ENTER FACE

Immáron több mint két éve, hogy az Enterprise számítógép megjelent a hazai üzletek polcain. Önnek, aki – reméljük – elégedett tulajdonosa ennek a csodálatos masinának, nem kell részletesebben ecsetelnünk számos jó tulajdonságát, kiemelkedő képességeit.

Ennek ellenére nem mondható el, hogy rangjához méltóan foglalkoztak volna vele. Bizonyítja ezt többek között az is, hogy szinte alig van róla szakirodalom, s sajnos a meglévők is sok pontatlanságot tartalmaznak.

Ezen próbál segíteni az 'a' STUDIO most induló lapja, az

ENTER-FACE

amelynek ez az

ELSŐ SZÁMA

'a' STUDIO

Tartalom

'a' STUDIO – ENTER-FACE

Havilap ENTERPRISE-tulajdonosoknak

Kiadja az 'a' STUDIO
Felelős kiadó: Kopácsi Vilmos
A szerkesztőség vezetője: Sipos Attila

Scan: gafz, 2006

Engedélyszám: 581/1989
ISSN 0865-3240

MULTICOLOR GMK

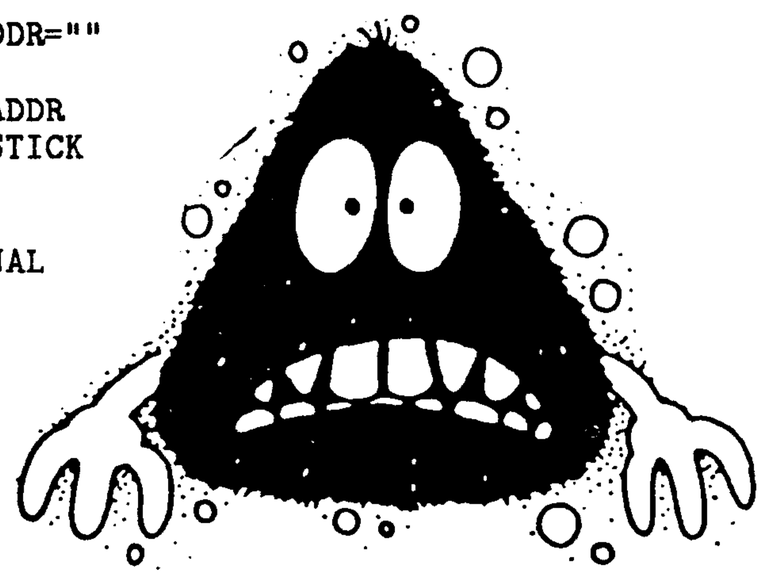
UTORG 476/89

STUDIO
Számítástechnikai Kisszövetkezet
Bp. XI. Hunyadi János u. 1.
Postacím: 1525 Bp. Pf. 105.
Tel.: 811-550

Két jó hír a BASIC-PLUS tulajdonosoknak!

AZ ELSŐ, hogy lapunkban rendszeresen vissza fogunk térni az ENTERPRISE-PLUS témájára!
Tudjuk, hogy sajnos előfordulnak benne hibák, ezek egy részét most az alább közölt kis programmal korrigáljuk. A program begépelése után a joystick problémamentesen működik. A demo-rész pedig a menükészítésben segíthet: itt ugyanis a MENU parancs paraméterezésénél arra kell ügyelnünk, hogy az első szám 1-el nagyobb legyen, mint a leghosszabb menüszöveg hossza.
A MÁSODIK, hogy azok a vásárlók, akik korrektül kitöltve beküldték a PLUS-szal kapott vevőazonosító kártyát, azok részére ebben az évben INGYEN elküldjük az ENTER-FACE idén megjelenő további számát!

```
100 PROGRAM "JOY_MENU.BAS"
110 !
120 EXT "HUN"
130 !
140 TEXT 40
150 !
160 ALLOCATE 51
170 !
180 !
190 !
200 !
210 CODE START_ADDR=""
220 ASS
230   ORG START_ADDR
240 LAB READ_JOYSTICK
250   XOR A
260   OR L
270   JR Z,INTERNAL
280   RR L
290   LD C,4
300   JR C,JOY1
310   LD C,9
320 LAB JOY1
330   LD B,5
340   LD DE,0
350 LAB CIKLUS
360   LD A,C
370   OUTT (HEX(HB5)),A
380   IN A,(HEX(HB6))
390   CPL
400   RRA
410   RR E
420   DEC C
430   DJNZ CIKLUS
440   RR E
450   RR E
460   RR E
470   EX DE,HL
480   RET
490 LAB INTERNAL
500   LD B,0
510   LD C,30
520   RST HEX(H30)
530   DEFB 16
540   LD A,D
550   LD C,L
560   LD B,9
570   RST HEX(H30)
580   DEFB 11
590   LD L,C
600   RET
610 EAS
620 !
630 DEF JOY(SORSZ)
640   LET JOY=USR(READ_JOYSTICK,SORSZ)
650 END DEF
660 !
670 !
680 !
690 !
700 !
710 MENU 14,10,25,"JOYSTICK TEST","    BELSŐ",
      "   KüLSŐ 1","   KüLSŐ 2"
720 DO
730   LET VALASZ=MENU
740 LOOP WHILE VALASZ<0
750 CLR_MENU
760 DO
770   LET B=JOY(VALASZ-1)
780   IF B<>0 THEN PRINT B,
790 LOOP
800 END
```



Komámasszony, hol az...

Részlet egy hamarosan megjelenő könyvből

[...] Az első feladat: tanítsuk meg a gépet arra, hogy egy tetszőleges szöveges változóban tárolt szövegben egy másik karaktersorozat valamennyi előfordulására ráleljen, és a pozíciókat visszaadja. Erre való a POS() függvény, amely elvégzi ezt a feladatot az első előfordulásig, ill. egy másik változata egy megadott pozíciótól kezdve a következő, ugyancsak első előfordulásig keres. Ha nem talál azonosságot, 0 értéket ad vissza. A POS() függvénynek ezt a két változatát kellene egy ún. felhasználói függvénybe összeszerkeszteni, hogy az az előbb definiált kívánalomnak megfelelően működjön.

Kezdjünk neki:

```
10 DEF keres(forras$,keresett$)
```

A forras$-ban keresem a keresett$-t. De ha sokszor van benne, akkor azt a sok mutatót hová tegyem? Egy függvénynek akármennyi bemenő adata (változója, paramétere) lehet, de csak egy kimenete, egy eredménye. Fejlettebb nyelvekben egy valamilyen szempontból összetartozó adatok egy-egy csoprtját definiálhatom listaként, halmazként, rekordként stb. Ott egy függvénynek lehet egy olyan halmaz az eredménye, amelynek az elemei például egy szöveg különböző pontjaira mutatnak. Vigyázni kell azonban a típuskeveredésre, ami a BASIC-ben annyit tesz kb., hogy nem adhatok össze egy sztringet és egy lebegőpontos számot. A gépi kódban programozók (ASSEMBLY-ben) nem sokat törődnek az efféle világnézeti kérdésekkel. Megküzdenek viszont egy-egy bonyolultabb algoritmus géprevitelével. Hát semmi sem tökéletes ezen a világon...

Mi viszont írjunk egy eljárást. Egy olyan eljárást, amelyik felismeri, hogy egy rajta kívülálló tömbbe kell majd elhelyeznie az általa megtalált mutatókat. Az első sor tehát máris módosul:

```
10 DEF keres(forras$,keresett$,REF tomb)
```

Meg kell nézni, mekkora ez a tömb, amit kívülről kapunk!

```
20 teteje=UBOUND(tomb):alja=LBOUND(tomb)
```

A kettő közé kell elférniük a mutatóknak. Megkeressük az első előfordulást.

```
30 mutato=POS(forras$,keresett$)
40 IF mutato=0 THEN              ! ha a mutató értéke 0, akkor
                                   nincs benne keresett$.
50 EXIT DEF                      ! Azaz nincs már itt semmi
                                   keresnivalónk, tűnjünk el!
60 ELSE                          ! ha mégis...
70 tomb(alja)=mutato:alja=alja+1  !Az első találat már célba
                                    ért. Az alja a következő
                                    rekeszre mutat.
80 DO                            ! cirkulálj barátom!
90   mutato=POS(forras$,keresett$,mutato)
```

A 90-as sorban elvben a következő előfordulást keresnénk meg, de azt hiszem ebben a formában sosem találnánk meg!

Legyen a forra$="123456___789___abc" keressük meg benne a"___" részletet

Ez láthatóan a 7-ik és a 13-ik pozíción kezdődik. Az első keresésnél a mutató értéke 7 lesz. Az újabb keresést a 7-es pozíción kezdi, ott meg is találja így aztán értéke megint 7 lesz, az idők végtelenségéig.

A 90-as sor tehát:

```
90 mutato=POS(forras$,keresett$,mutato+1)
```

Itt most a +1 elég. Látni fogjuk, hogy nem minden esetben elég!

```
100   IF mutato=0 THEN EXIT DEF  ! Ugyanaz, mint az 50-es
                                   sorban
110   tomb(alja)=mutato:alja=alja+1
120   LOOP UNTIL alja>teteje      ! Itt a másik kilépési
                                   feltétel: ha elfogy a tömb, nem
                                   keresünk tovább!
```

A 120-as sorban kiiktattuk egy goromba üzenet lehetőségét, ám nem lehetünk igazán nyugodtak. Ha kisebb méretű tömböt adunk át az eljárásnak, mint az előfordulások száma, nem keresi meg mindet, csak annyit, amennyi a „zsákjába" fér, és erről nem is értesít minket. Most mindenesetre próbáljuk működőképessé tenni az eljárást, de jegyezzük fel az észrevételünket.

```
130 ! Ha tomb < találatok, nem értesít róla!
```

Van itt még valami dogunk! Zárjuk le az eljárást és listázzuk ki. A gép elrendezi a kiírást és ha valahol valamit (ciklus, IF-szerkezet, stb.) elfelejtettünk lezárni, az eljárás alja nem fog visszakanyarodni a sorszámhoz.

```
 10 DEF KERES(FORRAS$,KERESETT$,REF TOMB)
 20   LET TETEJE=UBOUND(TOMB):LET ALJA=LBOUND(TOMB)
 30   LET MUTATO=POS(FORRAS$,KERESETT$)
 40   IF MUTATO=0 THEN
 50     EXIT DEF
 60   ELSE
 70     LET TOMB(ALJA)=MUTATO:LET ALJA=ALJA+1
 80     DO
 90       LET MUTATO=POS(FORRAS$,KERESETT$,MUTATO+1)
100       IF MUTATO=0 THEN EXIT DEF
110       LET TOMB(ALJA)=MUTATO:LET ALJA=ALJA+1
120     LOOP UNTIL ALJA>TETEJE
130     ! Ha tomb < találatok, nem értesít róla!
140   END IF
150 END DEF
ok
```

Látszólag most jónak tűnik. Próbáljuk ki!

```
START
ok
a$="123456___789___abcd"
ok
CALL keres(a$,"___")
 *** Error in DEF parameter.
```

Elhamarkodtam ezt a dolgot! Hát persze, hogy hibát talál a DEF sor paraméterei között, ha egyszer már elhatároztam, hogy oda beírom annak a tömbnek a nevét, amelyikben szeretném visszakapni a mutatókat! De nincs is ilyen tömb! Hát csináljunk egyet! A fenti példában csak két előfordulás számlálható meg, tehát bőven túlméretezem, ha 10-re dimenzionálom. Ezt a programba kell elhelyezni, parancsként nem fogadja el!

```
1 DIM mutatok(10)
```

Elvben, most müködnie kell. Nézzük:

```
START
ok
a$="123456___789___abcd"
ok
CALL keres(a$,"___",mutatok)
ok
```

Nagyszerü! Nézzük meg, mit helyezett bele!

```
PRINT mutatok(1)
13
ok
```

Nono! Itt most valami hiba van! Elárulhatom, hogy most nem a program hibás, sem a gép. Bennem van a hiba. Nézzük meg, miért!

```
PRINT alja
 *** Variable not initialised.
```

Erre visszatérünk!

4 – 'a' STUDIO

```
PRINT teteje
 *** Variable not initialised.
```

Erre is.

```
PRINT LBOUND(mutatok)
0
ok
```

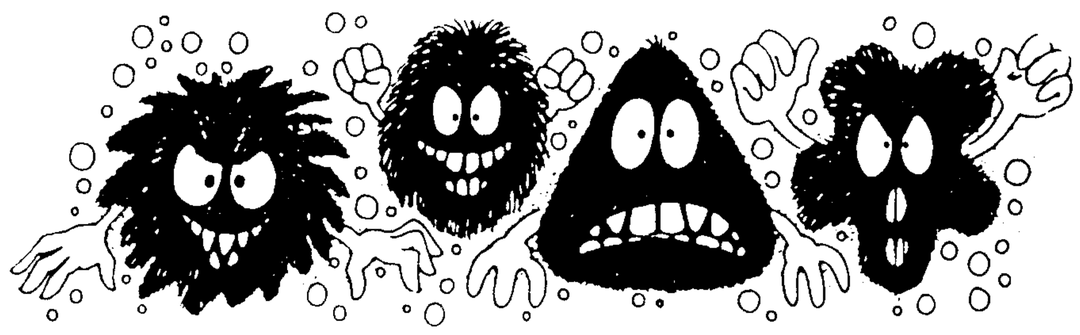
Hát ez a magyarázat! Milyen alapon feltételezem én, hogy egy tömb első eleme egy! (Más gépeknél így lehet!) De az ENTERPRISE-on, ha DIM-mel dimenzionálok tömböt, akkor a 0 az első eleme. Ha viszont NUMERIC-kel, akkor tetszés szerintire tudom beállítani az alsó és a felső indexhatárt is. Az LBOUND() és UBOUND() függvények ezt kérdezik vissza.

```
PRINT mutatok(0)
7
ok
PRINT mutatok(1)
13
ok
```

Így már stimmel! De térjünk vissza az alja és teteje változókra amit biztosan használ az eljárás, hiszen beletettük, most meg mégsem tudjuk lekérdezni a tartalmukat. Ez a két változó lokális, azaz helyi az eljáráson belül. Értékük csak addig marad meg, amíg az eljárás végrehajtás alatt van. Ha azonban az eljárás meghívása előtt, azaz rajta kívül létrehozok egy ugyanilyen nevű változót akkor abba fogja beleírni a tömb alsó és felső határának értékeit, azaz az eljárás lefutása után is lekérdezhetem. Ez a másik lehetőség, hogy eljárások és függvények kommunikáljanak a környezetükkel, azaz a globális változók használata, vagyis a környezet és az eljárás egyformán látja a változókat és mind a kettő írni is és olvasni is tudja azokat. És mi van akkor, ha én nem akarom, hogy egy eljárás vagy függvény belekotorásszon az adataimba? Arra is van lehetőség. Az előző eset, mint látható volt, nem tette igazán lokálissá az említett változókat. De ha az eljárás elején a NUMERIC kulcsszó után felsorolom őket, már nyugodtan használhatok az eljáráson (függvényen) kívül is ugyanilyen nevű változókat, nem fogja megváltoztatni azok értékét.

Próbáljuk ki!

```
2 alja=0:teteje=0
START
ok
a$="123456___789___abcd"
ok
CALL keres(a$,"___",mutatok)
ok
PRINT mutato(0)
7
PRINT alja
2
PRINT teteje
10
```



Ez működik. Az alja azért 2, mert már a következő rekeszre mutat. Folytassuk a próbát:

```
15 NUMERIC alja,teteje
START
ok
a$="123456___789___abcd"
ok
CALL keres(a$,"___",mutatok)
ok
PRINT alja
0
PRINT teteje
0
```

Amennyit a 2-es sorban adtunk neki. Ezt a sok vacakot annyiszor írtam már le, hogy akár programba is tehettem volna. Nézzük meg, hogy nézne ki:

```
3 a$="123456___789___abcd"
4 call keres(a$,"___",mutatok)
START
ok
PRINT mutatok(0)
7
PRINT mutatok(1)
13
```

Írjuk bele ezt a két printet is:

```
5 PRINT mutatok(0)
6 PRINT mutatok(1)
 START
7
13
ok
```

Most már nekem nem kell csinálnom semmit, csak megnyomnom a START gombot, és megjelenik a 7-es, meg a 13-as. Egy óra múlva már nem fogok rá emlékezni, hogy mi ez a 7-es és ez a 13-as. A következő részben ezt is megbeszéljük.

Kovács Zoltán

Egy grafikai program

```
100 PROGRAM "lines"
102 RANDOMIZE
110 NUMERIC L,M,U,V,R,A,B,C,D,CONT
120 LET BI=64
130 LET I=0:LET X=320
140 GRAPHICS HIRES 4
150 SET #101:PALETTE 0,1,2,3
160 SET LINE MODE 1
170 RANDOMIZE
180 LET X=RND(1279)
190 LET Y=RND(719)
200 LET L=RND(1279)
210 LET M=RND(719)
220 LET U=15:LET V=7:LET CONT=RND(200)+100
230 GOSUB 580
260 PLOT X,Y;L,M
270 LET A$=INKEY$
280 IF A$<>"" THEN
290   PING
300   WAIT DELAY 5
310   GOTO 650
320 END IF
330 IF X+A>1270 OR X+A<0 THEN
340   LET A=-A
350   SET #101:INK RND(3)+1
360 END IF
370 IF Y+B>710 OR Y+B<0 THEN
380   LET B=-B
390   SET #101:INK RND(3)+1
400 END IF
410 IF L+C>1270 OR L+C<0 THEN
420   LET C=-C
430   SET #101:INK RND(3)+1
440 END IF
450 IF M+D>710 OR M+D<0 THEN
460   LET D=-D
470   SET #101:INK RND(3)+1
480 END IF
490 LET X=X+A:LET Y=Y+B
500 LET L=L+C:LET M=M+D
510 LET CONT=CONT-1
520 IF CONT=1 THEN
530   PING
540   WAIT DELAY 5
550   GOTO 660
560 END IF
570 GOTO 260
580 SET #101:INK RND(3)+1
590 LET A=RND(U)-V+16
600 LET B=RND(U)-V+8
610 LET C=RND(U)-V+16
620 LET D=RND(U)-V+8
640 RETURN
650 REM ujra
660 PRINT "UJRA","VEGE"
670 DO
680   LET A$=INKEY$
690 LOOP WHILE A$=""
700 IF A$="u" THEN
710   CLEAR SCREEN
720   GOTO 140
730 END IF
740 IF A$="v" THEN GOTO 760
750 GOTO 670
760 TEXT
```

6 – 'a' STUDIO

MINER ZZZIP-PÁLYÁZAT

Az alábbi játékprogram az 'a' STUDIO ZZZIP-pályázatára érkezett be. Igen kellemes, jól megírt játék, reméljük, Nektek is tetszeni fog.
A játék lényege: egy kis emberkével végig kell járni a szobákat. Amerre elhaladsz, a padló más színűre változik. Ha mindenütt befestetted alattad az utat, valahol a szobában megjelenik egy kulcs, Ha ezt felveszed, továbbjutsz a következő terembe. 20 pálya van, s bizony nem könnyű végigjárni őket. A Shift+< gomb hatására 99 életed lesz.
A program minden különösebb probléma nélkül lefordítható a ZZZIP Compilerrel, mindössze egyetlen helyen jelez fordítási hibát. Hadd bízzuk Olvasóinkra ennek a megoldását, annál is inkább, mert ez a probléma bizony másoknál is gyakran előfordul!
Egyúttal hadd hívjuk fel a figyelmet arra, hogy a PÁLYÁZAT MÉG TART! Várjuk programjaitokat, a legjobbat le is közöljük lapunk következő számaiban!
Tehát a jelszó: ZZZIP!

