,17,165,140,56,229,164,133
450 DATA140,165,163,229,83,133,163,56,76,53,195,24,38,141,70,83,102,164,136
460 DATA208,223,96,165,197,201,13,240,250,96,0,0,0,0,1,1,1,2,2,2,3,3,4,4
470 DATA4,5,5,6,6,7,8,8,8,9,9,10,10,11,12,12,13,14,1,2,4,8,3,5,9,3,6,10,7
480 DATA11,5,6,12,7,13,7,14,15,9,10,12,11,13,11,14,15,13,14,15,15,3,61,42
490 DATA0
500 FOR I=49152 TO 50057
510 READ A
520 POKE I,A
530 B=B+A
540 NEXT I
550 IF B<>111129 THEN PRINT"HIBA A DATA SOROKBAN!!!":END
560 SYS49152
```

3. lista

TOP LISTA

	játék programok	IBM	AMIGA	C-128	C-64	C+4(16)	SPECTR.	ENTERP.	TVC	APPLE
1.	Bard's Tale III.		●		●		●			●
2.	Technocop		●		●		●			
3.	Grand Prix C.		●	●	●					●
4.	Rocket Ranger		●	●	●					●
5	Stealth Fighter			●	●					
6.	Ocean Ranger		●	●	●					
7.	Robocop		●	●	●		●			●
8.	Rizikó		●							
9.	Zak McKracken		●	●	●					
10	Battle Chess	●	●							●

	felhasználói programok	IBM	AMIGA	C-128	C-64	C+4(16)	SPECTR.	ENTERP.	TVC	APPLE
1.	dBase IV.	●								
2.	Ventura Publish.	●								
3.	Geos 2.0			●	●					
4.	Giga Paint				●					
5.	Paintbrush	●								
6.	Amica Paint				●					
7.	News Room	●			●					
8.	Art Studio 1.2				●		●			
9.	Quick C 5.0	●								
10	Giga Cad +				●					

Listánkat felhasználói, illetve játékprogramokból állítjuk össze. A legjobbakat, legérdekesebbeket a beküldött javaslatok alapján rangsoroljuk. Ehhez kérjük az olvasók közreműködését. C64-re, ZX-Spectrumra, Enterprise-ra, TVC-re, Atarira és IBM-re készült programrangsorokat várunk havonta.

Címünk:
Mikroszámítógép Magazin
Szerkesztősége
1371 Budapest, Pf. 433
Diákszerkesztőség

Dinamikus tömbök kezelése Pascalban

A Pascal **array** típusa köztudottan statikus fogalom, azaz a tömb típusának definíciójakor a tömb méreteit **konstansok** segítségével kell megadni. Ezt a megszorítást a nyelvet kidolgozó Wirth professzor valószínűleg a gépi megvalósítás egyszerűsítése érdekében vezette be. Ugyanakkor a Pascalban is megvan a dinamikus (azaz a program futása idején vezérelt) tárkezelés lehetősége, ami nemcsak tömbök, hanem tetszőleges típusú változók tárolására is használható.

Az alábbiakban bemutatjuk, hogyan lehet a Pascal dinamikus tárkezelési eszközeivel hagyományos szerkezetű tömböket, például vektorokat vagy kétdimenziós mátrixokat kezelni.

A mátrixok elemei minden esetben valós számok, mind sor-, mind oszlopindexük 1-től indul. A Turbo Pascalnak azt a megszorítását, miszerint a tömbök méretének már **fordítási időben** ismertnek kell lennie, úgy kerüljük meg, hogy a tömböket a dinamikus memóriában helyezzük el. Az elemeket egy mutatóval és az általunk számított indexekkel érjük el. Ha egy **m** sorból és **n** oszlopból álló tömböt akarunk létrehozni, akkor egy tetszőleges, egydimenziós tömbre vonatkozó mutatóra **(p)** van szükség. A helyfoglalást (elemenként 6 bájt kell a valós számoknak) a következő eljáráshívással végezhetjük el:

getmem (p,6*m*n);

Célszerű a mátrix sorainak és oszlopainak számát a mátrixszal együtt tartani, annak első két elemeként. Ebben az esetben két elemmel többnek kell helyet foglalni:

getmem (p,6*(m*n+2))

Ha 1<=i<=m és 1<=j<=n, a mátrix [i, j] indexű elemére a

p^[(i-1)*n+j+2]

változóval hivatkozhatunk.

Ha 3<=k<=m*n+2, a vektor k. eleméhez tartozó tömbindexek:

i=(k-3) div n+1 és **j=(k-3) mod n+1.**

Ezek után viszonylag egyszerűen felépíthetjük azt a programot, amely változó méretű tömbök létrehozását, tárolását, szemléjét és módosítását valósítja meg *(1. program)*. A program fontosabb eljárásai a következők:

regi Egy létező tömb beolvasása. Csak a tömböt tartalmazó állomány nevét kell megadni, mivel az állomány első két eleme a sorok, illetve az oszlopok száma. Először ezt olvassuk be, majd lefoglaljuk a tömb helyét, és behozzuk az elemeket.

ujtomb Új tömb létrehozása a dinamikus memóriában. Először a méreteket kell megadni, majd az elemeket sorfolytonosan, amit a beolvasáskor megjelenő indexek is mutatnak.

mod_szem A tömb egy-egy elemének szemléje és esetleges módosítása. Az indexhatárok betartását ellenőrizni kell.

lista A tömb méreteinek és elemeinek kiírása a képernyőre sorfolytonosan, a képernyő méreteihez igazodni igyekező, táblázatos formában.

mentes A tömb kivitele egy állományban. Az állomány **.dat** kiterjesztésű, nevét a felhasználó adja meg. Az állomány első két elemeként a tömb sorainak, illetve oszlopainak száma kerül kivitelre.

A programban minden numerikus bevitelt ellenőrzött módon az **egin,** illetve a **valin** eljárás segítségével olvasunk be. Új tömb létrehozásakor vagy beolvasásakor az előző tömb (ha volt ilyen) által elfoglalt memóriát a **freemem** eljárással szabadítjuk fel.

Az egydimenziós „alibi tömböt" az egyszerűség kedvéért kételeműnek választottuk.

A program nem ellenőrzi, hogy a létrehozandó vagy beolvasandó tömb számára van-e elegendő hely a dinamikus memóriában. Ennek beépítését — gyakorlás céljából — az olvasóra hagytuk. Ha ez az ellenőrzés is működik, a program minden kezelési vagy alkalmazási hibától védett lesz.

A programmal létrehozott dinamikus tömbökkel sokféle művelet végezhető. Mi csupán a mátrixalgebra néhány alapvető algoritmusával foglalkozunk, melyeket eljárások formájában közlünk.

A mátrixok körében elemi műveletnek számít a szorzás. Az eredménymátrix **i.** sorának **j.** elemét a szorzó **i.** sorának és a szorzandó **j.** oszlopának skalárszorzata adja. Ebből következően a szorzó oszlopainak és a szorzandó sorainak azonos számúaknak kell lenniük. Az eljárás feltételezi, hogy a hívó programban szerepel az 1. program **tomb** és **mutato** típusdefiníciója, valamint egy **ok** nevű, logikai változó deklarációja, amely összeférhetetlen mátrixok esetén **false,** különben **true** értéket kap. Az eljárás három paramétere a két tényező, illetve az eredmény mutatója.

A mátrixszorzáshoz egyetlen segédeljárást, a **skal** skalárszorzó algoritmust használjuk, amelyet az itt szükségesnél általánosabb formában deklaráltunk, ugyanis a sor-, illetve az oszlopvektorokat tartalmazó mátrixok mutatóit is paraméterként adtuk meg, az eljárás esetleges egyéb alkalmazására gondolva. Mint látjuk, a sor—oszlop indexszel való címzésről az egyindexes címzésre való áttérés elég bonyolult kifejezéseket eredményez. Ezt a dinamikus tömbkezelés árának kell tekinteni. Az eljárás természetesen gyorsítható, ha például a tömbök méretét egésszé alakító **trunc** függvény hívásait kiemeljük a ciklus elé, amint ezt a **matszor** törzsében tettük *(2. program)*.

A lineáris algebra másik gyakori feladata az egyenletrendszerek megoldása. A többféle ismert algoritmus közül mi az egyik legismertebbet és legáltalánosabbat, a **Gauss-féle eliminációt** választottuk *(3. program)*. A módszer lényege az, hogy az egyenletek megfelelő konstansokkal való szorzása és összeadása útján az együtthatómátrixot „háromszögesítjük", azaz kiirtjuk belőle a főátló alatti elemeket. Az említett konstansokat az előző sor első meglévő (ami a diagonális) és az aktuális sor kiirtandó elemének hányadosa alapján határozzuk meg.

A már háromszögű mátrixból az ismeretleneket az utolsó egyenlettől indulva, felfelé haladva, sorban meg lehet határozni.

Az eljárás **egyidejűleg** több olyan egyenletrendszer megoldását végzi el, melyeknek **együtthatómátrixa közös,** csak a jobb oldali vektorai különböznek. Ezt úgy lehet elérni, hogy egyetlen jobb oldali vektor helyett egy jobb oldali mátrixot kezelünk, melynek minden oszlopa egy-egy jobb oldali vektort reprezentál. Az eredmény egy **megoldásmátrix,** amelynek oszlopai adják az egyes megoldásvektorokat. A megoldás létezéséhez egyrészt az együttható és a jobb oldali mátrix méretbeli összeférhetősége, másrészt az kell, hogy az együtthatómátrix ne legyen szinguláris.

A program jobb áttekinthetősége érdekében bevezettük a **c1** és **c2** függvényeket, amelyek az együttható, illetve a jobb oldali és a megoldásmátrixra az (i, j) sor-oszlop indexből kiszámítják a megfelelő vektorbeli indexet. Az eljárás az 1. programban definiált **tomb** és **mutato** típust, valamint a hibajelzésre szolgáló **ok** logikai változó deklarációját feltételezi. Amennyiben az együtthatómátrix szingularitása az elimináció során kiderül, az **ok**-ot false-ra állítjuk, és az **exit** segítségével azonnal kilépünk.

A lineáris algebra többi műveletét — például a mátrixinverziót — a fenti programokhoz hasonlóan, részben azok felhasználásával valósíthatjuk meg.

BAKOS TAMÁS

1. PROGRAM

```
program tombok;

{Numerikus adatokat tartalmazó mátrixok kezelése}

type tomb = array [1..2] of real; {Ez csak azért kell,
                                   hogy legyen mire mutatni}
     mutato = ^tomb;
     str12 = string[12];

var p: mutato;
    m, n, n1: integer;
    numfile: file of real;

{$i menu}
{A Magazin 1989/5. számában  megjelent menu eljárást (21.
oldal) kell beilleszteni a hozzátartozó típusdefinicóval
együtt}

const m1: strtomb = ('TöMB BEOLVASASA LEMEZRöL',
                     'UJ TöMB',
                     'MODOSITAS/SZEMLE',
                     'LISTA',
                     'TöMB KIVITELE LEMEZRE',
                     'VéGE','','','','');

      m5 =            Folytatás tetszőleges billentyűre';

procedure wait; {Egy billentyűre vár}
var x, y: integer;

begin x:=wherex; y:=wherey;
      gotoxy(1,25); write(m5);
      repeat
      until keypressed;
      delline; gotoxy(x,y)
end;

procedure uzen(s:str80); {üzenetet ir a 23. sorba}
begin gotoxy(1,23);
      writeln(s);
      wait
end;

function van_e(nev: str12): boolean;
{Ha a paraméterként adott file létezik}
{TRUE, különben FALSE}
var f1: file;
begin assign(f1,nev);
      {$i-} reset(f1); {$i+}
      van_e:=(ioresult = 0);
      close (f1)
end;


procedure valin(var a: real);

{Valós számok ellenőrzött bevitele}

var x, y, h: integer;
    s:string[20];

begin x:=wherex; y:=wherey;
      repeat gotoxy(x,y); clreol;
             readln(s); val(s,a,h)
      until h=0
end;

procedure egin(var a: integer);

{Egész értékek ellenőrzött bevitele}

var x, y, h: integer;
    s:string[20];

begin x:=wherex; y:=wherey;
      repeat gotoxy(x,y); clreol;
             readln(s); val(s,a,h)
      until h=0
end;

procedure ujtomb;  {Uj tömb létrehozása}
var i, x, y: integer;

begin clrscr;
      if p<>nil {Ha volt már tömb, felszabadítjuk a
                 helyét}
      then freemem(p,6*(m*n+2));
      write('Sorok száma: '); egin(m);
      write('Oszlopok száma: '); egin(n); writeln;
      getmem(p,6*(m*n+2));    {Memória foglalás, minden
                               valós érték 6 byte}
      for i:=1 to m*n do
          begin x:=wherex; y:=wherey;
                write('[',(i-1) div n+1,',',(i-1) mod
                      n+1,']= ');
                valin(p^[i+2]);
                if i mod 4 = 0
                then writeln
                else gotoxy(x+20,y)
          end;
        p^[1]:=int(m); p^[2]:=int(n);
end;


procedure lista;   {A tömb elemeinek kiírása}
var i, j: integer;

begin clrscr; write('Sorok száma: ',m);
      writeln('  Oszlopok száma: ',n);
      for i:=1 to m do
          begin writeln;
                for j:=1 to n do
                    begin write(p^[(i-1)*n+j+2]:13:2);
                          if ((i-1)*n+j) mod 60=0
                          then wait;
                          if (j mod 6)=0
                          then writeln;
                    end
          end;
      wait
end;

procedure regi;          {Meglevő tömb beolvasása}
var s: str12;
    i: integer;
    x: real;

begin write('Az állomány neve [.DAT]: ');
      readln(s);
      if van_e(s+'.dat')
      then begin assign(numfile, s+'.dat');
                 reset(numfile);
                 read(numfile,x); m:=trunc(x);  {A tömb
                                                 méretei
                                                 valós}
                 read(numfile,x); n:=trunc(x); {számként}
                 getmem(p,6*(m*n+2));
                 for i:=3 to m*n+2 do
                     read(numfile,p^[i]);
                 close(numfile)
           end
      else uzen('   Nincs ilyen adatállomány!')
end;

procedure mod_szem;   {Egy elem szemléje/módosítása}
var i, j: integer;
    c: char;

begin clrscr; write('Sorindex: '); egin(i);
      write('Oszlopindex: '); egin(j);
      if (i<=m) and (i>0) and (j<=n) and (j>0)
      then begin writeln('Jelenlegi érték:
                         ,p^[(i- 1)*n+j+2]:10:2);
                 write('Módosul (i/n)? ');
                 read(kbd, c);
                 if upcase(c)='I'
                 then begin write('Uj érték: ');
                            valin(p^[(i-1)*n+j+2])
                      end
           end
      else uzen('   Hibás index!')
end;

procedure mentes;     {A lánc kivitele num.dat -ba}
var i: integer;
    s: str12;
    x: real;
begin write('Az állomány neve [.DAT]: ');
      readln(s);
      assign(numfile, s+'.dat');
      rewrite(numfile);
      for i:=1 to m*n+2 do
          write(numfile,p^[i]);
      close(numfile)
end;

{főprogram törzse:}
begin p:=nil;
      repeat clrscr; writeln('NUMERIKUS TöMBöK
KEZELéSE':52);
             menu(n1,6,30,m1,false);
             case n1 of
                  1: regi;
                  2: ujtomb;
                  3: mod_szem;
                  4: lista;
                  5: mentes
             end
      until n1=6
end.
```

2. PROGRAM

```
procedure matszor(p1,p2: mutato; var p3: mutato);

{Mátrixok szorzása. p1 és p2 a két tényező, p3 az eredmény
mutatója.
Az eljárás beállítja az ok nevű globális változót is.
Ok=true, ha a szorzás
elvégezhető, ok=false, ha a mátrixok méretei nem
összeférhetők.}

var i1, i2, i3, m, n, k: integer;
    s: real;

procedure skal(p1,p2: mutato; i,j: integer; var s: real);

{Egy sor és egy oszlop vektor skalár szorzata.
p1 a sorvektort, p2 az oszlopvektort tartalmazó mátrix
mutatója,
i a sorvektor, j az oszlopvektor indexe. Az eredményt s -ben
kapjuk.}

var i1: integer;

begin s:=0;
      for i1:=0 to trunc(p1^[2])-1 do
          s:=s+p1^[(i-
1)*trunc(p1^[2])+3+i1]*p2^[j+trunc(p2^[2])*i1+2]
end;

{matszor törzse:}
begin if trunc(p1^[2])=trunc(p2^[1])
{összeférhetőek-e?}
      then begin ok:=true;
                 m:=trunc(p1^[1]); n:=trunc(p1^[2]);
                 k:=trunc(p2^[2]);
                 getmem(p3,6*(m*k+2));      {helyfoglalás
                                              az
                                            eredmény
                                             számára}
                 p3^[1]:=p1^[1]; p3^[2]:=p2^[2];
                 i3:=3;
                 for i1:=1 to m do
                     for i2:=1 to k do
                         begin skal(p1,p2,i1,i2,s);
                               p3^[i3]:=s;
                               i3:=i3+1
                         end
           end
      else ok:=false {az összeférhetetlenséget ok jelzi.}
end;
```

3. PROGRAM

```
procedure linegy(p1,p2: mutato; var p3: mutato);

{Lineáris egyenletrendszer megoldása eliminációval. p1 az
együttható, p2 a jobboldali mátrix mutatója, p3 a megoldás
mátrixra mutat. Az eljárás az ok globális logikai változót
true -ra állítja, sikeres megoldás esetén és false -ra, ha a
mátrixok nem kompatibilisek, vagy az együttható mátrix
szinguláris.}

var m1, n1, m2, n2, i, j, k: integer;
    d, f: real;

function c1(i, j: integer): integer;  {a két dimenziós
indexből}
begin c1:=(i-1)*n1+j+2          {lineáris indexet számító}
end;                                  {függvények}

function c2(i, j: integer): integer;
begin c2:=(i-1)*n2+j+2
end;

{linegy törzse:}
begin m1:=trunc(p1^[1]); n1:=trunc(p1^[2]);
      m2:=trunc(p2^[1]); n2:=trunc(p2^[2]);
      ok:=true;  {a méretek átvétele és a kompatibilitás}
      if (m1=n1) and (m1=m2)        {ellenőrzése}
      then begin getmem(p3,6*(m2*n2+2));
                                    {helyfoglalás a megoldás}
                 p3^[1]:=p2^[1]; p3^[2]:=p2^[2]; {mátrix
                                                  számára}
                 for i:=1 to m1 do
                     begin d:=p1^[c1(i,i)];
                                    {az együttható mátrix}
                           if d<>0
                                    {háromszögesítése}
                           then for j:=i+1 to m1 do
                                    begin
                                      f:=p1^[c1(j,i)]/d;
                                       for k:=i to m1 do
                                          p1^[c1(j,k)]:=
                                          p1^[c1(j,k)]-
                                        f*p1^[c1(i,k)];
                                       for k:=1 to n2 do
                                         p2^[c2(j,k)]:=
                                          p2^[c2(j,k)]-
                                        f*p2^[c2(i,k)];
                                    end
                           else begin ok:=false;
                                      exit
                                end
                     end;
                 for k:=1 to n2 do
                     p3^[c2(m2,k)]:=
                           p2^[c2(m2,k)]/p1^[c1(m1,n1)];
                 for i:=m1-1 downto 1 do      {a gyökök
                                               számítása
                                                alulról}
                     begin d:=p1^[c1(i,i)];   {felfelé
                                               haladva}
                           for j:=1 to n2 do
                               begin p3^[c2(i,j)]:=0;
                                     for k:=i+1 to m1 do
                                         p3^[c2(i,j)]:=
                                         p3^[c2(i,j)]+
                                         p1^[c1(i,k)]*
                                         p3^[c2(k,j)];
                                     p3^[c2(i,j)]:=
                                     (p2^[c2(i,j)]-
                                     p3^[c2(i,j)])/d
                               end
                     end
           end
      else ok:=false
end;
```

BASIC-bővítések Commodore 16-ra

I. rész

A C16-os a rendkívül jól megszerkesztett BASIC-interpreterjével jogosan vívott ki magának előkelő helyet a világ mikroszámítógép-piacán, annak ellenére, hogy végül gyártásával leálltak. Létezik helyette a Commodore Plus/4, amely megnövekedett RAM-területével és beépített programjaival nyújt többet elődjénél.

A C16 azonban a számítástechnikával még csak most ismerkedők részére nagyon jó „iskolagép".

Ha viszont valaki komolyabb munkára akarja fogni gépét, és ennek megfelelő programot kíván készíteni, hamar Murphy egyik törvényével találja szemben magát: „A programok hosszának csak a rendelkezésre álló memóriaterület szab határt!".

És valóban, a programozók hajlamosak addig nyújtózkodni, ameddig a „memóriatakaró" ér. Hogyan lehet ezt a bizonyos takarót megnyújtani? Egyszerű a válasz: bővíteni kell! Két dolgot lehet bővíteni: hardverúton a memóriát (ez nem kevés pénzbe kerül), szoftverúton pedig a gép intelligenciáját (ami viszont időbe kerül). Bár az idő pénz, mégis azt hiszem, hogy a gép lehetőségeit jobban kihasználó interpreter még a memóriabővítővel ellátott gépekre is ráfér.

Aki C64-en már kipróbált legalább egy BASIC-bővítést (például a Simon's BASIC-et), az tudja, mennyivel könnyebb, egyszerűbb úgy programozni, ha az eredetinél többet tudó interpreterrel dolgozhatunk. Miért ne lehetne a C16 BASIC-jét is még jobbá, teljesebbé tenni?

TOKENEK

Nézzük meg az *1. ábrán* látható BASIC programot! Felismerhetők a kinyomtatott listán a szabályos BASIC-parancsok, ismerjük jelentésüket . . .

De vajon a gép memóriájában valóban így — betűről betűre — helyezkedik el a lista? Nézzük meg a *2. ábrát!* Láthatjuk, hogy nyoma sincs az általunk ismert BASIC-szavaknak. Akkor pedig hogyan tudja a gép, hogy mit akarunk végrehajtani?

Úgy látszik, az interpreter készítői is ismerték az előbb említett Murphy-törvényt, és takarékoskodtak. Takarékoskodtak a hellyel és az idővel. A hellyel azért, mert a BASIC-parancsok hosszú neve helyett csak egy bájtot tárolnak a memóriában, így minden BASIC-parancsnak, -utasításnak, -függvénynek egy-egy bájt felel meg, ami jelképezi, helyettesíti őket a tárolt programban. (Ezt a behelyettesítést hívjuk tokenizálásnak, a helyettesítő bájtot tokennek.) Az idővel azért takarékoskodtak, mert egy bájt felismerése, azonosítása sokkal rövidebb ideig tart, mint egy parancs betűről betűre való megvizsgálása.

Kérdés, hogy a gép honnan fogja venni a BASIC-parancshoz tartozó bájtot, illetve végrehajtáskor és listázáskor hogyan tudja meg, hogy ahhoz a bájthoz melyik név, parancs tartozik?

Nézzük meg a $818E-től kezdődő memóriaterületet. Itt a gép által ismert BASIC-szavakat találjuk *(3. ábra)*. A token — amelyik majd a parancsot a programban helyettesíteni fogja — pontosan az a szám lesz, ahányadik helyen a BASIC-parancs áll ebben a táblázatban (+128. A miértre majd visszatérek.)

Most már tudjuk, hogy honnan veszi majd elő az interpreter a tokent, illetve honnan tudja listázáskor a parancs nevét kiírni, azaz detokenizálni. De honnan tudja, hogy ahhoz a parancshoz hol találja meg a megfelelő programrészt, szubrutint?

Tekintsük meg a $8383-mal kezdődő memóriarészt. Itt találjuk a BASIC-parancsokhoz tartozó címeket szokásos alsó bájt—felső bájt sorrendben *(4. ábra)*.

Térjünk vissza arra, hogy miért kell a parancsnak a táblázatban elfoglalt helyéhez 128-at hozzáadni és ezt venni a token értékének? Azért, mert így a már tárolt programban az interpreter pontosan meg tudja különböztetni a tokent a számoktól, betűktől és az egyéb írásjelektől, valamint így könnyebb a programot értelmezni.

Az interpreter készítői gondoltak arra is, hátha valaki nem elégszik meg a gép adta lehetőséggel, és újabb BASIC-parancsokkal kívánja azt kibővíteni. Hagytak egy tokent a bővítés számára is. Ez az ún. felhasználói token. Értéke: $FE. Ez után a token után az interpreter még egy bájtot keres, és ezt tekinti a tényleges felhasználói tokennek.

KÁDÁR SÁNDOR

1. ábra

```
10 let a=10
20 let b=5
30 let c=a*b
40 print "10*5 =";c
50 end
```

2. ábra

```
>1000 00 0c 10 0a 00 88 20 41 :....... a
>1008 b2 31 30 00 16 10 14 00 :210.....
>1010 88 20 42 b2 35 00 22 10 :. b25.".
>1018 1e 00 88 20 43 b2 41 ac :... c2a,
>1020 42 00 33 10 28 00 99 20 :b.3.(..
>1028 22 31 30 2a 35 20 3d 22 :"10*5 ="
>1030 3b 43 00 39 10 32 00 80 :;c.9.2..
>1038 00 00 00 00 00 00 00 00 :........
```

3. ábra

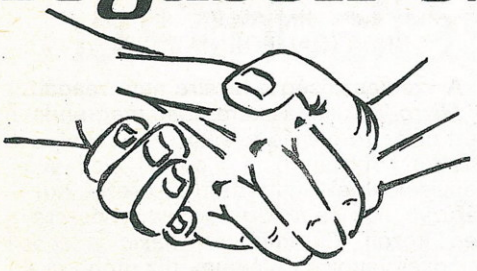
```
>818e 45 4e c4 46 4f d2 4e 45 :endforne
>8196 58 d4 44 41 54 c1 49 4e :xtdatain
>819e 50 55 54 a3 49 4e 50 55 :put#inpu
>81a6 d4 44 49 cd 52 45 41 c4 :tdimread
>81ae 4c 45 d4 47 4f 54 cf 52 :letgotor
>81b6 55 ce 49 c6 52 45 53 54 :unifrest
>81be 1f 52 c5 47 4f 53 55 c2 :oregosub
>81c6 52 45 54 55 52 ce 52 45 :returnre
>81ce cd 53 54 4f d0 4f ce 57 :mstoponw

>81d6 41 49 d4 4c 4f 41 c4 53 :aitloads
>81de 41 56 c5 56 45 52 49 46 :aveverif
>81e6 d9 44 45 c6 50 4f 4b c5 :ydefpoke
```

4. ábra

```
8383  d9 8c    $80,$8cda  end
8385  c9 ad    $81,$adca  for
8387  93 92    $82,$9294  next
8389  af 8d    $83,$8db0  data
838b  ed 90    $84,$90ee  input#
838d  07 91    $85,$9108  input
838f  9a 96    $86,$969b  dim
8391  4e 91    $87,$914f  read
8393  7b 8e    $88,$8e7c  let
8395  1c 8d    $89,$8d4d  goto
8397  bb 8b    $8a,$8bbc  run
8399  e0 8d    $8b,$8de1  if
839b  99 8c    $8c,$8c9a  restore
839d  2b 8d    $8d,$8d2c  gosub
839f  82 8d    $8e,$8d83  return
83a1  0a 8e    $8f,$8e0b  rem
83a3  d7 8c    $90,$8cd8  stop
```

Programozási fogások és melléfogások



A júliusi számban megjelent írásomban a *Form Writer* „Lista" nevű programjának hibáit elemeztem. A program a *Mikrovilág 1989/7. számában* jelent meg, s a helyreigazítás sokáig váratott magára. Jó három hónap után, a *14. számban* végre közölték a javítást, amely nem tér el lényegesen az általam javasolttól. Nem valószínű, hogy volt valami szerepem benne, hisz a *Magazin* júliusi és a *Mikrovilág* említett számának megjelenése között eltelt 10 nap — a hosszú nyomdai átfutási idő miatt — bizonyára kevés lenne erre.

Ugyancsak a *Mikrovilág* 14. számában jelent meg a *Mikromágia* című program, mely 20 különböző, más programba is beépíthető rutint fűz egybe. Az *1/a. listán* két részletet láthatunk belőle: a főprogramot megelőző részt és az óraütéseket bemutató 20-as számú rutint.

A 100-as soron kezdődő főprogram, amely egy 21. önálló rutinnak is tekinthető, menü segítségével teszi lehetővé a választást, majd ON GOTO utasítással a megfelelő rutinra ugrat. A főprogram elemzésével itt nem foglalkozom, csupán annyit jegyzek meg, hogy az említett ON GOTO helyett célszerűbb lenne az ON GOSUB használata. A főprogramban kap értéket az IS$ változó is, a „MÉG EGYSZER I/N ?" szöveggel.

A megjegyzéssorok után a gondosan a program elejére helyezett szubrutinokat átugró GOTO következik. Erről a témáról korábban már volt szó, a közeljövőben visszatérek rá. A 90-es soron kezdődő szubrutint most nem használjuk, csak a teljesség kedvéért szerepel. A 80-as soron kezdődő egy üzenet kiírása után egy billentyűleütésre *vár.*

Az óraütés rutin a 2000-es sortól kezdődik. Hogy a főprogramból a GOTO — az itt közölt listán nem látható módon — a megjegyzés sorra ugrik, illetlenségnek tartom, de erről majd egy más alkalommal írok részletesebben. A 2015-ös sor második utasítása felesleges, hiszen ha g$ értéke üres sztring — azaz nincs leütött billentyű —, a 2018-as sorról e nélkül is a GET utasításra ugrik a program. Ugyanez a helyzet a 2083-as sorban is, azzal az eltéréssel, hogy ott a következő sor

1/a. lista

```
   10 REM ********************************
   11 REM *                              *
   12 REM *  M I K R O M A G I A   C-64  *
   13 REM *                              *
   14 REM *  HARNA M. ANDRAS    SZEGED   *
   15 REM *                      1989    *
   16 REM ********************************
   50 GOTO 100
   80 PRINT "{DOWN}{2 RIGHT}{2 SPACES}VIS
      SZA A MENURE: {RVSON} BARMELYIK GOM
      B {RVSOFF}"
   82 POKE 198,0:WAIT 198,1
   84 RETURN
   90 POKE 214,SO:POKE 211,OS:SYS 58732
   91 RETURN
      ...
 2000 REM    ***   ORAUTES (GONG)   ***
 2005 PRINT "{CLR}{DOWN}{10  SPACES}20.{2
      SPACES}ORAUTES (GONG)"
 2010 PRINT "{4 DOWN}{2 SPACES}GONG{2 SPA
      CES}1 / 2{3 SPACES}KEREM A SZAMOT B
      EUTNI !"
 2015 GET G$:IF G$="" THEN 2015
 2017 IF G$="1" THEN 2050
 2018 IF G$<>"2" THEN 2015
 2020 PRINT "{2 DOWN}{10 SPACES}{RVSON} G
      ONG{2 SPACES}2. {RVSOFF}"
 2030 FOR T=1 TO 600:NEXT
 2033 S=54272
 2036 FOR I=1 TO 24:POKE S+I,0:NEXT
 2040 POKE S+1,15:POKE S+5,12:POKE S+15,6
      :POKE S+24,14
 2043 FOR N=1 TO 5:POKE S+4,21
 2046 FOR T=1 TO 2500:NEXT :POKE S+4,20
 2048 FOR T=1 TO 1000:NEXT :NEXT :GOTO 20
      80
 2050 PRINT "{2 DOWN}{10 SPACES}{RVSON} G
      ONG{2 SPACES}1. {RVSOFF}"
 2054 S=54272
 2056 FOR I=1 TO 24:POKE S+I,0:NEXT
 2058 POKE S+1,35:POKE S+5,12:POKE S+15,6
      :POKE S+24,14
 2060 FOR N=1 TO 5:POKE S+4,21
 2065 FOR T=1 TO 2500:NEXT :POKE S+4,20
 2070 FOR T=1 TO 1000:NEXT :NEXT
 2080 PRINT "{DOWN}"IS$
 2083 GET G$:IF G$="" THEN 2083
 2085 PRINT :PRINT
 2086 IF G$="I" THEN 2000
 2090 IF G$<>"N" THEN 2083
 2093 PRINT :PRINT
 2095 GOSUB 80:GOTO 100
```

1/b. lista