A listában szereplő aláhúzott betűket az ALT billentyűvel együtt kell begépelni!

```
  1 !   ************************
  2 !   *                      **
  3 !  *   E-software 1989    **
  4 !  *  Baráth Endre        **
  5 ! *   Ajka Korányi u.22.**
  6 ! *                     **
  7 !*   H U N G A R Y !   **
  8 !************************
100 TEXT 40
101 LET JO=0
102 !POKE 49144,201 !(*protect*)
103 ENVELOPE NUMBER 1;0,100,100,2;0,-100,-100,30
110 STRING Q$*155
120 SET STATUS OFF
130 SET KEY STOP OFF: !POKE 49144,201 !(*protect*)
140 SET BORDER 1: SET #102: PALETTE 1,22,3,4
150 PRINT #102,AT 10,6: "E-software 1989...
    Baráth Endre"
160 SET #102: INK 3
170 PRINT #102,AT 3,10:"n   n n n   n nnnn nnnn "
180 PRINT #102,AT 4,10:"nn nn n nn  n n    n   n"
190 PRINT #102,AT 5,10:"n n n n n n n nnn  nnnn "
200 PRINT #102,AT 6,10:"n   n n n n n n    n  n "
210 PRINT #102,AT 7,10:"n   n n n n n n    n   n"
220 PRINT #102,AT 8,10:"n   n n n  nn nnnn n   n"
230 RESTORE 5700
240 READ Q$
250 FOR X=1 TO 175
260   PRINT AT 10,35:Q$(X:X)
270   PRINT AT 10,2:CHR$(160)
280   IF JOY(JO)=16 THEN 310
290 NEXT
300 GOTO 230
310 CLEAR TEXT:SET #102:PALETTE 1,88,89,90,91
320 PRINT #102,AT 5,15:"** MENU **":PRINT #102,AT 8,13:"*
    JOY 0   *":PRINT #102,AT 9,13:"*   JOY 1   *"
321 PRINT #102,AT 5,15:"** MENU **":PRINT #102,AT 10,13:"*
    JOY 2   *":PRINT #102,AT 11,13:"*   START   *"
330 LET PO=7:LET LIVE=3
331 CLEAR SOUND:POKE 47385+PO*16,255
332 SOUND PITCH 30,DURATION 99,ENVELOPE 1,SOURCE 0,
      SYNC 1
333 SOUND PITCH 30.1,DURATION 99,ENVELOPE 1,SOURCE 1,
      SYNC 1
334 POKE 47385+PO*16,254:WAIT 1
340 POKE 47385+PO*16,254
350 IF JOY(JO) BAND 4 AND PO<10 THEN POKE
      47385+PO*16,88:LET PO=PO+1
351 IF INKEY$="≥" THEN LET LIVE=99:SOUND PITCH 50,
      ENVELOPE 1,SOURCE 2,STYLE 30
360 IF JOY(JO) BAND 8 AND PO>7 THEN POKE
      47385+PO*16,88:LET PO=PO-1
361 IF JOY(JO)=16 THEN 371
362 IF JOY(0)=16 AND JO<>0 THEN RUN
370 GOTO 340
371 IF PO=7 THEN LET JO=0:GOTO 331
372 IF PO=8 THEN LET JO=1:GOTO 331
373 IF PO=9 THEN LET JO=2:GOTO 331
374 IF PO=10 THEN 380
380 GRAPHICS HIRES 16
381 SET PALETTE 1,1,1,1,1
390 ENVELOPE NUMBER 1;0,100,100,1;0,-100,-100,20
400 !
410 SET CHARACTER 143,24,60,86,255,171,126,36,36,102
411 SET CHARACTER 144,129,255,129,129,255,129,
      129,255,129
420 SET CHARACTER 137,0,0,56,108,116,124,56
430 SET CHARACTER 140,0,0,4,166,127,166,4
440 SET CHARACTER 138,124,68,68,68,124,16,28,16,28
450 SET CHARACTER 154,24,60,90,255,219,102,60,60,102
460 SET CHARACTER 139,0,0,16,56,124,0,255,0,255
461 SET CHARACTER ORD("q"),251,243,251,243,251,243,
      251,243,251
470 STRING X$(10,20)*1
480 SET LINE MODE 3
490 LET I=1:LET GI=-1:LET MI=0:LET TAI=1
500 LET LEV=0:LET MOR=LIVE
510 CLEAR GRAPHICS:SET PALETTE 0,0:LET LEV=LEV+1
520 SELECT CASE LEV
530 CASE 1
540   RESTORE 3240
550 CASE 2
560   RESTORE 3350
570 CASE 3
580   RESTORE 3460
590 CASE 4
600   RESTORE 3570
610 CASE 5
620   RESTORE 3680
630 CASE 6
640   RESTORE 3790
650 CASE 7
660   RESTORE 3900
```

```
670 CASE 8
680   RESTORE 4010
690 CASE 9
700   RESTORE 4120
710 CASE 10
720   RESTORE 4230
730 CASE 11
740   RESTORE 4340
750 CASE 12
760   RESTORE 4450
770 CASE 13
780   RESTORE 4560
790 CASE 14
800   RESTORE 4670
810 CASE 15
820   RESTORE 4780
830 CASE 16
840   RESTORE 4890
850 CASE 17
860   RESTORE 5000
870 CASE 18
880   RESTORE 5110
890 CASE 19
900   RESTORE 5220
910 CASE 20
920   RESTORE 5330
930 CASE ELSE
940   GOTO 5430
950 END SELECT
960 LET VAN=0:LET FEL=0:LET BBB=0
961 SET PALETTE 1,4,4,4,4,4,4,4
962 SET #102:PALETTE 1,253
963 PRINT #102,AT 2,15:"LIFE:";MOR
970 FOR X=1 TO 10
980   READ A$
990   SET INK 2:SET CHARACTER 128,255,255,255,255,255,
        255,255,255,255
1000  SET CHARACTER 159,239,239,0,253,253,253,253,0,239
1020  PRINT #101,AT X,1:A$
1030  SET INK 4:SET CHARACTER 128,255,234,132,30,21,1
1040  SET CHARACTER 159,16,16,255,2,2,2,2,255,16
1060  PRINT #101,AT X,1:A$
1070  FOR Y=1 TO 20
1080    LET X$(X,Y)=A$(Y)
1081    IF X$(X,Y)=" " THEN 1140
1090    IF X$(X,Y)="ü" THEN LET VAN=VAN+1:GOTO 1140
1092    IF X$(X,Y)="q" THEN LET TA1=X:LET TA2=Y:GOTO 1140
1100    IF X$(X,Y)="j" THEN LET K1=X:LET K2=Y:
          LET X$(X,Y)=" ":GOTO 1140
1110    IF X$(X,Y)="o" THEN LET MA=X:LET M=Y:
          LET X$(X,Y)=" ":GOTO 1140
1120    IF X$(X,Y)="i" THEN LET GA=X:LET G=Y:
          LET X$(X,Y)=" ":GOTO 1140
1130    IF X$(X,Y)="l" THEN LET NY=X:LET N=Y:
          LET X$(X,Y)=" "
1140  NEXT
1150 NEXT
1160 SET INK 6:PRINT #101,AT K1,K2:"j":PRINT #101,
       AT GA,G:"i":PRINT #101,AT NY,N:"l":PRINT #101,
       AT MA,M:"o":PRINT #101,AT TA1,TA2:"q":LET TAI=1
1170 LET A=3:LET S=19:SET LINE MODE 3:SET INK 4:
       PRINT #101,AT MA,M:"o":
       PRINT #101,AT GA,G:"i":PRINT #101,AT NY,N:"l":
       CALL EMB
1180 CALL EMB:LET A=3:LET S=19:CALL EMB
1190 SET PALETTE 192,3,5,255,11,251,21
1200 IF GA=A AND G=S THEN GOTO 3000
1210 IF NY=A AND N=S THEN GOTO 3000
1220 IF MA=A AND M=S THEN GOTO 3000
1230 IF JOY(JO) BAND 4 AND X$(A+1,S)="p" THEN CALL EMB:
       LET A=A+1:CALL EMB:GOTO 1300
1240 IF JOY(JO) BAND 8 AND X$(A-1,S)="p" THEN CALL EMB:
       LET A=A-1:CALL EMB:GOTO 1300
1250 IF JOY(JO) BAND 2 AND X$(A,S-1)<>"o" THEN CALL EMB
       LET S=S-1:CALL EMB:GOTO 1300
1260 IF JOY(JO) BAND 1 AND X$(A,S+1)<>"o" THEN CALL EMB
       LET S=S+1:CALL EMB:GOTO 1300
1270 FOR VA=1 TO 10
1280   IF JOY(JO)<>0 THEN EXIT FOR
1290 NEXT
1300 IF X$(A+1,S)=" " THEN CALL EMB:LET A=A+1:CALL EMB
1310 IF X$(A+1,S)="k" THEN GOTO 3000
1320 IF BBB=1 AND K1=A AND K2=S THEN GOTO 510
1330 IF X$(A+1,S)="ü" THEN LET X$(A+1,S)="-":
       SET INK 15:PRINT #101,AT A+1,S:"ü":
       SET INK 4:LET FEL=FEL+1:CALL VIZS
1340 IF JOY(JO) BAND 16 AND X$(A+1,S)<>" " THEN GOTO 14
1350 IF GA=A AND G=S THEN GOTO 3000
1360 IF NY=A AND N=S THEN GOTO 3000
1380 IF MA=A AND M=S THEN GOTO 3000
1381 IF TA1>A-1 AND TA2=S THEN GOTO 3000
1390 CALL MANO
1400 CALL GOL
1410 CALL NYI
1415 CALL TAPOS
1420 GOTO 1200
1430 DEF EMB
1440   PRINT #101,AT A,S:"z"
1450 END DEF
1460 !!! UGRAS
1470 IF MA=A AND M=S THEN GOTO 3000
1480 IF X$(A-1,S)=" " THEN CALL EMB:LET A=A-1:CALL EMB
1490 IF GA=A AND G=S THEN GOTO 3000
1500 IF NY=A AND N=S THEN GOTO 3000
1510 IF MA=A AND M=S THEN GOTO 3000
1520 IF X$(A-1,S)=" " THEN CALL EMB:LET A=A-1:CALL EMB
1521 GOTO 1350
1530 !! END UGRAS
1540 DEF MANO
1550   PRINT #101,AT MA,M:"o"
1560   LET M=M+I
1570   IF X$(MA+1,M+I)=" " THEN LET I=-I
1580   PRINT #101,AT MA,M:"o"
1590 END DEF
1600 DEF GOL
1610   PRINT #101,AT GA,G:"c"
1620   LET GA=GA+GI
1630   IF X$(GA+GI,G)<>" " THEN LET GI=-GI:SOUND
         ENVELOPE 1,SOURCE 3,DURATION 1,INTERRUPT
1640   PRINT #101,AT GA,G:"c"
1650 END DEF
1660 DEF NYI
1670   PRINT #101,AT NY,N:"l"
1680   LET N=N+1
1690   IF N=20 THEN LET N=2:SOUND STYLE 30,ENVELOPE 1,
         PITCH 60,DURATION 60,SOURCE 1
1700   PRINT #101,AT NY,N:"l"
1710 END DEF
1800 DEF TAPOS
1810   PRINT #101,AT TA1,TA2:"q"
```

8 – 'a' STUDIO

```
1820   LET TA1=TA1+TAI
1830   IF X$(TA1+TAI,TA2)<>" " THEN LET TAI=-TAI:
         PRINT #101,AT TA1,TA2:"q":EXIT DEF
1840   !  PRINT #101,AT ta1,ta1:"q"
1850 END DEF
3000 !!! SZIVas
3010 SOUND DURATION 10,STYLE 30,ENVELOPE 1
3020 SOUND DURATION 10,STYLE 30,ENVELOPE 1
3030 SOUND DURATION 10,STYLE 30,ENVELOPE 1
3040 SOUND DURATION 10,STYLE 30,ENVELOPE 1
3050 LET MI=0
3060 FOR OO=1 TO 5
3070   PRINT #101,AT A,S:"z"
3080   PRINT #101,AT A,S:"z"
3090 NEXT
3100 LET MOR=MOR-1
3110 PRINT #102,AT 2,15:"LIFE:";MOR
3120 IF MOR=0 THEN RUN
3130 GOTO 1180
3140 !!! END SZIVAS
3150 !1
3160 DEF VIZS
3170   IF FEL<VAN THEN EXIT DEF
3180   PING
3190   LET BBB=1
3200   SET INK 1
3210   PRINT #101,AT K1,K2:"j"
3220   SET INK 4
3230 END DEF
3240 DATA "nnnnnnnnnnnnnnnnnnnn"
3250 DATA "np   l  q          n"
3260 DATA "npüüüüüüü n   üüü   n"
3270 DATA "np       n pü   ünnn"
3280 DATA "np     p n p     nnn"
3290 DATA "nnniüüüpnnnp o   jnn"
3300 DATA "nn     p   püüüüün n"
3310 DATA "n   üüüüüüüp       n"
3320 DATA "n          p       n"
3330 DATA "nüüüüüüüüüüüüüüüüüün"
3340 !2
3350 DATA "nnnnnnnnnnnnnnnnnnnn"
3360 DATA "n l                n"
3370 DATA "n l    l j    i    n"
3380 DATA "n    üüüüüpüüüüü    n"
3390 DATA "n   ü    p    ü    n"
3400 DATA "n        p         n"
3410 DATA "nüü    üüüüüü    üüün"
3420 DATA "nüüü  ü     ü  üüüün"
3430 DATA "nüüüüü          üüüüün"
3440 DATA "nnnnnnnnnnnnnnnnnnnn"
3450 !3
3460 DATA "nnnnnnnnnnnnnnnnnnnn"
3470 DATA "n                l  n"
3480 DATA "n püü       üü      n"
3490 DATA "n p  üü  üü    p   n"
3500 DATA "n p    üü      püüün"
3510 DATA "n p      i     p   n"
3520 DATA "n p     p    o p   n"
3530 DATA "n p ü  üpü  üüüü   n"
3540 DATA "n p l   p        j n"
3550 DATA "nüüüükkküükkküüüüüün"
3560 !4
3570 DATA "nnnnnnnnnnnnnnnnnnnn"
3580 DATA "n               l   n"
3590 DATA "n     l            n"
3600 DATA "nüüüüüüüün          n"
3610 DATA "n n     n o      ü n"
3620 DATA "n nnnn  nüünnn     n"
3630 DATA "n n   i       n  n ün"
3640 DATA "njnüüü   üüüün  n  n"
3650 DATA "n                ünü n"
3660 DATA "nüüüüüüüüüüüüüüüünüün"
3670 !5
3680 DATA "nnnnnnnnnnnnnnnnnnnn"
3690 DATA "nl   l          p   n"
3700 DATA "n    üüüüüüü  ü p   n"
3710 DATA "n   üi       jo p   n"
3720 DATA "nn     nnn  nnnp nnn"
3730 DATA "n  nn ü n  ü n p   n"
3740 DATA "n  nn   n    n p   n"
3750 DATA "n ün ü ün ü nnnp   n"
3760 DATA "n  n    n   n np   n"
3770 DATA "nüünkküknüüünnnnüküün"
3780 !6
3790 DATA "nnnnnnnnnnnnnnnnnnnn"
3800 DATA "n        l         n"
3810 DATA "n     p      o  j  n"
3820 DATA "n üüüüpüü i  üüüü  n"
3830 DATA "n     p          n"
3840 DATA "n üüüüpüü üüü    üün"
3850 DATA "n     p   nnnü  ünnn"
3860 DATA "n üüüüpüü    nüün  n"
3870 DATA "n    lp       nn   n"
3880 DATA "nnnnnnnnnnnnnnnnnnnn"
3890 !7
3900 DATA "nnnnnnnnnnnnnnnnnnnn"
3910 DATA "nl                 n"
3920 DATA "n                j n"
3930 DATA "nn  üüüüüüüüüüüüüüün"
3940 DATA "n n    l     l     n"
3950 DATA "n  n  p  o  p      n"
3960 DATA "n   n püüüüüp  üüüün"
3970 DATA "n  i  p     p  nnnnn"
3980 DATA "n     p     p      n"
3990 DATA "nnnüüünnnnnnnnüüüüüün"
4000 !8
4010 DATA "nnnnnnnnnnnnnnnnnnnn"
4020 DATA "nl                p  n"
4030 DATA "n      o          püün"
4040 DATA "n  üüüüüüüün    üüp  n"
4050 DATA "n ünnnnnnnn      püün"
4060 DATA "nün        nüüüüüp  n"
4070 DATA "nn               püün"
4080 DATA "n       i        p  n"
4090 DATA "n       l j      p  n"
4100 DATA "nüünnnnnnnnnnnnnnnnn"
4110 !9
4120 DATA "nnnnnnnnnnnnnnnnnnnn"
4130 DATA "n  l  p   p        n"
4140 DATA "n     p o p      j n"
4150 DATA "n     püüüp     üüüün"
4160 DATA "n i   p p   püü    n"
4170 DATA "n  n  p p   p      n"
4180 DATA "n     p p   p      n"
4190 DATA "nüü     p   p      n"
4200 DATA "n l                n"
4210 DATA "nkkkkkkkkkkkkkkkkkn"
4220 !10
4230 DATA "nnnnnnnnnnnnnnnnnnnn"
4240 DATA "np   l  p l  n     n"
```

```
4250 DATA "npnnn   pn   o     n"
4260 DATA "npn     pn nnnnn   n"
4270 DATA "npn nnnnpn         n"
4280 DATA "npnnn  np    n i   n"
4290 DATA "np  n  np  n n   j n"
4300 DATA "np  n   pn n     nnn"
4310 DATA "np  n   pnnnn      n"
4320 DATA "nnnünnnnnnnnnnnnnnn"
4330 !11
4340 DATA "nnnnnnnnnnnnnnnnnnn"
4350 DATA "nl  l   p          n"
4360 DATA "n üüüüüüüpüüüüüüüüü n"
4370 DATA "n       p      n   n"
4380 DATA "n üüüüüüüpüüüüüüü n nn"
4390 DATA "n       p j   npn  n"
4400 DATA "nüüüüüüüüpüüüüüünpnüin"
4410 DATA "n       p     np   n"
4420 DATA "n    o  p     np   n"
4430 DATA " nüüüüüüüüüüüüüünüüün "
4440 !12
4450 DATA "nnnnnnnnnnnnnnnnnnn"
4460 DATA "n   l              n"
4470 DATA "n       o          n"
4480 DATA "n       nnnnn      n"
4490 DATA "nüü     n   n ü  üüün"
4500 DATA "n       n i n      n"
4510 DATA "nüüü  ün   nü      n"
4520 DATA "n      nnnnnnn      n"
4530 DATA "n   l    j         n"
4540 DATA "nnnnnnnnnnnnnknnnnnn"
4550 !13
4560 DATA "nnnnnnnnnnnnnnnnnnn"
4570 DATA "n             l    n"
4580 DATA "n       l          n"
4590 DATA "n p       üpü      n"
4600 DATA "n p       p        pn"
4610 DATA "n pi nnn üpü  ünnnpn"
4620 DATA "n p   n   p     n pn"
4630 DATA "n p   n  nnn    n pn"
4640 DATA "n p jnnn  o    nnnpn"
4650 DATA "nnnnn n nnüüünn n nn"
4660 !14
4670 DATA "nnnnnnnnnnnnnnnnnnn"
4680 DATA "n               l    n"
4690 DATA "n         nn         n"
4700 DATA "n üüü   n  n   üüü n"
4710 DATA "n       n  j n      n"
4720 DATA "nüüüüü   nni  üüüüün"
4730 DATA "n    o  ünn         n"
4740 DATA "n   üüüü nn üüüü    n"
4750 DATA "n    l              n"
4760 DATA "nüüüüüüüüüüüüüüüüüün"
4770 !15
4780 DATA "nnnnnnnnnnnnnnnnnnn"
4790 DATA "n     l  l          n"
4800 DATA "n üü          nnn    n"
4810 DATA "n     üüü    n nnn  n"
4820 DATA "n  üü        n   nnnn"
4830 DATA "n    ü       n ü    n"
4840 DATA "n        üüü n   i  n"
4850 DATA "n  üü ü      nü    ün"
4860 DATA "n        n   o  n j n"
4870 DATA "nnnnnnnn nnnnnn nnnn"
4880 !16
4890 DATA "nnnnnnnnnnnnnnnnnnn"
4900 DATA "n              l    n"
4910 DATA "n                   n"
4920 DATA "nü i  o n        ü  n"
4930 DATA "n     nnnnn      ü  n"
4940 DATA "nüü  ü  n  ü  üüüüüün"
4950 DATA "n   ü   n  ü ü      n"
4960 DATA "n  ü    n   ü    j n"
4970 DATA "n   l           üüün"
4980 DATA "nnnnnnnnnnknnnnnnnnn"
4990 !17
5000 DATA "nnnnnnnnnnnnnnnnnnn"
5010 DATA "n     l             n"
5020 DATA "n                   n"
5030 DATA "n  nüü       üüü    n"
5040 DATA "n  n           o    n"
5050 DATA "n  nü soft üiüüüüüü "
5060 DATA "n  n                n"
5070 DATA "n  nüü   level 17   n"
5080 DATA "nj l                n"
5090 DATA "nnnnnnnnknnnnnnnnknn"
5100 !18
5110 DATA "nnnnnnnnnnnnnnnnnnn"
5120 DATA "n  l                n"
5130 DATA "n        p   nnn    n"
5140 DATA "n        p   npn    n"
5150 DATA "n       üpü   p     n"
5160 DATA "n    üüü p üüüp nnnn"
5170 DATA "n       üpü  jp     n"
5180 DATA "n   l    p i npn    n"
5190 DATA "n    o   p   nnn    n"
5200 DATA " üüüüüüüüüüüüü nnüüün"
5210 !19
5220 DATA "nnnnnnnnnnnnnnnnnnn"
5230 DATA "n    l           l  n"
5240 DATA "nüüüüüü             n"
5250 DATA "n     n  i  p  n n n"
5260 DATA "n  n  on   np nn    n"
5270 DATA "n   nnnnnnn pnn  n n"
5280 DATA "n           p n  n n"
5290 DATA "n           p  n nnn"
5300 DATA "n     üü    p     jn"
5310 DATA "nnüüüüünnüüüüüünnnnnn"
5320 !20
5330 DATA "nnnnnnnnnnnnnnnnnnn"
5340 DATA "n                l  n"
5350 DATA "n     üü   üü        n"
5360 DATA "n    ü  ü p  p      n"
5370 DATA "n j     ü p ip      n"
5380 DATA "nnn    ü  p  p   nnn"
5390 DATA "n      ü    o       n"
5400 DATA "n     üüüü  üü      n"
5410 DATA "n l                 n"
5420 DATA "nküüüüüüküüüküküküün"
5430 GRAPHICS HIRES 16
5440 SET PALETTE 0,1,2,3,4,5,6,7
5450 SET INK 2:PLOT 0,719,
5460 PRINT #101:"   nn  üü n    n "
5470 SET INK 1
5480 PRINT #101:"  n  nü  üüü üüi  "
5490 SET INK 2
5500 PRINT #101:"  n   üüüüüü ü üii "
5510 SET INK 3
5520 PRINT #101:"  n nnü  üü   üi  "
5530 SET INK 4
5540 PRINT #101:"   nn ü  üü   üii "
```

10 – 'a' STUDIO

```
5550 SET INK 5
5560 PRINT #101:"  k  j j oo lll   "
5570 SET INK 6
5580 PRINT #101:" k k j j o  l  l  "
5590 SET INK 7
5600 PRINT #101:" k k j j oo lll   "
5610 SET INK 1
5620 PRINT #101:" k k j j o  l l   "
5630 SET INK 2
5640 PRINT #101:"  k   joo  oo l  l  "
```

```
5650 FOR X=1 TO 200
5660   SET PALETTE 0,X,X+1,X+2,X+3,X+4,X+5,X+6
5670 NEXT
5680 IF JOY(J0)=0 THEN 5680
5690 RUN
5700 DATA "E-software programs: Pac-man.....
      Fred-1,2,3.....A kuldetes.....
      A kiraly koronaja.....TRON.....
      Entermusic.....etc..."
```

PUBLIC DOMAIN PUBLIC DOMAIN

FONTOS! Aki el akarja kerülni a lapban található listák fáradságos bepötyögését, kazettán is hozzájuthat!
HOGYAN? Ha vásárol egy gyári játékkazettát, a másik oldalára átmásoljuk a programokat.
FONTOS! S ráadásul még kedvezményt is kap a kazetta árából!
TEHÁT írjatok nekünk!

DE ha már bepötyögte a gépbe, akkor választhat az alábbi PUBLIC DOMAIN-programok között:

Academy	Alien-8	Joe Blade	Knight Lore
Antiriad	Asterix	Matchday-2	Mercenary
Aticatac	Ball Breaker	Mercenary-2	Micronaut One
Barbarian	Black Lamp	Midnight	Motos
Bobby Bearing	Buggy Boy	Movie	Nebulus
Colossus Chess	Cyclone	Nether	Night-Shade
Death Wish 3	Deviants	PSSST	Popeye
Driller	Eartlight	Rasputin	Rebellstar
Elite	Exolon	Renegade	Robin of the Wood
Exploding Fist I, III	720°	Robot Messiah	Saboteur
Frost-byte	Game over	Sabrewulf	Savage-1, -2, -3
Garfield	Great Escape	Shootout	Slap-fight
Gunfirght	Hang-on-1	Spindizzy	Starfox
Highnoon	Highway Encounter	Ten Pin	Timegate
Hobbit	Hulk	Titanic-1, -2	Tomahawk
Human Torcht	Hundra	UCM	Underwurlde
Hypaball	Jack the Nipper-1,2	Uridium	Xeno
Jetman	Jetpac		

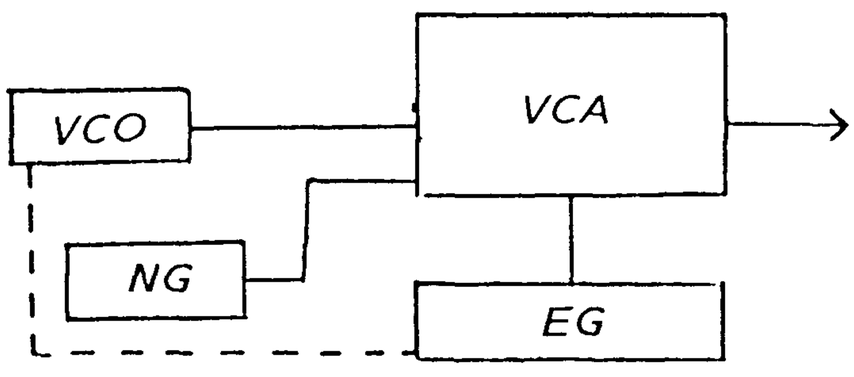
ENTERPRISE-MIDI

Sok Enterprise felhasználó nemigen tud mit kezdeni, ha hangszíneket akar – vagy kell – gyártania gépével. Ezért összefoglaljuk a legfontosabb tudnivalókat a gép hangszintézisével kapcsolatban, kiegészitve néhány elengedhetetlenül szükséges hangszínprogramozási alapismerettel.

Legjobb, ha a szintetizátorok működését vesszük alapul az Enterprise-zal való hangszín-editálás magyarázatához.

Egyszerű hangzások szintéziséhez is legalább négy egység szükséges:

a VCO (Voltage Controlled Oscillator),
az EG (Envelope Generator),
a VCA (Voltage Controlled Amplifier),
és az NG (Noise Generator).



1. ábra

Az EG vezérli a VCA-t, amelybe a VCO és az NG jelét vezetjük. (1. ábra) Ha az EG-vel a VCO-t is vezéreljük, az amplitúdóváltozással párhuzamosan frekvenciaváltozás is történik. A szintetizátorok VCO-kimenetén általában négyféle hullám jelenik meg. Ezek:

a szinusz (2. ábra)
a négyszög (3. ábra)
a fűrész (4. ábra)
és a négyszögimpulzus (5. ábra),

melynek kitöltési tényezője változtatható. A kitöltési tényező a négyszögimpulzus magas és mély összetevőinek aránya.

Az Enterprise-ban a VCO-nak megfelel az SWG (Square Wave Generator), amely – sajnos – csak négyszögjelet képes előállítani. Az NG és az EG a szintetizátorokéhoz hasonló. Az NG a 6. ábrának megfelelő jelet produkál.

Hangszín-editáláskor elsősorban arra kell ügyelnünk, hogyha konkrét hangszerek hangszíneit kívánjuk utánozni, az biztos, hogy ezt csak bizonyos frekvenciatartományokban tehetjük meg. (Ld. 7. ábra) Mivel az Enterprise SWG-je csak négyszögjelet produkál és ezzel a jelformával oly sokféle hangszín nem állítható elő, igénybe kell vennünk gépünk ringmodulátorait. A ringmodulátor bemenetére két SWG jelét kapcsolva a kimeneten a frekvenciák összege és különbsége jelenik meg. Így állíthatók elő fémes, gongszerű hangzások. Három SWG van a gépben (0-2) plusz egy NG (3). Ezekkel négyszólamú zenét szerkeszthetünk, hacsak nem ringmodulálunk. Ekkor természetesen a szólamok száma fogy. Az alábbi ringmodulációk lehetségesek:

– 1&3 ring
– 0&2 ring
– 2&0 ring
– 2&1 ring.

Az EG egység használata már kicsit bonyolultabb. Az EG felelős a hangzást kísérő amplitúdó- és frekvenciaváltozások kialakításáért, a hangszín burkológörbéjéért. (8. ábra) A szintetizátorokban általában négyfázisú EG található – innen az ADSR elnevezés. (9. ábra) Az Enterpriseban a fázisok száma 256 (0-255), de ez a 256 fázis mégsem olyan sok, mint első olvasásra tűnik, hiszen az egyes fázisoknak csak az idejét, az oszcillátorok fázishoz tartozó frekvenciáját, a bal és jobb oldal amplitúdóját (az Enterprise sztereó!) tudjuk állítani. Így sok kis téglalapból kell összeállítanunk azt a görbét (10. ábra attack fázisa), amit a szintetizátoroknál egy potenciométer-tekeréssel elintézhetünk. Az Enterprise-ban követett módszer azonban lehetőséget ad bonyolultabb burkológörbék szerkesztésére is.

Néhány hangszer burkológörbéjének leírása:

Gitár: Igen rövid (vagy zérus) felfutási idő, majd a teljes amplitúdójú rezgés fokozatosan csillapodik. (11. ábra)

Angolkürt: Hosszabb felfutási idő, rövid ideig teljes amplitúdó, majd a sustain fázis következik, végül gyors lecsengés. (12. ábra)

Trombita: Az angolkürtéhez hasonló felfutás nem teljes amplitúdóra, majd töréspont, innen újabb emelkedő szakasz, utána a sustain fázis, végül gyors lecsengés. (13. ábra)

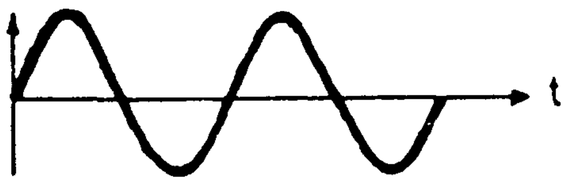
Orgona: A felfutási idő zérus, a teljes amplitúdójú sustain fázis után a lecsengés ismét zérus. (14. ábra.)

Zongora: Rövid felfutási idő után szintén rövid sustain teljes amplitúdón, végül a rezgés fokozatosan csillapodik. (15. ábra)

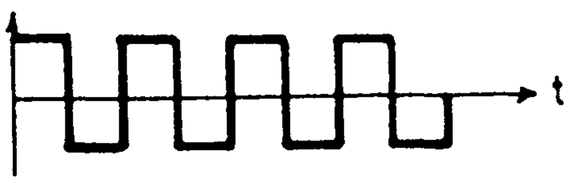
Az ismertetett burkológörbéket könnyen kialakíthatjuk az Enterprise-zal.

(folytatjuk)

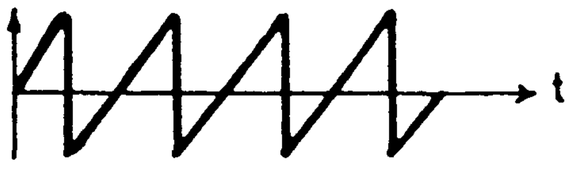
2.



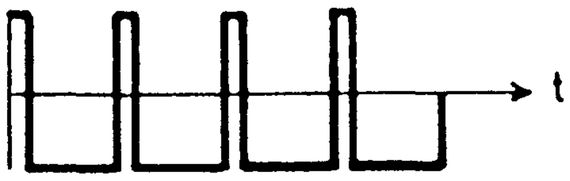
3.



4.

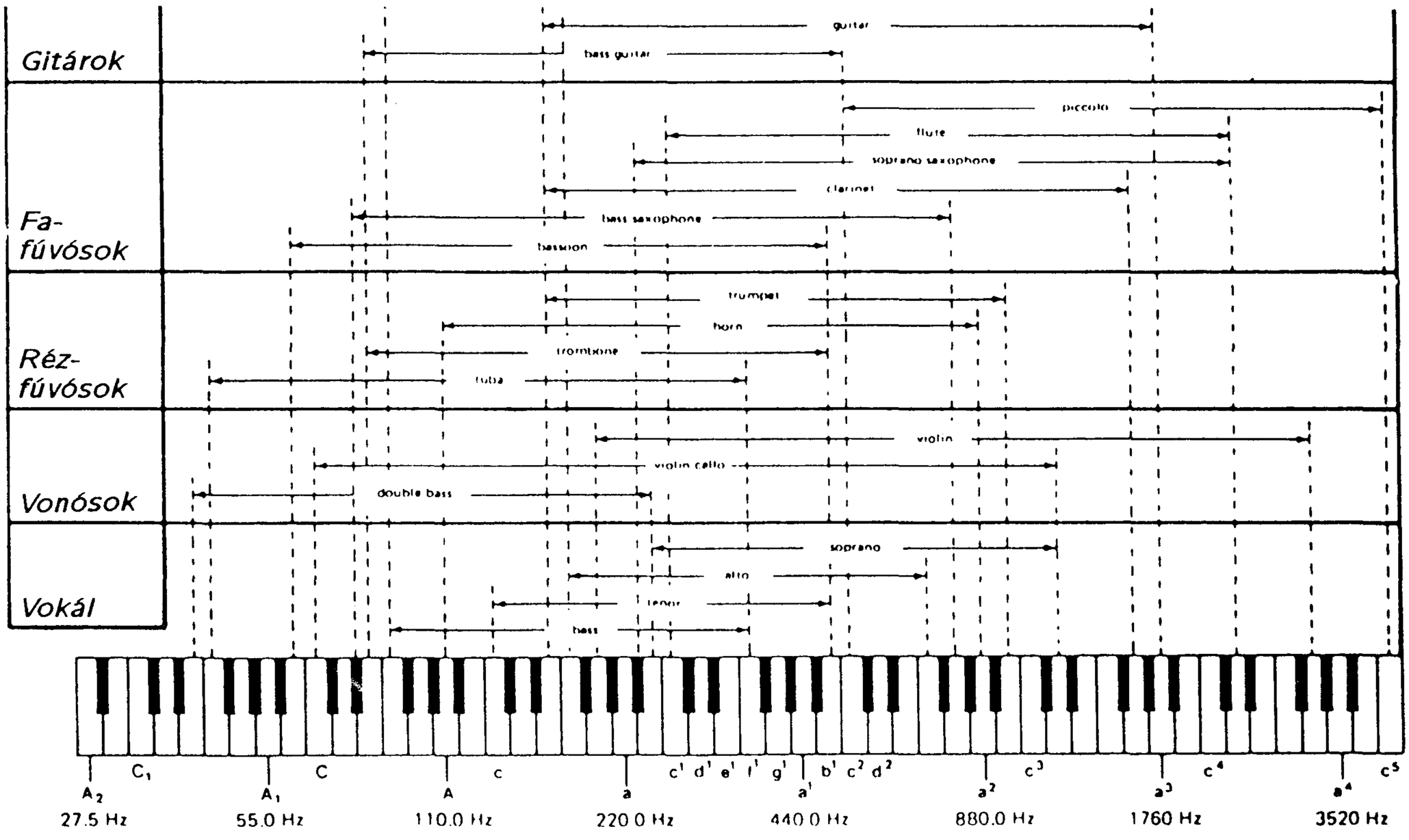


5.

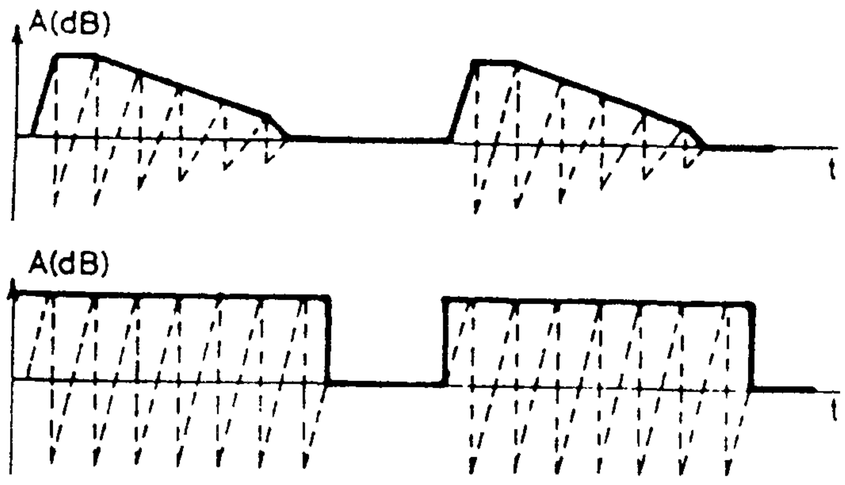


6.

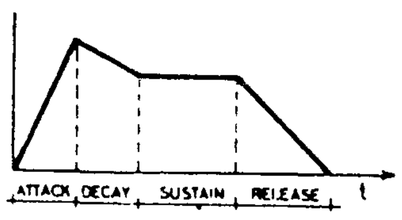
7.