```
      ...
 2000 rem    ***   orautes (gong)   ***
 2005 print "{clr}{down}{10  spaces}20.{2
      spaces}orautes (gong)"
 2010 print "{4 down}{2 spaces}gong{2 spa
      ces}1 / 2{3 spaces}kerem a szamot b
      eutni !"
 2015 get g$
 2020 if g$<>"1" and g$<>"2" then 2015
 2025 print "{2 down}{10 spaces}{rvson} g
      ong{2 spaces}"g$". {rvsoff}"
 2030 for t=1 to 600 : next
 2035 s=54272
 2040 for i=1 to 24 : poke s+i,0 : next
 2045 poke s+1,15-20*(g$="1") : poke s+5,
      12 : poke s+15,6 : poke s+24,14
 2050 for n=1 to 5 : poke s+4,21
 2055 for t=1 to 2500 : next : poke s+4,2
      0
 2060 for t=1 to 1000 : next : next
 2065 print "{down}"is$
 2070 get g$
 2075 if g$="i" then 2000
 2080 if g$<>"n" then 2070
 2085 print : print
 2090 gosub 80 : goto 100
```

PRINT utasításait is el kell hagyni. Ezek törlése azért célszerű, mert a jelenlegi alakban is nem kívánt hatásuk lehet.

Megfigyelhetjük, hogy a 2033—2048 és a 2054—2070 sorok egyetlen számtól és a 2048 sor végén álló GOTO utasítástól eltekintve pontosan megegyeznek. Ez bizony szószaporítás, a rosszabbik fajtából. Aki ilyesmit elkövet, az — mint alább látni fogjuk — egyéb csalafintaságokra is képes. Ilyenkor általában szubrutint szokás alkalmazni, esetünkben egyszerűbb az *1/b. listán* javasolt megoldás. A rutint — talán kissé öncélúan — átsorszámoztam, hogy a sorok sorszámai egyenletes lépésközben kövessék egymást. A 2045-ös sorban

15—20*(g$="1")

kifejezés némi magyarázatot igényel. A zárójelben egy logikai kifejezés van, melynek értéke, ha igaz, akkor −1, különben 0. Ennek megfelelően a POKE-olandó érték 35 lesz, ha g$="1", és 15 lesz g$="2" esetén. Ezt a módszert használtam — akkor magyarázat nélkül — a júliusi szám 2. listáján is.

* * *

Néhány évvel ezelőtt az amerikai *RUN* 1986. májusi számában fedeztem fel a *2/a. listán* látható programot. Bár a BASIC betöltőben szemet szúró csacskaságok egy kissé zavartak, a kísérő cikkben ígért lehetőségek felkeltették az érdeklődésemet: ideális segítség a DATA sorok tömeges beírásához. A „balra-nyíl"-billentyű lenyomására a „DATA" szó íródik a képernyőre, az adatokat elválasztó vesszőket pedig a „könnyen megcélozható" szóköz billentyűvel lehet kiírni.

Begépeltem a programot, használtam is néhány hétig vagy hónapig, aztán találtam helyette kényelmesebbet, jobban kezemhez állót. Lassan el is felejtettem az egészet.

Mielőtt folytatnám, vegyük szemügyre az említett csacskaságokat, melyeket annak idején változatlanul rögzítettem, javításuk eszembe se jutott. A 150-es sorban a leggyengébb fejszámoló is ki tudja számítani a konstansokkal végzett művelet eredményét. (A még gyengébbek kedvéért elárulom, hogy ez pontosan 678.) A GOTO-s megoldás helyett jobb lenne a hatékonyabb FOR-NEXT ciklust alkalmazni, bár a lassulás az adatok kis száma miatt elenyésző. Nem lenne túl nehéz feladat a ciklus végértékének kiszámítása sem. Hangsúlyozom, hogy mindezek jelentéktelen szépséghibák, amelyek sem a program hatékonyságát, sem a begépelendő szöveg mennyiségét nem befolyásolják lényegesen. Az ellenőrző összeg hiánya egyáltalán nem hiba, mert az eredeti listán BASIC sorellenőrző kódok is vannak. Szintén nem hiba a 190-es sor végén elhagyott idézőjel sem, erről a fogásról az előző részben már írtam.

Már el is feledkeztem az egészről, amikor a *PC Mikrovilág* 1987. október 14-i számának mellékletében *Az a sok DATA sor!* cím alatt a *2/b. listán* látható programra bukkantam. A beharangozó szöveg így kezdődött: „Harna M. András sok gépi kódú programot másolhatott ki különböző lapokból, mert elhatározta, hogy megkönnyíti a munkát."

A 150-es sorban lévő értékadás valahonnan ismerősnek tűnt. A listát összehasonlítva a RUN-beli eredetivel, meggyőződtem az egyezésről. A *„szerző"* egy kicsit rontott is a programon: a 140-es sorban álló üres ciklus feleslegesen húzza az időt, semmilyen pozitív hatása nincs. Itt — BASIC ellenőr hiányában — az ellenőrző összeg is elkelne.

A nyilvánvaló plágium láttán levelet írtam a Mikrovilág szerkesztőségének, melynek idézem egy részletét:

„A legutóbbi (1987/18) számban jelent meg egy DATA-beírást segítő program. Beküldője, *Harna M. András,* nagyon sok gépi kódú programot másolhatott ki különböző lapokból, míg arra vetemedett, hogy egyiket saját neve alatt közöltesse. Még „szerencse", hogy a *Copyright* jelzést nem tette a REM sorokba. Szeretném, ha *a lap hasábjain* mielőbb közölnék vele, hogy nem illik idegen tollakkal ékeskedni. Sokkal szebben mutatott volna az ő neve és címe helyett például a következő:

JIM ALLEN
BREA (CALIFORNIA), 1984"

A szerkesztőség a levélre nem reagált. A *Mikro '88* kiállítás keretében megrendezett újságíró-olvasó találkozón — ahol olvasóként vettem részt — a CWI jelen lévő képviselőjének említettem az esetet, hozzáfűzve, hogy a *Mikrovilágban* rendszeresen közölt Commodore *Basic ellenőr* mindkét változata plágium. (Ez utóbbira a *Magazin* idei áprilisi számában is utaltam.) Folytatás: semmi! Azóta is rendszeresen *sajátjukként* közlik a *Basic ellenőrt,* s a műholdas műsorban minden számban többször is beleütközhet az olvasó a szigorú szövegbe: „Felhívjuk a figyelmet, hogy lapunk bármely részének másolása, a másolatok terjesztése jogsértés." Nem furcsa, hogy csak nekik vannak szerzői jogaik?

Végezetül két megjegyzés:

A korábbiaktól eltérően az itteni programlisták *formailag* nem pontosan egyeznek meg a forrásként felhasznált programlistákkal, az eltérés az idézőjeleken belüli vezérlő karakterek ábrázolásában és a programsorok kiigazításában van.

Kivételes eset, hogy a *2/a.* és *2/b.* listán teljes programot idéztem, a „csacskaságok" szempontjából lényegtelen DATA sorokkal együtt. Most a „koppintás" tényét kívántam bizonyítani, melynek felismerése éppen az említett csacskaságoknak köszönhető.

BARNA LÁSZLÓ

2/a. lista

```
110 REM        COMMA GENERATOR
120 REM      VERSION 1.2 5/18/84
130 REM          BY JIM ALLEN
150 A=679-1
160 A=A+1
170 READ D:IF D=0 THEN 190
180 POKE A,D:GOTO 160
190 PRINT CHR$(147)TAB(80)"{RVSON}SYS679
    {RVSOFF} TO ENABLE
200 NEW
210 DATA 120, 169, 180, 141, 20, 3
220 DATA 169, 2, 141, 21, 3, 88
230 DATA 96, 165, 197, 201, 60, 240
240 DATA 17, 162, 17, 142, 185, 2
250 DATA 201, 57, 240, 29, 162, 29
260 DATA 142, 194, 2, 76, 49, 234
270 DATA 169, 14, 141, 185, 2, 169
280 DATA 157, 141, 119, 2, 169, 44
290 DATA 141, 120, 2, 169, 2, 133
300 DATA 198, 208, 232, 169, 5, 141
310 DATA 194, 2, 169, 157, 141, 119
320 DATA 2, 162, 5, 189, 250, 2
330 DATA 157, 119, 2, 202, 208, 247
340 DATA 169, 5, 133, 198, 208, 205
350 DATA 68, 65, 84, 65, 0
```

2/b. lista

```
 50 REM ********************************
 55 REM
 60 REM   DATA BEIRAST SEGITO PROGRAM
 65 REM
 70 REM          HARNA M. ANDRAS
 73 REM           SZEGED, 1987
 75 REM
 80 REM ********************************
100 PRINT"{CLR}{3 DOWN}"TAB(16)"DATA - M
    ANKO"
110 PRINT"{3 DOWN} ← = DATA{3 SPACES}{RV
    SON} SPACE {RVSOFF} = ,"
140 FOR T=1 TO 3000:NEXT
150 A=679-1
160 A=A+1
170 READ D:IF D=0 THEN 190
180 POKE A,D:GOTO 160
190 PRINT"{2 DOWN} INDITAS: {RVSON} SYS6
    79 {RVSOFF} "
200 NEW
210 DATA120,169,180,141,20,3
220 DATA169,2,141,21,3,88
230 DATA96,165,197,201,60,240
240 DATA17,162,17,142,185,2
250 DATA201,57,240,29,162,29
260 DATA142,194,2,76,49,234
270 DATA169,14,141,185,2,169
280 DATA157,141,119,2,169,44
290 DATA141,120,2,169,2,133
300 DATA198,208,232,169,5,141
310 DATA194,2,169,157,141,119
320 DATA2,162,5,189,250,2
330 DATA157,119,2,202,208,247
340 DATA169,5,133,198,208,205
350 DATA68,65,84,65,0
```

A LORIGRAPH rajzolóprogram hasznosítása

Sokszor hallottam már azt a véleményt, hogy az Enterprise számítógéphez készült rajzprogramok semmire sem használhatók, mert a rajzok csak a rajzprogramba tölthetők vissza. Ennek megcáfolására készült két rövid programom, amelyek a LORIGRAPH rajzprogrammal tervezett rajzok és karakterek betöltését végzik el BASIC programunkba. Ezekkel a rutinokkal színesebbé tehetjük programunkat.

Az *1. listán* látható az előre elkészített rajzot betöltő program. Ezt például úgy hasznosíthatjuk, hogy a rajzot mintegy címképként töltjük be a tényleges program előtt, s a további töltési idő alatt a képernyőn láthatjuk.

Ahhoz, hogy a program működését megértsük, meg kell ismernünk a LORIGRAPH-fal kimentett rajz felépítését. Mivel a rajzprogram megengedni a színmód, a felbontás, a képernyőméret és a színek megváltoztatását, s újbóli betöltésnél ezek beállításáról automatikusan gondoskodik, joggal gondolhatjuk, hogy a képpontadatokon kívül ezeket is kimenti a háttértárolóra. Ez valóban igaz. A kimentett rajz első 13 bájtja tartalmazza ezeket az adatokat, a következő sorrendben: 0. bájt — a videomódot (HIRES;LORES); 1. bájt — a színmódot (2,4,16,256 szín); 2. bájt — a rajzlap vízszintes méretét; 3. bájt — a rajzlap függőleges méretét; 4—11. bájt — a nyolc palettaszín értékét; 12. bájt — a fixbias értékét.

Ezek az adatok akkor is kiíródnak, ha csak kétszínű rajzot készítettünk, s a többi színre nem is lenne szükség! Ez valójában megkönnyíti a rugalmas alkalmazást.

E rövid ismertető után lássuk a program működését:

- 30 Lefoglaljuk a gépi kódú rutinok számára a tárban a helyet.
- 40 Tartalmazza azt a rutint, ami a rajz adatait fogja beolvasni. Assembler-megfelelője a *2. listán* látható.
- 50—70 A rajzot ténylegesen betöltő rész. Assembler-megfelelője a *3. listán* látható.
- 80 Az adatoknak definiál egy tömböt.
- 90—140 A fájl nevét adjuk át a betöltő gépi kódú részének. Az első bájt a név hosszát adja meg, a többi a név ASCII kódja. Azért választottam a LORI3 nevet, mert a LORIGRAPH ezen a néven menti ki a rajzokat. Ez természetesen megváltoztatható, de gondoskodni kell arról is, hogy a fájl nevével egyezzen. A fájl neve maximum 7 karakter lehet ebben az esetben, mert a 8-as tárcímet a rendszer használja, s ide nem szabad POKE-okkal írnunk!
- 150 Aktivizálja a típust meghatározó rutint. Ez a rutin a 16000-es tárcímtől kezdve helyezi el a beolvasott adatokat.
- 160—240 Az előbbiekben beolvasott adatoknak megfelelően állítja be a videofeltételeket.
- 250 Megnyitjuk a videocsatornát. Azért 100 fölötti számmal, hogy a program végén levő RUN parancs ne zárja le.
- 260—270 A színeket állítja be az eredeti adatok alapján.
- 280 Megjelenítjük a videolapot.
- 290—340 Kiszámítja a videolapunk által elfoglalt RAM-terület nagyságát, s elhelyezi a gépi kódú rutinunk számára is hozzáférhető helyen. (Ezt BASIC-ben egyszerűbb volt megírni.)
- 350 Aktivizálja a betöltő rutint, s betölti a rajzot.
- 360 Betölti és futtatja a megadott BASIC-programot.

1. lista

```
 10 PROGRAM "LORITOLT.BAS"
 20 REM KEYSOFT-1989.02.25-
 30 ALLOCATE 73
 40 CODE TIPUS=HEX$("3E,6A,11,00,00,F7,
    01,3E,6A,01,0D,00,11,80,3E,F7,06,C9")
 50 CODE LOADER=HEX$("F3,DB,B1,F5,DB,
    B2,F5,DB,B3,F5,FB,3E,78,06,03,F7,0B,
    F3,3E,FF,D3,B3,3D,D3,B2")
 60 CODE =HEX$("3D,D3,B1,FB,C5,D1,ED,
    4B,00,00,3E,6A,F7,06,3E,6A,F7,03")
 70 CODE =HEX$("F3,F1,D3,B3,F1,D3,B2,
    F1,D3,B1,FB,C9")
 80 NUMERIC ADATOK(12)
 90 POKE 0,5
100 POKE 1,76
110 POKE 2,79
120 POKE 3,82
130 POKE 4,73
140 POKE 5,51
150 CALL USR(TIPUS,0)
160 LET TAR=16000
170 FOR I=0 TO 12
180   LET ADATOK(I)=PEEK(TAR)
190   LET TAR=TAR+1
200 NEXT I
210 SET VIDEO MODE ADATOK(0)
220 SET VIDEO COLOR ADATOK(1)
230 SET VIDEO X ADATOK(2)
240 SET VIDEO Y ADATOK(3)
250 OPEN #120:"VIDEO:"
260 SET #120:PALETTE ADATOK(4),ADATOK(5),
    ADATOK(6),ADATOK(7),ADATOK(8),ADATOK(9),
    ADATOK(10),ADATOK(11)
270 SET #120:BIAS ADATOK(12)
280 DISPLAY #120:FROM 1 AT 1 TO ADATOK(3)
290 LET RAM=ADATOK(2)*ADATOK(3)*18
300 IF ADATOK(0)=4 THEN LET RAM=RAM/2
310 LET HIGHB=INT(RAM/256)
320 LET LOWB=RAM-256*INT(RAM/256)
330 POKE 0,LOWB
340 POKE 1,HIGHB
350 CALL USR(LOADER,0)
360 RUN "PROGRAMNEVE"
```

Futtatandó programunkat a következőképpen célszerű kezdeni.

```
10 CLOSE #120
20 TEXT
```

Ennek köszönhetően felszabadul a rajz által lefoglalt videomemória.

A programmal betölthető rajz mérete a teljes képernyő (x=42; y=27) méretéig terjed. Ennél nagyobb y-méretet nem fogad el, a 280-as sorban hibát jelezne, mert nagyobb kép nem jeleníthető meg egyszerre.

A szalagon a fájlok sorrendjének a következőnek kell lennie: a képbetöltő rutin, a kép adatai, a futtatandó program. Lemezegységes rendszernél a sorrend nem érdekes.

Beírásnál figyeljünk a gépi kódú rész pontos gépelésére. Angol gépnél a "#" karakter helyett " " jelet kell írni.

A következő számban a LORIGRAPH-fal készített karakterek felhasználását ismertetem.

Vicsotka Gyula

2. lista

```
LD A,106          ;A csatorna száma
LD DE,0           ;Mutató a fájl névre
EXOS 1            ;Csatorna nyitása funkció
LD A,106          ;A csatorna száma
LD BC,13          ;A beolvasandó bájtok száma
LD DE,16000       ;A tárolási cím
EXOS 6            ;Blokk olvasása funkció
RET               ;Visszatérés a BASIC-hez
```

3. lista

```
DI                ;Letiltja a megszakitást
IN A,(0B1h)       ;Z-80-as 1-es lap értékét
PUSH AF           ;menti el a verembe
IN A,(0B2h)       ;Z-80-as 2-es lap
PUSH AF
IN A,(0B3h)       ;Z-80-as 3-as lap
PUSH AF
EI                ;Ujra engedélyezi a megszakitást
LD A,120          ;A BASIC-ben megnyitott VIDEO cs.
LD B,3            ;Speciális funkció hivás, ami
EXOS 11           ;visszaadja a VIDEO cs. RAM-ját
DI                ;Lapozás előtt mindig célszerű
LD A,0FFh         ;A VIDEO RAM-ok belapozása
OUT (0B3h),A      ;a megfelelő Z-80-as szegmensekre
DEC A
OUT (0B2h),A
DEC A
OUT (0B1h),A
EI
PUSH BC           ;BC és DE felcserélése
POP DE
LD BC,(0)         ;BASIC-ből POKE-lt érték olvasása
LD A,106
EXOS 6            ;Blokk olvasási funkció
LD A,106
EXOS 3            ;A csatorna lezárása
DI
POP AF
OUT (0B3h),A      ;Az eredeti Z-80-as szegmensek
POP AF            ;visszaállitása
OUT (0B2h),A
POP AF
OUT (0B1h),A
EI
RET               ;Visszatérés a BASIC-hez
```

Enterprise — vállalkozás, avagy miért plusz a PLUS?

A CentrumNagyker — amely sokáig kizárólagos hazai forgalmazója volt az Enterprise számítógépeknek és tartozékoknak — a több mint kétéves szakmai ismeretanyag és információs tömeg birtokában saját fejlesztésbe kezdett. A kifejlesztett termékeket maga menedzseli, forgalmazza, terjeszti — stílszerűen az Enterprise nevéhez: vállalkozik is! Nemrégen az EP márkaboltokban megjelent az Enterprise PLUS SOFTCART fantázianevű firmware-termék (IC-be égetett program). Alapját az a felismerés adta, hogy az eltelt két év alatt kifejlődött egy magasabb szintű igény, amely most már nem elégszik meg a felhasználói programok és utilityk kazettás változatainak szűkös lehetőségeivel.

Egyes fejlesztési munkák során a program betöltése kazettáról nehézséget okozhat a felhasználónak. A körülményes állománykezelés és a sok vesződséggel járó fordítás még a legtürelmesebb fejlesztő kedvét is hamar elveszi. Az Enterprise bal oldali csatlakozási felülete, a ROM BAY így szinte kínálta a megoldást a SOFTCART elnevezésű újdonság bevezetéséhez. Ugyanakkor az is tény, hogy jelenleg csak egy szűk réteg engedheti meg magának az EX-DOS lemezvezérlő kártya és vele együtt egy mono vagy dual floppy használatát. Nem beszélve arról, hogy a Centrum Áruházak számítástechnikai választékából még mindig hiányzik a fordítók és utilityk lemezes verziója.

ROM BAY

Mindenki előtt ismert, hogy az Enterprise egyelőre 4 × 16 kbájtos ROM-ot képes kezelni a bal oldali cartridge csatlakozója felől. Ennek a ténynek az ismeretében — és annak az információnak tudatában, hogy a felhasználók kezdettől fogva kedvezően fogadták a BASIC interpreter modulban történő elhelyezését — született a SOFTCART. A forgalmazó információja szerint 2000 darab PLUS cartridge kerül folyamatosan a boltokba. A 2950 forintos irányár meglepően olcsó a korábbi drága kiegészítők és tartozékok áraihoz képest. Az ár alapján úgy tűnik, sikerült a termék gyártása során igen kedvező arányt kialakítani az import és a hazai alkatrészek között.

Enterprise PLUS cartridge

Az esztétikus, vákuumfóliás kiszerelésben forgalomba kerülő termék az alábbiakat tartalmazza. Egy műanyag cartridge-dobozt, egy háromfoglalatos NYÁK-lapot, egy 27128 jelű, japán EPROM-ot, egy használatba vételi engedélyt és végül egy öntapadós színes cartridge-címkét — ez utóbbi kettő a használati utasításban van benne.

Az EP PLUS jelölésű gyári EPROM a középső foglalatban van elhelyezve. A bevezető NYÁK-sínhez közelebb eső másik szabad foglalatba is csak 16 kbájtos EPROM helyezhető el, a harmadik foglalat 1 db 32 kbájtos EPROM fogadására alkalmas.

Üzembe helyezés

Az EP PLUS cartridge üzembe helyezésére három lehetőség kínálkozik. A vevőnek azonban minden esetben célszerű magával vinnie az eredeti BASIC cartridge-et az áruházba, így mindjárt a helyszínen kipróbálhatja a terméket. A legkevesebb probléma azokkal az angol cartridge-ekkel van, amelyekhez csak egy foglalat tartozik. Ilyen a forgalomba hozott gépek 86 százaléka. A gépek 3 százaléka olyan speciális, kétfoglalatos NYÁK-kal került az üzletekbe, amely tartalmazza az angol BASIC interpreter PROM-ot és egy 128 kbájtos EPROM-ot, a német BASIC interpretert. A fennmaradó 11 százalék esetén azonban a 27256-os EPROM szolgál arra, hogy annak alsó részében a német, a felső részében az angol BASIC interpreter helyezkedjen el.

Az Enterprise PLUS-nak csak az angol BASIC interpreterre van szüksége. Az EPROM ki- és beszerelését végezheti az eladó (ki vannak képezve) vagy a vevő, amennyiben szakember. De ha sem az eladó, sem a vevő nem meri behelyezni az EPROM-ot a foglalatba, akkor az EP márkaszerviz, illetve az IC-k cseréjét a gyártó Titán Kisszövetkezet (Bp. XIV., Nagy Lajos király útja 110—112.) percek alatt elvégezheti.

Azok, akiknek beégetett EPROM-juk van, vagy leadják az EPROM-ot és kapnak egy angol BASIC PROM-ot, vagy a CA. FLÓRIÁN Alkatrész Shopban vásárolhatnak külön egy angol BASIC PROM-ot, és akkor megmarad a 27256-os EPROM-juk is. Újszerű a firmware-termékben, hogy valamennyi EPROM-nak önálló gyári sorszáma van, amelyet a forgalmazó a vevő nevével együtt számítógépes nyilvántartásba vesz. Így kívánja biztosítani, hogy a folyamatos fejlesztések eredményei eljussanak a régi vevőkhöz. Ez utóbbi tényt a két fél közötti használatba vételi engedély szabályozza.

Mitől plusz a PLUS?

1. Az EP PLUS EPROM három olyan szoftvert is tartalmaz, amelyek a gép már meglévő jó tulajdonságait még tovább fokozzák: a WP (Word Processor Plus) 2.1 jelű verzióját, az Enter Video 2.3 jelű verzióját és a BASIC EXtension I—IV. 2.0 jelű verzióját.

Az új szövegszerkesztő, a WP 2.1 verziója az időközben felfedezett hibákat korrigálta. Megoldást talált a 40/80 karakteres képernyőkezelés problematikájára és az ékezetes magyar ábécé használatára is. A szövegszerkesztő bővített verziója révén a szerkesztett szövegek zavartalanul kiírathatók EPSON RX-80-as, illetve Datacoop BABY printeren. A sokak által ismert Enter Video révén direkt módon hívhatók meg a VSAVE, VLOAD, VDUMP javított kiterjesztések. A BASIC EXtension 2.0 a Pascal szintű BASIC 200 utasítását újabb 65 utasítással és függvénnyel egészítette ki.

Külön figyelmet érdemel az EP PLUS JOY_MOD: változója, amely alkalmassá teszi a számítógépet egy a közeljövő-

ben megjelenő 10 DIN-es szabványú numerikus billentyűzet kezelésére. A DATUM_FLAG: egy öröknaptár paraméterezésére szolgál, a DEFAULT A$ pedig beállítja a mentéshez/betöltéshez használt alapértelmezésű eszköz nevét.

2. A forgalmazó ígérete szerint már az ősszel megjelennek az eddig csak kazettán forgalmazott felhasználói programok firmware-változatai, illetve azok továbbfejlesztett moduljai. Így egy cartridge-en belül további programfejlesztési lehetőségek nyílnak a felhasználók számára. Mivel egyelőre a forgalomba kerülő firmware-termékek egyike sem lesz nagyobb, mint 32 kbájt, a harmadik üres foglalatba az alábbi programok ültethetők majd be:

Zzzip BASIC Compiler	3.0
IS-LISP Compiler	1.0
IS-FORTH Compiler	2.0
HiSOFT PASCAL Compiler	1.0
SEMI-SOFT ASMON	1.3
„HYDE" DEBUGGER	1.0
CBM Interface	4.1
UWP (Universal Word Processor)	3.0

3. „Kétnyelvűvé" teszi a számítógépet.

4. A szabaddá váló üres cartridge-dobozzal és a PLUS cartridge-dzsal egy időben forgalomba hozott 2×32 kbájtos EPROM-bővítő (1198,— Ft) NYÁK-kal szinte korlátlan lehetőséget biztosít a felhasználónak, akit most már legfeljebb csak a pénztárcája korlátoz abban, hogy hány felhasználói programot működtet.

5. A gyártók a termék külső megjelenésében is egy plusz szolgáltatással jelentkeztek: az öntapadós színes cartridge-címkével, valamint az Index Regiszterrel, amelyre fel lehet írni az adott EPROM-ok, illetve PROM tartalmát.

A fejlesztési sorozat első gyümölcse

Már gőzerővel fejlesztik az EP PLUS filozófiáját alapul vevő, GAMECART elnevezésű termékcsalád tagjait. Egy a közeljövőben megalakuló kft. segítségével közel 20 játékprogram jelenik meg cartridge-ban, s így megoldhatónak látszik a játékprogramok bérleti rendszerének kialakítása. Hiszen az egyre növekvő kazettaárak miatt jelentősen visszaesett a forgalom a Centrum Áruházak hálózatán belül. Az említett játékok bérlési lehetőségével minden bizonnyal nemcsak a mamár közel 200 program választéka fog növekedni, hanem az árbevétel is. Az elképzelések szerint egy bizonyos bérleti idő elteltével — illetve az EUROSOFT Club tagjainak kedvezményes áron — felkínálják megvételre a GAMECART márkájú játékmodulokat.

A fejlesztések másik területe a felhasználóorientált témakörök megvalósítása egy cartridge-ban. Bizonyára örömmel szereznek tudomást a zenerajongók a MIDI interfész fejlesztéséről. Az aktív memóriakártya elvén működő MEDICART, JUSTISCART, PETROLCART, XCART elnevezésű célorientált alkalmazási módok a pacientúrával is rendelkező orvosoknak, ügyvédeknek, szállodáknak nyújthatnak speciális szolgáltatásokat. Ezek azonban igen drága berendezéseikkel és egyéb járulékos eszközeikkel belátható időn belül aligha kerülnének nálunk alkalmazásra.

Minősítés

Kedvezőnek mondható az ár/teljesítmény/használhatóság összefüggések vonatkozásában az EP PLUS cartridge. Emeli a szolgáltatások értékét a használatba vételi engedélyben vállalt kötelezettségek újszerűsége. Hiányoljuk azonban, hogy különféle billentyűzetek generálásakor az eltérő betűtípusokat nem lehet öntapadós matricával felülragasztani. Kissé szokatlannak tűnik az eredeti EP BASIC cartridge-ból a BASIC interpreter PROM kivétele, illetve az a tény, hogy a cartridge nélkül kockázatos elindulni vásárolni. Jelentős hibaforrás lehet az egy vonalról 8 vonalra dekódoló demultiplexor, a szovjet gyártmányú K555ID7. Úgy véljük, jobb lett volna például a Texas SN74LS138N vagy az SN74ALS138N. Szintén zavaró a góliát méretű magyar gyártmányú 100nF-os szűrőkondenzátor. A NYÁK-lap felületi kiképzése és az eredetivel szinte azonos megjelenésű műanyag doboz minőségi munka.

Mindent egybevetve, ha a forgalmazó és a fejlesztő betartja ígéretét, hogy még az ősszel és a jövő év tavaszán a VSZM Enterprise Klubban egy a felhasználókkal megtartandó szakmai bemutató alkalmával felvetett problémákat, programlistákat és a hiányok pótlását egy újabb verzióban az eredeti EPROM-ba beégetve kiadja, minden bizonnyal kárpótolhatja az EP-felhasználók kissé megcsappant, reményét vesztett és hitehagyott táborát.

A forgalmazótól kapott legfrissebb információ szerint az eddig eladott PLUS cartridge-ek száma már a 300-at is meghaladja.

— PT —

BASIC sorbeírás

A futás közbeni sorbeírásról, programmódosításról 1989/3. számunkban közöltünk megoldást.

Török János beküldött egy sorbeíró programot. Igaz, ez kevesebbet tud, de a megoldás rövidebb. Ötlete közvetlenül kapcsolódik az előző, *Programból alkalmazott parancsmód* című cikkünkhöz.

Az alábbi programot bebillentyűzve, közvetlen parancsokat adhatunk a gépnek. Az esetleges szintaktikai hibáktól nem „száll el", azokat hibajelzésként megüzeni. A program könnyen beilleszthető saját készítésű munkáinkba.