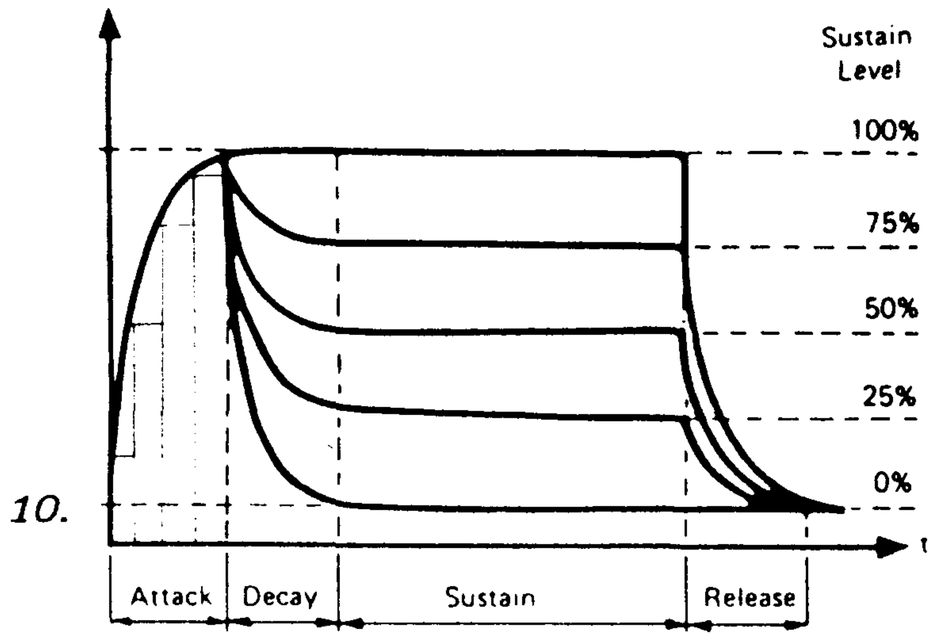
8.



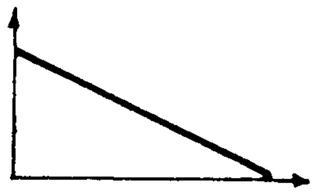
9.



10.

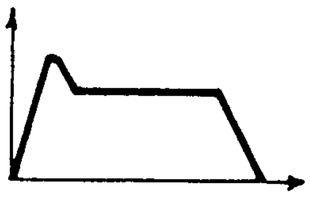


11.



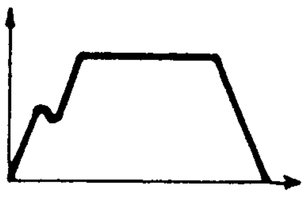
Gitár

12.



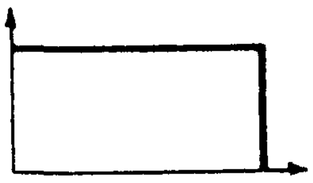
Angolkürt

13.



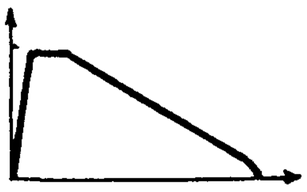
Trombita

14.



Orgona

15.



Zongora

EXOS-bővítések

A következő néhány hasábon olyan programok találhatók, amelyek tovább növelik az ENTERPRISE BASIC használhatóságát.

Az első három rutin úgynevezett rendszerbővítő, de ettől a kifejezéstől senki se rémüljön meg, mert a létrehozásukhoz és a kezelésükhöz elegendő minimális BASIC ismeret. A módszer a következő:

1. Az alább közölt programok begépelése, természetesen egyenként. 2. A begépelés után 'START' vagy 'RUN' paranccsal indítsuk el a programot, amely elkészíti szalagra vagy lemezre a rendszerbővítő modult.

Ezután már csak ezt a rendszerbővítőprogramot kell betölteni. Tehát például ha a "CLX" látványos képernyőtörlését akarjuk felhasználni egy BASIC programban, akkor valahol a program elején ki kell adnunk a következő parancssort: EXT "LOAD CLX.EXT". Majd minden olyan helyen, amikor szükség van a kép törlésre, akkor az EXT "CLX" paranccsal aktivizálhatjuk a bővítőt.

1. Látványos képernyőtörlés

Az elkészített rendszerbővítő a BASIC-ben oly módon törli le a GRAPHICS paranccsal megnyitott képernyőt, hogy a képet tartalmazó bájtok véletlenszerű sorrendben tűnnek el. A program használatakor azonban két dologra vigyázzunk: A program szak a sztenderd grafikus képernyőt, vagyis a #101-es csatornát törli le, s csakis 'HIRES' üzemmódban használható!

```
 10 PROGRAM "CLX.BAS"
 20 OPEN #36:"CLX.EXT" ACCESS OUTPUT
 30 FOR N=1 TO 268
 40   READ A
 50   PRINT #36:CHR$(A);
 60 NEXT
 70 CLOSE #36
 80 STOP
100 DATA 0,7,208,0,0,0,0
110 DATA 0,0,0,0,0,0,0
120 DATA 0,0,60,191,128,98,1
130 DATA 41,225,110,192,98,181,71
140 DATA 207,240,140,10,0,0,132
150 DATA 192,15,112,67,69,130,201
160 DATA 33,147,11,2,1,0,128
170 DATA 236,101,57,28,205,38,243
180 DATA 112,128,96,46,24,2,129
190 DATA 175,224,19,0,240,183,100
200 DATA 6,159,192,54,3,84,38
210 DATA 26,127,16,196,0,192,152
220 DATA 105,252,76,16,1,130,97
230 DATA 167,241,96,80,2,104,178
240 DATA 91,174,86,218,192,190,219
250 DATA 88,149,219,45,181,144,32
260 DATA 124,101,3,0,222,224,179
270 DATA 195,44,14,203,3,185,128
280 DATA 108,103,225,124,240,230,31
290 DATA 145,221,164,56,198,0,221
300 DATA 21,69,240,7,153,237,166
310 DATA 176,30,52,214,42,246,154
320 DATA 200,76,128,0,128,0,206
330 DATA 210,28,74,0,11,0,0
340 DATA 195,147,240,141,86,5,16
350 DATA 0,156,195,240,33,125,186
360 DATA 25,110,141,128,0,6,232
370 DATA 170,26,0,5,2,14,147
380 DATA 227,152,126,246,64,25,197
390 DATA 142,210,92,36,0,5,158
400 DATA 22,34,6,136,170,15,0
410 DATA 23,74,129,49,176,0,2
420 DATA 63,28,59,86,13,147,219
430 DATA 77,96,122,105,172,65,13
440 DATA 53,147,117,194,209,87,147
450 DATA 217,32,0,0,0,1,128
460 DATA 0,10,0,0,0,0,0
470 DATA 0,0,0,0,0,0,0
480 DATA 0,0
```

2. Külső joystick illesztő

A következő rendszerbővítő program segítségével a külső (external) joystick úgy viselkedik, mintha belső (internal) lenne. Tehát mozgatja a kurzort a BASIC-ben, a WP-ben, az EXDOS-ban, az IS-DOS-ban, a HISOFT PASCAL-ban. A tűzgomb (FIRE-BUTTON) használható SHIFT billentyűnek.

A 7-es, 10-es és 11-es rendszerváltozót kezeli a program (KEY CLICK, KEY RATE, KEY DELAY).

Megjegyezzük, hogy a program nem feltétlenül működik egyes német gépeken.

```
 10 PROGRAM "JOY.BAS"
 20 OPEN #36:"JOY.EXT" ACCESS OUTPUT
 30 FOR N=1 TO 481
 40   READ A
 50   PRINT #36:CHR$(A);
 60 NEXT
 70 CLOSE #36
 80 STOP
100 DATA 0,7,137,1,0,0,0
110 DATA 0,0,0,0,0,0,0
120 DATA 0,0,31,0,215,34,1
130 DATA 250,224,64,42,98,181,64
140 DATA 36,160,35,54,15,0,3
150 DATA 137,236,216,50,0,14,44
160 DATA 179,96,160,0,20,64,36
170 DATA 31,63,192,37,96,0,70
180 DATA 8,0,189,193,13,22,10
190 DATA 188,158,201,9,134,151,44
200 DATA 0,43,37,94,184,96,49
210 DATA 90,163,231,248,4,36,0
220 DATA 8,207,96,7,184,33,162
230 DATA 193,100,143,129,11,150,0
240 DATA 234,116,128,45,196,0,214
250 DATA 218,204,101,53,128,0,32
260 DATA 0,0,70,101,128,61,194
270 DATA 170,241,148,213,0,7,2
280 DATA 25,35,169,152,0,183,97
290 DATA 71,160,2,184,4,0,5
300 DATA 102,199,224,2,208,88,0
310 DATA 1,2,129,89,177,212,0
320 DATA 244,179,39,142,165,128,1
330 DATA 28,242,230,7,140,165,72
340 DATA 0,128,18,50,148,160,3
350 DATA 230,128,202,81,0,50,87
360 DATA 2,0,152,234,76,128,15
370 DATA 6,82,84,0,85,37,0
380 DATA 23,204,5,187,32,124,207
390 DATA 25,72,176,5,120,202,66
400 DATA 128,50,204,34,128,8,240
410 DATA 174,6,2,22,4,44,0
420 DATA 17,245,38,111,101,134,70
430 DATA 0,244,16,148,80,8,25
440 DATA 42,151,195,37,114,248,48
450 DATA 3,25,70,64,0,131,160
460 DATA 116,204,95,173,216,13,181
470 DATA 137,93,94,211,88,179,87
```

➤➤➤

```
480 DATA 70,195,211,77,98,201,25
490 DATA 42,151,195,37,114,249,94
500 DATA 50,136,192,12,147,200,49
510 DATA 166,181,109,173,131,227,246
520 DATA 88,72,32,244,100,143,159
530 DATA 227,40,108,0,175,100,143
540 DATA 156,236,152,191,255,62,255
550 DATA 231,207,252,247,255,158,191
560 DATA 243,207,254,120,255,206,255
570 DATA 249,219,255,58,255,231,79
580 DATA 252,231,255,155,223,243,143
590 DATA 252,0,0,0,0,4,0
600 DATA 10,51,123,254,160,0,0
610 DATA 12,148,79,44,128,0,0
620 DATA 6,130,252,0,74,39,150
630 DATA 68,2,1,1,216,202,114
640 DATA 57,154,77,230,225,0,196
650 DATA 92,49,6,136,4,2,1
660 DATA 0,128,64,69,60,29,12
670 DATA 167,35,113,132,216,32,37
680 DATA 27,207,39,51,161,164,198
690 DATA 107,16,17,14,70,147,177
700 DATA 148,228,13,16,8,4,2
710 DATA 1,0,129,0,32,24,142
720 DATA 71,3,129,1,4,220,100
730 DATA 50,156,142,98,2,113,164
740 DATA 202,108,57,153,77,194,193
750 DATA 1,16,202,110,54,152,78
760 DATA 70,176,110,0,10,0,0
770 DATA 0,0,0,0,0,0,0
780 DATA 0,0,0,0,0
```

3. Óra

Ez a rendszerbővítő egy órát működtet a státuszsorban, amely a betöltés után csak egy hideg reset után tűnik el a memóriából. Kilenc új EXOS változót generál és használ, amelyek a következők (a változók tartalmának alapértelmezése zárójelben):

Száma Magyarázat
247 : (0) = az óra látszik a státuszsorban, 255 = nem látszik
248 : A státuszsor-beli pozíció (6).
249 : 0 = az óra jár, (255) = nem jár
250 : 0 = az óra lenullázása
251 : Állítsuk nullára, mielőtt ASK-kal lekérdezzük az időt!
252 : Óra, maximum = 23.
253 : Perc, maximum = 59.
254 : Másodperc, maximum = 59.
255 : 2/100 másodperc, maximum = 98.

Egy BASIC program az alábbi parancssorral tudja lekérdezni az időt:

```
SET 251,0 : ASK 252 H : ASK 253 M : ASK 254 S : ASK 255 P
: PRINT H,M,P
```

Ha a rutint BASIC-be töltjük be, akkor az F7 és F8 billentyűkkel ('TOGGLE 249', illetve 'TOGGLE 250') leállíthatjuk, illetve lenullázhatjuk az órát.

Ha hibás időt akarunk beírni valamelyik változóba, akkor a következő hibaüzenet jelenik meg:

*** Invalid use of CLOCK
BASIC-ben: EXSTRING$(9126)

```
 10 PROGRAM "CLOCK.BAS"
 20 OPEN #36:"CLOCK.EXT" ACCESS OUTPUT
 30 FOR N=1 TO 3*256+225
 40   READ A
 50   PRINT #36:CHR$(A);
 60 NEXT
 70 CLOSE #36
 80 STOP
100 DATA 0,7,192,3,0,0,0
110 DATA 0,0,0,0,0,0,0
120 DATA 0,0,60,191,128,140,40
130 DATA 8,2,62,123,36,218,
140 DATA 165,121,92,67,33,64,67
150 DATA 37,21,128,128,58,86,31
160 DATA 62,210,99,0,162,188,2
170 DATA 80,2,22,153,96,49,128
180 DATA 5,2,203,12,132,30,228
190 DATA 113,240,61,68,128,43,171
200 DATA 201,236,145,243,237,38,48
210 DATA 35,129,88,82,248,0,124
220 DATA 255,93,8,1,175,113,0
230 DATA 134,40,1,28,68,103,33
240 DATA 25,204,70,24,22,30,223
250 DATA 207,193,0,16,124,23,12
260 DATA 3,67,130,128,32,248,118
270 DATA 24,3,15,159,202,34,184
280 DATA 248,196,146,109,73,32,3
290 DATA 208,4,20,255,2,36,70
300 DATA 15,196,9,60,240,6,2
310 DATA 8,192,225,7,226,4,238
320 DATA 87,87,147,217,43,113,0
330 DATA 73,218,91,130,128,66,168
340 DATA 38,4,237,41,192,80,33
350 DATA 20,4,2,39,133,159,236
360 DATA 145,232,128,152,5,16,2
370 DATA 23,66,128,42,188,158,22
380 DATA 127,178,71,83,168,5,110
390 DATA 32,5,14,167,152,11,185
400 DATA 0,50,157,192,33,135,88
410 DATA 234,116,128,191,134,2,0
420 DATA 155,185,0,50,156,0,34
430 DATA 169,170,2,34,154,192,34
440 DATA 169,162,2,34,154,64,33
450 DATA 134,136,132,0,0,17,76
460 DATA 64,17,20,195,1,17,76
470 DATA 32,17,20,193,1,17,75
480 DATA 64,17,20,179,1,12,46
490 DATA 135,162,0,40,20,64,25
500 DATA 12,4,199,162,0,40,20
510 DATA 80,13,12,1,64,162,128
520 DATA 64,96,20,62,63,3,128
530 DATA 5,22,73,201,94,79,100
540 DATA 159,133,231,114,186,188,158
550 DATA 201,127,0,196,3,243,194
560 DATA 220,64,15,98,181,71,207
570 DATA 240,8,72,0,17,145,160
580 DATA 47,112,67,69,130,201,127
590 DATA 1,88,12,86,168,4,0
600 DATA 5,9,136,99,224,8,105
610 DATA 124,32,0,131,2,98,48
620 DATA 131,220,242,254,2,136,1
630 DATA 227,231,248,4,234,0,8
640 DATA 198,96,23,184,33,162,193
650 DATA 87,147,217,45,22,11,37
660 DATA 252,5,16,3,143,15,227
670 DATA 243,0,117,4,128,72,224
680 DATA 224,0,70,110,0,178,95
690 DATA 192,134,0,234,33,128,173
700 DATA 196,2,22,218,204,101,63
710 DATA 64,0,32,0,0,70,121
720 DATA 0,61,194,163,227,72,4
```

➢➢➢

```
730 DATA 12,8,8,202,16,23,184
740 DATA 44,124,105,0,129,193,1
750 DATA 25,74,2,247,5,171,198
760 DATA 80,108,4,28,8,100,142
770 DATA 167,232,2,220,64,43,16
780 DATA 192,16,19,243,24,4,39
790 DATA 59,136,4,32,16,9,128
800 DATA 86,126,64,137,206,235,129
810 DATA 0,88,108,0,21,159,144
820 DATA 34,115,186,224,64,12,27
830 DATA 0,5,103,228,8,157,252
840 DATA 36,16,0,85,231,113,212
850 DATA 219,0,127,12,5,4,177
860 DATA 84,204,0,17,76,208,1
870 DATA 84,200,0,17,76,144,0
880 DATA 11,192,30,237,45,204,0
890 DATA 3,194,140,206,121,81,131
900 DATA 193,224,240,121,188,69,46
910 DATA 0,15,10,35,57,229,68
920 DATA 15,7,131,193,230,241,20
930 DATA 173,0,118,150,229,32,1
940 DATA 225,70,103,60,168,193,224
950 DATA 240,120,60,222,34,147,192
960 DATA 7,133,17,156,242,162,7
970 DATA 131,193,224,243,120,138,73
980 DATA 128,14,163,192,2,221,128
990 DATA 42,123,47,199,81,196,0
1000 DATA 44,0,47,134,93,99,168
1010 DATA 242,0,71,39,177,129,66
1020 DATA 168,230,0,9,117,156,196
1030 DATA 103,33,25,220,70,235,21
1040 DATA 70,144,0,75,172,230,35
1050 DATA 57,8,206,226,55,88,170
1060 DATA 47,128,2,93,103,49,25
1070 DATA 200,70,119,17,186,197,81
1080 DATA 84,0,18,235,57,136,206
1090 DATA 76,145,243,252,101,19,0
1100 DATA 25,35,231,59,36,0,0
1110 DATA 0,0,4,0,10,48,40
1120 DATA 2,160,0,0,20,134,76
1130 DATA 39,144,201,116,225,7,252
1140 DATA 235,255,157,63,243,159,254
1150 DATA 114,255,206,63,249,195,255
1160 DATA 55,255,230,239,252,219,255
1170 DATA 155,63,243,95,254,103,255
1180 DATA 205,63,250,128,0,207,247
1190 DATA 251,252,0,0,0,0,0
1200 DATA 0,0,0,0,0,0,0
1210 DATA 0,22,3,1,128,192,96
1220 DATA 48,24,12,6,0,2,25
1230 DATA 48,158,67,37,136,14,198
1240 DATA 83,145,204,210,111,55,8
1250 DATA 6,34,225,136,52,64,32
1260 DATA 16,8,4,2,1,80,128
1270 DATA 160,117,49,27,13,38,49
1280 DATA 1,16,222,109,48,154,77
1290 DATA 194,2,153,188,204,116,59
1300 DATA 152,78,70,81,0,168,26
1310 DATA 32,16,8,4,2,1,1
1320 DATA 20,220,116,50,156,142,7
1330 DATA 35,73,204,202,32,57,25
1340 DATA 76,38,193,1,208,210,109
1350 DATA 50,136,12,102,195,121,140
1360 DATA 214,13,16,8,4,2,1
1370 DATA 0,129,0,32,24,142,71
1380 DATA 3,129,1,4,92,32,39
1390 DATA 26,76,166,195,153,148,220
1400 DATA 44,16,17,12,166,227,105
1410 DATA 132,228,107,6,131,84,165
1420 DATA 66,121,28,142,76,34,136
1430 DATA 6,67,65,202,132,26,13
1440 DATA 82,149,9,228,114,57,48
1450 DATA 138,32,25,13,70,10,16
1460 DATA 104,80,146,110,59,24,77
1470 DATA 134,147,32,128,234,115,50
1480 DATA 136,13,230,97,1,12,152
1490 DATA 79,33,146,241,0,10,0
1500 DATA 0,0,0,0,0,0,0
1510 DATA 0,0,0,0,0,0
```


4. Scroll

Az alábbi BASIC program jobbra, illetve balra görget bármíly grafikus képernyőt. A rutin meghívása előtt a következő adatokkal kell feltöltenünk a VAR változókat:

POKE VAR+0,X : A grafikus lap első sora, amelyet görgetni akarunk.
POKE VAR+1,X : A lap utolsó sora.
POKE VAR+2,X : Turbo on/off. 0 = on, <> 0 = off.
POKE VAR+3,X : Bájt átdobás a sor elejére on/off. 0 = átdob. <> 0 = 0-ás bájtot szúr be.