```
   1 ! TOROK JANOS
 100 PROGRAM **SORBEIRAS**
 110 FOR Z=1 TO 16
 120   SET FKEY Z CHR$(222)&CHR$(177)&C
HR$(13)&"RUN140-"&CHR$(13)
 130 NEXT
 140 TEXT
 150 INPUT PROMPT "A BEIRANDO SOR:   ":
A$
 160 PRINT "NYOMJ MEG EGY FUNKCIOBILLEN
TYUT:"
 170 IF INKEY$<>CHR$(222) THEN 170
 180 PRINT A$
 190 STOP
```

ENTERPRISE TOTÓ

Az 1989/9. számunkban megjelent II. forduló megfejtése a következő:

2 1 X 1 2 1 X 1 X 2 X 2 X 1

Minden kedden 17-től 20 óráig
HCC ENTERPRISE klub
a VSZM
Közösségi Házban
(Bp. XI., fehérvári út 120.)
Klubvezető: Romvári Gábor
Telefon: 810-950/473

Mi a manó?

Egyre több a panasz, hogy a tápegység csatlakozó végének ki-be huzogatása miatt (néhány program RESET gombbal nem törölhető ki, például a Heathrow ATC.) a tartólamella meglágyul, és ilyenkor már gyakorivá válik az érintkezési hiba. A csere, illetve javítás is csak egy ideig oldja meg a problémát. Javasoljuk, használják a tápegység feszültségének ki/be kapcsolóját, egy japán alkatrészekből összeállított, piros LED kijelzővel ellátott olyan mikrokapcsolót, amely nemcsak az Enterprise gépekhez, hanem a Sinclair gépek számos típusához is jó. Ára 768 forint.

●

Hamarosan befejeződnek a tárgyalások az EP Márkaszerviz és a Centrum Nagyker között, aminek eredményeképpen várhatóan a Centrum Flórián Áruházban (csak itt!) számítógépalkatrész-részleg nyílik a műszaki osztály keretén belül. Eldöntésre vár, hogy az EP-alkatrészek közül melyeket forgalmazzák, és még azt sem tisztázták, hogy az EP-n kívül lesz-e alkatrész más — például TVC — számítógépekhez is.

●

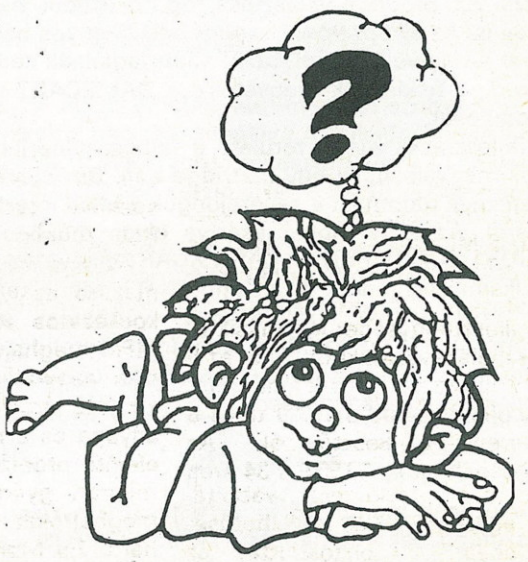
Az egyre erősödő dollárárfolyam miatt egyre kevesebb az esély arra, hogy a már katalógusunkban is meghirdetett DYRAS Boostert, az „aktív mini hangdobozt" és a CUMANA mono, illetve dual shasse-t tápegységgel importálhassuk — hangzott az egyik legfrissebb információ a Centrum Nagykerből.

●

Elfogyott és már csak az áruházakban kapható az eredeti Enterprise eredeti gyártmányú EX-DOS lemezvezérlő kártyája.

●

A forgalmazótól kapott információ szerint az RF hangmodulátorból a készletnek több mint a felét eladták már. Ez a kis berendezés, amelyet a márkaszervizek szerelnek bele a számítógépbe, arra szolgál, hogy a tévékészüléken hallgathassuk az Enterprise számítógép hangját. A berendezés ára szereléssel együtt 998 forint.



Gaetsch Günterné rajza

HARDVER BOOM

Merevlemez-illesztő egység

Melyik az igazi?

Napjainkban sok PC-gyártó hirdeti termékét a saját rendszerében alkalmazott merevlemez-illesztő és kódolási módszer megjelölésével, mivel az egész rendszer teljesítőképességének alapvető meghatározója a merevlemezes háttértár. (Érdekes a szóhasználat: az USA-ban a harddisk [merevlemez] Európában a winchester a szokásos elnevezése. Mi inkább az előbbit részesítjük előnyben, mert magyarosabb.) Olyan rövidítéseket használnak, mint az RLL (Run Lenght Limited), ESDI (Enhanced Small Device Interface) és az SCSI (Small Computer System Interface).

Hogyan működnek ezek az illesztők, és milyen módon határozzák meg a merevlemezegység teljesítményét? A következőkben ezeket próbáljuk összefoglalni, és az esetleges vásárlásoknál a döntéshez segítséget adni.

Maga a merevlemez-meghajtó, amit mi hard-disknek nevezünk, tartalmazza a mechanikát, a vezérlő elektronikát. A számítógép központi egysége és a meghajtó közötti adatátvitelt, illetve ennek vezérlését végzi az illesztő- (interfész-) kártya.

ST506: Az első szabvány

A merevlemezek alkalmazása a mikroszámítógépekben nem túl régi múltra tekint vissza. Bár néhány régebbi gépben is használták (S—100 buszt alkalmazó rendszerekben és az Apple II-ben), robbanásszerű elterjedése a 80-as évek elején megjelent 5 1/4 inches, a Shugart Technology (ma Seagate Technology) által gyártott merevlemezes egységekkel kezdődött. Ez az 5 Mbájtos — viszonylag kis kapacitású — ST506 típusjelű meghajtó volt.

Az ST506 illesztőegységét két másik illesztő felhasználásával alakították ki: az 5 1/4 inches floppymeghajtóknál használt SA450 illesztőjéből és az SA1000 típus 8 inches merevlemez-illesztőjéből. Az SA450-hez hasonlóan az ST506 is 34 eres kábelt használt a vezérlőjelek továbbítására, ami az egységek soros felfűzhetőségét (daisy-chain) is biztosította; az SA1000-esből pedig átvették a 20 eres, az illesztő és a meghajtó közötti adatáramlást lehetővé tevő „radiális" kábelt.

Az ST506-os illesztőjét 5 Mbit/s adatátviteli sebességre tervezték. Ez nem volt olyan gyors, mint az SMD illesztő (nagyszámítógépeknél alkalmazott rendszer), de elég gyors volt az akkori mikroszámítógépek sebességéhez.

Az eredeti ST506-os illesztővel kapcsolatos probléma — hasonlóan a floppymeghajtókhoz —, hogy az író-olvasó fej a sávokon egyenként lépked, pontos időzítések szerint. Mivel a léptetési sebesség korlátozott, ezért nem vezérelhető gyorsabban, mint amilyen gyorsan a fej lépkedni tud.

Ennek a problémának a megoldására fejlesztették ki az ST412 meghajtót, ami a „pufferelt keresés"-t alkalmazza. Ahelyett, hogy a vezérlő egyenként, időzítve, azaz a fejmozgás korlátja miatt lelassítva fogadná a léptető jeleket, inkább egy tárolóba olvassa be az impulzusokat. Ezután dönti el, hogy milyen gyorsan és milyen módon végzi el a kívánt sávra állást.

Az RLL kódolás

Bár az ST506-os szabvány sok alkalmazásnál megfelelőnek bizonyult, a merevlemezes meghajtó költsége magas volt. Ezért a gyártók keresték azokat a módszereket, hogyan lehet minél több adatot tárolni az ST506 meghajtón. Sok cég az IBM által szabadalmaztatott, ún. RLL tömörítési technikát alkalmazta, amelynek révén 50 százalékkal nőtt a tárolási kapacitás és a sebesség. Az RLL alkalmazása speciális vezérlőegységet igényel. (Az RLL technikát külön ismertetjük.)

Az első időkben az RLL kódolás alkalmazása az ST506-os meghajtókon elég kockázatos volt. Az elterjedt, szokásos MFM meghajtókhoz képest az RLL technika nagyobb precizitást igényel a működtető áramkörök, a lemez mágneses anyaga és a mechanika tekintetében. Ez eleinte problémákat okozott, de jelenleg minden gyártó kínál már ilyen típusú meghajtókat. A tipikus ST506/RLL meghajtó 7,5 Mbit/s-os adatátviteli sebességet tesz lehetővé, és mivel több adatot tárol egy sávon, ezért valószínűleg az író-olvasó fej mozgása is kevesebb.

Továbbfejlesztett RLL-meghajtók

Az RLL módszer alkalmazása csupán a lemezegység kapacitásának növekedését eredményezte, az adatvezetéken terjedő impulzussorozatok 5 MHz-es frekvenciáját nem változtatták meg. Néhány gyártó azonban ezt is megpróbálta 6,7 MHz-re megnövelni. Ezek a megoldások — az ARLL (Advanced RLL) és ERLL (Enhanced RLL) — mintegy száz százalékkal növelték a helyet és az adatátviteli sebességet az eredeti ST506-oshoz viszonyítva.

Az ARLL és ERLL rendszerek gyártása még problematikusabb, mint az RLL technikáé, mivel az e technika nyújtotta lehetőségek szélső határán mozognak. Ilyen sebességeknél a meghajtók nagyon érzékenyek a környezeti hőmérséklet változásaira, a gyártási tűrésekre és a kábelek hosszára.

Ezek miatt az okok miatt kétszer is célszerű meggondolni ilyen meghajtók vásárlását, és ha az általuk nyújtott nagyobb sebesség és kapacitás biztosította előnyökre szükségünk van, akkor célszerűbb az ESDI vagy az SCSI típusú meghajtók vásárlása.

ESDI (Enhanced Small Disk Interface)

1983-ban a meghajtó- és vezérlőgyártók elhatározták, hogy egy szabványos, megbízható, az ST506-os meghajtóillesztőnél sokkal fejlettebb és teljesítőképesebb interfészt alakítanak ki. Először a MAXTOR merevlemezgyártó cég kezdeményezte az ESDI szabvány kifejlesztését.

Az ESDI kábelkialakítása az ST506-éval megegyező, mégis számos előnyt biztosít, és a kompakt lemezmeghajtóknál is ezt az illesztést alkalmazzák.

Miben más akkor az ESDI? A legfontosabb változtatás az, hogy az adatszeparátort (ez az egység választja le az író-olvasó fejről érkező jelekből az adat- és időzítőjeleket) a vezérlőkártyáról magára a meghajtóra tették át. Ez két előnnyel járt: egyrészt eddig a hosszú kábelen átvitt jel sokat torzult, s ezért nehezebb volt kezelni, másrészt az adatszeparátort — mivel

most már a meghajtóra került — optimalizálni lehetett az adott mechanikára és a lemez mágneses anyagára. Mivel az ESDI kábelén analóg jelátvitel nem történik, ezért könnyen elérhető a 10 Mbit/s-os adatátviteli sebesség, és az elméleti határ 24 Mbit/s.

Az ESDI interfésznél a vezérlőjelek is jól átgondoltak. Bár az író-olvasó fej léptetése az ST506-osnál alkalmazott módszer szerint is történhet, az ESDI vezérlőnek a kívánt sávszámot egy bináris számként is megadhatjuk, és a vezérlő automatikusan végrehajtja a sávra állást. Más ESDI parancsok lekérdezhetik a konfiguráció jellemzőit — például hogy a meghajtó kompakt lemezt használ, volt-e lemezcsere —, és diagnosztikai teszt végrehajtására is utasíthatnak.

SMD

A Control Data Corporation (CDC) által kifejlesztett illesztőt nagy kapacitású és cserélhető merevlemezes egységek (=Bernoulli-boxok) illesztésére fejlesztették ki. Az IPI szabvány bevezetéséig az SMD volt a szabvány a nagy kapacitású, 5 1/4 inchnél nagyobb méretű lemezeknél.

Az ESDI-hez hasonlóan, az adatszeparátor a meghajtóegység vezérlőkártyáján van, és így 14,4 Mbit/s átviteli sebességet biztosít. Az SMD—E jelű fejlettebb verziónál ezt 20 Mbit/s-ra növelték. Mivel más szabványok jobban elterjedtek, ezért SMD meghajtókat ritkán alkalmaznak mikroszámítógépes rendszerekben.

SCSI

Ezt az interfészt az 1970-es években dolgozták ki, a számítógép és egy intelligens lemezmeghajtó vezérlője közötti kapcsolat megvalósítására. A Shugart Associates cég először a SASI (Shugart Associates System Interface) néven vezette be. Lehetővé teszi, hogy a számítógép bájt szélességű adatbuszon és néhány egyszerű vezérlőjellel kommunikáljon a meghajtóval.

A számítógépgyártók számára ez a rendszer számos előnyt jelent. A sokfajta vezérlőtípust (az ST506, SMD stb.) eggyel helyettesíti, és lehetővé teszi, hogy a számítógépes rendszer összeállítója csupán egy intelligens vezérlőt és a kapcsolódó meghajtókat használjon. Az összeállítónak nagyon keveset kell tudnia a kapcsolódó meghajtók elektromos és fizikai jellemzőiről.

Ez az eszközfüggetlenség más típusú perifériagyártók számára is vonzó. SCSI illesztést használnak mágnesszalagos egységeknél, floppymeghajtóknál, Bernoulli-boxoknál, RAM-diszk egységeknél.

Az évek során az SCSI nagymértékben továbbfejlődött. Az eredeti SASI illesztő 1,5 Mbájt/s-os adatátviteli sebességgel működött, a továbbfejlesztett SCSI ezt már 4 Mbájt/s-ra növelte. Az új, SCSI-2 specifikációja, amelyet már sok készülékgyártó elfogadott, és hamarosan ANSI szabvánnyá válik, már 10 Mbájt/s-os sebességet tesz lehetővé és opcionálisan 16 és 32 bites adatvonal-szélességgel is képes működni.

Az SCSI-2 összes lehetőségét kihasználva, az elméletileg elérhető maximális adatátviteli sebesség 40 Mbájt/s — jóval több, mint amit ma a legtöbb mikroszámítógép ki tudna használni. A fejlődés azonban ezt a határt is valószínűleg hamarosan eléri.

Az SCSI illesztés műszaki leírása vaskos köteteket tesz ki. A legfontosabb tulajdonsága az, hogy sokkal több meghajtótípus illesztésére alkalmas, mint az előbbiekben leírtak.

Jelenleg a különféle készülékek gyártói kismérvű inkompatibilitással gyártják az SCSI illesztőket, de kidolgozás alatt áll egy szabványos, CAM-nek (Common Access Method = közös elérési módszer) nevezett eljárás e probléma megszüntetésére.

Összegezve: az SCSI előtt nagy jövő áll. Bizonyára ez az oka, hogy az olyan cégek, mint az Apple, a Sun, a NeXT és mások, kizárólag ezt az illesztést alkalmazzák merevlemezes egységeiknél.

IPI= Intelligens periféria-interfész

Nagy rendszerek (IBM, CDC, Unisys) részére kifejlesztett illesztés. Jellemzői a nagy kábelhossz (125 méterig!), a nagy számú lemezmeghajtó kezelése és a nagyon nagy adatátviteli sebesség (80 Mbit/s felett).

Az IPI többszörözött vezérlőket használ, amelyek nagymértékben intelligensek, és elrejtik a meghajtók fizikai jellemzőit. Nem valószínű, hogy a közeljövőben a mikroszámítógépekben használni fogják, de ki tudja?

Az eredeti adatbit csoport	2,7 RLL kódolás (0-szünet 1-pulzus)
0 0	1 0 0 0
0 1	0 1 0 0
1 0 0	0 0 1 0 0 0
1 0 1	1 0 0 1 0 0
1 1 0 0	0 0 0 0 1 0 0 0
1 1 0 1	0 0 1 0 0 1 0 0
1 1 1	0 0 0 1 0 0

A 2,7 RLL kódolási táblázat

A megfelelő illesztő

Az ST506-os illesztő alkalmazása jó választásnak tűnik, mivel a legtöbb merevlemezes meghajtóegység ezt használja. Ha valaki nagyobb tárolási kapacitást akar, célszerű az RLL vezérlők és meghajtók közül választani. Mind a vezérlőnek, mind a meghajtónak RLL típusúnak kell lennie! A vezérlők kiválasztásánál azt is figyelembe kell venni, hogy más típusú kártya kell az XT és más az AT gépekbe, a lemez periféria portjainak eltérő címzése miatt.

Ha valaki új gépet vásárol vagy a meglévőt bővíteni akarja, és még nincs merevlemezegysége, az ESDI vagy az SCSI vezérlők választását javasoljuk a maximális teljesítmény eléréséhez. Ha valakinek SCSI illesztője van, adott a lehetőség mágnesszalagegység, kompakt diszk vagy más típusú meghajtóegység illesztésére.

Végül hangsúlyozni kell, hogy egy rendszer tényleges teljesítményét nem csupán a lemezillesztők határozzák meg, hanem a szoftver, maga a központi egység és a gyorstár (cache memory) megléte is befolyásolja.

Az egyes rendszerek legfontosabb jellemzőit a könnyebb áttekinthetőség érdekében táblázatban foglaljuk össze.

Merevlemez típusok összehasonlító táblázata

Típus	Vezetékek Daisy-chain	Radiális	Adatút szélesség (bit)	Távolság (m)	Átviteli frekvencia (MHz)	Adatátv. seb. (Mbájt/s)
ST506/412	34	20	1	3	5	0.625
ST506/412/RLL	34	20	1	3	5	0.9375
ESDI	34	20	1	3	10	1.25
SMD	60	26	1	15	14.4	1.8
SMD-E	60	26	1	15	24	3
SASI	50	-	8	3	1.5	1.5
SCSI	50	-	8	25	4	4
SCSI-2	50+68	-	8+24	25	10	10-40
IPI-3	50	-	16	125	5	10
Enhanced IPI	50/100	-	16+16	>60	12.5	50

RLL és társai

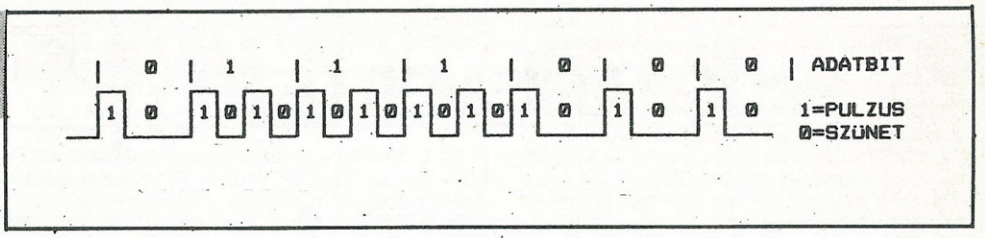
Ahhoz, hogy az RLL kódolás lényegét megértsük, először tekintsük át a lemezeknél használt, ma alkalmazott kódolási eljárásokat.

Frekvenciamoduláció (FM)

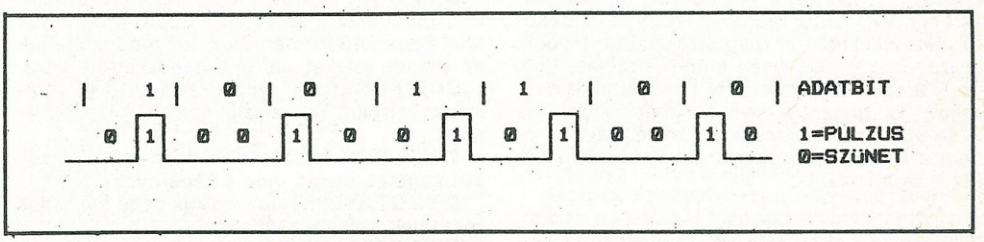
Az adatok a lemezen impulzusok és szünetek sorozatainak formájában rögzítődnek. Az FM-kódolásnál minden 0 és 1 értékű adatbit pulzust és szünetet tartalmazó csoportokba van rögzítve. Például, ha a szünetet egy impulzus követi, akkor az adatbit 0, ha kettő, akkor az adatbit 1. Az itt és a továbbiakban említett impulzus más néven az órajel, amely időzítési célokra is szolgál. Mivel minden adatbithez egy óraimpulzus tartozik, a vezérlő áramkör nagyon könnyen előállítja a jelből az adatokat.

Az *1. ábra* mutatja azt, hogy miért hívjuk ezt a módszert frekvenciamodulációnak. A csupa 1-esekből álló adatsorozatban kétszer annyi impulzus van, mint a csupa 0-ból állókban, így átlagosan 1,5 pulzus van bitenként.

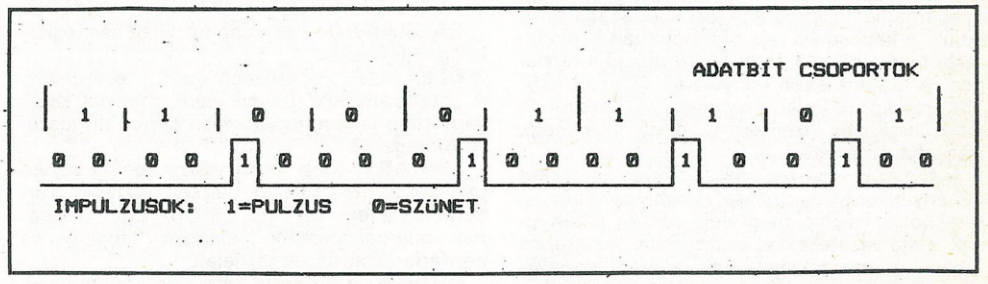
Nagyon egyszerű annak a meghatározása, hogy mennyi adat helyezhető el a lemezen: elég helynek kell lennie az impulzusok között, hogy azok megkülönböztethetők legyenek. Az FM-kódolás mindig elég helyet biztosít. Az adatbitek maximális száma éppen fele a maximálisan elhelyezhető impulzusok számának. Ha kevesebb impulzussal kódolnánk az adatokat, akkor több adatot tudnánk elhelyezni a lemezen.



1. ábra. Az FM kódolásnál minden bit vagy egy pulzussal és egy szünettel (0), vagy két egymást követő pulzussal (1) van kódolva



2. ábra. Az MFM kódolásnál a pulzusok között legalább egy szünet van. Mivel a lemezre rögzíthető adatmennyiség a pulzusok közelségétől függ, ezért kétszer annyi adat helyezhető el a lemezen, mint az FM kódolásnál



3. ábra. Itt látható, hogy kódolható egy bitalakzat a 2,7 RLL kódolás szerint. Minden kódcsoport 4-8 félbit hosszúságú és 2-4 adatbitet képes kódolni. Az alakzat hossza függ az eredeti adatbitektől, de a pulzusok megoszlása garantálja a fenti 2,7 maximális és minimális futási hosszt

Módosított frekvenciamoduláció (MFM)

Az MFM módszernél a kódolási szabály a következő. Az 1 adatbitet a szünet utáni impulzus reprezentálja, a 0-t a következő két alakzat jelzi: az impulzus után egy szünet, ha nem volt impulzus az előző adatbit végénél, illetve két szünet, ha az előző bit impulzussal fejeződött be.

Az MFM biztosítja, hogy mindig legalább *egy* szünet lesz az impulzusok között (ami azt jelenti, hogy sokkal közelebb helyezhetők el a jelek az egybeolvadás veszélye nélkül), de *háromnál* nem több (ami még lehetővé teszi az órajel visszaállítását). Ez átlagosan 0,75 impulzust jelent bitenként, feltételezve, hogy a 0-t jelölő kétfajta alakzat előfordulása egyforma. Emiatt, mivel kétszer akkora kapacitást biztosít, szokták az ezt használó diszkeket dupla sűrűségű meghajtóknak is nevezni *(2. ábra)*.

Az ST506-os meghajtó eredetileg MFM kódolást használt. Van esetleg olyan kódolási módszer, amivel még jobban meg lehet növelni a tárolási sűrűséget? A kérdés igenlő megválaszolásához vizsgáljuk meg a következő módszert. Vezessük be a futási hossz (run lenght) kifejezést a rögzített jelekben egymást követő legrövidebb és leghosszabb szünetek jelölésére.

Az FM technikánál a minimális futási hossz értéke 0 (lehetséges, hogy ne legyen szünet az impulzusok között), és a maximális futási hossz értéke 1 (a szünet után mindig van órajel). Ilyen módon röviden az FM 0,1 korlátozott futási hosszú, vagy röviden: 0,1 RLL.

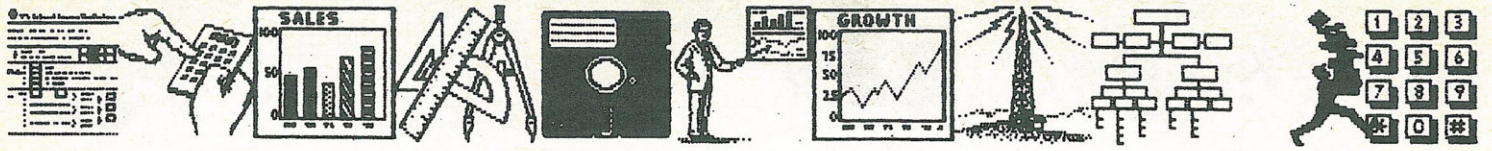
Hasonló módon az MFM-nél mindig legalább egy szünet van az impulzusok között, de háromnál nem több, azaz 1,3 RLL.

Általában a minimális érték adja meg, hogy az adatok milyen sűrűn helyezkedhetnek el a lemezen, míg a maximális érték meghatározza a vezérlő áramkör időzítési pontosságát (hiszen akkor is elő kell állítani az órajeleket, amikor szünet van), valamint emiatt a lemez forgási sebességét is szűkebb határon belül kell közel állandó értéken tartani.

Pontosabb időzítés

Az a kódolási eljárás, amit mi RLL-nek ismerünk, a 2,7 RLL *(3. ábra)*. Ez sokkal összetettebb kódolási szabályokat alkalmaz a bitek impulzussorozattal való leírásához és az előző adatbit értékét is felhasználja. Az alapelv azonban ugyanaz: kevesebb impulzust használ, de sokkal pontosabb időzítéseket igényel.

(A Byte 1989. februári számában megjelent cikk nyomán)



PROGRAMISMERTETŐ

OrCAD/SDT

Kapcsolási rajzot készítő program

Villamos kapcsolási rajzok készítésében nagy segítséget nyújthat a számítógép. A kapcsolási rajz készítése lényegében az alkalmazott elemek rajzszimbólumainak és az összekötő vezetékeknek a megrajzolásából tevődik össze. Ez a tevékenység automatizálható úgy, hogy a szimbólumkészletet könyvtárakban tároljuk, és onnan a kívánt elemet előhívjuk. A szimbólumok nemcsak a körvonalrajzot, hanem a kivezetéseket, azok nevét és számozását is tartalmazzák.

Tervezés „rajzlapon"

Az OrCAD Systems Corporation által forgalmazott OrCAD/SDT (Schematic Design Tools) nevű program az előbbiekben körvonalazott feladatot végzi el. A programmal különböző méretű „rajzlapokon" dolgozhatunk. A képernyőn mindig a rajzlap egy tetszőleges részletét látjuk. A kapcsolási rajz szimbólumait könyvtárakból vehetjük elő és helyezhetjük el a rajzlapon. A legfontosabb könyvtárak: TTL; CMOS; LSI-áramkörök (mikroprocesszorok, memóriák stb.); eszközök (tranzisztor, dióda, kapcsoló, transzformátor stb.).

A tervezés úgy történik, hogy az elemeket néhány billentyűnyomással meghívjuk a könyvtárakból, a rajzlap megfelelő részén elhelyezzük, majd vezetékekkel összekötjük. Az azonos funkciójú vezetékeket egy vastag vonalba egyesítve, buszvezetékeket is kialakíthatunk.

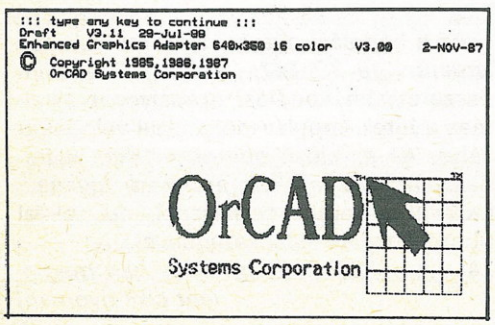
A bonyolult vagy terjedelmes kapcsolási rajzok hierarchikus rendszerbe szervezhetők. Ilyenkor több lap tartalmazza a teljes kapcsolási rajzot.

Az elkészített kapcsolási rajz mátrixprinteren kinyomtatható, plotteren kirajzoltatható és alkatrészlista is készíthető. Előállítható az alkatrészeket összekötő ún. kötési lista is. Ilyenkor bizonyos összekötési alapszabályokat is ellenőriz a program.

Az elkészült rajz és alkatrészlista együttesen a teljes dokumentáció fontos, könnyen javítható része.

Az OrCAD egy programcsomag, amely több különböző funkciójú programból, könyvtárfájlokból és perifériákat kezelő meghajtó (driver) programokból áll. A következőkben röviden felsoroljuk a programok nevét és funkcióját.

Az OrCAD program bejelentkezési képe



DRAFT: a tulajdonképpeni rajzolást és szerkesztést végzi.

DECOMP: a DRAFT által használt rajzjeleket konvertálja a szövegszerkesztő programok által kezelhető formátumba. Így módosíthatjuk az eredeti jeleket, és újakat hozhatunk létre.

COMPOSER: a szövegszerkesztővel elkészített rajzokat konvertálja át a DRAFT számára.

TREELIST: hierarchikus szervezésű rajzok struktúráját jeleníti meg a képernyőn.

ANNOTATE: a hierarchikus vagy flat fájlok részeinek azonosítását végzi.

PRINTALL: nyomtatás.

PLOTALL: rajzolás plotteren.

PARTLIST: alkatrészlista.

ERCHECK: alapvető elektrotechnikai hibák kiszűrése.

NETLIST: egyéb rajzolóprogramok által elolvasható, kapcsolási rajzokat tartalmazó fájlokat állít elő.

BACKANNO: a tervben használt azonosítókat gyűjti össze.

CLEANUP: rajztechnikai ellenőrzés. A „duplikátumok" (olyan elem, melyből kettő vagy több is van, ugyanolyan címkével) kiszűrését végzi el.

LIBRARY: több alkönyvtárra bontva ebben találhatjuk meg az alkatrészkönyvtárakat. Tartalmazza a legtöbb TTL-, CMOS-, ECL-, memória-, mikroprocesszor-, diszkrét-, analóg- és periféria-alkatrészek rajzjeleit.

DRIVER: a program különféle képernyővel, printerrel, plotterrel ellátott rendszeren képes futni. Ezeket a meghajtóprogramokat tartalmazza ez a csoport.

Mivel maga a rajzolás a DRAFT programmal történik, használatával részletesebben is meg kell ismerkedni. A program — viszonylagos bonyolult felépítése ellenére is — egyszerű, menükialakítása mintaszerű, és használat közben szinte minden felmerülő kérdésre választ kapunk.

Előnyök

A DRAFT számos előnye közül kiemelendő, hogy több mint 2700 elemből álló, bővíthető alkatrészkönyvtárat tartalmaz. A TTL elemek De Morgan-ekvivalensének előállítása egy gombbal lehetséges. Öféle lapméret állítható be, a legnagyobb mérete 44 × 34 inch (1117,6 × 863,6 mm). Max. 5000 mélységű, vagyis gyakorlati szempontból korlátlan hierarchikus szervezés érhető el, 5 perspektíva (zoom) használata lehetséges. Több mint 100 felhasználó által definiálható utasításmakró kapcsolódhat a programhoz, lehetséges sztringek keresése, és kiadhatók a DOS parancsai is a program működésének ideiglenes felfüggesztésével.

A program indítása többféleképpen történhet. A rendszer konfigurálását a DRAFT/C parancs begépelésével kell kezdeni. Ekkor megjelenik egy menü, meg kell adni a fájlok elérésének útvonalát, a használni kívánt alkatrészkönyvtárak neveit, és néhány hardveradatot. Ha ezt a megadott konfigurációt elmentjük, legközelebb már ezzel indul a program.

A DRAFT/M indítás hatására a program az egér működését letiltja. Normál esetben az elindítás egyszerűen: DRAFT, <ENTER>. Ezután megjelenik egy kérdés: LOAD FILE?. Ha új rajzlapot akarunk kezdeni, akkor <ENTER>, ha régit akarunk beolvasni, akkor be kell írni a kívánt fájl nevét, és utána kell leütni az <ENTER> billentyűt.

A program menüvezérelt, a menü főmenüre és almenükre bontható. Az éppen aktuális menü az <ENTER> gombbal hívható le. Parancsból kilépni az <Esc>-pel lehet. A parancsok megadása háromféle formában lehetséges:

— a parancs kezdőbetűjével,

— kurzornyilakkal kijelölve a menüben a megfelelő parancsot, és <ENTER>,

— egérrel, mint az előbb, ekkor a bal oldali gomb az egéren az <ENTER>, a jobb az <Exc>.

Mielőtt az utasításokkal kezdenénk foglalkozni, ismerkedjünk meg a kapcsolási rajzok lehetséges struktúráival.

ONE SHEET: egylapos. Egyszerű rajzokhoz alkalmas.

FLAT FILE: többlapos, az egyes lapok mellérendelt viszonyban vannak. Például egy mikroprocesszoros rendszer rajzánál az egyik lapon van a központi egység, a másikon a memória, egy harmadikon a periféria. A rajzokat a cím-, táp- és adatbuszok, valamint egyéb jelvezetékek kötik össze.

HIERARCHY FILE: szintén többlapos, de van egy főlap, és vannak ez alá rendelt lapok. A főlapon blokkvázlatszerűen van feltüntetve a többi. Ezt gyakorlatilag tetszés szerinti mélységig fokozhatjuk, vagyis egy alacsonyabb szintű lapnak szintén lehetnek alárendelt lapjai. Bonyolult, sok alkatrészt tartalmazó, összetett áramköri rajzok áttekinthető elkészítését teszi lehetővé.

Utasítások

A rajzlap egy kijelölt területét blokknak nevezzük. Néhány parancsnál szükség van blokkok megjelölésére. Ezek a BLOCK Move, Drag, Save, Export, DELETE Block és a PLACE Sheet. Ha ezek közül kiválasztjuk valamelyiket, a következő menüből választhatunk: Begin Find Jump Zoom escape. A kívánt blokk egyik sarkához kell mozgatni a kurzort, s kiadni a Begin (kezdet) parancsot. Ekkor a menüben a Begin helyett End jelenik meg, és ezzel kijelöltük a terület egyik sarkát. A kurzornyilak segítségével keretezhetjük a megfelelő blokkot, s ekkor az End (vége) utasítást kell kiadni. A területet ezzel definiáltuk, s folytathatjuk a megkezdett műveletet. A fenti menü többi utasításának szerepe:

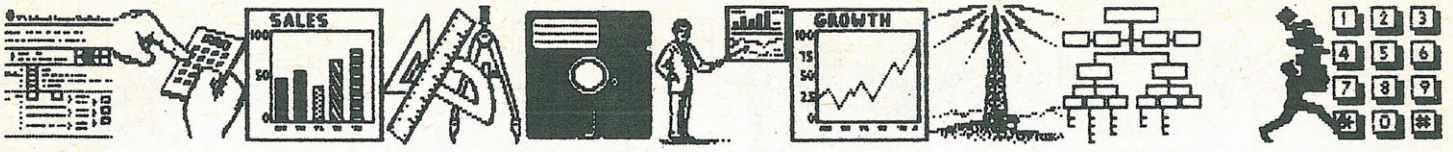
Find: megkeresi a rajzon a megadott sztringet, és oda állítja a kurzort.

Jump: ugrás a (későbbiekben ismertetendő módon definiált) pozíciókra.

Zoom: a rajzlap különböző léptékű megjelenítéséhez a képernyőn különböző kicsinyítést, illetve nagyítást kérhetünk.

Ezek után vegyük sorra a főmenü utasításait. Hogy a hierarchikus szervezést láthatóvá tegyük, egyes sorokat bekezdéssel szedtünk.

AGAIN: legutolsó utasítás ismétlése.

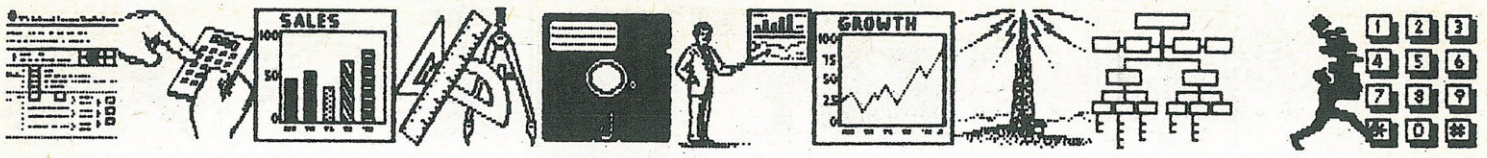


BLOCK: blokkműveletek.
Move: blokkmozgatás az összekötettések fenntartása nélkül.
Drag: blokkmozgatás az összekötettések fenntartásával („gumiszál").
Fixup: a „Drag" blookmozgatás során összekuszálódott vezetékek rendezése.
Alparancsai: a **Pick** (felvesz) és a **Drop** (eldob).