Ezek után a rutint a következőképpen hívhatjuk meg:

CALL USR (LEFT,CSATORNA+256*bájtok-száma-amit-görgetni-akarunk)

```
100 ALLOCATE 203
110 CODE VAR=HEX$("0,0,0,0")
120 CODE SCROLL=HEX$("7D,6,2,F7,B,DF,15,28,4,3E,4,
    BA,C0,ED,5B")&WORD$(VAR)&HEX$("79,87,87,87,
    81,3D,BA,D8,BB,D8,7A,BB,D8,4C,C5,D5,7D,6,3,
    F7,B,E1,D1,DF,3A")&WORD$(VAR+2)
130 CODE=HEX$("B7,20,5,F3,3E,FE,D3,BF,DB,B1,F5,
    DB,B2,F5,DB,B3,F5,3E,C0,A0,3E,FD,20,3,3D,CB,
    F0,D3,B1,3C,D3,B2,3C,D3,B3,D5,E5,5A,16,0,3A")
    &WORD$(SCROLL+137)
140 CODE=HEX$("FE,2B,28,8,7D,B7,60,69,20,7,18,
    11,7C,3C,60,69,2B,6,8,F,30,1,19,CB,13,CB,12,
    10,F6,D1,EB,C1,8,3A")&WORD$(VAR+3)
150 CODE=HEX$("8,78,41,4F,D,7C,95,3C,F5,C5,D5,
    62,6B,46,23,C5,6,0,ED,B0,C1,EB,8,B7,20,1,47,
    70,8,3D,20,EB,D1,C1,F1,10,E3,F1,D3,B3,F1,D3,
    B2,F1,D3,B1,3A")&WORD$(VAR+2)&HEX$("B7,C0,
    3E,4,D3,BF,FB,C9")     >>>
```

```
160 CODE LEFT=HEX$("1,B0,23,18,3")
170 CODE RIGHT=HEX$("1,B8,2B,78,32")&WORD$(SCROLL+137)
    &HEX$("79,32")&WORD$(SCROLL+142)&HEX$("C3")
    &WORD$(SCROLL)
180 !
190 ! SCROLL DEMO
200 !
210 SET 26,255:SET 27,0:SET 22,5:SET 23,2:SET 24,40:
    SET 25,3
220 CLOSE #102
230 FOR N=1 TO 2
240   OPEN #N:"VIDEO:"
250   FOR J=0 TO 3
260     DISPLAY #N:AT-2+N*3+J*7 FROM 1 TO 3
270   NEXT J
280   SET #N:PALETTE 0,N*73+36,N*73
290   SET #N:LINE STYLE 13
300   PLOT #N:0,96;1200,96
310   PLOT #N:0,8;1200,8
320   PLOT #N:0,80,
330   SET #N:INK 2
340 NEXT N
350 PRINT #1:"'a' STUDIO"
360 PRINT #2:"SCROLL"
370 POKE VAR,0
380 POKE VAR+1,26
390 POKE VAR+2,255
400 POKE VAR+3,255
410 DO
420   CALL USR(LEFT,1+256*1)
430   CALL USR(RIGHT,2+256*1)
440 LOOP UNTIL INKEY$<>""
```

'a' STUDIO 'a' STUDIO 'a' STUDIO 'a' STUDIO

Válasz néhány megválaszolatlan kérdésre

Aki kérdez: az Olvasó
Aki válaszol: Szűcs Zoltán

K: A közelmúltban vásároltam egy 'ENTERPRISE' számítógépet. Milyen lemezmeghajtót vásárolhatok hozzá?
V: Bármilyen, a SHUGART 410 szabványnak megfelelő meghajtó használható.
K: ???
V: Majdnem minden 3.5"-os és 5.25"-os meghajtó rendelkezik a SHUGART 410 néven közismert összeköttetési szabványnak megfelelő illesztővel és kivezetéssel.
K: Ez mind szép, de hogyan keressem az üzletekben?
V: Ez elszántság kérdése. A mindenre elszántak kereshetik a következő módon: "Keresek

BASF 6162,
Epson SMD100, Epson SMD120,
Epson SMD130, Epson SMD140,
Sony MPX-026R,
Teac FD35B,
Teac FD35F típusjelű 3.5", valamint
BASF 6128,
Mitsubishi M4853,
Teac FD55BR-506-U
Teac FD55FV-13-U,
Shugart SA465 típusjelű 5.25"-os

floppy disk driver-t."

A kevésbé elszántak, akik már ismerik a magyar kereskedők számítástechnikai tájékozottságát, a következőképpen kereshetik:
"Olyan floppyt keresek, ami jó az IBM PC/XT gépekbe!"
Ez utóbbi esetben nem hozzuk kellemetlen helyzetbe az eladót, és a siker garantált. Lesz floppy.

K: Ezek szerint az 1.2 Mb floppyt is kezeli az 'ENTERPRISE'?
V: Igen, 720 Kb-os formátumban tudja használni. Ennek a következő oka van:
Az 'EXDOS' lemezkezelő egysége tartalmaz egy WD1770-es floppy controller-t. Ez az IC kompatibilis az FD179X IC-kel, tehát csípőből tudja az IBM 3740 szabvány szerinti 'szoft szektoros' felírást, formattálást. Az 'EXDOS' lemezkezelő szoftvere pedig felülről kompatibilis az MS-DOS 2.0 verziójával.
Ez utóbbi teszi lehetővé, hogy az 'ENTERPRISE' adatkompatibilis legyen az IBM PC/XT/AT és ezekkel kompatibilis gépekkel.
K: Mit jelent ez az 'adatkompatibilitás'?
V: Nagyjából azt, hogy megbízhatóan tudják egymás lemezeit kezelni.
K: Ez hogy lehetséges?
V: Nézzük meg a lemez szervezését! ➢➢➢

A BOOT szektor:

```
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   0123456789ABCDEF
0000  EB FE 90 45 58 44 4F 53 31 2E 33 00 02 02 01 00   ...EXDOS1.3.....
0010  02 70 00 A0 05 F9 03 00 09 00 02 00 00 00 C9 00   .p..............
0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0040  56 4F 4C 5F 49 44 00 68 30 21 3A 00 00 00 00 00   VOL_ID.h0!:.....
0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0060  00 00 00 00 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
0070  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
0080  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
0090  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
00A0  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
00B0  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
00C0  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
00D0  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
00E0  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
00F0  E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5   ................
```

```
B: 2 Side/Disk, 80 Track/Side,  9 Sector/Track    DISK SIZE: 720 Kilobyte
Logical sector:   0  Side:0  Track: 0  Sector: 1  Max. logic. sector:1439
```

A boot sector egyes byte-jainak jelentése:

```
00-02 :EB FE 90  -> 8086 assembler kódként értelmezve JMP SHORT $
                                                     NOP
03-0A :EXDOS 1.3 -> A DOS neve és verziószáma
0B-0C :00 02     -> Egy SZEKTOR mérete bytokban kifejezve. Itt 512
0D    :02        -> Egy LOGIKAI BLOKK ennyi SZEKTOR-ból áll.
0E-0F :01 00     -> A FAT-ot megelőző SZEKTOR-ok száma.
10    :02        -> A lemezen található FAT másolatok száma.
11-12 :70 00     -> A fő DIRECTORY-ban maximum ennyi bejegyzés lehet.
13-14 :A0 05     -> A lemezre maximum ennyi SZEKTOR írható fel.
15    :F9        -> A lemez típusa.
16-17 :03 00     -> A FAT egy másolatának mérete.
18-19 :09 00     -> Az egy SÁVON elhelyezkedő SZEKTOROK száma.
1A-1B :02 00     -> A lemezoldalak száma.
1C-3F :          -> EXDOS-ban nem használt.
40-46 :          -> Lemezazonosító bevezető.
47-4A :          -> Az EXDOS által formattáláskor generált 32 bites
                    véletlenszám, a lemez azonosításához.
```

V: Látható, hogy a BOOT szektor a lemez első szektora. Byte-jainak jelentése a 00-1BH-ig azonos az MS-DOS-nál és az EXDOS-nál. Mivel a lemezkezelő mindkét gépen e szektor alapján azonosítja a lemezt, tulajdonképpen nem is kell tudnia arról, hogy ezt nem a saját rendszere hozta létre.

K: Nem vagyok meggyőzve, hiszen a lemezkezeléshez ennél több is kell.

V: Így igaz, de erről majd legközelebb.

Magyar üzemmód az Enterprise-on

Bizonyára sok Enterprise-tulajdonos gondolt már arra, hogy hasznos lenne a német vagy angol üzemmód mintájára működő magyar kiterjesztés. A most következő program ezt próbálja megoldani. A program a kétnyelvű gépek sajátosságait használja ki, ezért azoknak, akik nem ilyen géppel rendelkeznek, némileg módosítaniuk kell a programot. A program ún. rendszerbővítőként lett kialakítva, ezért ismertetése előtt rövid szinopszist adunk a bővítőkről.

A bővítőkről általában

Az Enterprise bővítőként kezel minden olyan file-t, amely fejének (első 16 byte-jának) első byte-ja 00, a második byte-ja 06 v. 07 értékű. Az első esetben abszolút, a második esetben relatív rendszerbővítőről beszélőnk. A kettő közötti különbség csak annyi, hogy az abszolút bővítő mindig a 0C00AH címen kezdődik, míg a relatív esetében az EXOS az aktuális RAM-állapottól függően valahová, de mindig a 3. lapra helyezik le a file értékes byte-jait (a 17. byte-tól kezdve).

A bővítők talán az Enterprise leghatékonyabb szolgáltatása, ugyanis különböző EXOS-rutinok hívják automatikusan őket, ha saját hatókörükben már nem tudják feladatukat megoldani. Így például ha az EXOS-nak 127-nél nagyobb számú rendszerváltozót kellene kezelnie, a "Rendszerváltozók írása, olvasása vagy átkapcsolása" nevű EXOS-funkció (EXOS_VARS, kód: 16=10H) ez esetben végigvizsgálja az összes élő rendszerbővítőt, hogy készek-e fogadni EXOS-változókat kezelő parancsokat. A bővítők attól függően, hogy milyen hatókörben kellene szolgáltatást végezniük, az EXOS-tól egy speciális, 1 és 8 közé eső értéket (akciókódot) kapnak paraméterként, mely alapján a bővítő eldöntheti, hogy képes-e az aktuális feladat végrehajtására, vagy sem. A bővítő az akciókódtól függően további paramétereket is kaphat a B, illetve a DE regiszterekben. Az akciókód mindig a C regiszterben kerül átadásra. Jelen példánkban (EXOS-változó kezelése) a bővítők 4-es akciókóddal aktivizálódnak. Ha egy bővítő az aktuális feladatot nem képes elvégezni, akkor a BC, illetve a DE regiszterpárok híváskor érvényes értékeit megőrizve kell visszatérnie, ha viszont képes a feladat elvégzésére, akkor ezen regiszterekben a feladat által meghatározott paramétereket kell átadnia az őt hívó programnak. Ez változhat az akciókódtól függően, de a C regiszterben (vagyis az akciókódnak) mindig 0-nak kell lennie, jelezve, hogy a többi bővítést nem kell a feladat végrehajtása miatt meghívni; illetve az A regiszterben az állapotkódnak kell lennie, ami a végrehajtás során esetleg felmerülő hibára utal (ha nincs hiba: A=0).

A jelen bővítés jellegzetességei

Mikor a szövegszerkesztőből, illetve a BASIC interpreterből parancsmódból, vagy programmódból EXT paranccsal bővítést aktivizálunk, akkor a bővítőket mindig a "Bővítések vizsgálata" nevű EXOS-funkció (EXP_INQ, kód: 26=1AH) hívja meg. Ez a funkció két akciókóddal tud körbejárni: 2-essel, ha meghatározott bővítést aktivizálni kell, 3-assal, ha help(segítő)-üzenetet kell a bővítés(ek)nek kiírnia. Tehát a magyar üzemmódot beállító bővítésnek csak e két akciókóddal kell futnia. További paraméterek is adódnak át, melyek mind a két akciókód esetén ugyanazok: a DE regiszterpár a parancs-stringre, melynek első byte-ja egy hosszúságbyte, a B regiszter viszont a parancs-string első szavának (az első betűközig tartó karaktersorozatnak) a hosszát tartalmazza. Fontos megjegyezni, hogy a DE által mutatott byte értéke és a B regiszter tartalma eltérhet, ugyanis lehetőség van arra, hogy a parancs-string a bővítés nevén kívül, a név után betűközzel elválasztva más, paraméterként fungáló karaktersorozatot is tartalmazzon.

Ez esetben a B regiszter a hívó bővítésnév hosszát tartalmazza, a DE által mutatott byte értéke viszont a további karakterek számát is magában foglalja.

A hívó EXP_INQ funkció átalakítva adja át a parancs-stringet: help-hívás esetén a stringről lemetszi a kezdő "HELP" szót és az azt követő szóközöket; tehát ezek nem kerülnek átadásra. A maradékból pedig, illetve a direkt hívás esetén a változatlan parancsfüzérből az első szót, de csak azt, nagybetűsíti.

A program használata

A program betöltés után nem aktivizálódik (ehhez a 8-as akciókódot is kellene fogadnia), csak inicializálódik: a megfelelő táblázatba beíródik a címe és RAM-rezidenssé válik. (Ennek eredményeképpen még meleg-reset esetén is élő marad.) Aktivizálása úgy történik, mint a beépített bővítéseké, vagyis BASIC-ből parancsmódban a bővítés nevét ("HUN") egy kettőspont mögé írva, ENTER -rel lezárva, parancsmódban az EXT utasítás argumentumaként, illetve a szövegszerkesztőből a 8-as funkcióbillentyű megnyomása után a "HUN" karaktersorozattal válaszolva.

A program a német üzemmód kiterjesztéseként dolgozik, tehát ha előzőleg angol üzemmódban volt a gép, a többi karakter is átíródik. A program ebben az elrendezésében az ALT billentyű lenyomása mellett szolgáltat ékezetes karaktereket a következőképp:

ALT H - É	ALT X - é	ALT C - Ő	ALT S - ő
ALT I - Í	ALT Y - í	ALT J - Ú	ALT Z - ú
ALT F - Ó	ALT V - ó	ALT D - Ű	ALT T - ű

Az Á és az á karakterek változatlanok:

ALT E = Á ALT U = á

A rövid Ö, ö, Ü, ü a német billentyűzet-kiosztásnak megfelelően érhető el. A karakterkiosztás azért így lett átalakítva, mert így kényelmesen elvégezhető a kisbetű–nagybetű átalakítás.

A program felvétele

Azoknak, akiknek van olyan asszemblerük, amelyik képes EXOS modulfejet generálni a file elé, csak be kell írniuk az assembly programot, lefordítani, és 6-os típusú modulfejjel (XABS: abszolút rendszerbővítő) elmenteni (1. sz. lista). Akiknek nem ilyen az asszemblerük, a lefordított program elé közvetlenül egy 16 byte-os fejet kell írniuk, ahogy az a BASIC programban szerepel, és az egészet együtt kimenteni. (2. sz. lista)

Azoknak, akik semmilyen asszemblerrel nem rendelkeznek, a következő BASIC programot kell begépelniük, és futtatniuk. Sikeres lefutás esetén nekik is működőképes programjuk lesz. (3. sz. lista)

A BASIC program úgy lett kialakítva, hogy ugyanazokat a cimkéket tartalmazza, amelyeket az assembly lista, így a BASIC programban is követhető a program menete. A BASIC listán még két plusz gépi kódú rutin is szerepel. A SAVE_HEAD rutin a 106-os csatornán létrehoz egy kimeneti file-t, mely neve a FILE_NAME cimke után áll. Aki módosítani akarja a nevet, vegye figyelembe, hogy a CHR$ függvény argumentumában a mindenkori név hosszának kell szerepelnie. Ez a rutin írja még ki a fej 16 byte-ját is, melynek első két byte-ja az abszolút rendszerbővítő modul típusbyte-jai, a 3. és a 4. a file hosszát adja a fej byte-jai nélkül. A maradék 12 byte ebben az esetben nem használt. Aki módosít az itt közölt programon, és BASIC programmal veszi fel a file-t, ne felejtse el az új programhossznak megfelelően átírni a PRG_LNG változó értékét!

A SAVE_FILE rutin menti ki a létrehozott csatornára a program-file-t és ezután megszűnteti a csatornát. A két rutin assembly listáján követni lehet a megvalósítás menetét, így a rutinokat bárki beilleszthetí saját programjaiba.

A BASIC program interaktívan kijelzi a felhasználó által elvégzendő feladatokat, jelzi a program sikeres lefutását, illetve hibajelzést ad, ha futás során hiba történt. A hibajelzés száma a megfelelő EXOS hibakódra utal.

A sikeres programfutás után a BASIC program akár kitörölhető is lenne, de azért nem árt elmenteni, hátha gépelési hiba miatt a felvett program rosszul működne.

Maga a program

A START címke utáni öt utasítás megvizsgálja az akciókód értékét. Ha ez nem 2 (parancsfüzér) vagy nem 3 (help-füzér), akkor visszatér a hívó programhoz a regiszterek értékének megőrzése mellett. Erre azért van szükség, mert a további bővítések vizsgálatakor egyébként azok hibás értékeket kapnának meg. 2-es akciókód esetén a RUN címkétől kezdve elindul a karakterkészlet megváltoztatása; 3-as akciókód esetén a következő byte-okra adódik a vezérlés. Itt először a program megvizsgálja, hogy a füzér hossza 0-e. Ha igen, akkor ez általános help-hívást jelent, melyre minden bővítőnek válaszolnia kell; ha nem 0, akkor konkrét help-hívás történt. Általános help-hívás esetén az A regiszterbe a COMHELP címke értéke kerül: a help-szöveg kiírásakor ez fogja majd meghatározni a kiíratandó karakterek számát. Innen általános hívás esetén egy előreugrás történik. Az átugrott részen a program meghív egy szubrutint, mely megvizsgálja, hogy a hívó füzér első szava "HUN"-e. Ha nem, alacsony zero-flaggel tér vissza. Ezután a C regiszter értékét (akciókód) nullára állítja be. Ezért szükséges általános help-hívás esetén kihagyni, mivel ekkor a többi bővítésnek is válaszolnia kell. A továbbiakban a program az alternatív regiszterekkel dolgozik, hogy általános hívás esetén is megőrződjenek a BC és a DE regiszterpárok értékei. Majd a 255-ös csatornára a "Blokk írása" EXOS-funkcióval (BLK_OUT, kód: 8) kiírja a HELPSTR címkénél kezdődő szöveget az A regisztertől függő hosszon. A rutin végén az A regiszter értékét kinullázza, jelezve, hogy a program sikeresen fejeződött be.

A CHEK_ID nevű szubrutin vizsgálja meg a parancs-stringet. Bemenő paraméterként a B regiszterben a parancsfüzér hosszát, a DE regiszterpárban a parancs-string címét várja. A rutin azt vizsgálja, hogy a parancsfüzés első szava a bővítés neve-e. Ha az, akkor magas zero-flaggel tér vissza. Itt arra kell vigyázni, hogy a bővítés nevét (mely a HUN_ID címkénél helyezkedik el) nagybetűs karakterekkel tároljuk, különben a vizsgálat mindig sikertelen lesz!