Save: a blokk elraktározása az erre a célra fenntartott memóriarészben. Ekkor a blokk nem törlődik a rajzról.
Get: a Save által tárolt blokkot veszi elő.
Import: blokk beolvasása lemezről.
Export: blokk mentése lemezre. Ezáltal más rajzok részeit újra tudjuk használni.
CONDITIONS: egy állapotmenüt jelenít meg, amelyen a munkalap, a hierarchia-puffer, a makró-puffer és a szabad tárterület nagysága található meg.
DELETE: törlés.
Object: egyes elemek törlése. A kurzornak a törölni kívánt elem „testén" kell elhelyezkednie.
Block: blokk törlése.
Undo: az utoljára kitörölt blokk visszaállítása.
EDIT: szerkesztés. A különböző alkatrészek nevét, értékét, azok elhelyezkedését, irányát, a hierarchikus lap jelének méretét, nevét, kivezetéseinek számát, típusát változtathatjuk meg. Lényegében minden alkatrészen és minden azonosító néven tudunk módosítani, csak újat nem tudunk beilleszteni ezzel az utasítással. A kiválasztás a kurzorral történik, utána kell az EDIT parancsot kiadni.
FIND: karaktercsoport keresése akár hierarchikus lapokon át is. Ha sikeres volt a keresése, a kurzor annál az alkatrésznél fog elhelyezkedni, amelyik az adott sztringet tartalmazza.
GET: ez teszi lehetővé új alkatrészek elővételét, forgatását, konvertálását, végül elhelyezését a megfelelő pozícióba. A típus kiválasztása kétféle módon történhet: ha tudjuk a kívánt alkatrész pontos nevét, akkor be kell írni a Get? prompt után, majd <ENTER>; ha nem tudjuk, akkor azonnal <ENTER>, ekkor menüszerűen választhatunk az alkatrészalkönyvtárak közül, majd azon belül választhatjuk ki a megfelelő típust.

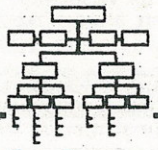
Menü

A kiválasztás után a következő menü jelenik meg:
Place: az alkatrész helyének véglegesítése, beillesztés.
Rotate: elforgatás balra 90 fokkal.
Normal: az eredeti irány visszaállítása. Ha De Morgan-ekvivalenst állítottunk elő, azt is visszakonvertálja.
Up: az eredeti irányhoz képest egyszeri balra forgatás.
Over: 180 fokos forgatás az eredetihez képest (két Rotate).
Down: 270 fokos balra forgatás az eredeti irányhoz képest.
Mirror: tükrözés a függőleges tengelyre.
Convert: csak akkor jelenik meg, ha az alkatrésznek De Morgan-ekvivalense is van. Ennek előállítására szolgál.
HARDCOPY: lehetővé teszi a rajz kinyomtatását DRAFT-ból. Plotterre rajzolni ezzel nem lehet.
JUMP: gyors mozgást tesz lehetővé a rajzterületen. Ugorhatunk meghatározott, általunk definiált pozícióba, léphetünk függőlegesen vagy vízszintesen a kurzor aktuális helyéhez képest az általunk megadott mértékben (+ szám jobbra, illetve lefelé, egy lépés 1/10 vagy 1/100 inch a beállítástól függő módon, lásd SET Grid References).
LIBRARY: alkatrészkönyvtár vizsgálata.
Directory: a kiválasztott alkatrészkönyvtár listáját jeleníti meg, ezt képernyőre, printerre vagy lemezre küldhetjük.
Browse: a kiválasztott könyvtár teljes tartalmát vagy csak az általunk kívánt részét jeleníti meg a képernyőn.
MACRO: utasítás-makró definiálható a funkciógombokra, az <Alt>, <Ctrl> és <Shift> gombok, valamint a különböző betűk kombinációira, az egér középső gombjára (ha 3 gombos), és még néhány nem használt billentyűre (Home, Pgup stb.).
Capture: makró létrehozása. Utána azt a billentyűt kell megnyomni, amelyikre definiálni szeretnénk a makrót, majd <ENTER>, utána folyamatosan kell azokat az utasításokat megadni, amelyeket egy makróba akarunk tenni. A végét az <M> billentyű jelzi.
Delete: makró törlése.
Initialize: az összes makró törlése.
List: az összes makró listázása.
Read: makró-fájl beolvasása.
Write: a memóriában levő összes makró kimentése egy fájlba.
PLACE: vezeték, busz, tápegység csatlakozása, modul port, címke elhelyezése.
Wire: vezeték. A következő menü jelenik meg: Begin Find Jump Zoom escape. A Begin paranccsal kezdhetjük el a vezetéket, a többi utasítás hatásáról már volt szó. A Begin határása újabb menü jelenik meg:
Begin: 90 fokos elfordulás esetén kell ezt választani, ekkor a vezeték rögzítődik az aktuális ponton, s onnan folytatható tovább.
End: vezetékrajzolás vége.
New: vezetékrajzolás vége, de nem lép vissza a főmenübe, hanem Beginnel újabb vonalat húzhatunk. A Find, Jump, Zoom, escape itt is megtalálható.
Bus: buszrajzolás. Hasonlóan történik a vezetékéhez.
Junction: „kitérő". A kereszteződések és a csatlakozások megkülönböztetésére szolgál. A Place alparanccsal helyezhető el.
Entry (Bus): csatlakozás a buszból.
/, \: a kilépés szögének változtatása.
Wire: csatlakozás a vezetékkel.
Bus: csatlakozás busszal.
Label: azonosítónév elhelyezése. Típusa lehet belső, buszazonosító és magyarázó.
Module Port: a hierarchikus lapok csatlakozásainak azonosítására szolgál. Meg kell adni a nevét, majd egy menüből ki kell választani a típusát (ki-, bemenet, kétirányú vagy nem specifikált).
Power: tápfeszültség csatlakozása.
Sheet: hierarchikus lap rajzjelének beillesztése. A Begin alparanccsal kell kezdeni és definiálni a területet. Utána a következő menü jelenik meg:
Add: csatlakozás hozzáadása.
Delete: csatlakozás törlése.
Edit: a csatlakozás nevének változtatása.
Name: az új lap nevének megadása.
Filename: a hierarchikus laphoz tartozó fájl neve.
Size: a lap rajzjelének méretét változtathatjuk meg.
Dashed Line: szaggatott vonal.
QUIT: kilépés, mozgás a hierarchiában.
Enter Sheet: lefelé mozgás a hierarchiában. A lap belsejébe kell vinni a kurzort, s úgy kiadni a parancsot. Vigyázat! Az aktuális lapot előtte el kell menteni, mert egyébként elvész.
Leave sheet: felfelé mozgás a hierarchiában, itt is menteni kell előbb.
Update File: mentés lemezre. Akkor alkalmazható, ha a munkalapnak már van neve.
Write File: mentés lemezre. Ekkor meg kell adni a fájl nevét.
Initialize: új rajz beolvasása vagy tiszta lap kezdése lehetséges. A LOAD FILE? kérdésre a program indításánál leírtak szerint kell válaszolni.
Suspend to DOS: a működés időszakos felfüggesztése és DOS-parancsok kiadásának lehetősége. Ezt az üzemet a »prompt jelzi. Visszatérés a programba: »EXIT.
Abandon Edits: kilépés a programból.
REPEAT: a legutoljára kiválasztott elemet rajzolja fel. A SET Repeat Parameters paranccsal meg kell adni a függőleges és a vízszintes lépés, valamint a paraméter növekményének értékét.
SET: különféle feltételek megadása.
Auto Pan at Edge: a képernyő határvonalán túl tartó mozgás lehetősége.
Backup File Made: háttérfájl készítésének engedélyezése.
Drag Buses: a BLOCK Drag művelet során a buszokat is „gumiszálként" húzza magával, ha engedélyezett.
Error Bell: hiba esetén hangjelzés kiadásának lehetősége.
Left Button: az egér bal oldali gombjának engedélyezése. Ez a parancsmenüknél az <ENTER>-t helyettesítheti.
Macro Prompts: engedélyezés esetén a makró-parancsok kiadásakor a képernyőn is megjelennek a makró utasításai.
Orthogonal: ha beállított, a vezetékek és a buszok csak vízszintesen és függőlegesen haladhatnak.
Show Pins: engedélyezi vagy letiltja a kivezetések sorszámának kiírását.
Title Block: előre definiált szövegmező automatikus felrajzolása.
Worksheet Size: a munkalap méretét lehet beállítani (5-féle).
X, Y Display: az X, Y koordináták kiírásának engedélyezése.
Grid Parameters: az 1/10 és 1/100 inches lépésközt lehet átkapcsolni, engedélyezhetjük 1/10 inch osztású rácspontok megjelenítését.
Repeat Parameters: a REPEAT parancsnál említett adatok meghatározása.
TAG: nyolc pocíziót határozhatunk meg a JUMP utasítás számára.
ZOOM: 1/1, 1/2, 1/5, 1/10, 1/20 arányi kicsinyítési lehetőség, a kívánt elem a képernyő közepére helyezhető.

Hogyan történik ezek után egy elkészített rajzfájl megrajzolása papíron? A CLEANUP programmal eltávolítjuk a rajzról a duplikátumokat. Az ERCHECK programmal ellenőrizzük a durva elektromos kötési hibákat. Az így kijavított kapcsolásirajz-fájl szolgál a PRINTALL, illetve a PLOTALL program bemeneteként. Ezekkel lehetséges az ábra kinyomtatása, illetve megrajzoltatása plotterrel.



VEGYES



Kiadványszerkesztés szövegszerkesztőkkel

Itt a WYSIWYG?

A szövegszerkesztést és irodai kiadványszerkesztést mindeddig különböző „tudományoknak" tekintették, amelyekhez más-más segédeszközök, programok kellenek. Sok ember számára azonban a szokásos szövegszerkesztés utáni következő lépés annak a lehetőségnek a kihasználása, hogy különböző betűtípusokat használjanak, grafikát illeszthessenek be a szövegbe, és lássák a képernyőn, mi fog megjelenni a papíron. Ezeknek a felhasználóknak a legjobb szövegszerkesztő csomagok mindazt biztosítják, ami a „gusztusos" levelekhez, feljegyzésekhez, ismertetésekhez kell. Mielőtt az alábbi öt nagyon jó szövegszerkesztőt átnéznénk, érdemes összeállítani azt a kívánságlistát, amelyet az ideális programnak ki kell elégítenie.

Sebesség

A szövegszerkesztés rendszerint nem számolás- és mágneslemez-igényes alkalmazás, mint a táblázat- vagy adatbázis-kezelés, ezért elvárható, hogy a szövegszerkesztők gyorsan fussanak a 8088 alapú személyi számítógépeken is. *Táblázatunk* az öt legjobbnak tartott szövegszerkesztő tesztelésének eredményét közli.

Grafika

Ideális esetben a szövegszerkesztőknek igazi WYSIWYG-hatást (azt kapod, amit látsz: What You See Is What You Get) kellene kelteniük a képernyőn, de a valóságban ezt csak az igazi kiadványszerkesztő csomagok közelítik meg, például a Ventura és a Pagemaker. Ha valóban olyan típusú karaktereket akarunk látni a képernyőn, amilyen a papírra kerül, hatékony grafikus társprocesszorra vagy 80386-os gépre van szükségünk. Az ismertetett öt program közül egyik sem rendelkezik igazi WYSIWYG-hatással, de a DisplayWrite 4 kivételével mindegyik megjeleníti a kinyomtatandó oldal megközelítő mását a képernyőn.

A Microsoft Windows 2 (Presentation Manager) interfész növekvő népszerűsége bizonyára nagy piacot teremt majd a jó WYSIWYG-képességű programoknak. A Windows Write, egy kis szövegszerkesztő program, amelyet ingyen adnak a Windows 2 és Windows/386 csomagokkal, már majdnem olyan méretű és formájú betűtípusokat jelenít meg a képernyőn, mint amilyenek a papírra kerülnek.

Más szövegszerkesztők is kaphatók, amelyek a Windows-környezetben működnek. Ilyen például a WinText és a Comfotext. A Microsoft sokat ígérő új terméke az év végére várható. A Windows-környezet használatának nagy előnye, hogy a programok más alkalmazásokból is tudnak grafikát átvenni, bár a legjobb termékek többsége is megoldja ezt a feladatot. Noha a Windows egy további lépést jelent az igazi WYSIWYG-hez vezető úton, a Windows Paint program kivételével a Windows raszteres formátumú betűtípusokat használ, vagyis a betűket pontsorozatokként tárolja a memóriában.

A következő években várható a PostScript képernyőalapú változatának megjelenése, és ekkor lehet csak igazi WYSIWYG-et használni. A PostScript titka a betűtípusok kezelésében rejlik: a szoftver ezeket bonyolult matematikai képletekkel reprezentálja.

Így lehetséges, hogy a PostScript nyomtatók rugalmasabbak a többinél, még a kiváló Hewlett-Packard Laserjetnél is. A Laserjet és a többi nyomtató raszteres formában tárolja a betűtípust, és csak a tárolt elrendezésben tudja reprodukálni. A PostScript betűtípusokat tud forgatni, nyújtani, dönteni. Ennek ára is van: a sebesség. A PostScript nyomtatók rendszerint 68000-es vagy 68020-as processzorral működnek, de a bonyolult kimenetek elkészítéséhez így is sok idő (sok perc) kell.

A mostani legjobb — kompromisszumos — megoldás a Ramfont-rendszer, amelyet a legújabb Hercules grafikus kártyák használnak. 256 karakter helyett (ennyit kezelnek az IBM kijelző adapterek a szövegmódban) 3000 karaktertípust tud megjeleníteni. Az új karakterek a szabványos ASCII-karakterek döntött, megvastagított vagy máshogy átalakított változatai. A karakterek alakját leíró mátrix RAM-ban van. A WordStar 2000 Plus, a WordPerfect 5 és a Word 4, amely a Ramfontot támogatja, a szövegmód sebességével működik, és lehetővé teszi a formátumok kialakítását a képernyőn, ami a grafikus mód sajátja.

A következőkben tekintsük át a fent említett szövegszerkesztők legfontosabb sajátságait.

DisplayWrite 4

Az IBM DisplayWrite 4 a népszerű Displaywriter új inkarnációja. A termék az előző változat kissé bővített kiadása, több menüt tartalmaz, és támogatja az egér használatát.

A szerkesztő és karakterkészítő funkciók többsége az ALT- és CTRL-billentyűkombinációkkal aktivizálható, ami kissé nehézkes. A szoftver segítőprogramja (help) azonban jól használható, szövegérzékeny.

Gyengéje a csomagnak, hogy nem tud más alkalmazásból grafikát átvenni, és csak az IBM nyomtatókat támogatja, kivéve a már ipari szabvánnyá vált PostScript lapleíró nyelvet, amely egyre több lézernyomtatón jelenik meg, így az IBM saját 4216 Personal Pageprinterén is.

A menüket kiválasztó ablakokkal, aláhúzott utasításbetűvel, utasításszámmal vagy egérrel lehet kiválasztani. A menükből almenükbe lehet lépni, de ezekből rendszerint nem lehet gyorsan visszatérni a legfelső szintre. Egyszerre csak egy szintet lehet visszalépni, ami időigényes. A csomag előnye a számos megjelenítő funkció támogatása, például vonalak húzása, a kimenet formátumának tervezése és a helyesírás ellenőrzése.

A DisplayWrite 4 jó szövegszerkesztő, de tudását elsősorban az IBM-környezetben lehet kiaknázni. A nagy gépek DisplayWrite/36 és /370 dokumentációs rendszereivel is tud fájlt cserélni.

Microsoft Word 4

Jól átgondolt, jól implementált termék. A WordPerfecttel együtt a Word foglalja el az első helyet az Egyesült Államokban és Európában is.

A Word 1 volt valószínűleg az első szövegszerkesztő, amely a PostScriptet támogatta. Az első változat bővítése nyomán a Word több ablak használatát teszi lehetővé, támogatja az egeret, majdnem WYSIWYG-megjelenítést hoz létre, ellenőrzi a helyesírást, kivonatokat készít hosszabb iratokból, van szinonimaszótára. A Word 4 gyorsabb, mint az előző verziók, és lehetővé teszi a makrók használatát is.

Két képernyőtípust támogat a Word: a szöveg- és a grafikus módot. A szövegmódban a tervezett karakterek kiemelten jelennek meg (erőteljesebben vagy színesen). Így lehet a leggyorsabban „átmenni" egy iraton.

A grafikus módban — bár a különböző méretű karakterek azonos magasságúak a képernyőn — a megvastagított, döntött betűk, az alsó és felső indexek ugyanúgy jelennek meg a kijelzőn, ahogy a papíron — ha a nyomtató támogatja ezeket a formátumokat. Ha nem, a Word megkísérel hasonló típusokat nyomtatni a nyomtatómeghajtók segítségével. A szöveg- és a grafikus mód között az irat tetszőleges helyén lehet váltani.

A grafikus mód hátránya, hogy lassúbb a szövegmódnál, hiszen az ASCII-karakterek helyett egyenként kell kezelni a képelemeket (pixeleket).

A Word 4-et a Pageview programmal együtt árulják. A Pageview a Windows 2 és a Windows/386 alatt fut, és lehetővé teszi az irat megjelenítését a képernyőn a nyomtatást imitáló formában. Segítségével a Windows Clipboardból át lehet vinni grafikát a Word-iratokba, és itt lehet őket méretezni.

Ha a grafikával együtt tesszük el a Word-iratot, a grafikát nem nézhetjük meg a Wordből. Ehelyett az iratban egy utasítássor jelenik meg, amely az iratot egy különálló grafikus fájllal kapcsolja össze (ez rendszerint ugyanabban a résztartalomjegyzékben van, és a fájl neve is megegyezik az eredetivel, csak a toldaléka különbözik). Ha a Worddel nyomtatjuk ki az iratot, a grafika helyén üres folt jelenik meg.

A Pageview használata jó ötlet, de a programot rosszul implementálták. A Pageview a Windows-zal megadott nyomtatómeghajtókat használja az összeszerkesztett iratok formátumának elkészítésére és nyomtatására. Az IBM Proprinteren például a Word képes közel betűminőségű (NLQ=near letter quality) nyomtatásra, de a Windows Proprinterhez adott nyomtatómeghajtója nagy rasztergrafikus betűket készít. Ezt elkerülendő, a Pageview-t nyomtatómeghajtó-fordító táblázatokkal látták el, amelyek a Word meghajtóit Windows-meghajtókká alakítják át. Ha pedig a Wordöt a Windows alatt futtatjuk, a Microsoft azt tanácsolja,

hogy zárjuk le a Worddel készített iratot, mielőtt ugyanazt a Pageview-val néznénk meg: ez lelassítja a munkát.

Mindezzel együtt a Word 4 valószínűleg a legjobb szövegszerkesztő, amellyel irodai kiadványszerkesztési szintű terméket lehet létrehozni. A többi szövegszerkesztőtől eltérően 64-féle betűtípust lehet vele néhány tizedmilliméteres pontossággal pozicionálni.

Sentinel Software WordPerfect 5

A WordPerfect 4.2 az a szövegszerkesztő, amelyből a legtöbbet adták el a világon. Nagyon jó a szövegfeldolgozása és gyors. Az új verzióba grafikabeépítő modult is tettek, így a program képes átvenni például a Lotus PIC fájlokat, a CGM metafájlokat, a Windows Paint és a GamPaint fájlokat. Az átvett grafikát aztán lehet méretre szabni, és meg lehet becsülni a képernyőn, hogy a kinyomtatott lap hogyan fest majd.

A Word 4-hez hasonlóan a WordPerfect 5 abszolút egységeket használ, hogy a szöveg és a grafika helyét kiszámítsa. Ez különösen a lézernyomtatóknál hasznos, amelyek esetleg egyazon sorban különböző méretű karaktereket nyomtatnak.

Ha egy szöveget több szerző ír, fontos, hogy a mások által beiktatott változtatásokat megnézhessük. A WordPerfect lehetővé teszi a szöveg „rejtett" formattálását, vagyis az írás megjelenik a képernyőn, de nem kerül a nyomtatóra. Egy másik funkció összehasonlítja a képernyőn mutatott iratot azzal, amelyik a mágneslemezen van, és megjelöli a szövegben a különbségeket.

WordStar 2000 Plus Release 3 Personal Edition

A program nagyon hatékony, számos jól használható funkciója van: például telekommunikáció-szervezés, címlista (mail merge), hatalmas szinonimaszótár. A kinyomtatandó oldalt előbb át lehet nézni a képernyőn, és lehet grafikát átvenni más programokból, például a Lotus 1-2-3-ból vagy a Symphonyból. Támogatja a Hewlett-Packard Laserjet és a PostScript nyomtatókat. A MultiMate Advantage II-vel mintegy 400 nyomtató használható!

Ennek ellenére a WordStar 2000 nem eléggé kidolgozott munka. Kiterjedten használja az overlay-programokat, hogy a maximális konfigurációt jelentő 8 Mbájtos mágneslemez-terület felhasználásával megbirkózzon, ami igencsak lelassítja a munkát.

A WordStar 2000 két kiegészítő programot használ, a PC Outline-t és a ShowTextet. Az előbbi kivonatot készít a megadott szövegből, de jobban lehetne használni, ha a szövegszerkesztő része lenne, nem pedig hozzácsatolt program. A ShowTextnek semmi köze sincs a WordStar 2000 irataihoz. Arra való, hogy nagy, jó minőségű karaktereket készítsen különböző nyomtatókon. A programmal leginkább írásvetítő fóliákra célszerű nyomtatni.

MultiMate Advantage II

Az Ashton-Tate szövegszerkesztője nagyon jó, különösen „bolondbiztos": a kitörölt szövegeket jóval a tévedés után is vissza lehet szerezni.

A program legnagyobb hibája, hogy laporientált, ami többek között azt jelenti, hogy nem lehet a képernyőn egyszerre két lapot nézni. Egymás melletti oszlopokat viszont lehet, és a program számos nyomtatót is támogat. A lapok tervezése nem megfelelő, ezért az Ashton-Tate kibocsátotta a Byline-t, amely valódi laptervező program, és jól együttműködik a MultiMate Advantage II-vel.

Hogy melyik program a legjobb, nehéz eldönteni. A DisplayWrite 4-gyel lehet átvenni az iratokat a nagy IBM gépekről. A Word 4 gyors, kvázi-WYSIWYG megjelenítése és a WordPerfect 5 sok hasznos funkciója segít leginkább megközelíteni az irodai kiadványszerkesztést.

(A Floppy lap nyomán)

Néhány teszt eredménye

Teszt	*DW4*	*WS 2000+*	*WS Word4*	*WP4.2*	*MMAII*
Fájlméret (bájt)	*64 512*	*54 706*	*50 176*	*50 000*	*53 760*
Betöltés (s)	*5,00*	*1,50*	*1,00*	*1,00*	*2,00*
Keresés és helyettesítés (s)	*18,00*	*5,70*	*6,00*	*6,00*	*9,00*
Szövegtörlés (s)	*3,00*	*1,00*	*0,10*	*1,00*	*3,00*
Iratnyomtatás (s)	*63,00*	*67,00*	*35,00*	*2,00*	*155,00*

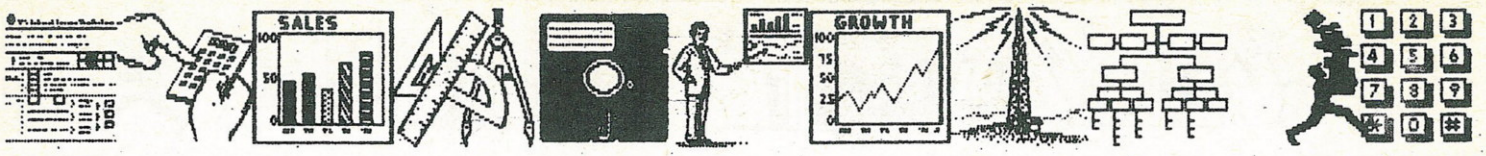
A szövegszerkesztők legfontosabb jellemzői

A számítógépeket használóknak a legjobb szövegszerkesztő programok mindazt biztosítják, ami a formás levelek, feljegyzések, dokumentációk készítéséhez szükséges. Mivel egyre újabb és jobb szöveg- és kiadványszerkesztők kaphatók, a vevőnek érdemes összeállítania azt a „tulajdonságlistát", amelyet az ideális programnak ki kell elégítenie.

A következőkben összefoglaljuk, hogy milyen fontosabb tulajdonságokat kell figyelembe venni egy szövegszerkesztő kiválasztásánál, és a jobb tájékozódás kedvéért zárójelben az eredeti angol kifejezést is közöljük.

Jellemzők	**(Features)**
Helyesírás-ellenőrzés	**(Spelling checker)**
Automatikus elválasztás	**(Hyphenate)**
Szinonimaszótár	**(Thesaurus)**
Indexhasználat	**(Indexing)**
Tartalomjegyzék	**(Table of contents)**
Lábjegyzetek	**(Footnotes)**
Fejezetvégi jegyzetek	**(Endnotes)**
Fejléc	**(Headers)**
Alsó oldalfelirat	**(Footers)**
Más szöveg beillesztése	**(Documentation merge)**
Stílusforma	**(Style sheets)**
Többszörös formátumjelölő	**(Multiple rulers)**
Szószámlálás	**(Word count)**
Nem nyomtatandó megjegyzések	**(Non-printing comments)**
Grafika	**(Graphics)**
Körlevélírás	**(Mailmerge)**
Automatikus fájlmentés	**(Autosave)**
Az eredeti fájl automatikus megőrzése	**(Auto backup)**
Makróhasználat	**(Macros)**
Számítási lehetőségek	**(Math)**
Feltételes lapváltás	**(Conditional page breaks)**
Hasábok használata	**(Columns)**
Oldal megnézése	**(Page preview)**
Több ablak, szöveg kezelése egyszerre	**(Multiple windows)**
Hány nyomtatómeghajtóval rendelkezik	**(Number of supported drivers)**
Postscript-támogatás	**(Postscript support)**
Felhasználó által definiált karakterek	**(Soft font support)**
Felhasználó által írt meghajtók	**(User defined printer drivers)**
Minimális memória	**(Minimal memory)**
Minimális lemez-, illetve meghajtóigény	**(Minimal disks)**
Egérhasználat	**(Mouse support)**
Kapcsolat más szövegszerkesztőkkel	**(Document import/export)**

Az adott szövegszerkesztő vizsgálatánál célszerű ellenőrizni, hogy a fentiekben felsorolt tulajdonságokkal rendelkezik-e a program. Annál előnyösebb, minél több az igen válasz. A végleges választásnál természetesen egyéb, a program kezelhetőségére, felhasználóbarátságára jellemző tényezőket és szubjektív szempontokat is figyelembe kell venni.



AJÁNDÉK

DOSEDIT: a DOS-parancsok kiadását segítő program

Bár az IBM DOS rendszerét alapszinten közvetlen parancsok kiadásával is lehet használni, a fájlokkal kapcsolatos műveleteket jól használható DOS-héjak segítik. Ilyen a Norton Commander, a PcTools és a Pathminder. Ezek a programok tekintélyes méretűek, de ehhez viszonyul a tudásuk is: összetett és bonyolult fájl- és egyéb műveleteket végezhetünk velük. Mivel a DOS parancsszinten nagyon gyengén támogatja a begépelt parancsok szerkeszthetőségét, ezért születtek meg az olyan segédprogramok, amelyek ezt lehetővé teszik. Három ismertebb program ezek közül: a Norton DOS Edit, a CED és a DOSEDIT. Ezek közül most részletesebben a Jack Gersbach által írt DOSEDIT szabad szoftvert mutatjuk be.

A program hívása: DOSEDIT fájlnév. Ez a fájlnév tartalmazza a parancsokhoz tartozó rövidítések összerendelési listáját. A DOSEDIT-ben használt szerkesztőbillentyűk eltérnek a szabványos DOS-ban használtaktól, és az átlagos felhasználó számára sokkal természetesebbek. A funkcióbillentyűket a program nem használja. *Táblázatunk* a szerkesztőbillentyűk használatának leírását tartalmazza.

A beszúrásos üzemmód automatikusan kikapcsolódik az ENTER és az ESC billentyűk megnyomásával.

Két 256 bájtos körkörös verem tárolja az új vagy a szerkesztett parancsokat. Az egyik verem a DOS parancsmódban aktív, a másikat a külső parancsok vagy alkalmazói programok használják.

Egy parancs végrehajtásakor a veremmutató az aktuális parancs és a következő közé mutat. A előző parancs visszahívható a felfelé nyíl billentyű megnyomásával. Ha az aktuális parancs új, akkor ez bekerül a verembe, és a veremmutató értéke az utolsó és az éppen beírt parancs közé mutat. A legelsőnek bevitt parancs elvész, ha a verem megtelik. Az aktuálisan kijelzett parancs az ENTER billentyűvel aktivizálható.

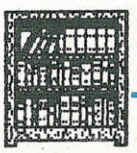
Nagyon hasznos tulajdonsága a DOSEDIT-nek, hogy a parancsokhoz egy rövidítést rendelhetünk, és a parancs aktivizálásához csupán ezt a rövidítést kell begépelni (például dir a: helyett da). A hozzárendeléseket egy különálló fájlban kell elhelyezni, amit a DOSEDIT akkor fog beolvasni és használni, amikor elindítjuk. A fájl felépítése:

RÖVIDÍTÉS HELYETTESÍTETT KIFEJEZÉS

Jobbra nyil	A kurzort egy pozicióval előreviszi
Balra nyil	A kurzort egy pozicióval visszalépteti
Ctrl-Jobbra	A kurzort a következő szóra állitja
Ctrl-Balra	A kurzort az előző szóra állitja
Balshift-tab	A kurzort az előző tabulátorpozicióra állitja
Jobbshift-tab	A kurzort a következő tabulátorpozicióra állitja
Home	A kurzort a sor elejére állitja
End	A kurzort a sor végére állitja
Del	A kurzorpozición lévő karaktert törli
Back Space	A kurzortól balra lévő karaktert törli
Esc	A teljes sor törlése
Ctrl-Home	Törlés a kurzortól a sor elejéig
Ctrl-End	Törlés a kurzortól a sor végéig
Felfelé nyil	A veremből az előző utasítás előhivása
Lefelé nyil	A veremből a következő utasítás előhivása
Ctrl-PgUp	Az aktuális verem teljes tartalmának törlése
Ctrl-PgDn	Az aktuális, kijelzett veremtartalom törlése
Ins	Beszúrás ki/be kapcsoló. Ha aktív, a kurzor más méretű.
Ctrl-A	Kijelzi a parancs-hozzárendelések listáját.
Ctrl-Z	Fájl vége jel. Az F6 billentyűt helyettesiti.
F1-F10	Nem használt

A két betűcsoport között minimum egy betűköznek kell lennie. Minden sort ENTER-rel kell zárni. A fájlt a fájlvége jellel kell befejezni. Ha az elkészült fájlt a DOSEDIT nem tudja értelmezni, hibaüzenetet küld és nem használja. A rövidítések maximális hossza 8 karakter, és a fájlnevekben használatos bármely karakter használható. A helyettesített kifejezés hossza elvileg tetszőleges lehet, de a parancs szövegtárolójában túlcsordulás léphet fel a behelyettesítéskor. Ha ez történik, egy üzenet jelenik meg, és üres sztringgel tér vissza. A hozzárendelések csak a DOS parancsmódjában aktívak. Példának egy *mintafájlt* közlünk:

```
d    dir
da   dir a:
dc   dir c:
dw   dir/w
fa   format a:
w    cd c:\w4
o    cd c:\orcad
3    cd c:\orc3
p    pcb
s    cd c:\sm
e    edit
```



LEXIKON

Szótár a merevlemez-illesztőkhöz

Adatszeparátor-áramkör. Átalakítja és létrehozza a lemezmeghajtó író-olvasó fejéről jövő impulzusból az adat- és órajeleket.

ARLL Advanced-Run-Length-Limited kódolás. Az RLL kódolás továbbfejlesztett változata, amely további sebesség- és adattárolási kapacitásnövekedést tesz lehetővé.

CAM Common Access Method. Fejlesztés alatt álló szabvány, ami lehetővé teszi, hogy a különböző számítógépeken a programozók ugyanazon forráskód segítségével vezéreljék az SCSCI berendezéseket.

ERLL Enhanced-Run-Length-Limited encoding. Lásd ARLL.

ESDI Enhanced-Small Device Interface. Ez az illesztés csak lemezmeghajtóknál használatos. Az ST506 illesztésnek a továbbfejlesztése, amelynél az adatszeparálás a meghajtón lévő kártyán történik, és lehetővé teszi, hogy a vezérlő meghajtónak bináris alakú parancsokat küldjön egy párhuzamos buszon keresztül.

FM Frequency Modulation. A legegyszerűbb, de a legkevésbé hatékony adattárolási eljárás a lemezen, merevlemezeknél ténylegesen soha nem használták.

IPI Intelligent Peripheral Interface. Nagyszámítógépeknél alkalmazott szabványos illesztés, amely nagy kábelhosszakat, elosztott vezérlést és nagy adatátviteli sebességet tesz lehetővé.

MFM Modified Frequency Modulation. Ez a kódolási technika — más néven dupla sűrűségű tárolás — kétszer annyi adat tárolását teszi a sávokon lehetővé, mint az FM.

Pufferelt keresés. A lemezmeghajtóknál az író-olvasó fejeket léptető impulzus gyorsabban érkezik a meghajtóba, mint ahogy a fej képes mozogni. A beérkezett impulzusokat a meghajtó tárolja, és ezután a fej olyan gyorsan mozog a kívánt pozícióba, ahogy ez lehetséges.

RLL Run-Length-Limited encoding. Az MFM technika továbbfejlesztése. Az RLL speciális módszert alkalmaz a lemezeken lévő adatok visszaállítására, amivel még nagyobb felírási sűrűség érhető el. A legtöbb rendszer 2,7 RLL, néhány 1,7 RLL kódolást használ.

SCSI Small Computer System Interface. Ezt a párhuzamos buszt alkalmazó szabványt kisszámítógépek lemezeinek, mágnesszalagegységeinek és egyéb perifériáinak illesztésére tervezték. A perifériák oldaláról intelligenciát tételez fel.

SMD Storage Module Device interface. Régi, nagyszámítógépes szabvány, lassan feledésbe merül nagy költsége és a gyors illesztőegységek megjelenése miatt.

ST506. A merevlemez-illesztők szabványa, amit a Seagate vezetett be az ST506 típusjelű, 5,25 inches meghajtójánál. Az illesztés ipari szabvánnyá vált.

BÖRZE

TUTTI

ELECTROCOOP KISSZÖVETKEZET

— Eredeti számítógépek
— Kompatibilis XT/AT, 386 konfigurációk
— Szünetmentes tápegységek
— Egyedi megrendelésfelvétel

Cím: Bp., Üllői út 81. 1091
Tel.: 334-354
Telex: 22-7230
Telefax: 149-869

ES EcoSoft

ENET

Lokális hálózat
IBM kompatibilis gépekre
Külön hardver
nem szükséges
Ára gépenként:
19 900 Ft + áfa
ECOSOFT KISSZÖVETKEZET
Tel.: 863-677

COBRA COMPUTER

ELEKTRONIKAI ÉS SZOLGÁLTATÓ KISSZÖVETKEZET
1097 Budapest IX., Illatos út 7. Telefon: 476-169/388

SZÁMLAKÉSZÍTÉSTŐL A KÖNYVELÉSIG

COBRACONTO

ügyviteli programrendszer moduljai:
— számlakészítő,
— számlanyilvántartó,
— bér- és jövedelemszámfejtő,
— főkönyvi könyvelő,
— anyagnyilvántartó,
— általános könyvelési programok.

KÉRJEN RÉSZLETES TÁJÉKOZTATÓT!

ARECO KFT.

a Mikropo KSZ fejlesztési témáinak jogutódja, felajánlja szabad fejlesztőmérnöki kapacitását. ARECO Informatika KFT.
Tel.: 427-453

ASY ELEKTRONIKA

A legújabb ajánlatunkból:
— ASY—16 SZUPERMIKRO számítógépek (12 terminál, UNIX operációs rendszer)
— IBM PC/XT-AT-kompatibilis számítógépek
— megrendelő által definiált betűkészlettel rendelkező billentyűzetek
— elektronikai elemek (integrált áramkörök, ellenállások, tranzisztorok)
— monitordobozok, műszerházak
— szoftvertermékek és fejlesztések

KERESKEDELMI IRODA:
1061 BUDAPEST
LISZT FERENC TÉR 10.
TEL.: 415-166, TELEX: 22-4378

NÁLUNK MÉG KAPHATÓ

Első könyvem a mikróról 30,— Ft
Első könyvem a programról 30,— Ft:

ISKOLÁKNAK OKTATÓKNAK SZAKKÖRÖKNEK

ZÜV

Kereskedelmi Iroda
1145 Budapest,
Szugló u. 9—15.
Telefon: 642-000/176-os,
177-es mellék

procontrol

pc®

IRÁNYÍTÁSTECHNIKA KISSZÖVETKEZET

MÁRKABOLT, BEMUTATÓTEREM
62/21-165, 28 985
Szeged, Kazinczy u. 8. Tel.: 12-259
Telex: 82-726

386 AT 25 MHz toronygép	299 000 Ft-tól
286 AT 16 MHz TURBO AT gép — 86 XT 12 MHz	98 000 Ft-tól
TURBO XT gép	69 000 Ft-tól
LAPTOP hordozható LCD XT gép + modem + akku	149 000 Ft
12" zöld monochrom monitor TUV M31	4 900 Ft
5 1/4" DS DD Floppy lemez	79 Ft

és még 2000 árucikkről árjegyzék, azonnali kiszolgálás.

BROTHER AX 15 típusú,
margarétafejes,
elektronikus írógép

Megvásárolható:
Budapest VI.,
Népköztársaság útja 2.
Tel.: 531-231

MIGERT

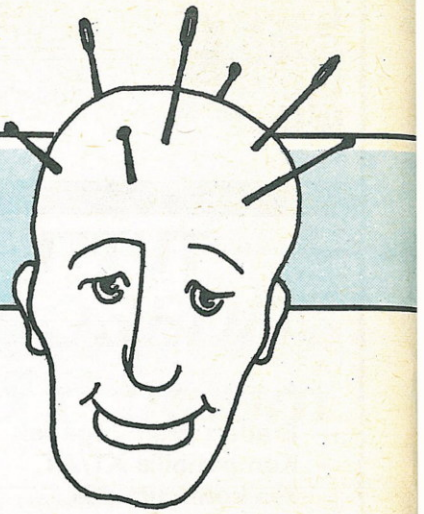
ECONORG SZÁMÍTÁSTECHNIKAI KÖZÖS VÁLLALAT

1146 Bp.,
AJTÓSI DÜRER SOR 10.
Levélcím:
1393 Pf.: 319.
Telefon: 421-974
Telex: 22-6544

SZAKÜZLETEK:

1. **Bp. VI., Szinyei Merse u. 1.**
   Tel.:127-628. Tx.: 22-6684
2. **9022 Győr, Lukács S. u. 18.**
   Tel.: 96-14880. Tx.: 02-4679
3. **3100 Salgótarján, Ady Endre u. 1.**
   Tel.: 32-10971. Tx.: 22-9380

Nyomtatók apróban



Kezdetben vala a derék öreg karos írógép, amely megbízhatóan kopogta ki az első számítógépeknél az eredményeket. Az elektronikus nyomtatás első továbbfejlesztései is az írógép elvét követték, lásd a sornyomtatókat. Mára eljutottunk a tintasugaras és a lézernyomtatókig, hogy csak a legjobbakat említsem. A legelterjedtebbek azonban kétségkívül a mátrixnyomtatók lettek, s talán maradnak is még egy darabig. Népszerűségüket a viszonylag egyszerű — tehát olcsón gyártható — mechanika mellett a képernyővel rokon rajzolási módnak, azaz a hasonló programozástechnikának és az egyszerű képernyő—papír leképezésnek (hardcopynak) köszönhették.

Mint a mesebeli lovagok a legyőzött sárkányokat fejeik száma után, a mátrixnyomtatókat a fejben levő tűk száma alapján értékelhetjük. A végletek: hallottam egy szomszédos országban gyártott 1(!) tűs nyomtatóról, és olvastam tesztet egy 48 tűsről is. Míg ez utóbbi minőségi mutatói (és ára) a lézernyomtatókhoz hasonlóak, az előbbi sebességét alighanem lap/nap dimenzióban adják meg. A legelterjedtebbek azonban a 7, 9 és 24 tűsek. Ez egyúttal egyféle minőségi rangsor is lehetne: a 7 tűs képtelen a lelógó szárú kisbetűket tisztességesen kinyomtatni, a 9 tűs már alá is húzza a szöveget, a 24 tűs pedig igen jó tempóban ír, írógépminőségben. De hogy mindebből milyen gonosz dolgok is fakadnak, arról később.

„Fogadd kódolatom!"

Mivel az egyszerű mechanikához okos szoftver dukál, és a központi egységet sem illik nagyon feltartani azzal, hogy mindent neki kell csinálnia, a nyomtatók maguk is számítógépek — mikroprocesszorral, RAM-mal, ROM-mal. (Elnézést kérek Sir Clive Sinclairtől, az ő filozófiája ennek szöges ellentéte — volt.) A „központi egységnek", mondjuk egy C64-nek nem kell tehát folyamatosan magyarázznia: most ez és ez a tű nyomtasson, most mozgasd a fejet, most a papírt — elég, ha annyit mond: „A" BETŰ — és a papíron már ott is van, a nyomtató legfeljebb annyit jelez: kész vagyok. Ez nagyon szépen hangzik, van azonban egy buktatója: vajon érti-e a nyomtató, hogy mit mondott a C64? És ha úgy is hinné, hogy érti, jó-e az nekünk is?

A kódkészlet véges és jól behatárolt: a nyolcbites gépeken nullától 255-ig terjed. De melyik kód jelenti az „A" betűt? Ráadásul a C64 előtt ott van még a láncban a szövegszerkesztő program, a nyomtató és a gép közé egy interfész kerülhet, és máris bábeli a nyelvzavar.

Vannak ugyan szabványok, olyanok, mint az ASCII (American Standard Code for Information Interchanges), amely inkább ajánlás, tehát még Amerikában sem kötelező, lásd például a Commodore esetét. Továbbmegyek: az még csak hagyján, hogy megértetjük az egyes nyomtatandó jeleket, de honnan tudja a nyomtató, hogy mettől meddig szedjen vastagon, dőlt betűkkel vagy húzzon alá? Erről az ASCII nem rendelkezik. Léteznek tehát külön nyomtatószabványok. Az egyik legelterjedtebb az Epsonnak, a világ legnagyobb nyomtatógyártó cégének ajánlása, az ESC/P (Epson Standard Code for Printers), ez szerencsére az ASCII-n alapul, annak kiegészítéseként. Néhány kivétellel az összes nyomtatóparancs egy ún. váltókódra [CHR$(27), ESCAPE] épül, például az Elite betűtípus az ESC/M [CHR$(27)+CHR$(77)] kódsorozattal választható ki. Ettől eltérő kódokat használ a NEC P6 széria, megint másokat az IBM ProPrinter.

Ez lenne a dolognak a szoftveroldala. De hogyan is állunk a madzagokkal meg a dugókkal? Itt megint a szabványok egész sora áll előttünk: a PC-kategóriában elfogadott a Centronics, az RS232 vagy a Commodore által favorizált IEEE—488. A szabványok meghatározzák a készülékek összekötéséhez szükséges csatlakozókat, azok bekötését, a jelszinteket. Ezeket aztán a gyártó cégek vagy betartják, vagy nem. Az Enterprise 128-nak csak a csatlakozója nem szabványos Centronics, a C64-esé úgyszintén nem és az operációs rendszere sem támogatja, de ezen a gondon még lehet egy segédprogram betöltésével enyhíteni. Támogatja viszont az RS232-t, csak éppen a jelszintjei és a csatlakozói nem szabványosak. Jó lesz tehát a forrasztópákát nem elpakolni, vagy kénytelenek leszünk csatlakozókért és interfészekért a kisiparosokat felkeresni.

Olcsó húsnak...

De hogy is néz ki ez a gyakorlatban? Tegyük fel ismét, hogy valakinek van egy C64-ese, és nyomtatót szeretne hozzá, vagy már van neki egy, de nem elégedett a Commodore által kínált 7 tűs MPS—801-essel, hiszen lassú, nyomtatási képe és szolgáltatásai szegényesek. Mit tehet? Vesz egy idegen márkájút. Ha szerencséje van, talál olyan, amelynél a csatlakozások és a kódok is illeszkednek. Ilyen azért kevés akad: kevés cég butítja erre a szintre egy-egy modelljét. Ajánlani merem a Seikosha VC jelű (nomen non est omen) típusait, vagy a kicsit drágább Epson LX—800-at. Ha ilyet nem talál a vásárló, akkor vesz egy igényeinek és pénztárcájának megfelelő típust. Most már tulajdonképpen mindegy, milyet. Vesz továbbá egy interfészt, amely jó esetben a kódkonverziót is elvégzi, és vesz egy új szövegszerkesztő programot, ami ezekkel együtt is működik. Egyet mondtam, három lett belőle.

Még mindig nem beszéltem egy igen lényeges dologról. Ha az illető számítógép-vásárló úr mondjuk magyar, és magyar nyelvű szövegeket szeretne írni, további nehézségeknek néz elébe. Jobb nyomtatók tízféle nemzeti karakterkészletet ismernek (ez ASCII-ajánlás), de fájdalom, a magyar nincs a tíz között. Néhány szövegszerkesztő program a szoftver útján pótolja a hiányt: a Deltex például az ékezetes betűket grafikus üzemmódban (lásd később), pontonként nyomtatja ki. Ez az eljárás csak addig volt tartható,

1. ábra