Az érdemi rutin a RUN címkénél kezdődik. Itt állítódnak át a karakterképek. Először névellenőrzés történik a CHEK_ID rutinnal, majd a rutin meghívja az EXP_INQ EXOS-funkciót "BRD" parancs-stringgel. Ennek eredményeképpen a gép német üzemmódra áll be. A ST_CHRS-nél kezdődő rutinrészlet beállítja az IX regiszterpár értékét és a VID_EDIT EXOS-rendszerváltozó értékét átírja a COM_CHAN BASIC rendszerváltozóba (248:521=0F8H:0209H). A program további részei a CHR_SHP táblázat szerint átírja a megfelelő karaktereket. A programhurok végét a táblázat utolsó, 0 értékű byte-ja jelzi. A program végén IX felveszi az eredeti értékét, az állapotbyte (A regiszter) és a C regiszter 0 értéket vesz fel, jelezve a feladat sikeres végrehajtását.

A karakterképek átállítása a WRITE_CHAR nevű BASIC könyvtári rutinnal történik (kód: 56=38H). A könyvtári rutingyűjteménybe az RST 10H assembly utasítással kell belépni, és a további byte-ok mindegyike egy-egy rutint szimbolizál. A 0-ás kódú rutin (END_LIB) a kilépést végzi a könyvtárból. A WRITE_CHAR rutin a COM_CHAN rendszerváltozó által mutatott csatornára írja ki az A regiszterben átadott karaktert. A helyes működéséhez az IX regiszterpár értékének 0200H-nak (512) kell lennie. A karakterek átállítása ún. escape-szekvenciával történik. Ez azt jelenti, hogy egy létező video-csatornára egy olyan karaktersorozatot kell kiírni, melynek első karaktere ESC karakter (kód: 27=1BH). A második karakter az escape-szekvencia funkcióját meghatározó kód.

A további karakterek paraméterek. Bizonyos BASIC gépi feltételek gépi kódból csak így programozhatók. A jelen esetben a szekvencia kódja 4BH (75), amit még 10 karakternek kell követnie: ezek közül az első a karakter kódja, a többi kilenc a karakterkép bitmátrixa. A táblázatban ez a tíz byte tárolódik le. A program ezt a tíz byte-ot (egy táblaelemet) egy ciklussal olvassa be és írja ki, majd megvizsgálja, hogy a következő táblaelem első byte-ja nulla-e. Ha nem, ismételten kiír egy ESC karaktert és egy 4BH karaktert, majd a következő táblaelem byte-jait. Látható, hogy a táblázat bővítéséhez a programon belül semmilyen változtatást sem kell eszközölni, csupán a táblázatot továbbírni és a végére egy nulla byte-ot tenni.

Tippek a program módosításához

Ha azt akarjuk, hogy bővítésünk a betöltés után azonnal aktivizálódjon, az akciókódok vizsgálatát végző rutinrészletet írjuk át úgy, hogy 8-as akciókód esetén a RUN címke utáni "LD DE, BRD_ID" utasításra ugorjon. (Vagyis úgy, mint 2-es akciókód esetén, csak kihagyva a parancs-string vizsgálatát.)

A help-üzenet szövegének megváltoztatásakor ügyeljünk a kiírandó karakterek számának átírására, és arra, hogy a sorokat mindig egy CR (kocsi-vissza, kód: 13=0DH) és egy LF (soremelés, kód: 10=0AH) karakter együtt zárja le!

Ha esetleg angol, illetve német üzemmódtól függően különböző üzeneteket szeretnénk kiírni, segítségünkre lehet az, hogy a 90H (144) számú EXOS rendszerváltozó értéke német üzemmód esetén 0, míg angol üzemmód esetén 255 értékű.

Ha valaki a karakterek kiosztását, vagy a karakterek képét meg szeretné változtatni, az a CHR_SHP táblázat átírásával könnyen tudja őket módosítani.

Mivel a program kétnyelvű gépre készült, az angol nyelvű gépek gazdáinak módosítaniuk kell a programot. Minthogy a két géptípus különbségeit nem ismerem, ehhez biztos tippet nem tudok adni. A legszembetűnőbb eltérés talán az, hogy az angol gépek nem rendelkeznek a BRD beépített bővítéssel, így ehelyett ajánlott a §§FONT nevű speciális EXOS-funkciót meghívni. Azoknak, akik nem rendelkeznek részletesebb EXOS-funkcióleírással, hasznos lehet a speciális funkció hívásának leírása: az A regiszterbe egy élő video-csatorna számát kell írni, a B regiszterbe pedig az alfunkció kódját, ami §§FONT esetén 4; és ezeket beállítva kell meghívni a "Speciális feladat végrehajtása" nevű EXOS-funkciót (SPEC-FUNC, kód: 11=0BH). Ez az alfunkció alaphelyzetbe állítja a karakterkészletet. Figyelni kell még arra is, hogy rövid Ö, ő, Ü, ü karakterek képét is ki kell angol gép esetén alakítani.

Az általam említett rutin- és rendszerváltozó nevek, terminológiák, és a szakirodalmakban szereplők között eltérések lehetnek. Ugyanis szakirodalmat nem használtam fel,

Racskó Tamás

2. lista

```
            org     144dh
file_name   equ     1439h
head        equ     12dbh
prg_lng     equ     0142h
file        equ     12ebh
sa_head     ld      a,6ah
            ld      de,file_name
            exos    02h
            or      a
            jr      nz,sa_hd_1
            ld      a,6ah
            ld      bc,0010h
            ld      de,head
            exos    08h
sa_hd_a     ld      l,a
            ld      h,00h
            ret
sa_file     ld      a,6ah
            ld      bc,prg_lng
            ld      de,file
            exos    08h
            or      a
            jr      nz,sa_fl_1
            ld      a,6ah
            exos    04h
sa_fl_1     ld      l,a
            ld      h,00h
            ret
            end
```

1. lista

```
;  MAGYAR EEKEZETES KARAKTER-KESZLET
;        (C) 1988 RACSKO TAMAS
;
                org     0c00ah
comchan         equ     0209h
comhelp         equ     1fh
hunhelp         equ     4fh
;
start           ld      a,c
                sub     02h
                jr      run
                dec     a
                ret     nz
help            ld      a,(de)
                or      a
                ld      a,comhelp
                jr      z,hlp_2
                call    chek_id
                ret     nz
                ld      a,hunhelp
                ld      c,0
hlp_2           exx
                ld      c,a
                xor     a
                ld      b,a
                dec     a
                ld      de,helpstr
                exos    08h
                exx
                xor     a
                ret
;
chek_id         push    de
                push    bc
                ld      hl,hun_id
                ld      a,b
                inc     b
id_loop         cp      (hl)
                jr      nz.id_end
                inc     hl
                inc     de
                ld      a,(de)
                djnz    id_loop
id_end          pop     bc
                pop     de
                ret
;
hun_id          db      03h
                defm    'HUN'
brd_id          db      03h
                defm    'BRD'
;
run             call    chek_id
                ret     nz
                ld      de,brd_id
                exos    1ah
st_chrs         push    ix
                ld      ix,0200h
                ld      bc,001dh
                exos    10h
                ld      a,d
                ld      (comchan),a
                ld      hl,chr_shp
st_nxt          ld      a,(hl)
                cp      0
                jr      z,st_end
                ld      a,1bh
                rst     10h
                db      38h,00h
                ld      a,4bh
                rst     10h
                db      38h,00h
                ld      b,0ah
st_loop         push    bc
                push    hl
                ld      a,(hl)
                rst     10h
                db      38h,00h
                pop     hl
                pop     bc
                inc     hl
                djnz    st_loop
                jr      st_nxt
st_end          pop     ix
                xor     a
                ld      c,a
                ret
helpstr         defm    'HUN   Hungarian'
                defm    ' character-set'
                db      0dh,0ah
                defm    '      version 1.0'
                db      0dh,0ah
                defm    '      '
                db      80h
                defm    ' 1988 Racsko Tam'
                db      95h,'s'
                db      0dh,0ah,0dh,0a,
;
chr_shp         db      98h
                db      0ch,18,h3ch,66h,7eh
                db      60h,3ch,00h,00h
                db      88h
                db      0ch,18h,7eh,60h,7ch
                db      60h,7eh,00h,00h
                db      99h
                db      0ch,18h,00h,38h,18h
                db      18h,3ch,00h,00h
                db      89h
                db      0ch,08h,3ch,18h,18h
                db      18h,3ch,00h,00h
                db      96h
                db      0ch,18h,3ch,66h,66h
                db      66h,3ch,00h,00h
                db      86h
                db      0eh,3ch,66h,66h,66h
                db      66h,3ch,00h,00h
                db      93h
                db      36h,24h,3ch,66h,66h
                db      66h,3ch,00h,00h
                db      83h
                db      36h,3ch,66h,66h,66h
                db      66h,3ch,00h,00h
                db      9ah
                db      0ch,18h,00h,66h,66h
                db      66h,3ch,00h,00h
                db      8ah
                db      18h,42h,66h,66h
                db      66h,3ch,00h,00h
                db      94h
                db      36h,24h,00h,66h,66h
                db      66h,3ch,00h,00h
                db      84h
                db      33h,44h,66h,66h,66h
                db      66h,3ch,00h,00h
                db      00h
;
end
```

3. lista

```
 1 PROGRAM "HUN_MODE.BAS"
 2 !
 3 ! MAGYAR UZEMMOD ENTERPRISE-ON
 4 !       1988 Racsko Tamas
 5 !
 6 ALLOCATE 450
 7 NUMERIC PRG_LNG,A
 8 STRING NL$
 9 TEXT: LET PRG_LNG=322:LET NL$=CHR$(13)&CHR$(10)
10 CODE HEAD=HEX$("00,06)&WORD$(PRG_LNG)&
     HEX$(00,00,00,00,00,00,00,00,00,00,00,00")
11 CODE FILE=HEX$("79,D6,02,28,37,3D,C0")
12 CODE HELP=HEX$("1A,B7,3E,1F,28,08,CD,2C,C0,
     C0,3E,4F,0E,00")
13 CODE HLP_2=HEX$("D9,4F,AF,47,3D,
     11,84,C0,F7,08,D9,AF,C9")
14 CODE CHEK_ID=HEX$("D5,C5,21,3E,C0,78,04")
15 CODE ID_LOOP=HEX$("BE,20,05,23,13,1A,10,F8")
16 CODE ID_END=HEX$("C1,D1,C9")
17 CODE HUN_ID=CHR$(3)&"HUN"
18 CODE BRD_ID=CHR$(3)&"BRD"
19 CODE RUN=HEX$("CD,2C,C0,C0,11,42,C0,F7,1A")
20 CODE ST_CHRS=HEX$("DD,E5,DD,21,00,02,01,1D,00,
     F7,10,7A,32,09,02,21,D3,C0")
21 CODE ST_NXT=HEX$("7E,FE,00,28,19,3E,1B,
     D7,38,00,3E,4B,D7,38,00,06,0A")
22 CODE ST_LOOP=HEX$("C5,E5,7E,D7,38,00,E1,
     C1,23,10,F5,18,E2")
23 CODE ST_END=HEX$("DD,E1,AF,4F,C9")
24 CODE HELPSTR=""
25 CODE="HUN   Hungarian character-set"&NL$:
     !3+1 BETUKOZ
26 CODE="      Version 1.0"&NL$:!6+1 BETUKOZ
27 CODE="      1988 Racsko Tamas"&NL$&NL$:
     !6+1+1+1 BETUKOZ
28 CODE CHR_SHP=""
29 CODE=HEX$("98,0C,18,30,66,7E,60,3C,00,00")
30 CODE=HEX$("88,0C,18,7E,60,7C,60,7E,00,00")
31 CODE=HEX$("99,0C,18,00,38,18,18,3C,00,00")
32 CODE=HEX$("89,0C,08,3C,18,18,18,3C,00,00")
33 CODE=HEX$("96,0C,18,3C,66,66,66,3C,00,00")
34 CODE=HEX$("86,0E,3C,66,66,66,66,EC,00,00")
35 CODE=HEX$("93,36,24,3C,66,66,66,3C,00,00")
36 CODE=HEX$("83,36,3C,66,66,66,66,3C,00,00")
37 CODE=HEX$("9A,0C,18,00,66,66,66,3C,00,00")
38 CODE=HEX$("8A,18,42,66,66,66,66,3C,00,00")
39 CODE=HEX$("94,36,24,00,66,66,66,3C,00,00")
40 CODE=HEX$("84,33,44,66,66,66,66,3C,00,00")
41 CODE=HEX$("00,00,00,00,00,00,00,00,00,00,00,00,00")
42 CODE FILE_NAME=CHR$(8)&"HUN_MODE"&
     HEX$("20,20,20,20,20,20,20,20,20,20,20,20")
43 CODE SAVE_HEAD=HEX$("3E,6A,11")&WORD$(FILE_NAME)
     &HEX$("F7,02,B7,20,0A,3E,6A,01,10,00,11")
     &WORD$(HEAD)&HEX$("F7,08,6F,26,00,C9")
44 CODE SAVE_FILE=HEX$("3E,6A,01")&WORD$(PRG_LNG)
     &HEX$("11")&WORD$(FILE)
     &HEX$("F7,08,B7,20,04,3E,6A,F7,04,6F,26,00,C9")
45 PRINT AT 6,1:"Helyezze be a kazettat"&chr$(241):PRINT"s
     kapcsoljon felvetelre!"&CHR$(241)
46 PRINT"Ha kesz, nyomja meg az <ENTER>-t!"&CHR$(241):PRINT
47 DO
48 LOOP UNTIL INKEY$=CHR$(13)
49 PRINT"Felvetel indul!"&CHR$(241):PRINT
50 PRINT"A fej kimentese..."
51 LET A=USR(SAVE_HEAD,0)
52 CALL EX_STAT
53 PRINT"A file kimentese..."
54 LET A=USR(SAVE_FILE,0)
55 CALL EX_STAT
56 PRINT:PRINT:PRINT"A magyar uzemmod programja
     felveve!":PRINT
57 END
58 DEF EX_STAT
59   IF A=0 THEN
60     PRINT TAB(5);"Rendben lezajlott!":PRINT
61   ELSE CAUSE EXCEPTION 9000+A
62   END IF
63 END DEF
```

Monitor

Előbb-utóbb minden ENTERPRISE-tulajdonos szembetalálja magát azzal, hogy az otthonában található fekete-fehér, vagy színes TV-készülék hosszabb távon nem jól használható sem a játékhoz, főleg nem a programozáshoz. Ez a cikk az ENTERPRISE és a kereskedelemben kapható különböző monitorok összekapcsolásával kapcsolatos tudnivalókat igyekszik összefoglalni azok számára, akik elszánták magukat saját display vásárlására.

A boltokban óriási számú monitor között választhatunk. A probléma csak az, vajon az illető monitor összekapcsolható-e gépünkkel vagy sem. Sok esetben az eladó sem tud konkrét segítséget nyújtani; ennek egész egyszerűen az az oka, hogy sajnos ebben az esetben sem egyetlen világszabvány létezik, hanem legalább egy tucat különféle, szinte minden gyártónál más és más. Nem csupán a csatlakozók különböznek egymástól, hanem a képi információt hordozó jelek is. Ezért tehát egy számítógép összekapcsolása egy monitorral elég komplikált feladat is lehet, s előfordulhat, hogy egyáltalán nem is sikerül.

Az ENTERPRISE-tulajdonosok viszonylag kedvező helyzetben vannak; gépük gyakorlatilag egyetlen a piacon, amely bármilyen monitorral összeköthető.

Az egyetlen komolyabb nehézséget a gép hátoldalán található, meglehetősen különleges kinézetű monitor-csatlakozó. Mint a többi EP-csatlakozás, ez is teljesen egyedi, egyetlen más gépnek sincs ilyen. A kereskedelemben szerencsére kapható ilyen csatlakozással ellátott kábel, amelynek másik vége valamely ismertebb monitorcsatlakozó-szabvánnyal van ellátva. Mégsem haszontalan, ha most ismertetjük e kábelek jellemzőit, mert előfordulhat, hogy a készen kapott mégsem működik a mi konfigurációnkkal.

Ahhoz, hogy saját szükségleteinknek megfelelő kábelt tudjunk építeni, tudnunk kell, az EP monitorkimenetén milyen jelek találhatóak. Tudnunk kell azt, ezek közül melyekre van szükségünk. Az 1. táblázat az EP jelkiosztását ismerteti, a hozzájuk tartozó színkódokkal. A B1-el jelölt csatlakozás a bal felső, az A1-el jelölt a bal alsó sarkot jelenti.

Ennek a táblázatnak, illetve a következőkben összefoglalt főbb tudnivalókkal már nekiláthatunk a kábel összeszereléséhez.

1. Az ún. cynch-csatlakozás: többnyire a monochrom, azaz egyszínű monitoroknál használják, ahol a teljes képi jelet egyetlen szálon továbbítják. Bekötési módja látható a 1. ábrán. Ha a monitornak beépített hangszórója is van, a hangot további jelhez egy második csatlakozásra lesz szükségünk. Ebben az esetben a jobb-, illetve baloldali jelet közösen forrasszuk rá a csatlakozásra, ellenkező esetben csak az egyik hangcsatorna lesz hallható a monitor hangszóróján. A föld-jelet (A2) mindkét esetben a dugó külső palástjához kell forrasztani.

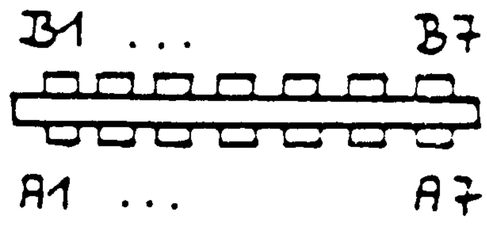
2. A 6- vagy 7-pólusú DIN-szabvány: a két szabvány nemcsak külsőleg tér el egymástól, hanem a különböző jeleket hordozó szálak számában is (2. és 3. ábra). Ezeket a csatlakozókat többnyire az RGB, illetve TTL-színes monitoroknál használják. Ebben az esetben az ENTERPRISE megfelelő pontjaihoz kell csatlakoznunk. Ha a színes kép életlen, vagy torz, akkor forrasszunk a piros, zöld és kék jelet hordozó szálakhoz egy-egy 100 ohmos, 1/8 watt teljesítményű ellenállást. Ha ezekután a kép fut a képernyőn, a CSYNC szálhoz is kell egy további 100 ohmos ellenállás. Ezekre azért van szükség, hogy az ENTERPRISE viszonylag magas jelszintjeit az érzékenyebb monitoréhoz igazítsák. Az ellenállások a DIN-csatlakozó dobozában is elférnek (4. ábra).

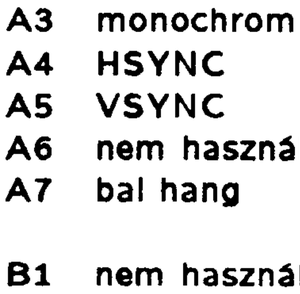
3. SCART-szabvány (5. ábra): ez a csatlakozás az utóbbi időkben hódított tért, főleg a videomagnetofonok miatt, néhány színes TV-készülék is már ezzel a 21 pólusú csatlakozással rendelkezik. Ha ilyen színes TV-nk van, akkor monitorra egyelőre nem is lesz szükségünk, mert a kívánt képminőséget ezzel is elérhetjük. Az összekötésnél ügyeljünk arra, hogy valamennyi föld-jelet az EP A2/B2-es pontjához forrasszuk. A képélességre és a képstabilitásra ugyanaz vonatkozik, mint a 2. pontban. A SCART-csatlakozásnak van még egy különlegessége: a 16 póluson 12 voltos feszültségnek kell lennie, ez kapcsolja át a TV-készüléket monitor üzemmódba. Az EP-n megvan ez a feszültség, a B6-os pólust kell tehát a SCART 8-as pólusához forrasztanunk. Ezt a 8-as pólust ezután egy 470 ohmos, 1/8 watt teljesítményű ellenállással kössük össze a 16-ossal (6. ábra).