```
Figyeld az ékezetest!
Figyeld az ékezetest!

   eéoöüű   eéoöüű

(Ez a DELTEX módszere)


Levél minőségű
Levél minőségű

Ez a TEXTER-é

aáeéiíoóöőuúüű
AAEÉIIOÓÖŐUÚÜŰ
```

amíg itthon is elérhetővé nem váltak az ún. (egészen vagy csaknem) levélminőségű (LQ, NLQ: Near Letter Quality) nyomtatók. A levélminőségű nyomtatás csak a karakteres módban él: nem működik a grafikusban. Az eredmény: „levélminőségű" levelünkben világítanak az ékezetes betűk *(1. ábra)*.

Rokonszenvesebb a Texter által alkalmazott megoldás: első menetben balról jobbra kiírja az ékezeteket, a másodikban visszafelé a betűket — így kevésbé lóg ki a lóláb. Teljes értékű megoldás a nyomtató karakterkészletének cseréje; ez az olcsóbb modellekben a ROM cseréjét jelenti, és azt, hogy a gépünk már a saját típusával sem lesz többé kompatibilis. Ez különben már a negyedik kiadás, pedig csak egy árva nyomtatót szerettünk volna. Drágább típusokon a teljes karakterkészletet átmásolhatjuk RAM-ba, és ott természetesen módosíthatjuk (DOWNLOAD opció). Az egészen kiválóakon ez az ügy a karakter-ROM kártyácskák cseréjére-dugdosására korlátozódik (Epson, NEC, Star NB 24-10, Citizen HQP—40). Csakhogy magyar ékezetes kártyát még nem láttam. Ott vagyunk tehát, ahol elkezdtük.

Vissza a gyökerekhez

Nyomtatónk lehetőségei nem merülnek ki száraz szövegek és táblázatok nyomtatásában. A betűvetésen kívül imponáló képessége, hogy tetszőleges, számítógépen készített műalkotásunkat is papírra viszi. Grafikus nyomtatást használnak rajzolóprogramjaink, a CAD és DTP programok. Az elv mindenféle mátrixnyomtatón közös: nem kész betűket kérünk tőlük, hanem folyamatosan megmondjuk, hogy az egymás alatt lévő tűk közül most éppen melyik „üssön" — majd a fej egyetlen képpontnyi elmozdítása után újra megadjuk az éppen soron lévő oszlop rajzát. Hogy ez az oszlop milyen magas, azaz hány képpontot tudunk egyszerre nyomtatni, az éppen attól függ, hogy hány tű van a fejben. Nyolcig nincs is semmi baj, mert annyit ki tud egyszerre mondani a nyolcbites számítógép. Ettől fölfelé viszont több bájt szükséges ahhoz, hogy egy-egy oszlopot meghatározzunk. Háromféle grafikus szabvány terjedt el: a 7 tűs, a 8 tűs és a 24 tűs.

Ez utóbbi az itthoni nyomtatóparkot ismerve nagy érdeklődésre sajnos nem tarthat számot, nézzük tehát az előző kettőt! A 7 tűs grafika az olcsóbb, az egyszerű nyomtatók sajátja. Nyomtatóparancsra sem sokra van szükség hozzá. Az első közli, hogy a következő kódokat grafikus oszlopkódokként kell értelmezni, egyúttal a soremelést is olyan értékre állítja, hogy a legalsó pont mellé az alatta levő oszlop legfelső pontja éppen odaérjen. Így lehet függőlegesen nagyobb kiterjedésű rajzokat készíteni. A grafikus kódok számítása is egyszerű: azon tűk kettes számrendszerbeli helyi értékét összeadjuk, amelyeknek ütniük kell *(2. ábra)*. Hogy ezt a nyomtató valóban grafikus bájtként értelmezze, hozzá kell még adni ismertetőjegyként 128-at. Így a kódok értéke 128 és 255 közé adódik. A 32 és 127 közé eső kódokat a nyomtató figyelmen kívül hagyja, a maradékot pedig — szokás szerint — vezérlőkódként értelmezi, így például a grafikus nyomtatást feloldó kódot. Könnyű belátni, hogy így mondjuk az „A" betű kirajzolásához öt-nyolcszor annyi adat kell. Hamar rövid lesz hát az olcsó gép szűk emlékezete, azaz tele lesz a puffer-RAM. Ezt elkerülendő vezettek be még egy kódot, a grafikus ismétlés kódját. Az azonos bitmintákat összefogva küldhetjük el a következő sorozattal: 1. a grafikus ismétlés kódja, 2. az ismétlések száma, 3. az ismétlődő kód. Három bájt összefogásával nyomtatási időt takarítunk meg, néggyel pedig már értékes puffer-RAM-ot.

A 9 tűs, igényesebb nyomtatók vízszintesen többféle sűrűségben is képesek rajzolni. Így a sűrűbben elhelyezett pontok egymáshoz, sőt egymásra érhetnek *(3. ábra)*. Ehhez hasonlóan a papírt is nagyon finom lépésekben (1/3 pont!) tudjuk továbbítani, és olyan vonalakat vagyunk képesek húzni, amelyeket elemi pontokból állítottunk ugyan össze, de azokat már nem tudjuk megkülönböztetni. Ezt használjuk ki az NLQ üzemben is.

Mivel egyszerre nyolc képpont adatát szeretnénk megadni, a 8 tűs grafika programozása az elviekben is különbözik. Az egyik lehetséges szabvány itt is az ESC/P. (A legtöbb gyár ezt vette át az Epson után.) Itt mind a nyolc bitre szükségünk van az oszlopkódhoz, ezért a parancsok is ennek megfelelően alakulnak: 1. ESCAPE kód, 2. a felbontás kódja, 3. a grafikus bájtok száma alacsony bájt—magas bájt sorrendben. Az ezt követő, megadott számú bájt kinyomtatása után következő kódok ismét karakterként nyomtatódnak — hacsak nem következik ismét parancsbájt.

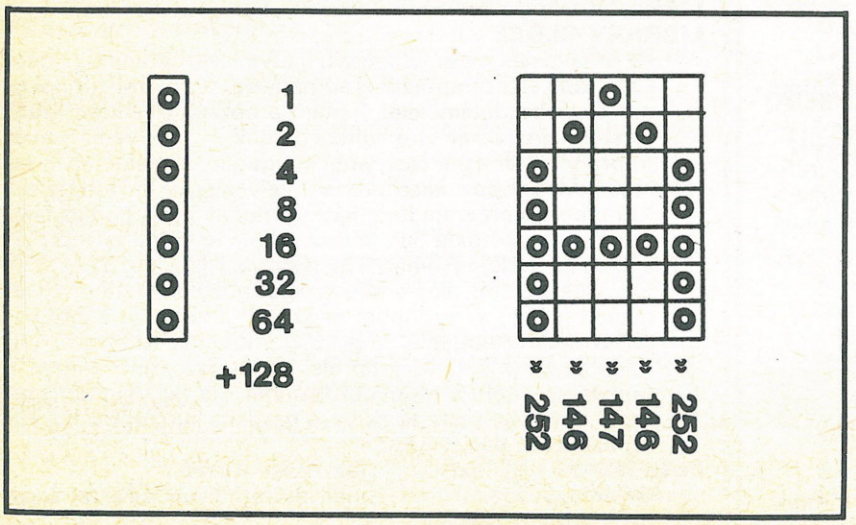
Igen ritkán írunk azonban programot direkt a nyomtatóra, sokkal gyakoribb feladat a számítógép képernyőjén vagy tárában már meglévő képnek a papírra másolása. A folytatásban ennek a problémáiról írok.

Egy renitens printer

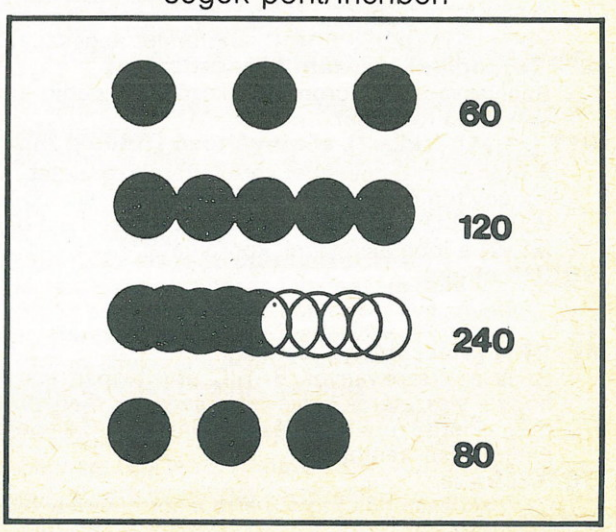
Elég sok példánya lelhető fel országszerte a Commodore MPS—802 típusú nyomtatónak (leánykori nevén VC—1526). E bájos teremtés cáfolni igyekszik az előző tételeket: 8 tűvel nyomtat, és semmiféle grafikára nem lehet rábírni. („Majd fogtok ti engem visszabutítani" – vallja.) Beható vizsgálat kiderítette azonban, hogy ez a típus is egy 9 tűs mechanikára épül. Miért hagyták hát ki azt a két tranzisztort, amivel az utolsó tű is munkára bírható, és miért maradt ki a nyomtató firmware-jéből a grafika? Ezt csak a Commodore programozói tudják. Az NSZK-ban kifejlesztettek egy programot, melyet EPROM-ba égetve, s azt a nyomtató ROM-jával kicserélve, grafikára képes, igazi 9 tűs nyomtatót kapunk, megszabadulva néhány további apró hibától. A programot és a kapcsolást bárkinek szívesen elküldöm.

ZOLTAI PÉTER

2. ábra. A grafikus oszlopkódok számítása



3. ábra. Különböző vízszintes pontsűrűségek pont/inchben



Az AmigaBasic utasításkészlete

END SUB
Az alprogramok lezárására szolgál.

y=EOF(fájlszám)
Logikai függvény, amely akkor ad „igaz" (−1) választ, ha a megadott fájl pointere az utolsó karakter utáni helyre mutat.

ERASE tömbváltozó1[,tömbváltozó2]...
Törli a tömbváltozókat, ezáltal memóriát szabadít fel.

y=ERL
Az utoljára bekövetkezett hiba sorszámát tároló rendszerváltozó.

y=ERR
Az utoljára bekövetkezett hiba kódját tároló rendszerváltozó.

ERROR hibakód
Kiírja a hibakódnak megfelelő hibaüzenetet. A hibakódok a 0..255 tartományba esnek.

EXIT SUB
Kilép az éppen futó alprogramból.

y=EXP(x)
Az argumentum exponenciális függvénye.

FIELD [#]logikai fájlszám, hossz1 AS sztringváltozó1 [,hossz2 AS sztringváltozó2]...
I/O buffereket hoz létre a megadott számú fájl részére, és ezeket a buffereket hozzárendeli a felsorolt sztringváltozókhoz. Ezután a változókat írva és olvasva írható és olvasható a fájl, a PUT# és a GET# utasítások felhasználásával.

FILES [útvonal]
Tartalomjegyzéket ír ki a megadott útvonalról, útvonalmegadás hiányában pedig az aktuális könyvtárról.

y=FIX(x)
Az argumentum egész részét adja vissza úgy, hogy egyszerűen elhagyja a tizedesponttól jobbra eső számrészt (éppen ezért nem azonos az INT(x)-szel).

FOR változó=x TO y [STEP z]
•
•
•
NEXT [változó1][,változó2]...
A jól ismert ciklusszervező utasításpár. A NEXT használatát egyszerűsíti, hogy nem kötelező kitenni a változónevet, ilyenkor a legközelebbi FOR lesz a ciklus feje. További könnyítés, hogy egy NEXT utasítással több ciklust is lezárhatunk.

y=FRE(x)
Kiírja a szabad memóriaterület nagyságát bájtokban.
x= −1: szabad rendszerterület nagysága
x= −2: szabad veremterület nagysága (GOSUB—RETURN, FOR—NEXT)
x>= −1: szabad BASIC adatterület nagysága

GET [#] logikai fájlszám [,rekordszám]
Beolvassa a fájl soron következő vagy pedig a megadott sorszámú rekordját.

GET (x1,y1)—(x2,y2), tömbváltozó [(index[,index]...)]
A megadott téglalappal körülhatárolt részletet mint grafikát egy tömbváltozóba menti el. A tömbváltozó tartalma a PUT utasítással jeleníthető meg újra. (x1,y1) a bal felső és (x2,y2) a jobb alsó sarok koordinátái.

GOSUB címke
•
•
•
RETURN [visszatérési címke]
Ez is egy szokványos BASIC utasításpár, viszont azzal, hogy a visszatérés helye tetszőlegesen megadható, programunk strukturáltsága és áttekinthetősége egészen minimálisra csökkenthető!

GOTO címke
Vezérlésátadás (ugrás) arra a sorra, amely előtt a megadott címke áll.

y$=HEX$(x)
A decimális argumentumot (max. 32 bites) hexadecimálissá alakítja.

IF kifejezés THEN/GOTO utasítások/címke ELSE utasítások/címke
IF kifejezés THEN utasítások
ELSEIF kifejezés THEN utasítások
ELSE utasítások
ENDIF
Az első szintaxis a hagyományos forma. A második alkalmazása viszont áttekinthetőbbé teszi programunkat, ezért bonyolultabb feltételvizsgálatok esetén javasolt a használata.

y$=INKEY
Az éppen lenyomott billentyűhöz rendelt karaktert megadó függvény.

INPUT [;]["szöveg";] változó1[,változó2]...
Kiírja a szöveget, majd a billentyűzetről beadott értékekkel tölti fel a felsorolt változókat.

y$=INPUT$ (n[,[#]logikai fájlszám])
Beolvas egy n karakter hosszúságú szakaszt a kijelölt fájlból.

INPUT #logikai fájlszám,változó1[,változó2]...
Feltölti a felsorolt változókat a megadott fájlból.

y=INSTR([n],x$,y$)
A függvényérték egy szám, amely x$-nek arra a karakterére mutat, hol az y$ először megtalálható benne. N helyén megadhatjuk a keresés kezdetét. Ha a keresés sikertelen, a függvény értéke 0.

y=INT(x)
Egészrész-függvény, az x-hez legközelebbi, nála kisebb, vagy vele egyenlő egész számot adja.

KILL fájlnév
Törli a megadott nevű fájlt a lemezről.

LBOUND(tömbváltozó[,dimenzió])
UBOUND(tömbváltozó[,dimenzió])
A tömbváltozó legkisebb (LBOUND) és legnagyobb (UBOUND) indexét adja meg a kiválasztott dimenzióban. Ha nem adjuk meg a dimenziót, az alapérték 1.

y$=LEFT$(x$,n)
A sztring első n karakterét megadó függvény.

y=LEN(x$)
A sztring hosszát adja meg.

[LET] változó=kifejezés
Értékadás. (A kulcsszó elhagyható.)

LIBRARY könyvtárnév
LIBRARY CLOSE
Az utasítással megnyithatunk, illetve lezárhatunk egy gépi kódú könyvtárat, aminek eredményeként a ROM-rutinokra nemcsak kezdőcímükkel, hanem a hozzájuk rendelt szabványos címkékkel is hivatkozhatunk. A könyvtár nevét ".library" kiterjesztéssel lehet megadni, és valamelyik lemezmeghajtóban elérhetőnek kell lennie a megfelelő ".bmap" kiterjesztésű fájlnak. (Például a "graphics. library" könyvtárhoz a "graphics.bmap" nevű fájl tartozik.)

LINE [[STEP (x1,y1)] — [STEP](x2,y2)[,[szín][,B[F]]]
Egyenes vonalat, illetve négyszöget rajzoló utasítás. Vonal esetén (x1,y1) a kezdőpont és (x2,y2) a végpont koordinátái. B opciót megadva négyszöget kapunk, amelynek bal felső sarka (x1,y1) és jobb alsó sarka (x2,y2). F esetén a téglalapot kitölti a megadott színnel. Ha (x1,y1) nem szerepel, akkor helyette az aktuális grafikus kurzorpozíciót veszi alapul az AmigaBasic.

LINE INPUT [;]["szöveg";]sztringváltozó
Olyan szöveges INPUT, amellyel bármilyen karakter bevihető (szóköz, vessző stb.).

LINE INPUT #logikai fájlok száma, sztringváltozó
Olyan szöveges INPUT#, amellyel bármilyen karaktert behozhatunk egy fájlból (szóköz, vessző stb.).

LIST [kezdősor][—[befejezősor]][,logikai eszköz]
Kilistázza a megadott programrészletet a kívánt kiviteli eszközön.

LLIST [kezdősor][— befejezősor]]
Nyomtatón listázza ki a megadott programrészletet vagy az egész programot.

LOAD [programnév[,R]]
Betölti a programot, és R opció esetén el is indítja.

y=LOC(logikai fájlszám)
Megadja, hogy a fájl hányadik rekordját olvastuk utoljára.

LOCATE [sor][,oszlop]
Beállítja a szöveges kurzor helyét a képernyőn.

y=LOF(logikai fájlszám)
Megadja a fájl hosszát.

y=LOG(x)
Természetes alapú logaritmusfüggvény.

y=LPOS(n)
Megadja az utoljára kiírt karakter sorszámát a sorpufferben. n egy álparaméter, értéke nem befolyásolja a függvény értékét.

LPRINT [kifejezések listája][;]
LPRINT USING u$,kifejezések listája[;]
Kinyomtatja a kifejezések értékét, USING kulcsszó esetén pedig az u$ által meghatározott formázott nyomtatást végez. (Lásd: PRINT USING)

LSET sztringváltozó=x$
Feltölti a FIELD segítségével valamely fájlhoz hozzárendelt puffert x$ tartalmával. x$-et a pufferben balra igazítva helyezi el. Ezután következhet a PUT# utasítás.

MENU oszlopszám,sorszám,állapot[,menüpont neve]
MENU RESET
Megnyit egy menüt, illetve menüsort, vagy RESET esetén törli az összes menüt. Az oszlopszám 1-től 10-ig terjedhet. (1-től 4-ig az AmigaBasic menüi találhatók.) A sorszám 0 . . . 19 lehet, a 0-ás a menüoszlop fejléce. Három lehetséges menüállapot van:
0: inaktív a menüoszlop/menüsor
1: aktív a menüoszlop/menüsor
2: kijelölt a menüsor (ki van pipálva)

y=MENU(n)
n=0 esetén a kiválasztott menüoszlop számát,
n=1 esetén pedig a kiválasztott menüsor számát adja vissza ez a függvény.

MENU ON/OFF/STOP
A menüpontok aktivizálásának figyelését, illetve az ebből eredő programelágazást (ON MENU GOSUB) engedélyezi, illetve tiltja.
ON: engedélyezi az ON MENU GOSUB működését
OFF: a rendszer nem figyeli a menütevékenységet
STOP: a rendszer tárolja a legutolsó menütevékenység adatait, de az ON MENU GOSUB-bal megadott rutinra csak egy MENU ON után adja át a vezérlést.

MERGE programfájlnév
A megadott *ASCII formátumú* programfájlt betölti és elhelyezi a memóriában levő program után.

y$=MID$(x$,n[,m])
Az x$ n-edik karakterpozíciójától számított m darab karakterét szolgáltatja ez a függvény. Inverz módon is használható: MID$(y$,n[,m])=x$. Ekkor az y$ egy részét felülírja x$-el az n-edik karakterpozíciótól, m hosszon.

y$=MKI$(16 bites numerikus kifejezés)
y$=MKL$(32 bites numerikus kifejezés)
y$=MKS$(egyszeres pontosságú numerikus kifejezés)
y$=MKD$(dupla pontosságú numerikus kifejezés)
A felsorolt függvények a különböző formátumú numerikus kifejezések értékét sztringgé alakítják.

y=MOUSE(n)
Az egér állapotának lekérdezésére szolgáló függvény.
n=0: A bal oldali billentyű lekérdezése. Ha a függvényérték negatív szám, azt jelenti, hogy pillanatnyilag le van nyomva. A kapott szám abszolút értéke (1 . . . 3) azt mondja meg, hogy a legutóbbi MOUSE(0) lekérdezés óta hányszor volt lenyomva.
n=1: Megadja az egér x koordinátáját a legutóbbi MOUSE (0) lekérdezés pillanatában.
n=2: Megadja az egér y koordinátáját a legutóbbi MOUSE(0) lekérdezés pillanatában.
n=3: Az x koordinátát adja vissza a bal gomb lenyomásának pillanatában, ha ezt az eseményt egy MOUSE(0) követte.
n=4: Az n=3 eset, csak az y koordináta az eredmény.
n=5: Ugyanazt eredményezi, mint az n=3 megadása, de ezzel a bal gomb *felengedésének* pillanatában fennálló koordinátákat is megkapjuk.
n=6: Az n=5 eset, csak az y koordináta az eredmény.

MOUSE ON/OFF/STOP
Az egér bal oldali billentyűjének figyelését és az ON MOUSE GOSUB utasítást tiltja, illetve engedélyezi.
ON: A rendszer figyeli a bal gombot, és annak lenyomásakor aktivizálódik az ON MOUSE GOSUB-bal kijelölt szubrutin.
OFF: A rendszer nem figyeli a bal gomb állapotát.
STOP: A bal gomb lenyomásának tényét tárolja a rendszer, és csak egy MOUSE ON kiadása után adja át a vezérlést az ON MOUSE GOSUB-bal deklarált rutinnak.

NAME régi fájlnév AS új fájlnév
Átnevezi a kijelölt fájlt.

NEW
Törli a memóriában levő programot és a változókat.

OBJECT.AX object-szám, gyorsulás
OBJECT.AY object-szám, gyorsulás
Object gyorsulásának beállítása a megadott irányban (az x vagy az y tengely mentén). Mértékegység: képpont/szekundumnégyzet. Tartomány: −32768 . . . +32767.

OBJECT.CLIP (x1,y1)—(x2,y2)
Meghatározza azt a téglalap alakú területet, amelyen belül az objectek mozoghatnak anélkül, hogy "kerettel történt ütközést" érzékelne a rendszer. OBJECT.CLIP kiadása nélkül az ablak szélei képezik a keretet.

OBJECT.CLOSE [object-szám1[,object-szám2 . . .]]
Megszűnteti a kijelölt vagy kijelölés nélküli esetben az összes objectet.

OBJECT.HIT object-szám[,saját maszk][,idegen maszk]
Mindkét maszk 16 bites formátumú szám. Az utasítással azt állíthatjuk be, hogy egy adott tárgynak mivel kell ütköznie ahhoz, hogy az ON COLLISION GOSUB által deklarált rutinra adódjon a vezérlés. Ha bármilyen ütközés bekövetkezik, a tárgy saját maszkja és a másik tárgy idegen maszkja között bitenkénti ÉS műveletet végez a BASIC, és ha az eredmény nem nulla, akkor aktivizálódik az ON COLLISION GOSUB. A két maszkot tehát ennek ismeretében, egyéni célunknak megfelelően kell megterveznünk.

OBJECT.ON [object-szám1[,object-szám2 . . .]]
OBJECT.OFF [object-szám1[,object-szám2 . . .]]
Láthatóvá (ON), illetve láthatatlanná (OFF) teszi az objecteket.

OBJECT.PLANES object-szám, bitsík maszk, bitsík-kapcsoló maszk
Mindkét maszk 8 bites szám. A maszkokkal az object bitsíkjait tetszőlegesen átrendezhetjük, ezáltal a rendelkezésre álló paletta minden színét hozzárendelhetjük az object minden részletéhez.

OBJECT.PRIORITY object-szám,prioritás
Beállítja az object prioritását (elsőbbségi szintjét), amelynek −32768 és +32767 közé kell esnie. A nagyobb szám nagyobb prioritást jelent, a nagyobb prioritású object átfedés esetén eltakarja a kisebb prioritásút.

OBJECT.SHAPE object-szám, sztring
OBJECT.SHAPE object-szám1,object-szám2
Az első szintaxis szerint a sztringből (mint bájtsorozatból) egy új objectet hoz létre. A második szintaxist alkalmazva az object1-et átmásolhatjuk az object2-be.

Tóth Zoltán

Grafika karakterkészletben

Ha nem túl bonyolult

Ismeretes, hogy a C64-nek többféle grafikai tulajdonsága van. A képeket vagy a nagyobb grafikákat általában grafikus üzemmódban szokták megjeleníteni.

A megjelenítés módja a karakteres megjelenítés. Ezt legegyszerűbben úgy érhetjük el, hogy például ha a kép $2000-nél kezdődik, a karakterkészlet kezdőcímét is $2000-re állítjuk. (POKE 53272,25). Ezután íratjuk ki a képernyőre a 256 karaktert. Ekkor azonban az egész képnek csak mintegy negyedrészét láthatjuk.

Ha a kép nem túl bonyolult, valószínűleg vannak olyan részei, amelyeknek ugyanaz a karakter felel meg. Minél több az ismétlődés, a képnek annál nagyobb része fér bele a karakterkészletbe. A közölt program az ismétlődéseket megkeresve — és ezeket egy táblázatba rögzítve — a képet mintegy „összetömörítve" pakolja le a karakterkészletbe. Csak azokat a képeket tudja kezelni, amelyek tömörítve elférnek 256 karakterben.

A *2. listán* a gépi kódú programrész található BASIC-betöltővel. Ennek futtatása után kell betölteni a BASIC részt (*1. lista*). A gépi kódú részben található a képet betöltő, illetve kimentő, valamint a lepakoló rutin.

A BASIC rész segítségével betölthetünk egy ART STUDIO formátumú képet.A sikeres tömörítés után beállíthatjuk a színeket, majd a karakterkészletet, táblázatot és a megjelenítő rutint kimenthetjük lemezre. Ez mint program betölthető és futtatható. (A lista 2600-3000-as sorában lévő grafikus karakterek rendre az F1, F3, F5, F7 billentyűk lenyomásával jelennek meg.)

A képek karakteres megjelenítésének több előnye is van. Jóval kevesebb a memóriafelhasználás. Egy kép 8 kbájt helyett max. 3 kbájtnyi területen elfér. Sokkal könnyebb az ábrák mozgatása is. Ha a képet scrollozni akarjuk, karakteres üzemmódban 8-ad annyi memóriaterületet kell mozgatnunk, mint grafikus üzemmódban, hiszen egy karakter 8 bájtnyi információt tartalmaz. Így elérhetjük a rasztericiklusonkénti képváltást, ami folyamatos mozgás látszatát kelti. További előny, hogy egy nem egész képernyőt betöltő ábra a képernyő bármely részén megjeleníthető, akár több helyen is.

Az ábrázolásmódnak azonban hátránya is van: kevesebb színt tudunk megjeleníteni. Multicolor grafikus üzemmódban minden karakterhelyen az alapszínen kívül az összes szín közül bármely három megjeleníthető. A karakteres módban mindenhol megegyezik az alapszín és még két szín (ezek a $D022, $D023 címek). A harmadik szín csak nyolcféle lehet, viszont karakterenként különböző.

Mérlegelve az előnyöket és a hátrányokat, úgy véljük, hogy a karakteres ábrázolásmód használata mindenképpen hasznos.

Nagy Attila—Magyar Attila

1. lista

```
10 REM *** KEP BETOLTESE ***
20 GOSUB 1000:PRINT"■"
30 PRINT:PRINT:INPUT"KEP NEVE    ";A$
40 H=LEN(A$)
45 IF H=0 OR H>16 OR A$="$" THEN 30
50 POKE 49326,H
60 FORI=1TOH
65 POKE52991+I,ASC(MID$(A$,I,1)):NEXT
70 SYS 49318
80 GOSUB 1040:A=VAL(A$)
90 IF A>2 THEN 30
100 PRINT:PRINT"(H)IRES VAGY (M)ULTI ?"
110 GOSUB 1100
115 IF A$<>"H" AND A$<>"M" THEN 110
120 V=200:IF A$="M" THEN V=V+16
140 REM *** KEP BEKAPCSOLASA ***
150 POKE49431,V:SYS 49420
160 GOSUB1100:GOSUB 1000:PRINT"■"
180 REM *** PAKOLAS ***
190 SYS 49152:IF PEEK(780)=0 THEN 220
200 PRINT"NEM FER BELE !!!!":END
220 REM *** SZINEK BEALLITASA ***
230 POKE53272,155:POKE53270,V
235 S=53280:H=10216
240 GOSUB 1100
250 IF A$=CHR$(13) THEN370
260 IF A$="■" THEN X=S:Y=H:GOTO350
270 IF A$="■" THEN X=S+1:Y=H+1:GOTO350
280 IF A$="■" THEN X=S+2:Y=H+2:GOTO350
290 IF A$="■" THEN X=S+3:Y=H+3:GOTO350
300 IF A$<>"■" THEN 240
310 A=(PEEK(55296) OR 240)-1
320 IF V=216 AND A<248 THEN A=255
330 POKEH+4,A:POKE780,A:SYS49595
340 GOTO 240
350 A=PEEK(X)-1:POKEX,A:POKEY,A:GOTO240
370 REM *** KIMENTES ***
380 GOSUB1000:PRINT"■":POKE49571,V
390 PRINT:PRINT:INPUT"FILE-NEV    ";A$
400 H=LEN(A$)
410 IF H=0 OR H>16 OR A$="$" THEN 390
420 POKE 49363,H
430 FORI=1TOH
440 POKE52991+I,ASC(MID$(A$,I,1)):NEXT
450 SYS 49344
460 GOSUB 1040:A=VAL(A$)
470 IF A>2 THEN 390
480 END
1000 REM *** INICIALIZALAS ***
1010 POKE56576,151:POKE53272,21
1015 POKE646,15:POKE53280,0:POKE53281,0
1020 POKE53265,27:POKE53270,200:RETURN
1040 REM *** DISK STATUS ***
1050 PRINT:PRINT"DISK STATUS: ";
1060 OPEN1,8,15:INPUT#1,A$,B$,C$,D$
1070 PRINTA$","B$","C$","D$:CLOSE1
1080 RETURN
1100 REM *** VARAKOZAS BILLENTYURE ***
1110 GETA$:IF A$="" THEN 1110
1120 RETURN
```

2. lista

```
1 FORI=0TO 460:READ A$
2 A1$=LEFT$(A$,1):A2$=RIGHT$(A$,1)
3 A1= ASC(A1$):A2=ASC(A2$)
4 A1=A1-48:IF A1>15 THEN A1=A1-7
5 A2=A2-48:IF A2>15 THEN A2=A2-7
6 A=A1*16+A2:POKE49152+I,A:B=B+A
7 NEXT
8 IF B<>63341 THEN PRINT"HIBAS ADAT !"
9 END
10 DATA A2,0C,A0,00,A9,24,84,FA,85,FB
11 DATA 98,91,FA,C8,D0,FB,E6,FB,CA,D0
12 DATA F6,A2,01,A0,00,A9,24,84,FA,85
13 DATA FB,A9,28,84,FC,85,FD,A9,40,84
14 DATA FE,85,FF,A9,00,85,F9,A0,00,B1
15 DATA FE,D1,FC,D0,08,C8,C0,08,D0,F5
16 DATA 4C,61,C0,E6,F9,E4,F9,F0,06,20
17 DATA 9A,C0,4C,2F,C0,20,9A,C0,A0,00
18 DATA B1,FE,91,FC,C8,C0,08,D0,F7,E8
19 DATA E0,01,D0,03,A9,01,60,20,79,C0
20 DATA A5,FB,C9,27,D0,09,A5,FA,C9,E8
21 DATA D0,03,A9,00,60,20,86,C0,4C,2B
22 DATA C0,A0,00,A5,F9,91,FA,E6,FA,D0
23 DATA 02,E6,FB,60,A5,FE,18,69,08,90
24 DATA 02,E6,FF,85,FE,A9,00,85,FC,A9
25 DATA 28,85,FD,60,A5,FC,18,69,08,90
26 DATA 02,E6,FD,85,FC,60,A2,08,A0,00
27 DATA 20,BA,FF,A9,00,A2,00,A0,CF,20
28 DATA BD,FF,A9,00,A2,00,A0,40,20,D5
29 DATA FF,60,A2,00,BD,62,C1,9D,95,23
30 DATA E8,E0,6B,D0,F5,A2,08,20,BA,FF
31 DATA A9,00,A2,00,A0,CF,20,BD,FF,A2
32 DATA 00,A0,30,86,AE,84,AF,A2,95,A0
33 DATA 23,86,AC,84,AD,A9,61,85,B9,20
34 DATA D5,F3,A5,BA,20,0C,ED,A5,B9,20
35 DATA B9,ED,A9,01,20,DD,ED,A9,08,20
36 DATA DD,ED,A0,00,20,24,F6,60,A9,96
37 DATA 8D,00,DD,A9,3B,8D,11,D0,A9,C8
38 DATA 8D,16,D0,A9,F5,8D,18,D0,A2,00
39 DATA BD,40,5F,9D,00,7C,BD,40,60,9D
40 DATA 00,7D,BD,40,61,9D,00,7E,BD,28
41 DATA 62,9D,E8,7E,BD,38,63,9D,00,D8
42 DATA BD,38,64,9D,00,D9,BD,38,65,9D
43 DATA 00,DA,BD,20,66,9D,E8,DA,E8,D0
44 DATA CD,AD,28,63,8D,20,D0,AD,29,63
45 DATA 8D,21,D0,60,0D,08,C4,07,9E,28
46 DATA 32,30,36,34,29,00,00,00,00,A2
47 DATA 0C,A9,00,A0,24,85,FA,84,FB,A9
48 DATA 6C,A0,08,85,FC,84,FD,A0,00,B1
49 DATA FC,91,FA,C8,D0,F9,E6,FB,E6,FD
50 DATA CA,D0,F2,A9,1B,8D,11,D0,A9,97
51 DATA 8D,00,DD,A9,9B,8D,18,D0,A9,C8
52 DATA 8D,16,D0,A2,04,BD,E8,27,9D,20
53 DATA D0,CA,10,F7,AD,EC,27,20,5A,08
54 DATA 4C,57,08,A2,00,9D,00,D8,9D,00
55 DATA D9,9D,00,DA,9D,E8,DA,E8,D0,F1,60
```

Pozíciós bevitel GW BASIC-ben

Hogy a kép szép legyen

Hasonlóan a kisebb gépekhez, a PC-kategóriában is a kezdők számára legkönnyebben elsajátítható programnyelv a BASIC. Az igényesebb kivitelezés során hamar felmerül a szép, szerkesztett szöveges kép igénye, ehhez pedig szükség van pozíciós input utasításra. Ezt a feladatot oldja meg az alábbi, GW BASIC 3.11 verzióban írt rutin.

A főprogram és a 200-tól kezdődő szubrutin között az információcsere öt változó, a CS, CO, X$, XL és X segítségével történik. A szubrutin számára a CS, CO, X$ beviteli, az XL és X beviteli-kiviteli változó. A beviteli változók értékeit a hívó programban, a rutin meghívását megelőzően kell beállítani. A CS, CO változókba kerül a kurzorpozíció sora és oszlopa; ez a közölt *listán* látható főprogramban — az első tíz sorban — éppen 20 és 10. Az X$ és X változó szolgál a billentyűzetről beolvasandó szöveg, illetve szám befogadására. A kétféle esetet az X változó bemeneti értékének 0-ra vagy 1-re állításával különböztetjük meg. A közölt példában számot kívánunk fogadni, ezért a 140-es sorban X=0. Az XL változó bemeneti értéke maximalizálja a beolvasandó karakterfüzér hosszát; kimeneti értéke a tényleges hosszt adja meg szöveg fogadásakor — számbevitelnél természetesen nincs értelme.

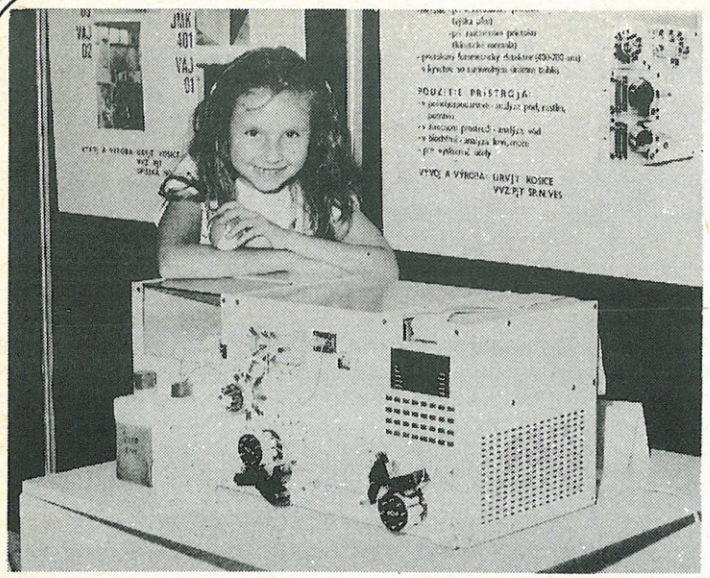
A szubrutinban a kurzor pozicionálása a „LOCATE sor, oszlop" utasítással történik. A billentyűzetről beolvasott karakterek ettől a pozíciótól kezdődően jelennek meg a képernyőn. Billentyűzettel a kurzort nem lehet mozgatni, csak a BACKSPACE billentyűvel törölhetjük az utolsó karaktert. Mind a szöveg, mind a szám a RETURN billentyűvel zárandó le, itt van a szubrutin vége is. A számjegykarakterek között pont és mínusz (—) jel is lehet. A rutin egyszerűen kibővíthető oly módon is, hogy képes legyen például normál alakban megadott szám fogadására.

Megjegyzendő, hogy egy PRINT utasítás végrehajtása után a kurzorpozíció a következő sor első karakterére mutat, ezért gondoskodni kell a helyreállításáról.

A közölt példában, a főprogramban tehát szám beolvasása történik, a képernyő 20. sorának 10. pozíciójától legfeljebb 4 karakter hosszban, azaz a —999 és a 9999 határolta számtartományban.

Dr. Szikszai Csaba