Ezek után feltehetően elboldogulunk már magunk az ENTERPRISE és a monitor összekapcsolásával, még akkor is, ha ne adj' isten egy negyedikféle csatlakozással rendelkezik is.

1. táblázat

	Funkció	Színkód
A1	zöld	zöld
A2	föld	fekete
A3	monochrom kompozit	lila
A4	HSYNC	nincs bekötve
A5	VSYNC	nincs bekötve
A6	nem használt	
A7	bal hang	narancs
B1	nem használt	
B2	föld	fekete
B3	kék	kék
B4	vörös	vörös
B5	CSYNC	barna
B6	kapcsoló fesz. 12V	sárga
B7	jobb hang	fehér



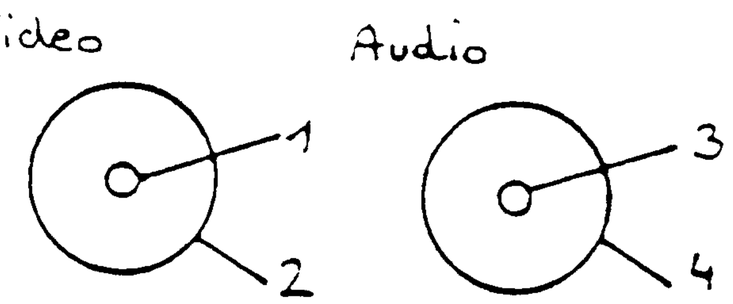


2. ábra

	Funkció	Színkód
1	vörös	vörös
2	zöld	zöld
3	kék	kék
4	CSYNC	barna
5	föld	fekete
6	nem használt	

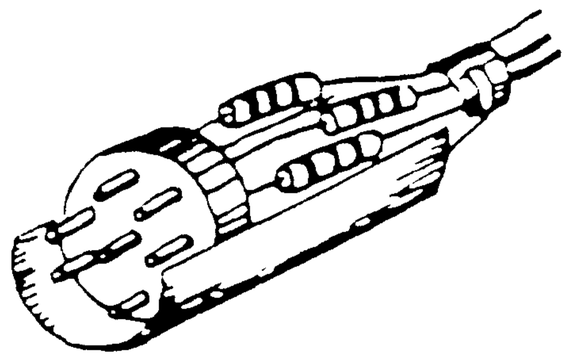
3. ábra

	Funkció	Színkód
1	föld	fekete
2	CSYNC	barna
3	zöld	zöld
4	nem használt	
5	kék	kék
6	hang	narancs/fehér
7	vörös	vörös



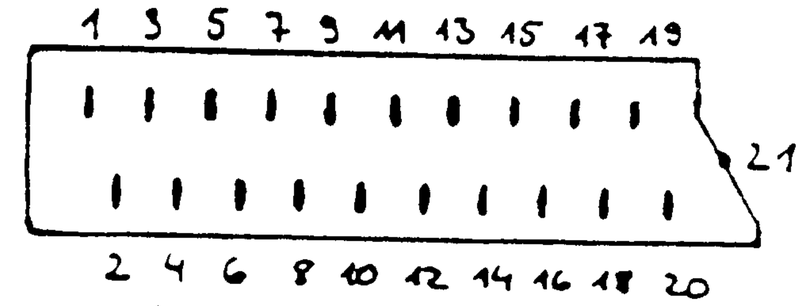
1. ábra

	Funkció	Színkód
1	Video kompozit	lila
2	föld	fekete
3	hang	narancs/fehér
4	föld	fekete

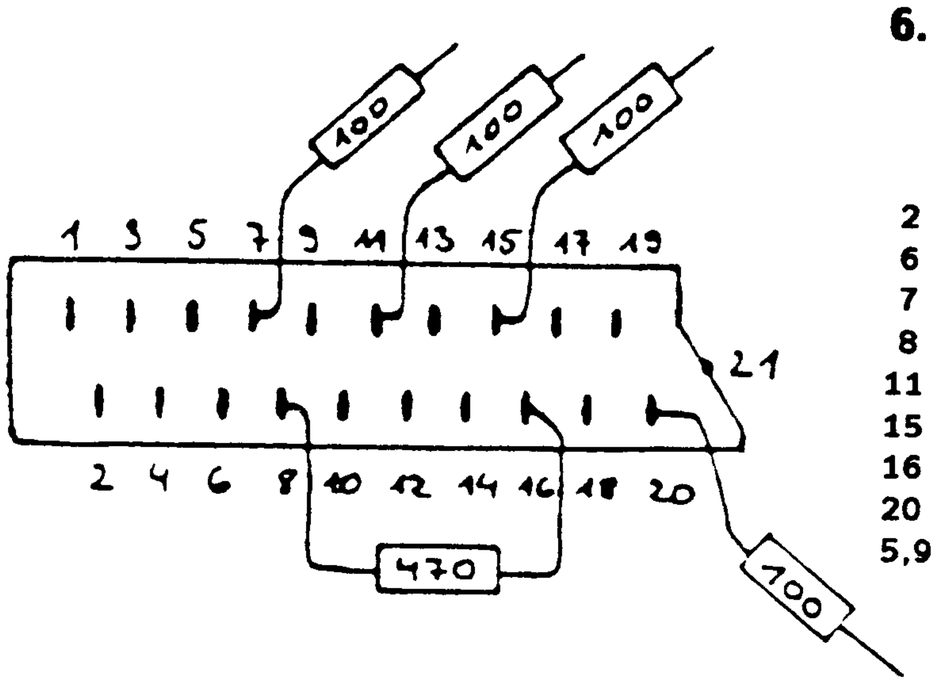


4. ábra

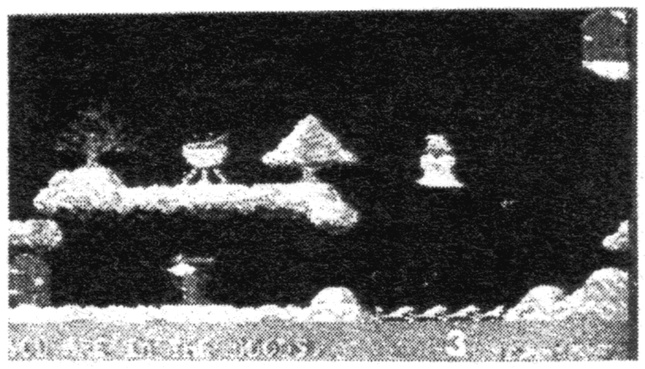
5. ábra

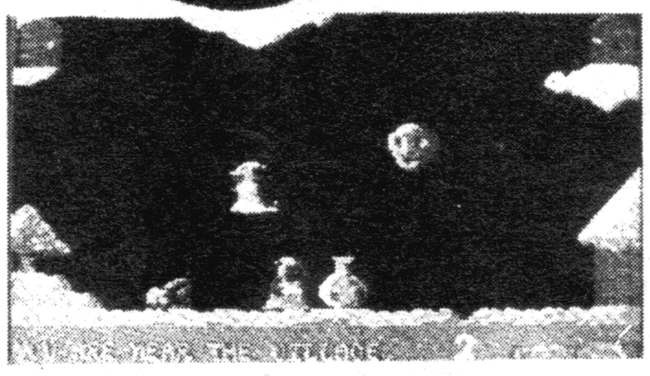


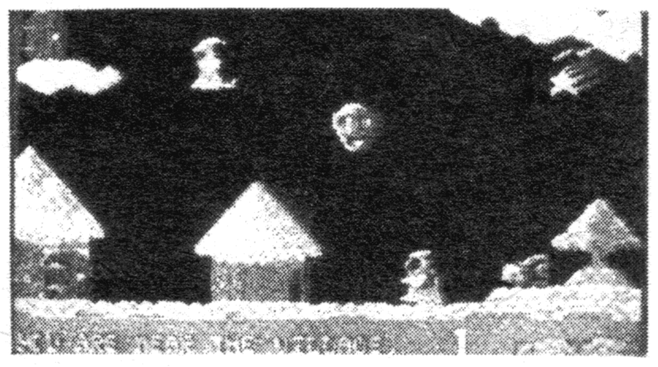
6. ábra

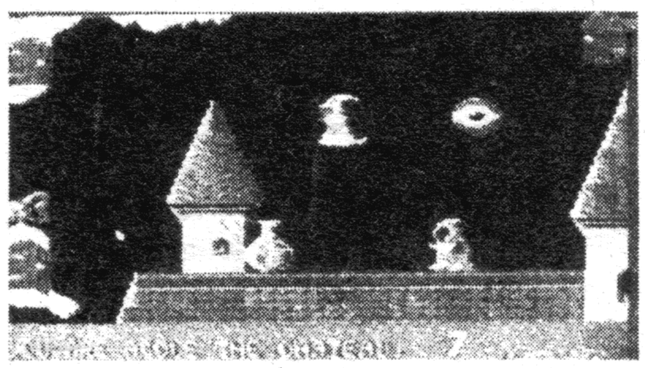


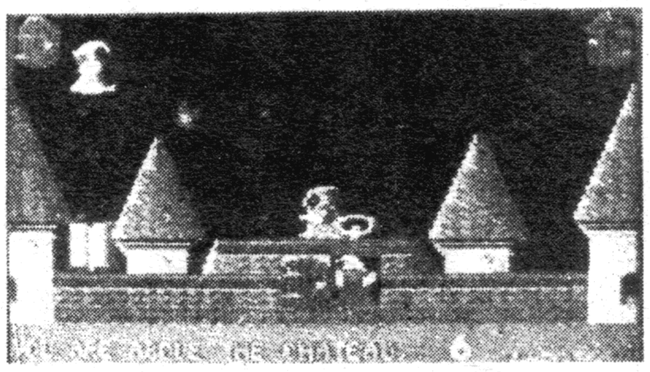
	Funkció	Színkód
2	Jobb hang	fehér
6	bal hang	narancs
7	kék	kék
8	kapcsoló fesz. 12V	sárga
11	zöld	zöld
15	vörös	vörös
16	kapcsoló bemenet	
20	CSYNC	barna
5,9	föld	fekete

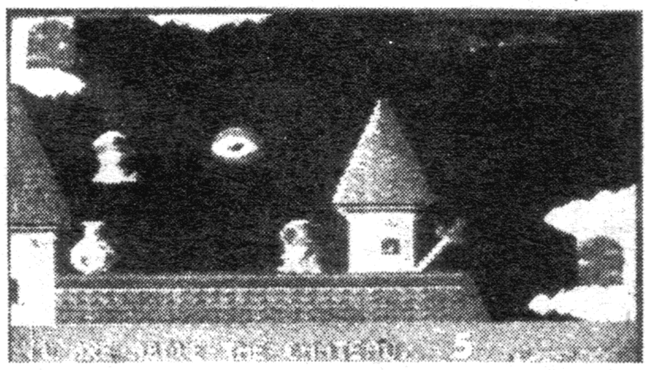


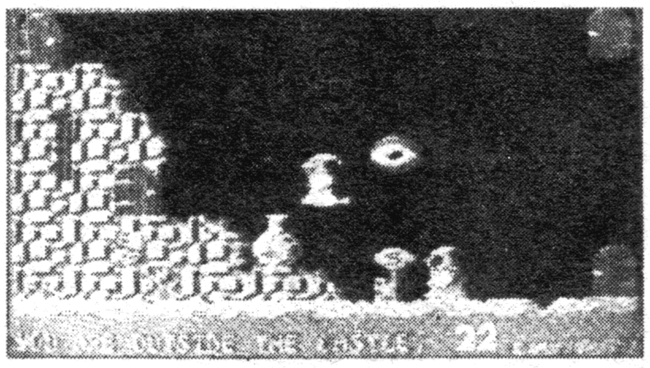


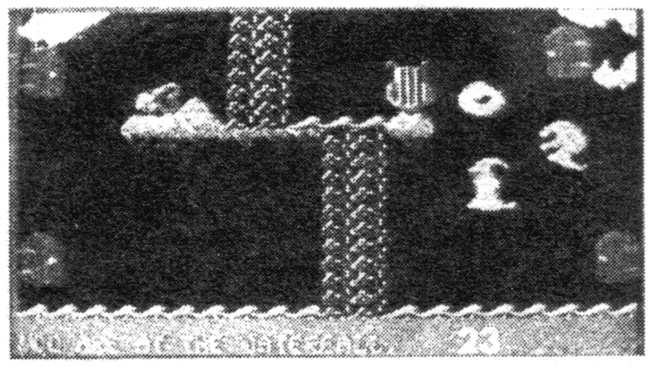


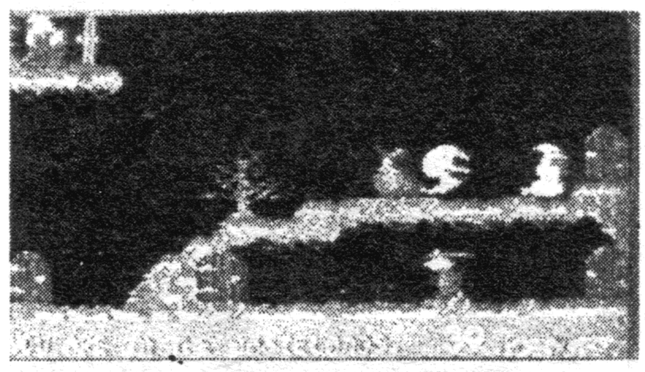


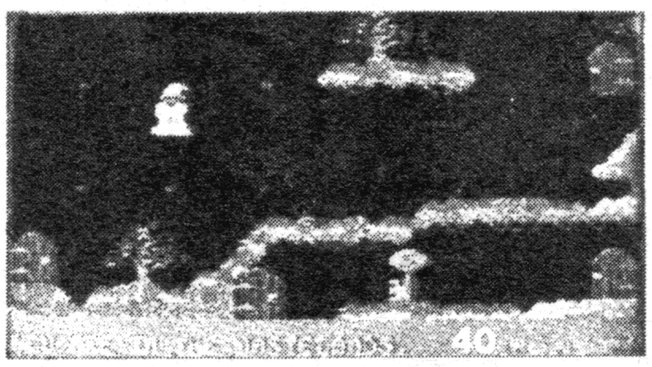


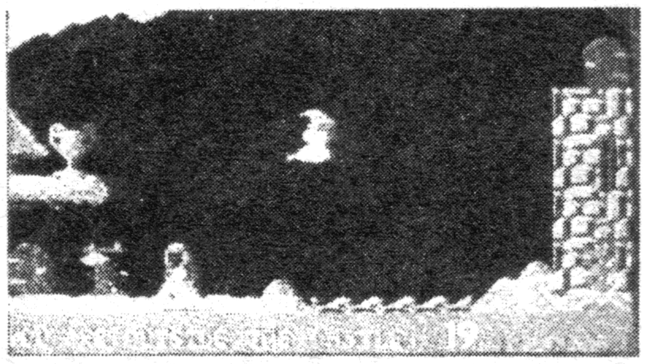


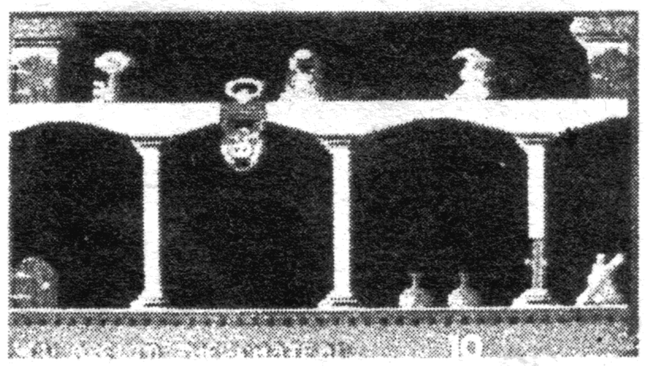


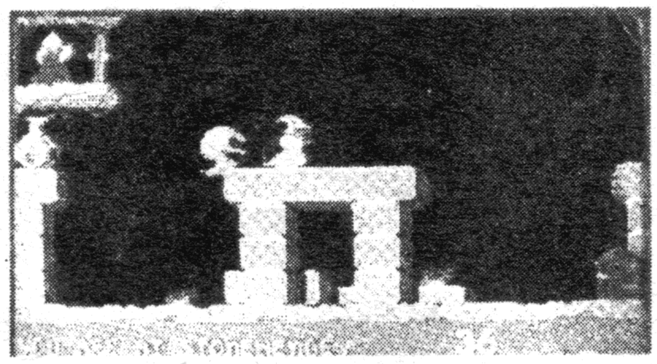


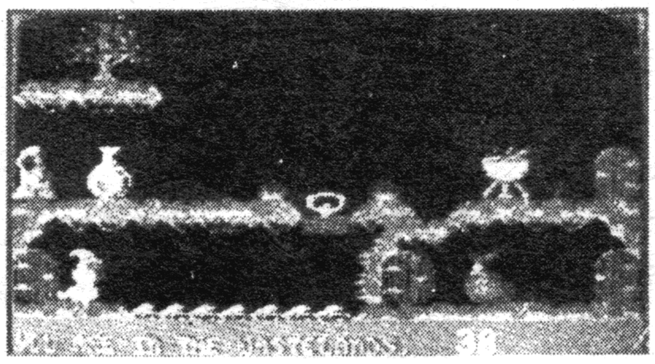


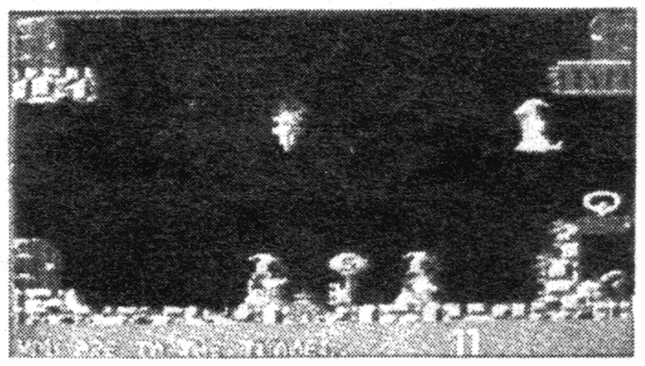


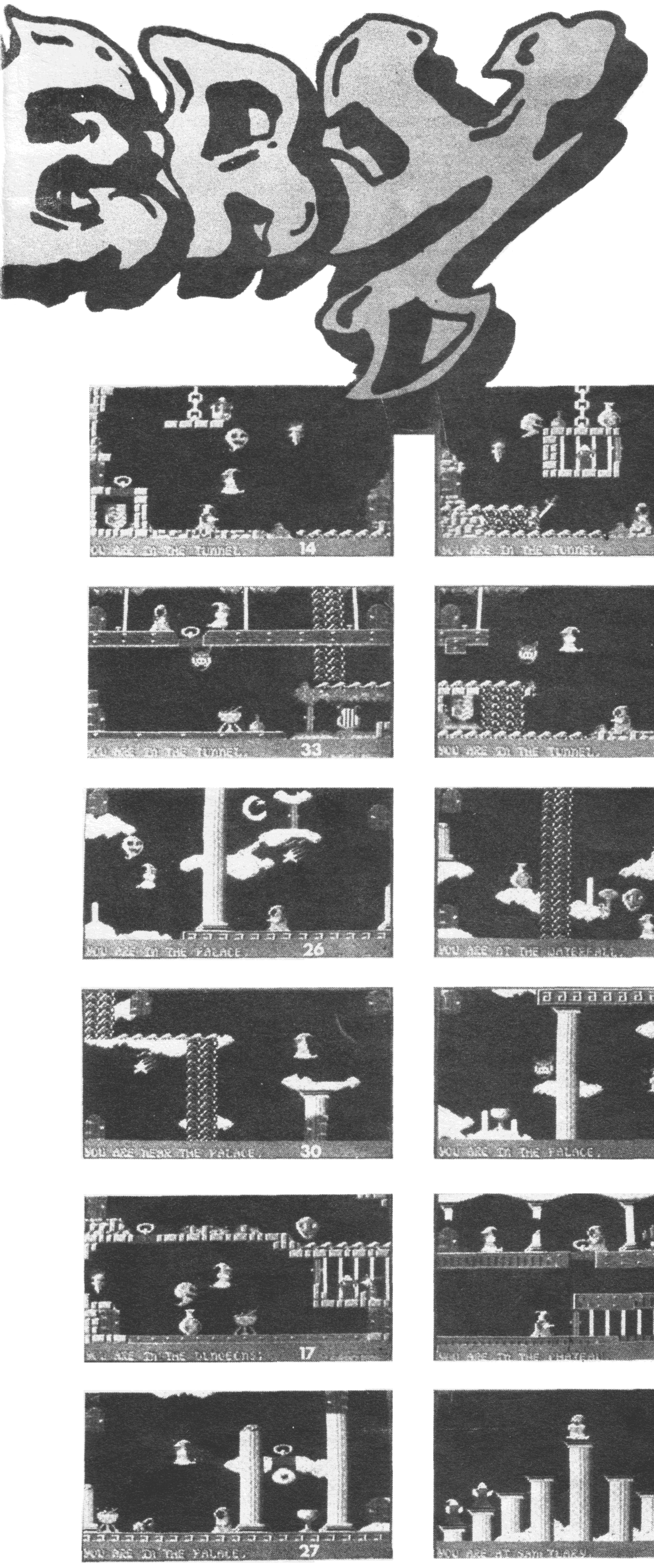











YOU ARE IN THE TUNNEL. 14
YOU ARE IN THE TUNNEL. 13
YOU ARE IN THE TUNNEL. 12
YOU ARE IN THE TUNNEL. 33
YOU ARE IN THE TUNNEL. 32
YOU ARE IN THE ENTRANCE. 16
YOU ARE IN THE PALACE. 26
YOU ARE AT THE WATERFALL. 25
YOU ARE AT THE TUNNEL DEATH. 35
YOU ARE NEAR THE PALACE. 30
YOU ARE IN THE PALACE. 29
YOU ARE IN THE PALACE. 28
YOU ARE IN THE DUNGEONS. 17
YOU ARE IN THE PALACE. 9
YOU ARE IN THE DUNGEONS. 8
YOU ARE IN THE PALACE. 27
YOU ARE AT SAM TOWER. 31
YOU ARE IN THE DUNGEONS. 18