```
100 REM ******************************
110 REM *  pozinput pelda szamra     *
120 REM ******************************
130 CLS
140 CO=10:CS=20:X=0
150 XL=8
160 GOSUB 330
165 LOCATE 2,50
170 PRINT X
180 END
200 REM ********************
210 REM *  pozinput rutin *
220 REM ********************
230 REM  Valtozok: CS - kurzor sora
240 REM            CO - kurzor oszlopa
250 REM            X$ - input string
260 REM            XL - string hossza
270 REM                 input: maximum
280 REM                 output: tenyleges ertek
290 REM          X  - adatjelzö es input szam helye
300 REM               0 : szamot, 1 : szoveget varunk
310 REM
320 REM Munkavaltozok: SM,OM,X1$,X1,LM,LL
330 X$="":SM=CS:OM=CO:LM=0:X1$="":LL=X
340 LOCATE SM,OM
350 X1$=INPUT$(1):X1=ASC(X1$)
360 IF LM<>0 THEN IF X1=13 THEN 540  ELSE IF X1=8 THEN 470
370 IF LM<XL THEN IF X1>=48 AND X1<=57 OR X1=46 OR X1=45  THEN 400 ELSE IF X1>32
 AND  X1<127  AND X=1   THEN 400
380 GOTO 350
390 REM fel
400 PRINT X1$
410 OM=OM+1
420 LOCATE SM,OM
430 LM=LM+1
440 X$=X$+X1$
450 GOTO 350
460 REM del
470 OM=OM-1
480 LOCATE SM,OM
490 PRINT" "
500 LOCATE SM,OM
505 X$=LEFT$(X$,LEN(X$)-1)
510 LM=LM-1
520 GOTO 350
530 REM vege
540 IF X=0 THEN X=VAL(X$) ELSE XL=LM
550 RETURN
```



INJEKCIÓS ÁTÁRAMLÁS-ANALIZÁTOR, PIA—01

Az új, analitikus készüléket az a radioökológiai és magtechnikai felhasználással foglalkozó kassai intézet készítette el, amely a vezető csehszlovák intézetek közé tartozik a környezeti problémák kutatásában a radiometrikus és további módszerek felhasználásával. Az analizátor az átáramlási analízis módszerét használja; az átvizsgálandó folyadékot a vivőáramba fecskendezik be, és ott egy fotometrikus detekció által kiértékelt zónát képez. A készüléket a Palacky Egyetem természettudományi fakultásával együttműködve fejlesztették ki Kassán, és a prágai Károly Egyetem matematikai, fizikai fakultásával együtt elsősorban a környezeti próbák és a minőségi vizsgálatok analízisére, a mezőgazdasági és élelmiszer-nyersanyagok vizsgálatával foglalkozó és termelési ágazatok számára szánják.

A teljes mérési és kiértékelési rendszert a készülékben egy mikroprocesszor vezérli, amely a hidrodinamikus és az elektrolikus funkciós egységek átkapcsolásakor a teljes automatikus üzemre óránként 200 analízist is lehetővé tesz. Összehasonlítva egy másik analitikus módszerrel, az automatikus mérési eljárással, ez az új analizátor, a PIA—01, könnyen kezelhető és egyúttal objektív munkamódot ér el, jelentősen alacsonyabb analitikai munkaköltség mellett.

Az univerzális jelleg a kiválasztott módszernél és ezzel az egész készüléknél is egy sokkal szélesebb felhasználási skálát enged meg a környezeti kutatás területén kívül és a vizsgálatokon kívül is, amelyekre eredetileg a készüléket létrehozták. Ezzel az analizátorral többek között a klinikai biokémiában jó tapasztalatokat értek el, például a vérszérumok analízisénél, a biológiai anyagok, a vizelet stb. elemzésénél, az iparban a vegyi elemzéseknél, az élelmiszeriparban a tej, az italok, a hús, a zöldség és a gyümölcs ellenőrzésénél. Éppen ez az univerzális felhasználhatóság, valamint a progreszszív technikai megoldás az oka annak, hogy széles szakmai körökben nagy érdeklődés mutatkozott a nemrég befejeződött Incheba Nemzetközi Vásáron, ahol a készüléket először mutatták be a nemzetközi nyilvánosság előtt.

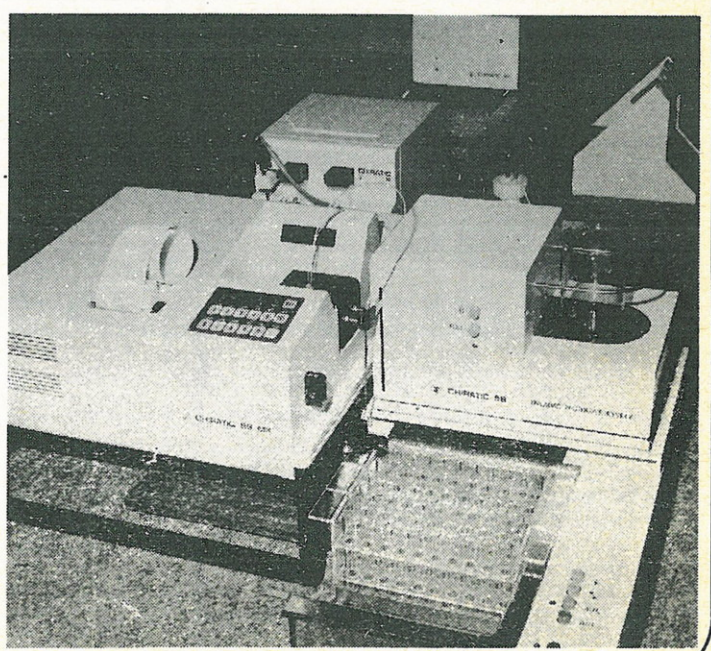
(Közelebbi információkat a KOVO, Prága Külkereskedelmi Vállalat ad.)

CHIRATIC 59 MK

A Chiratic 59 MK kinetikus fotométer, a laboratóriumi analízisek és a koncentrációs mérések, a biokémiai és orvosi laboratóriumok próbái számára szolgál. Az objektív diagnózisok meghatározásakor az analitikus módszerek egyre fontosabb jelentőséget kapnak, és ezért egyre növekszik terjedelmük, így az a lehetőség, hogy az egyes próbákkal való foglalatosság a kezelő beavatkozása nélkül, automatikusan is elvégezhető, ebben az esetben döntő kritériuma a készülékgyártásnak. A Chiratic 59 MK, amelyet egy cél-mikroszámítógép vezérel mikroprocesszorral, sikeres abban, hogy az átfolyó cellák vezérlőkészülékének kombinálásával a Chiratic 58-cal és a Chiratic 52-es próbaszállítóval egyszériás próbamegmunkálást lehetett elérni, amelynél csak minimális emberi kezelésre van szükség. Az automatikus mérés növeli a munka produktivitását a laboratóriumban, csökkenti a kezelői beavatkozást és a próbák összecserélésének lehetőségét, valamint csökkenti az infekció lehetséges veszélyét is.

A beadott információkat és mérési eredményeket a Chiratic 59 MK négy helyiértéken ábrázolja, és ezzel egyidejűleg dokumentációs célokra egy beépített termonyomtatóval ki is nyomtatja.

A Chiratic 59 MK készüléket a Chirana Stará Turá Konszern Vállalat állítja elő. Közelebbi információkat a műszaki adatokról, valamint az esetleges szállításokról a Chirana Konszern külkereskedelmi osztály közöl Piestanyban.



Az értékelőfüggvény végső formája

Az 1988/12-es számtól kezdődően részletesen bemutattuk az értékelőfüggvény összetevőit, ezek hatásait a játék különböző szakaszában. Megtekinthettük a különféle komponensek ábráját a féllépésszám függvényében. Most kialakítjuk a végső értékelést, bemutatjuk, hogy milyen mértékben érdemes az egyes komponenseket súlyozni.

Rendszeres olvasóink korábbi számainkban láthattak hasonló értékelőfüggvényt, de az nem volt ennyire precízen kidolgozva, kevesebb szempontot vett figyelembe. Igaz, a túlzottan aprólékos értékelés, a precíz megfogalmazás viszont sokkal több gépidőt vesz igénybe, ezért a sakkprogram készítőjének kell megtalálnia a két véglet közötti optimális megoldást az adott számítógép gyorsaságának a függvényében.

Az értékelőfüggvény a következőképpen néz ki:

F=1000*M+P+D+0.5*C+
L+2*R+Q+2*B+0.5*OW+OP

A betűk jelentése a következő (a zárójelben lévő hivatkozások azokra a számainkra utalnak, ahol azzal az összetevővel részletesen foglalkoztunk):

M : Anyagi érték (1988/12)
P : Pszeudolegális lépések (1989/1)
D : Különböző megtámadott mezők száma (1989/8)
C : Centrumkontroll (1989/2)
L : David Levy mozgékonysági törvénye (1989/3)
R : Bástyabónusz
Q : Vezérbónusz (1989/4)
B : Futóbónusz (1989/5, 1989/6)
OW: Megtámadott saját mezők száma (1989/8)
OP : Megtámadott ellenséges mezők száma (1989/9)

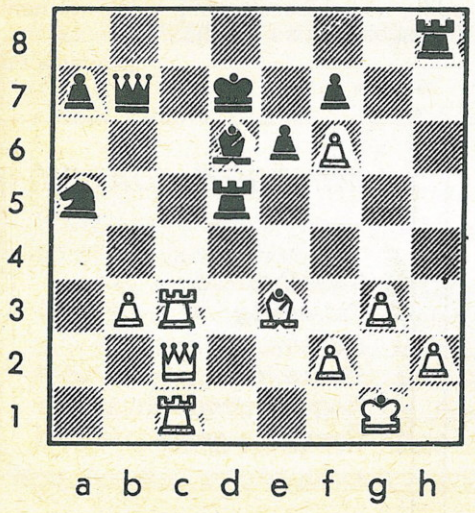
A bástya bónuszát a következőképpen számítjuk:

Fr=1.6*A—1.6*D+8*DR+8*0+3*S+
22*SR

A betűk jelentése:

A : Megtámadott mezők száma.
D : A saját király és a bástya sor- és oszlopkülönbségének minimuma.
DR : Kettős bástya a soron vagy az oszlopon.
O : Bástya a nyílt vonalon.
S : Bástya a félig nyílt vonalon.
SR : Bástya a 7. soron.

1. ábra. Kaszparov—Timoscsenko 1981 58. féllépés után



Most pedig nézzük meg a konkrét példán, milyen értékeket ad az értékelőfüggvényünk! Az 1. ábrán látható hadállásra a következő értékek adódnak:

Világos: M=2.7; P=39; D=40; C=148; L=20; R=48; Q=11; B=14; OW=41; OP=14.
Tehát F(világos)=3042.5

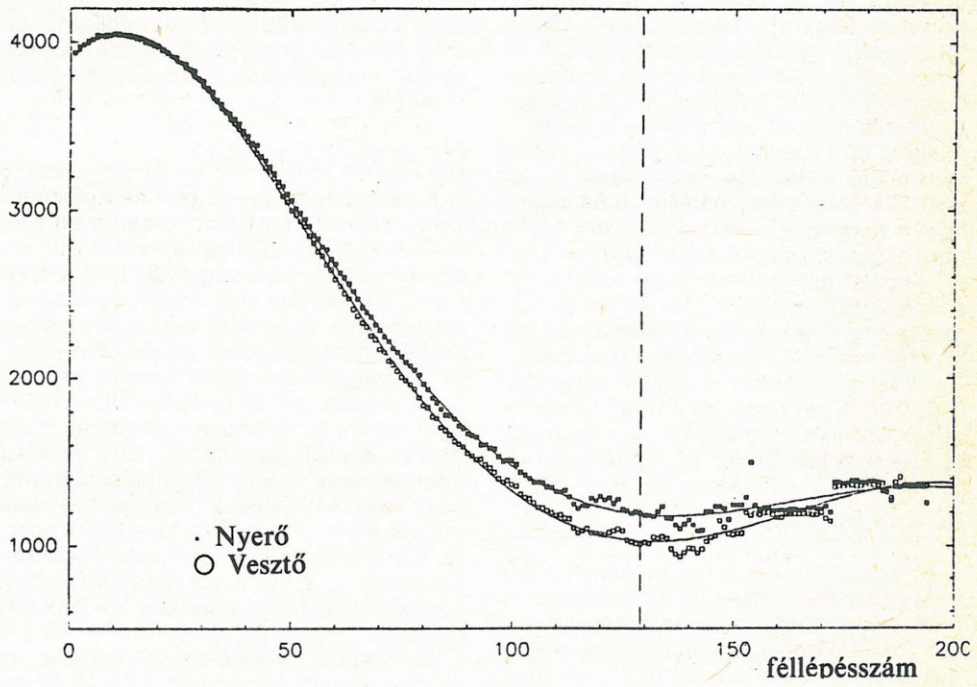
Sötét: M=2.8; P=54; D=41; C=178; L=25; R=56; Q=6; B=—1; OW=50; OP=15.
Tehát F(sötét)=3165.

A végső érték: F(világos)—F(sötét)=—122.5, ami azt jelenti, hogy sötétnek körülbelül egy gyalognyi előnye van.

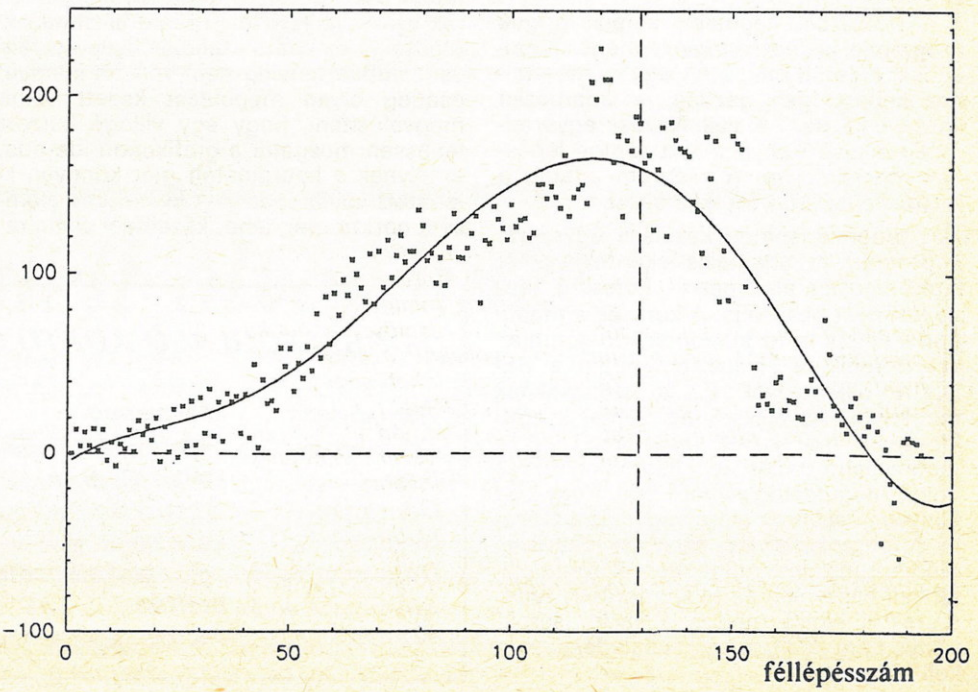
A 2. ábrán láthatjuk, hogyan alakul függvényünk átlagértéke a féllépésszám függvényében.

KOVÁCS P. ATTILA

Σ pont *2/a ábra*



Világos és sötét pontértékének különbsége *2/b ábra*



PROGRAMTERMÉK

Kémiai adatbázis Plus/4-re

A szólás régi: a fejem nem káptalan. A polihisztorok is erősen kihalófélben vannak. A tudomány fejlődésével csak az össznépi dilettantizmus fejlődik egyenes arányban. Minél több ismeretet halmoznak fel a tudósok, annál többet nem tudunk mi, közönséges halandók.

Egy mai diák legfontosabb feladata már nem az, hogy óriási adathalmazokat magoljon be memoriterként (bár itt is utalnom kell a bölcs kompromisszum szükségességére). Nem! Ennél sokkal fontosabb a tanulás technológiájának és a hiányzó adatok megszerzési módjának megtanulása. Fel kell készülni arra, hogy az iskolában már nem lehet egyszer s mindenkorra elsajátítani egy szakmát, hanem egész életünkben tanulni kell, ha lépést akarunk tartani a világ fejlődésével. Egy, a negyvenes években végzett elektromérnöknek például napjainkig már háromszor átírták az egész elektronikát az elektroncsőtől a tranzisztoron át a mikrochipekig. A felhalmozott adattömeg fejben tartása pedig éppenséggel kivihetetlen még egyetlen szűk szakterületen is. E sok adat tárolását és visszakereshetőségét tekintve egyre inkább a technika segítségére szorulunk.

Európa nyilvános!

Az adatbázisok már hazánkban is szinte a „levegőben lógnak". A tv hullámain — immár a műholdas adásokat is beszámítva — teletexttel akár egész Európa nyilvános adatbázisaiba betekintést kaphatunk. A Magyar Posta jelentős erőfeszítéseket tesz, hogy ennél jóval több szolgáltatást nyújtó adatbázisokat is elérhessünk. (A legszebb az lenne, ha ebből az oktatás is profitálhatna!)

A Novotrade legújabb kémiai tárgyú programja kapcsán nagyon megörültem, amikor kiderült róla, hogy egy — méreteihez képest igen gazdag — adatbázist kaptam kézhez. A vállalkozást egyértelműen a jövő irányába tett fontos lépésnek minősítettem. A program adatait a szokásos táblázat foglalja össze.

A programcsomag kezelése egyszerű. A program az adatbázis-lekérdezéseknél kizárólagosan alkalmazott korszerű menüvezérlést használja. A keresés a menüben a Plus/4 le-fel kurzorvezérlő gombjaival történik, a kiválasztás pedig a RETURN billentyűvel. Ez a szokásoknak megfelelő. Az elemek adatlapjai közötti lapozásra már kicsit szokatlan, hogy a jobbra-balra nyilat kell használni, bár a dolog megideologizálható úgy, hogy egy könyvben akarunk előre vagy hátra lapozni. A legszokatlanabb azonban a keresés módja a tartalomjegyzékben. A fel-le nyilat kell használni, de más interaktív rendszerektől eltérő módon a kívánt scrollirányt, nem pedig a logikai haladási irányt kell megadni.

A program dokumentációja meglehetősen lakonikus, s ezt a menüben található leírás sem bővíti ki túlzottan. Az „on-line help" mindenesetre tökéletesen helyettesíti a kazettaborítón található dokumentációt, és végül is elegendő a nem túl bonyolult algoritmusú lekérdezés elmagyarázására.

Háromfogásos menü

A programmal — az on-line helpen kívül — három menüelem közül választhatunk. Az első menüelem a *lapozás,* amely az elemek tulajdonságainak (két képernyőn 19 különféle adat) megtekintését teszi lehetővé. A 15 adat szerint kijelölhető sorrendről a *tartalomjegyzék* menüponttal intézkedhetünk. A tartalomjegyzék lehetővé teszi, hogy gyorsan kikeressünk egy elemet az adatbázisból, majd megnézzük annak adatlapjait. Egy adatlap megtekintése után továbblapozhatunk, vagy visszatérhetünk a tartalomjegyzékbe (sajnos nem a korábban kikeresett pontra, hanem az elejére), és újabb elem után kutathatunk. A harmadik menüpont *grafikus ábrázolást* tesz lehetővé Y=f(X) formában.

Az Y és X — ebben a sorrendben, ez először engem is becsapott! — 15-15 tulajdonság közül tetszőlegesen választható. Olykor igen meghökkentő alakú függvényeket kapunk. A probléma egyik oka talán a sok hiányzó adatban keresendő, a másik oka a skálázás teljes hiánya. Az ábrák csak kvalitatív elemzésre alkalmasak. Minthogy az adatok felírása ilyen sokféle variációban tényleg nem volt túl könnyű, esetleg olyan megoldást kellett volna megvalósítani, hogy egy villogó kurzort lehessen mozgatni a grafikonon ide-oda, amelynek a koordinátáit már könnyen ki lehetett volna jelezni. A kvantitatív elemzést persze meg lehet közelíteni olyan ravasz trükkel, hogy egyszerre két gépbe töltjük be a programot, majd miközben az egyik gépen kirajzolunk egy grafikont, a másikon az ordináta-tulajdonságra rendezett tartalomjegyzéket állítjuk be.

ESC= menekülés

A programnak a jelzetteken kívül több fogyatékossága alig van, kifejezett hibát pedig nem is találtam benne. Az adatok helyességét persze nem ellenőriztem. A grafikus ábrázolás gyengeségei között megemlítem még, hogy túl erősen befagyaszt egy ábrát, úgyhogy csak a menühívást jelentő ESC gomb menekít ki ebből a funkcióból. A kiléptetésre elég lett volna talán egy RETURN is, és jobb lett volna, ha a program visszatér a grafikus almenübe. Az ESC gomb funkciója ekkor sokkal tisztább lett volna.

Túl sok kívánnivalót a program esztétikai kivitele sem hagy maga után. Az adatok kiírása a teljes magyar ékezetes ábécével történik, de ezen túl a mértékegységek megadására speciális karaktereket is felvett a szerző a karakterkészletbe. Az adatok a SI-egységekben szerepelnek, ami manapság igazán ildomos. Kivétel talán a Celsius-fok, ami miatt néhány grafikon „átlóg" a negatív mezőbe. A táblázatok adatai érdekesek, de néhány termokémiai jellemző talán fontos lett volna még.

Külön kiemelendő, hogy a program igen gyorsan működik, ami a tanórai alkalmazásoknál igen kedvező tulajdonság.

A program nemcsak az általános iskolai munkához nyújt segítséget, hanem a középiskolákban is igen hasznos (felmerül a PC-s változat elkészítése, mivel a középiskolákban nagyon megindult a PC-vonal!). Jól használható az önképzésben, sőt a tanárok ismereteinek bővítésére is alkalmas. A szerző minden elismerést megérdemel, érezhetően elég kemény munkát végzett. Egyszer a Chemisys-projekt kapcsán nekem is meg kellett csinálnom egy hasonló adatbázist, és emlékszem, menynyi baj volt vele. Az adatoknak ugyanis van egy nagyon makacs tulajdonságuk: addig nem hibásak, amíg valaki fel nem fedezi bennük a hibát(!).

Zsadányi Pál

ÖSSZEFOGLALÓ ADATOK

Forgalmazó:	*Novotrade*
Terméknév:	*Elemek*
Szerző:	*Bálint György*
Géptípus:	*C Plus/4*
Hordozó:	*kazetta*
Dokumentáció:	*egy oldal a kazettaborítón*
Ár:	*345 Ft*

MINŐSÍTŐ ADATOK

Kezelhetőség:	*jó*
Teljesség:	*kiváló*
Dokumentáltság:	*jó*
Használhatóság:	*kiváló*
Ár/teljesítmény:	*kiváló*
Összbenyomás:	*kiváló*

ADOK—VESZEK —CSERÉLEK

Ebben a rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás: kereskedelmi tevékenységet folytatóknak gépelt soronként (60 karakter) 100,- Ft, másoknak az első sor 50,- Ft, minden további sor 20,- Ft. Kérjük, hogy a hirdetés díját az NJSZT OTP V.ker. fiókjánál (218-98055) vezetett 508-8609 rendezvényszámlára vagy 1054 Bp., Báthori u. 16. címére fizessék be, rózsaszínű postautalványon (jelölve, hogy apróhirdetés), a befizetést igazoló szelvényt pedig csatolják a hirdetéshez. Hirdetéseiket a szerkesztőség címére várjuk (1371 Bp., Pf.: 433). Az NJSZT tagjai továbbra is kedvezményesen hirdethetnek (az első sor ingyenes), de kérjük, hogy adják meg tagsági számukat. Azokat a hirdetéseket, amelyek a hónap első napjáig megérkeznek, már a hónap végén viszontláthatják lapunkban.

ADOK

Amiga 500 számítógép és MPS 1200 printer eladó. Szükség esetén programokat is tudok biztosítani. Raiz Tamás, Budapest, Győri u. 8. 1123. Tel.: 566-941

Amiga!!! THE WINNER IN HUNGARY!!! Legújabb programok minden mennyiségben. Laci Szőnyi, Budapest, Tavirózsa 5. 1161. Tel.: 848-471

Amiga 500-as (új) programokkal, Atari 800XL + floppy és Philips színes monitor kedvező áron eladó. Tel.: 42/18-632

Amiga programok nagy választékban eladók! Érdeklődni lehet: Keresztes Gábor, Budapest, Laky köz 11. 1142. Tel.: 643-452

Amiga! Legújabb külföldi programok! Balassagyarmat, Pf.: 118. 2661

C Plus/4-es, C16-os programokat adok (10 Ft/db) kazettán. Listát kérésre küldök. Keresem a RAMAN nevű programot. Agárdi Tibor, Kecskemét, Dankó P. u. 37. 6000

C Plus/4, magnó, 1551-es floppy, 2 db joystick, BASIC 7.0 cartridge, kb. 150 program, szakkönyvek eladók. Irányár: 45 000 Ft. Ujvári Róbert, Csongrád, Lenin u. 3. 6640

Commodore Plus/4 + magnó + szakkönyvek + 200-nál is több kitűnő program eladó. Ár megegyezés szerint. Tenkes Zoltán, Budapest, IV. ker. Erzsébet u. 29. I/10. 1043. Tel.: 895-348

C16, C Plus/4, C64, C128, Amiga 500-ra a legújabb 1989-es programok eladók. Keresztfalvi János, Budapest, Doberdó u. 4. 1034. Kérésre listát küldök!

C64-es lemezek eladók (200 db) 100 Ft/db (szuper programokkal). Kérésre listát küldök. 1351-es egér, Citizen 120D + interfész, magnó (6000, 25 000, 2000 Ft) eladók. Németi Ferenc, Budapest, Nagyenyed u. 8/a. 1182

C64-hez 40 db 3M, illetve Commodore lemez eladó. A rajtuk levő programokért ('86-'89) nem kell fizetni (+ leírások, gépi kódú könyvek). Czene Iván, Budapest, Farkasfa u. 29. 1165

C64 (16 000), C Plus/4 (12 000) + magnó + joystick + könyvek + programok olcsón eladók. Tel.: 33/11-599 (hétköznap!) Hétvégén: Jelen Imre, Esztergom, Töltés u. 1/17. 2500

C64-re programokat olcsón eladok. Várhegyi István, Nyírbátor, Erzsi u. 31. 4300

C64-es programokat adok és cserélek lemezen. Ugyanitt C16-os eladó magnóval, bővítővel, játékokkal. Tima Márk, Tárnok, Béla u. 17. 2461

C64 (új) magnóval, programokkal olcsón eladó. 41256-os IC-t beszámítok. Pálhidai László, Kecskemét, Lánchíd u. 7/A. II/6. 6000. Tel.: 76/46-284 (17 óra után)

C64, 1541-es floppy drive, Final III. cartridge, 78 db lemez programokkal, 100 db-os lemeztartó, joystick és szakirodalom eladó. Irányár: 40 000 Ft. Bognár Attila, Budapest, Práter u. 18. 1082. Tel.: 141-510

C64-hez cartridge-ok és hangdigitalizáló eladó. Varga Zsigmond, Budapest, Nádastó park 10. 1156. Tel.: 645-442

C64-re régi és új, színvonalas játékokat adok (15 Ft/db) és cserélek is minőségi programok esetén, kazettán. Válaszborítéték ellenében listát küldök. Béres Csaba, Fényeslitke, Vasút sor 10. 4621

C128 + 1570 diszk + magnó + Final cartridge III. és 1500 új program eladó. Almási György, Nyíregyháza, Kiss E. u. 50. 4400. Tel.: 42/18-392

C128D eladó! C128, C64, CP/M mód + 1571 beépített floppy. Már egy C64 + 1541 II. áráért az Öné lehet! Ajánlatokat: Radics Róbert, Kiskunfélegyháza, Szőlő u. 10/4. 6100. Tel.: 76/61-116

Enterprise 128, Spectrum-emulátor, egér, szakkönyvek, programok együtt 20 000 Ft-ért. eladók. Bak Attila, Budapest, Népszínház u. 40. II/4. 1081

Enterprise programokat nagyon gyorsan és olcsón adok, illetve cserepartnereket keresek. Széles programválaszték. Sándor József, Bonyhád, Bezerédi u. 41. III/5. 7150

Enterprise programokat adok, veszek, cserélek. Kiss József, Budapest, Bezerédi u. 24. 1181

Enterprise programokat adok, veszek, cserélek! Listát kérek! Molnár Imre, Budapest, Havanna u. 7. V/27. 1181

Enterprise programokat adok és cserélek. Válaszborítéték ellenében listát küldök! Nyigrényi József, Szekszárd, Wosinszky ltp. 29. III/8. 7100

Enterprise programokat adok-veszek, cserepartnereket keresek. Kb. 180-190 db kiváló minőségű programmal rendelkezem! Keresek megvételre egy jó állapotban lévő Spectrum 48 k-t a következő tartozékokkal: magnó, joystick, interfész. Ifj. Sala István, Budapest, Hársfa sétány 17. V.30. 1203

Enterprise 128 + magnó + programok + könyvek + joystick eladó. Ár: 16 500 Ft. Nagy Csaba, Nyírbátor, Édesanyák u. 39. 4300

Enterprise programokat adok és megfelelő partnerrel szívesen cserélek. Széles programválaszték a legújabb programokkal! Válaszborítékért listát küldök. Olcsó ár, gyors válasz! Cím: Barabás Barbara, Bonyhád, III. Posta, Pf.: 25. 7153

Enterprise programok eladók! 10-től 50 Ft-ig. Válaszborítékért listát küldök. Gyurta Viktor, Budapest, Szövőszék u. 14. VI. em. 1108. Tel.: 775-695

IBM PC XT 256 k memóriával, reális áron eladó. Galla Tamás, Kisvárda, Felszabadulás ltp. 3/b. fszt. 3. 4600

IBM PC-hez vadonatúj, 360 k-os floppycsatoló kártya csatolóval olcsón (4500 Ft) eladó. Turbo XT/R.O.C./ alaplaphoz és CGP kártyához leírás kapható. Tel.: 866-447

TV-Computer programokat adok-veszek-cserélek. Listát, válaszborítékot kérek! (750 db program) Molnár János, Szolnok, Jászi F. u. 10. VI/25. 5000. Tel.: 56/31-085

Spectrum emulátor 5000 Ft és CBM MFT programcsomag 1000 Ft-ért eladó. Érdeklődni lehet: Budavölgyi László, Budapest, Szövetség u. 43. I/14. 1074

ZX-Spectrum tartozékokkal eladó. Gulyás Ákos, Budapest, Endrődi u. 42/D. 1026

ZX-Spectrum+ számítógép, joystickkal, joystick interfésszel, programokkal és szakkönyvekkel eladó. Irányár: 12 000 Ft. Simon Levente, Tel.: 585-532, este.

80 k-s **Zx-Spectrum** 640 k-s lemezkapacitású floppy meghajtóval, sok játék- és rendszerprogrammal olcsón eladó. Belányi László, Vác, Gombási u. 36/A. 2600. Tel.: 27/10-133/301-es mellék (15 óráig)

Szinvonalas **ZX-Spectrum 48/128**-as játékprogramok olcsón eladók. Megfelelő partner esetén programcsere is lehetséges. Jelszavam: Gyorsan, olcsón, minőségit! Érdeklődni: Fonyódi Zoltán, Ajka, Petőfi Sándor u. 5. 8400

ZX-Spectrumra adok, veszek, cserélek programot. Címem: Devecz László, Lenti, Átkötő u. 5/D. 8960

ZX-Spectrumhoz 2 joystick-kimenetes interfész eladó. Reset gombot is tartalmaz. KEMPSTON, CURSOR és SINCLAIR típusú joystickra alkalmas. Érdeklődni: Miszori Róbert, Ajka, Újélet u. 6. 8400. Tel.: 89/11-284

ZX-Spectrum + floppy-illesztő - együtt vagy külön-külön -, játék- és felhasználói programokkal, szakirodalommal eladó. Ára: 9000 + 9500 Ft. Cím: Budapest, Balázs Béla u. 42. I/3. 1094. Érdeklődni lehet: 334-760/296-os telefonon.

Zx-Spectrum (48 k) tartozékokkal (Kempston interfész, joystick, magnó, 200 db program, sok könyv) olcsón eladó. Kisida Gábor, Miskolc, Vörösmarty u. 26. 3530. Tel.: 87-713

ZX-Spectrum (48 k), Kempston interfész + joystick, user port (PIO), szakkönyvek, többszáz program, MK magnó (kis hibával) 20 000 Ft-ért eladó. Varsányi Lászlóné, Sopron, Hajnóczy u. 13. 9400

ZX-Spectrum+ Kempston interfésszel, 2 joystickkal, 120 programmal, szakkönyvekkel sürgősen eladó! Bubori Gábor, Kaposvár, Damjanich u. 1/3. 7400

Eladom alig használt, öt hónapos Citizen 120-D nyomtatóm + ráadásként 30 lemeznyi jó minőségű, tetszőlegesen kiválasztott C64-es programot, lemezekkel együtt 27 000 Ft-ért. Péli Gábor, Tel.: 496-032

Eladó Interfész I. + Mikrodrive + 4 db cartridge. Árajánlatokat az alábbi címre kérek: Galyassy Tamás, Baja, Kodály Z. u. 12. 4/15. 6500

Fordításokat készítek a **64'er Magazin**ból. Több mint 3000 oldalnyi kész anyag! Kedvező árak. Textomat+, GEOS 1.2, Hi-Eddi, Giga CAD, Disc Wizard, Disc Demon stb. dokumentációk. Szolnoki Béla, Budapest, Pf.: 400. 1446. Válaszborítékot kérünk!

1-3 hetes törésű C64 játékok szuper olcsón, közvetlenül nyugatról. TTS, Szeged, Szamos u. 3/b. 6723

Joystick Szerviz a Flórián Áruházhoz közel! Javítás, magnófej beállítás. C64 játékprogramok kazettán és floppyn 10 Ft/db. Budapest III. Kerék u. 36. IV/24. Hétfőn és szerdán 17-től 19 óráig.

Jutányos feltételekkel vállalok középfokú német nyelvvizsgára való fölkészítést. C64 vagy C128 és floppy szükséges. Érdeklődni: Szolnoki Béla, Budapest, Pf.: 400. 1446

Nyelvtanulás C64-en! 5000 szavas szótár diszkem pillanatok alatt oda-vissza szótároz, oktat a Németi I-II, vagy Themen I-II leckéi szerint. Bővíthető, igény esetén Plus/4-re is. Ára 950 Ft. Megrendelhető: Kiss András, Kaposvár, Arany J. köz 12/II/3/1. 7400

ÚJ VADNYUGAT - magyar nyelvű kalandjáték C64-re. Ára lemezen 370 Ft, kazettán 340 Ft. (A hordozó árával együtt!) Megrendelhető: Rátkai István, Esztergom, Bocskoroskúti út 28. 2900

VESZEK

C64-hez lemezegységet veszek! A válaszlevelet ármegjelöléssel kérem az alábbi címre: Egri Imre, Békésszentandrás, Pacsirta u. 6. 5561. Ugyanott programcsere is, kazettán! Több mint 2200 programot tudok cserére felajánlani.

Keresem a Master64 és a C64 Forth (Mikroproducts) program dokumentációját, a GEOS programozói leírását angol vagy magyar nyelven, valamint C64-es felhasználói és játékprogramokat veszek. Listát várok! Szabó Péter, Jászberény, Korányi u. 1.

CSERÉLEK

C16, C Plus/4 és C64-es programjaimat kínálom fel cserére, szalagon és lemezen! Főleg oktató- és játékprogramok érdekelnek. Lehetőleg listát kérek! Várom a régi cserepartnereim jelentkezését is! Hika György, Szeged, Sárosi I/A. 6724. Tel.: 62/30-496 (17 óra után!) Csak cserélek!

C16, Plus/4 programokat cserélnék kazettán. Listát a következő címre kérek: Szabó Rita, Gyöngyössolymos, Lenin út 24. 3231

C64-re játék, felhasználói, demo programokat cserélek kazettán (900 program), lemezen (100 program). Sok TOP-listás program. Veszek stratégiai játékokat, leírásokat. Keresem a Micro-Prolog programot. Listát kérek! Várhegyi István, Nyírbátor, Erzsi u. 31. 4300

C64-es programokat cserélek vagy eladok (10 FT/db) kazettán. Listát kérek és adok. Balogh Zoltán, Kapuvár, Szent László u. 15. 9330

C64-es programokat cserélek és eladok lemezen és kazettán. Cserealapom 2000 program és 250 leírás. Szabó Csaba, Öcsa, Üllői út 28. 2364

C64-re színvonalas programokat (88/89) cserélek lemezen. Szíjártó Péter, Cegléd, Köztársaság út 14/B. III. 12. Tel.: 11-864

Enterprise programokat cserélek - elsősorban lemezen. Válaszboríték ellenében listát küldök. Ugyanitt EP-Spectrum-emulátor eladó. Szabó Tibor, Budapest, Gazdagréti tér 3. X/117. 1118

Enterprise programcsere. Listát kérek! Fehér Sándor, Gyulakeszi, Jókai u. 1/1. 8286

Enterprise programcsere. Keresem a Tank Command című programot. Csányi Viktor, Jászberény, Gárdos M. u. 17. 5100

Enterprise programokat cserélek, veszek, leírással adok. Listát kérek, válaszboríték ellenében listát küldök. Bujtor Gyula, Budapest, Zsilvölgy u. 6. 1209

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

Tisztelt olvasóink!

Igen sokan keresik fel szerkesztőségünket azzal a kéréssel, hogy a Mikroszámítógép Magazin régebbi számai közül szeretnének vásárolni.

Bár minden korábbi évfolyam minden száma már nem lelhető fel sem a szerkesztőségben, sem a kiadóban, úgy érezzük, lapunk hasábjain érdemes közzétennünk azoknak a számoknak a listáját, amelyek még megvásárolhatók a kiadóban (Neumann János Számítógéptudományi Társaság, Budapest V., Báthori u. 16.).

Meglévő számaink:
1985/4.
1986/6., 10., 11—12.
1987/1., 2., 4., 5., 6., 7., 8., 9., 10., 11., 12.
1988/8., 9., 11., 12.
1989/1., 2., 3., 5., 6., 7., 8.

Reméljük, sokan találják meg azt az információt, cikket, programot, amely segítséget nyújthat a munkában, illetve tanulásban.

KÖNYVEK — HÍREK — ÉRDEKESSÉGEK

Ráth György:
Adat- és programvédelem IBM PC-re
(Budapest, 1989. LSI ATSZ,
63 oldal. Ára: 97,— Ft)

A mikrogépes adat- és programvédelem története és fejlődése szorosan összefügg a mikrogépek fejlődésével. Sokszor vált döntővé egy-egy új mikrogép megjelenése a piacon, és messzemenően megváltoztatta még a tőle jelentősen eltérő gépekhez kapcsolódó védelmi stratégiákat is. Kezdetben a mikrogépes védelem kérdése fel sem merült, kevés gépet dobtak a piacra, és a kereskedelemben forgalmazott programok köre is szűk volt. Az adatvédelem nem is létezett még mikrogépekre, a viszonylag csekély kapacitású hajlékonylemezek, a nehézkesen kezelhető magnókazetták nem számítottak komoly adattárolóknak.

Mint a bevezetőből megtudhatjuk, az áttörést az Apple gépek megjelenése okozta, mivel egyre többen kezdték egymás között cserélni a játékprogramokat. Ez azonban figyelmeztetés is volt egyúttal, mivel egy egyszerű számítógépes játék lemásolása semmiben sem különbözik egy nagy felhasználói rendszer lemásolásától.

Ezért az Apple-II világában a programvédelmek és az azokat kijátszó módszerek rendkívül dinamikusan fejlődtek, és pár év alatt igen magas szintre jutottak. A szerző ezeknek a technikáját ismerteti.

Zimányi Magdolna—Kálmán László—
Fadgyas Tibor:
A LISP programozási nyelv
(Budapest, 1989. Műszaki Könyvkiadó,
428 oldal. Ára: 195,— Ft)

A LISP nyelv a mesterséges intelligencia kutatásának kezdettől fogva alapvető programozási nyelve, amely meghatározó szerepet játszik a kutatásban és az eredmények gyakorlati alkalmazásában is. A nyelv neve a „LISt Processing language" (listafeldolgozó nyelv) rövidítése. Míg a LISP nyelvvel közel egy időben született Algol 60 és FORTRAN nyelvet elsősorban a numerikus számítások igényei szerint alakították ki, a LISP nyelvet azért hozták létre, hogy szimbolikus (szimbólumokból álló) kifejezésekkel végzett műveletekre alkalmas nyelvet teremtsenek.

A LISP azért is különleges programozási nyelv, mert új elgondolások alapján tervezték meg. Ezek jelentőségét sok tekintetben csak ma tudjuk igazán értékelni. Ezáltal a LISP a funkcionális nyelveknek lett első és máig legfontosabb képviselője. A nyelv egyik legjelentősebb alkalmazási területe a mesterséges intelligencia kutatása, ahol a felmerülő feladatokat az jellemzi, hogy a megoldásukhoz összetett adatszerkezeteken bonyolult logikai eljárásokat kell végrehajtani. Ezekhez a követelményekhez a LISP nyelv természetes eszközként illeszkedik.

Boér Gyula—Dóra Gyula—
Fenyő László—Seres Attila:
Az IBM PC-k belső felépítése
(Budapest, 1989. LSI ATSZ,
393 oldal. Ára: 399,— Ft)

A kötet azoknak a szakembereknek készült, akik szeretnének részletesebben megismerkedni az IBM PC-k belső felépítésével, működésével. Ide tartozik az XT, AT, PCjr, az eredeti PC, az XT286 és valamennyi, ezekkel kompatibilis gép. A könyv a két legelterjedtebb típussal, az érdeklődésre leginkább számot tartó XT-vel és AT-vel foglalkozik elsősorban, de utalások találhatók más típusokra is.

Igaz, hogy bárki gond nélkül használhatja a számítógépét anélkül, hogy bármit tudna a belső felépítéséről, a hardverről, az egységek egymás közötti kommunikációjáról. Azonban a gép részletesebb ismerete sokat segíthet abban, hogy a felhasználó jobban kihasználja a PC-k belső lehetőségeit (például DMA-vezérlés, számlálók/időzítők használata, megszakításkezelés stb.).

A PC-alkalmazók között egyre nő azoknak a száma, akik a gépeket valamilyen nagyobb rendszerhez illesztik, vagy a gép bővítésével alakítanak ki mérésadatgyűjtő rendszereket, ipari szabályozóberendezéseket. Az ilyen jellegű feladatok rendszerint az XT vagy az AT bekiviteli csatornájára csatlakoztatható új modulok, adapterkártyák kifejlesztését is igénylik. A könyv részletesen ismerteti a be-kiviteli csatorna felépítését, jelforgalmát és azokat az általános tervezési szempontokat, amelyeket egy be-kiviteli bővítőmodul tervezésekor figyelembe kell venni.

A kötet azoknak a felhasználóknak is szól, akik egyszerűen kíváncsiságból, vagy azért gyűjtenek hardverismereteket, hogy jobban megismerjék azt az eszközt, amivel dolgoznak.

Berkes Jenő—Gonda László—Szabó Károly—Verebély Attila:
Adatátvitel számítógép-felhasználóknak
(Távközlési abc)
(Budapest, 1989. IPIK, 350 oldal.
Ára: 395,— Ft)

Könyvkiadásunkban nagyon hiányzik az a közérthető formában írt adatátviteli bevezető alapmunka, amely a mai számítógép-felhasználó számára megmagyarázza mind az elveket, mind a gyakorlati ismereteket (matematikai kifejtés és számítási módszerek mellőzésével). Ez a könyv kézikönyvszerűen adja meg az adatokat, és gyakorlati útmutatást nyújt jól rendezett és rendszerezett formában.

Az eddig megjelent művek erősen elméleti jellegűek, magas szintű matematikával dolgoznak, elsősorban a csomagkapcsolással foglalkoznak, kiemelten csak a telematikai szolgálatokat ismertetik. E hiánypótló adatátviteli szakkönyv nyelvét tekintve alapfokú, azonban szinte minden számítógép-felhasználónak hasznos, olykor nélkülözhetetlen ismereteket közöl.

A számítógépek robbanásszerű elterjedése mellett lényegében a távközlési kultúra fogja a jövőben meghatározni a hazai számítógéphasználat kulturáltságát is.

A kiadvány a magyar viszonyok figyelembevételével készült, tárgyköre kiterjed a távfeldolgozás, az egyszerű adatátviteli hálózatok és a komplex szolgálatok fogalmaira, elveire, eszközeire, eljárásaira, vagyis mindarra, amire a számítógép felhasználóinak a gyakorlatban szükségük van.

Kontrasztok

Lengyelországban éles ellentétet lehet észrevenni a számítógépek, illetve a közszükségleti cikkek kínálatában. Ez utóbbiak körében — a hústól a tejig — napi gond az elemi szükségletek kielégítése, számítógépet ezzel szemben bármikor kapni lehet. Az óriási kínálat elsősorban annak köszönhető, hogy a szédületes profitok nyomán ugrásszerűen megemelkedett az ebben a piaci szegmensben tevékenykedő magánvállalkozások száma. Ma már mintegy négyszáz ilyen cég működik, amelyek zöme az utóbbi három évben alakult.

Tíz óra alatt fordítja a Bibliát

A hollandiai BSO cégnél kidolgozás alatt álló, az elmúlt évben már hazánkban is bemutatott DLT projektnek az elkészült első rendszere angolról franciára fordít, az eszperantó, mint közvetítő nyelv közbeiktatásával. Ennek ismeretében, mintegy ehhez kapcsolódva nyújtott kétéves ösztöndíjat a Soros Alapítvány Mohai Lajosnak, hogy eszperantóról magyarra fordító programot készítsen. E program első változata ez év közepén készült el. Egy IBM PC/AT-n mért fordítási sebessége óránként 70 ezer szó, tehát a teljes Biblia fordítását 10 óra alatt végzi el. A rendszer ősszel nyilvános fordításon mutatkozik be, elsősorban abból a célból, hogy a szerző a rendszer fejlesztéséhez további támogatókat találjon.

Néhány, számítógéppel magyarra fordított mondat (amint látszik, „nyelvbotlás" azért akad még)

La arbohakisto elhakis la grandaJN kaj belaJN arboJN.
A favágó kihasította a nagy és szép fákat.

La patro AKUZIS sian filinoN PRO dibocca vivo.
Az apa vádolta saját leányáT kicsapongó életTEL.

Regu super LA BIRDOJ de la ccielo, KIUJ ESTU manggajjo por vi.
Kormányozzon az ég madarai fölött, kik legyenek étel érted.

La servisto alportis plej bonan bovidon de nia bovaro.
A szolga odahozta a gulyánk legjobb borjúját.

Via domo estas bela, sed MIA AMIKO, KIU estis cci tie, HAVAS pli belan domon.
A házad (van) szép, de a barátomnak, ki volt itt, van szépebb háza.

Trias

A háromdimenziós grafika hazai neves művelője, Kádár Edit a komoly hazai és nemzetközi piaci sikereket elért Gratis után az idén egy új, háromdimenziós grafikus rendszerrel, a Triasszal rukkolt ki. Ennek alapkoncepciója egy olyan általános célú grafikus tervezőrendszer kialakítása, amely az egyes alkalmazói igényeknek megfelelően egyszerűen adaptálható, rugalmasan kezelhető és bővíthető.

A Trias egy szoftverben három funkciót egyesít: 1. hagyományos CAD-funkciók háromdimenziós modellezéshez, szerkesztési feladatok, műszaki rajzok előállítása, módosítása, archiválása, tervtárak kialakítása; 2. egyfelől szövegszerkesztő egy grafikus adatbázis átalakítá-

sára, másfelől grafikus editor adatbázisban történő szövegmódosításhoz; 3. fejlesztő programcsomag CAD-alkalmazásokat készítő szakemberek számára egy tömör, hatékony grafikus interaktív nyelv támogatásával.

A Trias 3D CAD Stúdió által forgalmazott szoftver piaci sikere kedvező: megjelenésekor azonnal megkezdődött a forgalmazása a nyugati piacokon is.

Hova megy a működőtőke?

A japán Fujitsu, a világ hatodik legnagyobb félvezetőgyártó csoportja (éves forgalma 2,4 milliárd dollár), 400 millió dolláros beruházással lapkagyártó üzemet létesít Északkelet-Angliában. A döntés nagy jelentőségű a brit gazdaság számára, annál is inkább, mert néhány éve a legnagyobb japán félvezetőgyártó Nippon Electric Companynak (NEC), amelynek tavalyi forgalma elérte a 4,5 milliárd dollárt, ugyancsak lapkagyára működik Skóciában. Tony Newton kereskedelmi és ipari miniszter szerint a mostani beruházás új lendületet ad az angliai elektronikai iparnak, s az export bővülésével, valamint az import csökkenésével javítja majd a kereskedelmi mérleget.

Nagy-Britanniában eddig kereken száz japán vállalat mintegy 1,5 milliárd font összegű működőtőke-beruházást hajtott végre, s ezzel 25 ezer új munkahelyet teremtett.

Számítógépes város

2000-re a Tokió melletti Tiba prefektúrán megalakul a világ első számítógépes városa. A tokiói egyetem professzorának, Ken Szakamurának a tervei alapján megvalósuló település létrehozásába 130 cég kapcsolódott be, köztük olyan világszerte ismert nagyvállalatok, mint a Fuji, a Sony és az IBM, amelyek állják a nagyszabású beruházás mintegy 100 milliárd jenre rúgó költségeit.

Hogyan zajlik majd az élet a számítógépes városban? Az épületek világítását például számítógép szabályozza, attól függően, hogy nappal, éjszaka, pirkadat vagy szürkület köszönt a településre. A háztartási készülékek és szórakoztató elektronikai berendezések működtetését is a számítógép végzi. A lakóknak ezentúl nem kell olyan „fárasztó" foglalatosságokkal bíbelődniük, mint a vízcsap kinyitása. Elég, ha a mosdó fölött finoman megérintik a falat. Sőt még a vécét sem kell lehúzniuk, ezt is a számítógép végzi el helyettük.

A városkában működő irodákból eltűnnek majd a hivatalos levelek, iratok, nem lesz szükség többé titkárnőre, kézbesítőre. Az információ beszerzéséhez csak a személyi számítógéphez kell ülniük.

A számítógépes várost mindössze egy négyzetkilométernyi területre tervezik, ezer állandó lakossal és 6 ezer ingázóval. Hogy kiket csábítanak majd ide, arról egyelőre nem szólnak a hírek.

Közhasznú információk

A Művelődési Minisztérium közművelődési tanácsának kezdeményezésére még az idén létrejön az ország eddigi legszélesebb körű közhasznú információs hálózata. 130 pályázó közül 62 intézményt választottak ki, amelyek egy-egy IBM PC/XT-vel kompatibilis géphez jutottak. Ezek segítségével helyi információs rendszereket építenek ki, kezdve a mesteremberek nyilvántartásától egészen az olcsó szállások, férőhelyek regisztrálásáig. A gépeket az év végén kötik hálózatba telefonvonal segítségével. A hálózat nyitott, bárki bekapcsolódhat, ha van már számítógépe. A hálózat kiépítésének technikai megoldását a Közművelődési Információs Vállalat végzi.

Emeletes lapkák

Sokféle alkalmazási területen fontos követelmény a rendkívül gyors számítógépek alkalmazása a lehető legkisebb térfogattal és tömeggel, például űrjárműveken, radarberendezésekben stb. Az egyes elemek működési sebességét a VLSI-technika egy lapkán belül lényegesen megnöveli, de ezt korlátozza az ezeket összekötő vezetékekben a terjedési sebesség. Az egyes ilyen elemeket egy síkban elrendezve a terjedési idő többszöröse lehet az egyes elemek működési idejének.

Kézenfekvő megoldásként kínálkozott a háromdimenziós elrendezés: a VLSI-technikával készült lapkákat egymás fölé elhelyezve, ezek összeköttetései a lapok síkjára merőlegesek.

A Cray 2 szuperszámítógépben a morzsákat 10,5 × 20,5 centiméteres lapokon képezték ki, ezeket nyolcasával sűrűn egymás fölött helyezték el. Az egymás közötti néhány ezer összeköttetés így lényegesen megrövidült, ezeket azonban utólag kellett elkészíteni. A nagy elemsűrűség következtében a rendszert semleges kémhatású folyadékkal kell hűteni.

Távcsevegés

A fejlett tőkés országokban sok, agyvérzéstől vagy izomsorvadástól mozgáskorlátozott gyerek használ számítógépet, gyakran átalakított billentyűzettel vagy kapcsolókkal, hogy szüleivel vagy tanáraival kommunikáljon. Tom Holloway számítógéphálózat-szakértőt pedig az IBM azért vezényelte az egyik közép-londoni technikumba, hogy tanulmányozza a mozgáskorlátozottak számítógépes igényeit. A fenti két tény összekapcsolásával, az ő ötleteként született meg a Chatback (szabad fordításban viszontcsevegés) nevű, a mozgáskorlátozottak számára létrehozott hálózat. Ausztrália, Alaszka, Kanada, Új-Zéland — a gyerekek levelezőtársi kapcsolatot létesítenek egymással, de az Egyesült Királyságban is rendszeresen beszélgetnek egymással. Popzenekarokról, fociról, mi a kedvenc tévésorozatod? — elektronikus levelek száguldoznak ide-oda. A nagy távolságok operatív áthidalását ezek a gyerekek különösen értékelik, valósággal kiszabadulnak az állapotuk okozta fogságból.

Forradalom

A franciaországi Bourges-ban júniusban nemzetközi elektroakusztikai és számítógépzenei fesztivált rendeztek. A rendezőségtől Patachich Iván elektroakusztikus mű komponálására kapott megbízást, a francia forradalom 200. évfordulója alkalmából. Az 1789 (Liberal Fra) című kompozíció elkészült, a hangszalagra és ütőhangszerekre írt mű nagy sikert aratott.

Akupunktúra

Az idén jelent meg hazánkban az első, hazai fejlesztésű akupunktúrás orvosi diagnosztikai és terápiás program. Ez többek között tartalmaz egy részletes ajánlatot a WHO által javasolt 52 leggyakoribb betegség kezelésére. Tartalmazza továbbá a csatornák, pontok egzakt lokalizációját, a tű beszúrási technikáját, a pont beidegzését, valamint egyéb szükséges megjegyzéseket, továbbá a fülakupunktúra legfontosabb gyógyítóelemeit.

A Data Manager Kisszövetkezet megbízásából a szerzők — dr. Danis György és Kiss Gábor — már dolgoznak a program legújabb verzióján, amelynél a csatornák és pontok elhelyezkedését már rajzok segítségével is nyomon lehet követni. Ezenkívül a terápiás részt gyógynövény-alkalmazással is kiegészítik.

Korhely

Helyesírás-ellenőrző programot fejlesztett ki a SZÜV nyíregyházi számítóközpontja. A Korhely nevű program megadott szövegállományból egyenként olvasva a szavakat, rákérdez helyességükre, és az adott válasznak megfelelően vagy egy szótárba teszi, vagy lehetőséget ad a szó javítására. Ha az adott szó a szótárban megtalálható, azt helyesnek feltételezi. A szoftver bármilyen IBM PC/XT-vel kompatibilis gépen szerkesztett szöveg ellenőrzésére alkalmas, de a Venturával is együttműködik. Alkalmassá tették a Korhelyt a Commodore 64 — Robotron — Deltex rendszerrel való alkalmazásra is.

Az Olvasó írja

Kőszegi Béla, Budapest

Bár sajnos nem vagyok szakmabeli, mégis évek óta nagy érdeklődéssel olvasom (és gyűjtöm) lapjukat. Munkahelyemen (vegyész szabadalmi ügyvivő vagyok) a „gép ész" egy Terta TAP—34 formájában jelent meg. „Összeismerkedésünk" után szerény képességeihez mérten hasznos munkatársunkká vált és maradt egészen a legutóbbi időkig. A „pécéző" korszak 1985-ben köszöntött be, és tart mind a mai napig, igen megkönnyítve munkánkat.

Tudom, mindez érdektelen az önök számára, de talán magyarázza azt, hogy kívülállóként miért érdeklődöm a PC-k iránt. Mindezek miatt persze nem fogtam volna „gépet", hogy levelemmel zaklassam önöket. Ami írásra ösztökélt, az a számomra oly rég várt irányváltás, amit a pécézés térnyerése mutat a lap hasábjain.

Töredelmesen be kell vallanom, hogy a tavalyi VÁM—PÍR-os számuk után nem hagyott nyugodni a kérdés: ha másnak sikerült a Bécsi út túlsó végén megvásárolt elemekből összeállítani egy otthoni gépet, sikerülne-e nekem is? Levelezgetések, árajánlat-böngészések után az anyagi fedezet — két személy turistaellátmánya + némi devizaszámla — biztosítottnak látszott egy monokróm XT-re. Ez év tavaszán belevágtam, és áprilisban (egy „Bánhida, kapaszkodj!" felkiáltás után) elrévedő szemmel figyeltem, amint lefut a RAM-teszt a gépben. A létrejött konfiguráció végül is egy 640 kbájt RAM-ot tartalmazó turbó alaplapból, egy Super multi I/O, egy Auto switching dual graphics display and printer adapterből és egy winchester-meghajtó kártyából áll, egy 360 kbájtos floppyval és 20 Mbájtos winchesterrel. Megjelenítőként egy monokróm Thomson monitor szolgál. Mit mondjak? A cikk írójának igaza volt, és köszönetemet fejezem ki a cikkért. Amíg a hazai árak ennyire valószerűtlenek, a géphez jutásnak ez egy jól járható útját jelenti.

S ha már levelemmel zaklattam önöket, dupla vagy semmi alapon szeretném segítségüket kérni. Gondolom, problémámmal nem állok egyedül. A monitorral, illetve a monitor és a meghajtó kártya összekapcsolásával kapcsolatos a problémám. E két elem összekapcsolását vagy koax kábellel RC csatlakozókon át, vagy többeres csatlakozó kábellel 9 pólusú Canon csatlakozókon keresztül végzik. Barátságosabb az a kártya, amelyen mindkét csatlakozó megtalálható, míg vagylagos esetben a felhasználó kezét a gyártó megköti. Az én esetem olyan szempontból szerencsés, hogy mind a kártyán, mind a monitoron a 9 pólusú Canon csatlakozó található. Az egyes csatlakozópontok mibenléte is kiderül a gépkönyvekből. Segítségüket ahhoz kérem, hogy az egyes jelek mibenlétét — a nevükön túlmenően funkciójukat is — ismerjessék. Számomra a nehézséget az okozza, hogy a kártya kiépített monitorcsatlakozója csak mono (alfa/numerikus) üzemmódban „él". A szoftverből kapcsolható, illetve rövidzár-dugókkal (jumper) beállítható színes/grafikus üzemmódban a JP1 tüskesorra adja ki a kártya a képet, és erre kellene egy második Canon csatlakozót kötni. A tüskesor egyes tüskéire táblázatban közlik a rajtuk megjelenő jel nevét (Ground, +Red, +Green, +Blue, +Intensity, +Video, +H. Sync., +V. Sync.), de e jelek pontos ismerete nélkül nem tudom eldönteni, hogy ezek közül a monokróm monitor számára melyikeket kössem a Canon alj egyes csapjaira? A Ground és a két szinkronjelet nem probléma bekötni, de azt már nem tudom eldönteni, hogy a monokróm monitor a képet például a MODE BW80 vagy a MODE BW40 DOS-parancs után melyik többi jelből képes felépíteni. Tudom, a problémám elég primitív, de megoldása nélkül a gépet színes/grafikus üzemmódban nem tudom használni. Sajnos a lap monitorokkal foglalkozó egyik korábbi számában sem találtam erre felhasználható útmutatást. A hardverrel foglalkozó, számomra hozzáférhető szakirodalom (például a Műszertechnika XT-ről szóló hardverkönyve) a kérdést olyan magas szinten tárgyalja, hogy azon sem tudok elindulni.

Kérem, ha a konkrét megoldást bármi okból nem is tudják megadni, legalább a megfelelő (nálunk hozzáférhető) szakirodalmat nevezzék meg.

Örömmel láttam a legutóbbi számban a PC-hez építhető nyolc be- és kimenetet tartalmazó kártya leírását. Gondolom, sokan megépítik majd. Bízom benne, hogy továbbiak is fogják követni. Jó volt a sorozat a legújabb kártyacsodákról, de a hazai ismert elmaradottságunk mellett talán még a kommersz kártyák részletes ismertetéséből is sokat profitálhatnának az olvasók, főleg, ha gépépítési terveik vannak.

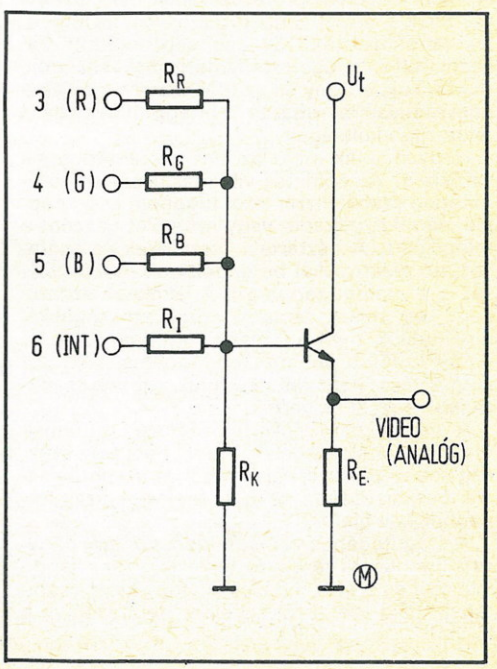
*

Észrevételeit örömmel fogadtuk. Ami a problémáját illeti, hardverszakemberekkel konzultáltunk, és arra a megállapításra jutottunk, hogy elméleti megoldás létezik, de a konkrét konfiguráció tulajdonságainak ismerete nélkül egyértelmű választ nem tudunk adni. A Canon csatlakozó 9 pólusú kiosztása színes monitorhoz (kártyához), illetve monokrómhoz a következőképpen alakul:

	Színes		Monokróm
1	GND	=	GND
2	GND	=	GND
3	RED	≠	NH*
4	GREEN	≠	NH*
5	BLUE	≠	NH*
6	INTENZ.	=	INTENZ.
7	NH*	≠	VIDEO
8	HOR.S.	=	HOR.S.
9	VERT.S.	=	VERT.S.

* *Az adott kivezetést nem használja.*

Egy elméleti megoldást mutatunk be az ábrán, amely a monokróm képernyőn az egyes színes jelek súlyozott megjelenítését teszi lehetővé. Az ábrán sem a tranzisztor, sem az ellenállások típusát, nagyságát, sem pedig a tápfeszültség értékét nem jelöltük: az áramkör megépítése és bekötése mindenképpen szakember közreműködését igényli, aki a konkrét kapcsolási rajzok alapján méretezi az áramkör elemeit. Az egytranzisztoros súlyozó áramkör bemenetére kell kötni a Canon csatlakozó 3, 4, 5 és 6 pólusait (R, G, B, Int), a kimenet az analóg video bemenetre kerül. Ha ilyen kimenet nincs a monitoron kívül, a TTL bemenetet kikerülve, a monitoron belül kell az analóg „bemenetre" kötni.



... a legyőzött sárkányokat fejeik száma,
a mátrixnyomtatókat
a fejben lévő tűk száma
alapján értékelhetjük.

Nyomtatók apróban című cikkünk
a 36. oldalon olvasható.

Sci-L

PROFESSZIONÁLIS
SZÁMÍTÓGÉPEK
PERIFÉRIÁJA

PANASONIC KX—P 1540 MÁTRIXNYOMTATÓ

98 000,— Ft + ÁFA
(18 havi garanciával)

- 24 betűs, kétirányú nyomtatású, terminálhoz is kapcsolható
- nyomtatási sebesség: 240 karakter/s
- 3 alap- és 6 opcionális karakterkészlet
- grafikus üzemmódban felhasználói diagramok, grafikonok, illusztrációk
- 96 ASCII karakterkészlet hagyományos vagy dőlt betűkkel
- Centronics párhuzamos és RS232C soros illesztő
- belső 13,5 kbájtos szabvány puffer
- 3 parancskészlet: 1. EPSON LQ—1500™
  2. IBM PROPINTER™
  3. DIABLO®

TOVÁBBI TERMÉKEINKRŐL KÉRJE ÁRJEGYZÉKÜNKET

SCI-L Számítástechnikai Informatikai Fejlesztő Leányvállalat Kereskedelmi Iroda
Budapest, Iskola u. 10. 1011
Telefon: 154-069, 350-180/180, 181, 182, 183, 184
Telex: 22-4599
Telefax: 35-39-15