Ötféle halált okozó szörny van, melyeknek bármelyikét megölhetjük valamelyik fegyverrel, amelyeket a játéktér különböző részein összeszedhetünk. Van egy varázslatos zsákok, lövöldözős csillag, amely megöli némelyiket, de azt ki kell találni, hogy melyik melyiknek a fegyvere. Sok más tárgy is felhasználható; így találunk kulcsokat, üvegeket és a gyertyatartót, melyek kinyithatják a bezárt szobákat és a csapóajtókat.

Ezekkel a mágikus tárgyakkal kiszabadíthatod az összes varázslót, de nagyon gyorsan kell mozognod, mert energiád rohamosan csökken.

Például a Kastélyban (Chateau) amint meglátod a varázsbárdot, szükséged van egy kulcsra, hogy kinyisd az ajtót és a kulcsot megkaparintva felveheted a mágikus bárdot, ám csak egy kulcs van a szobában. Tehát kell egy másik kulcs is, ezért a bal felső ajtón kell kimenned. A Kastély közelében szemben találod magad két szörnnyel, de találsz egy kulcsot és egy gyertyatartót. Menj vissza a varázsbárdhoz, az ajtót kinyitja a gyertyatartó. Amikor kinyílt az ajtó, visszamehetsz, felkaphatod a kulcsot, majd vissza abba a szobába, amelyben a varázsbárd található. Nyisd ki a csapóajtót, kapd fel a tartalékkulcsot, a bárdot és szökj ki a balkéz felöli ajtón az Alagútba. Tedd a bárdot a baloldali alsó ajtóhoz. Az Alagútban két szörnnyel találkozol, valamint találsz egy kulcsot, egy folyót és egy energiát adó üstöt, de sajnos két ellenféllel is. Így innen reménytelen vállalkozás kiszabadítani a varázslót, ezért lógj meg minél gyorsabban. Azon az ajtón keresztül elhagyhatod az alagutat, amin bejöttél.

Könnyebb kiszabadítani azt a varázslót, amelyik a Kastély feletti szobában van, ha megtalálod azt a betüskönyvet, amelyik ugyanabban a szobában van. Ha ezt megtaláltad, felfed egy másik ajtót, amelyen azonban nem tudsz kimenni. Miután elhagytad ezt a szobát a tetőn jobbra levő ajtón, egy olyan szobában találod magad, ahol lebeg egy szem, egy harcos, egy kard és egy üveg található. A harcost a karddal megölheted, ha akarod, de kifelé menet fel kell venned az üveget.

Így a Kastély közelében vagy ismét, két szörnnyel és két tárggyal (az üveggel és az aranyserleggel), de hogy ki tudd nyitni a varázsajtót, tovább kell vinned az erdő felé az üveget. Az üveg kinyitja az ajtót, utána a lövöldözős csillaggal öld meg mindkét szörnyet, töltsd fel az energiádat az üstnél, de nagyon fontos, hogy felkapd a gyertyatartót. Most már elindulhatsz a varázsajtó felé és – Tárulj Szezám! – megjelenik előtted egy másik varázsló, akit kiszabadíthatsz azzal az aranyserleggel, melyet két szobával előbb felvettél.

Mindegyik varázsló kiszabadítható, de csak a megfelelő tárgygyal. Például a Stonehenge-ben lévő barátodat a Varázsholddal tudod kiszabadítani. A hárfa szabadítja ki a palota kupolájában fogva tartott varázslót, és egy borospohár segít kihozni a varázslót a Wastelandsből.

Az utolsó két varázslót a legnehezebb kiszabadítani.

A kastélyban az egyik szobában található egy tekercs és egy bilincs. Vedd fel a bilincset, hagyd el a Kastélyt a jobboldali ajtón. Így ki tudod nyitni az ajtót, amely félúton van a sziklafalon. Amint kinyitottad az ajtót, térj vissza a Kastélyba és vedd fel a tekercset. Majd vissza a sziklaajtón keresztül a tömlöcbe, a tekerccsel szabadítsd ki a varázslót, használd a bárdot, amely megöli a szörnyet, amelyik ebben a szobában van. Menj ki a baloldali ajtón, menetközben vedd fel az üveget. Ez az üveg nyitja a csapóajtót, de előtte a karddal öld meg a harcost, aki az utadban áll. Kapd fel a kulcsot, mely a bal felső sarokban található, és a jobb alsó sarkon lévő ajtón keresztül menj be a Kinzókamrába. A kulccsal nyisd ki a csapóajtót. A varázszsákkal előtte tisztítsd meg a termet az ellenségektől. Hogy kiszabadíthasd a varázslót, vissza kell menned az úton a Kastélyba a címerért, aztán újra térj vissza a tömlöcbe, ahol felvetted a kulcsot.

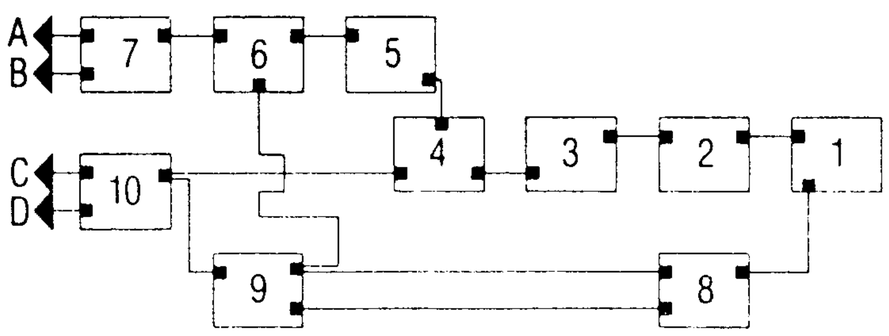
A baloldali ajtón keresztül menj be Wastelandbe, ahol megtalálod azt a koronát, amelyik kiszabadítja az utolsó varázslót. De ahhoz, hogy a tömlöcbe vissza tudj térni, előbb ki kell nyitnod a címerrel az ajtót. Ezután már csak le kell tenned a koronát a Kinzókamrába és a varázsló máris szabad.

Bármilyen sorrendben is, de mind a nyolc varázslót ki kell szabadítanod, s utána ne felejts el elmenni a Szentélybe, hogy viszontláthasd kedvenc cimboráidat, a varázslókat. Ezzel teljesítetted is a játékot.

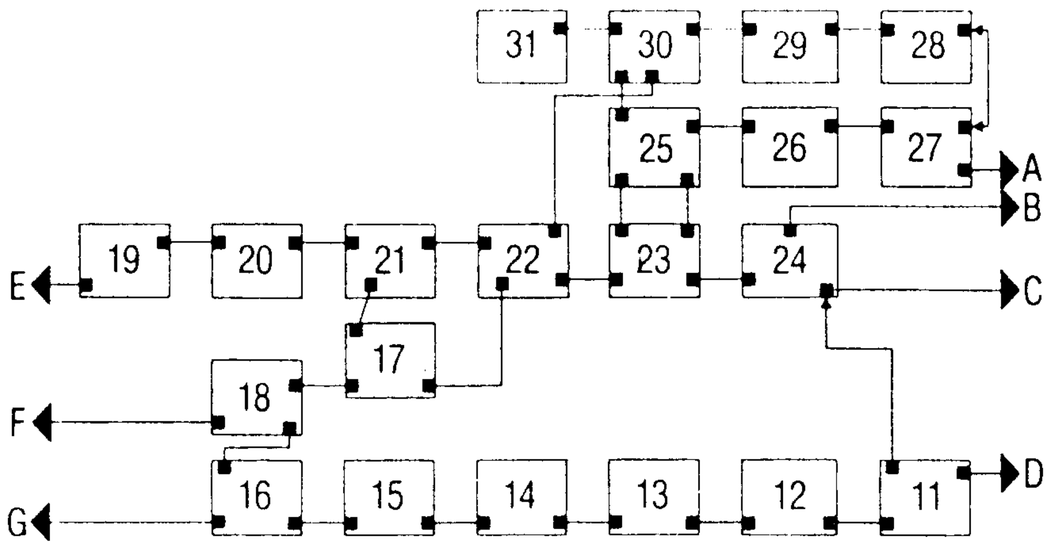
Ahhoz, hogy minél nagyobb pontszámot érhess el, a lehető legtöbb ellenséget kell megölnöd, lehetőleg minden termet járj végig, s lehetőleg vegyél fel minél több tárgyat.

Jó szórakozást!

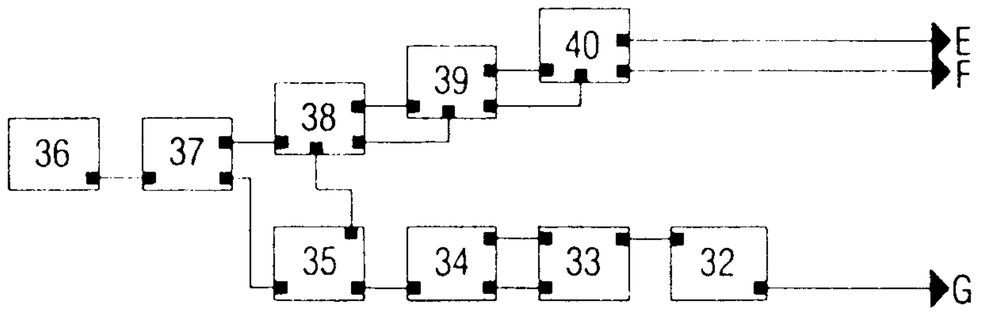
SECTION 1



SECTION 2



SECTION 3



Itt lenne az olvasói levelek helye. Kérünk tehát Benneteket, ha valami problémátok, kérdésetek, ötletetek van akár ENTERPRISE-programokkal, akár a lapban megjelent témákkal kapcsolatban, írjatok nekünk. A közérdekű levelekre e hasábokon válaszolni fogunk. Írjátok meg, miről szóljanak a következő lapszámok, s szívesen látunk kész cikkeket is.

Reméljük, a következő hónapokban elárasztanak bennünket a levelek s nem lesz üres ez a hasáb!

ROUNDSMAN

Útvonal minimalizáló program

A ROUNDSMAN (árúkihordó) feladata síkban véletlenszerűen elhelyezkedő célpontok körbejárása minden pont egyszeri érintésével a lehető legrövidebb útvonalon.

A program az ENTERPRISE-128 számítógép IS-BASIC programnyelvén ZZZIP compileres fordításhoz készült, az útvonalat MONTE-CARLO-szerű találgatásos módszerrel minimalizálja, a számítási folyamatot folyamatosan grafikusan ábrázolja.

A célpontokat tele kör alakúra módosított karakter (alt k) jelképezi.

Az N (2...40) db célpont koordinátáit az XA,YA numerikus tömbök tárolják. Az ADAT eljárás a véletlen értékeket ad a koordinátáknak, majd a minimum-maximum, valamint a karakterméret alapján a videolap méretére illeszti a koordinátákat. A körbejárási sorrendet az U$ sztring elemeinek kódjai adják, a V$ átmeneti munkatároló:

```
120 STRING U$*254,V$*254
```

Az EXOS operációs rendszer karakter- és blokkírási funkcióval kezeli a videolapokat. Nagy mennyiségű karakter, pont és vonal ismételt ábrázolásához a PLOT parancsnál hatékonyabb eljárás escape-szekvenciákkal felépített sztringek kiküldése a videocsatornába. A célpontkarakterek és az összekötő vonalak koordinátáit, valamint az escape-szekvenciákat tartalmazó sztringtömbök:

```
                vonal      ,pont
130 STRING VN$(40)*6,PT$(40)*7
140 LET ESC$=CHR$(27)
```

A koordinátákat a WORD$ függvény alakítja byteokká (alsó és felső helyiérték):

```
970 LET VN$(I)=ESC$&"A"&WORD$(XA(I))&WORD$(YA(I))
980 LET PT$(I)=ESC$&"A"&WORD$(XA(I)-16)
    &WORD$(YA(I)+16)&"•"
```

A pontokat mindíg természetes (1...N) sorrendben a PONT eljárás ábrázolja. A RAJZ eljárás a V$ sztring elemeinek megfelelő sorrendben köti össze vonalakkal a pontokat.

```
1530 PRINT #CX:ESC$&"s"&VN$(ORD(V$(1)))&ESC$&"S";
1540 FOR IR=2 TO N
1550 PRINT #CX:VN$(ORD(V$(IR)));
1560 NEXT IR
1570 PRINT #CX:VN$(ORD(V$(1)))&ESC$&"s"
```

Mivel az EXOS vonalrajzolással mozgatja a videosugarat, az ábra kezdő- és végpontjában a sugarat ki és be kell kapcsolni: ESC$&"s", ESC$&"S"

A plasztikus, ugrálásmentes ábrázolás érdekében két videólap nyitott:

A CL – látható lap van a képernyőn.

A CN – nem látható lapon készül az új rajz. A GRAF eljárás felülről lefelé soronként teríti a képernyőre az új rajzot, és felcseréli a CL,CN csatornaszámokat.

A MATRIX eljárás PITHAGORASZ tétellel számítja ki a pontok egymástól mért távolságait és a TAV kétdimenziós numerikus tömbben tárolja. A távolság számításánál a 9-cel osztás (1280, 1290. sor) akadályozza meg, hogy a további műveletekben a számok túllépjék a 32767-et (komplementálódás!)

A DIRECT eljárás minimalizálja az útvonalat. A TM az eddig elért minimális távolság. A T1 segédváltozó figyeli, hogy a -loc.min.- ciklusban történt-e újabb minimalizálás;

– ha igen, akkor folytatja a ciklust,

– ha nem, akkor befejezi a számítást.

Kiinduláskor a véletlenszerűen elhelyezkedő pontokat összekötő vonalak összebonyolódott hálózatot alkotnak. A számítás során ez a hálózat fokozatosan egy keresztezés és hurokmentes zárt görbévé egyszerűsödik. A program jelenleg három minimalizáló stratégiával működik:

STRAT_3, STRAT_4, STRAT_5

A stratégiák az U$-ból kiindulva a V$ sorrend-sztring elemeinek átrendezésével keresnek olyan új sorrendeket, melyeknél a T távolság kisebb az előzőekben elért TM minimumnál;

– ha igen, akkor az U$ és a TM változók felveszik a sorrend és a távolság értékeit.

A változást a TEST és a HOSSZ eljárás értékeli. Mindhárom stratégia N, ill. N-1-szer próbálkozik és minden művelet végén a sorrend-sztringet ciklikusan balra lépteti, pl:

```
1990 LET U$=U$(2:)&U$(1)
```

STRAT_3 (hurokfelbontás): A sztring H hosszúságú szakaszán az elemek sorrendjét megfordítja.

STRAT_4 (célpontcsere): A sztring két elemét felcseréli.

STRAT_5 (szakaszáthelyezés):A sztring H hosszúságú szakaszát változatlan sorrendben áthelyezi.

A számítási módszer hatékonyságát jellemzi, hogy a kiinduló feltételek mellett az elméletileg lehetséges útvonalak száma a célpontok számának faktoriálisa:

```
10! = 3.6288 E 06
20! = 2.4329 E 18
40! = 8.1592 E 47
```

A program továbbfejlesztésénél korlátlan tere nyílik új, hatékonyabb stratégiák kidolgozásának. Nagymértékben gyorsítja a minimum megtalálását, ha először kiszámíttatjuk a pontrendszer súlypontjából a pontokhoz húzott sugarak szögeit (polárkoordináták), majd a szögeket nagyság szerint rendező eljárással állítjuk be a kiinduló sorrendet.

Kóta Béla

```
100 PROGRAM "rounds.bas"
110 NUMERIC XA(1 TO 40),YA(1 TO 40),TAV(0 TO 40,
      0 TO 40),VX,VY,XM,YM,CX,PJ
120 STRING U$*254,V$*254
130 STRING VN$(40)*6,PT$(40)*7
140 LET ESC$=CHR$(27):LET EM$=CHR$(25)
150 LET CL=1:LET CN=2:LET T,TM,TA=0
160 LET VX=40:LET XM=32*VX-1:SET VIDEO X VX
170 LET VY=24:LET YM=36*VY-1:SET VIDEO Y VY
180 SET VIDEO MODE 1:SET VIDEO COLOUR 1
190 FOR CX=1 TO 2
200   OPEN #CX:"video:"
210   SET #CX:PALETTE 0,255:SET #CX:PAPER 0:
        SET #CX:INK 1:SET #CX:LINE MODE 0:
        SET #CX:LINE STYLE 0:CLEAR #CX
220 NEXT CX
230 !
240 SET CHARACTER ORD("•"),60,126,255,255,
      255,255,126,60,0
250 CALL PROG
260 !
270 PRINT EM$
280 RESTORE 225
290 FOR I=1 TO 12
300   READ X$
310   PRINT EM$;X$;CHR$(241)
320 NEXT I
330 PRINT EM$
340 PRINT EM$,,"FREE =";FREE
350 !
360 PRINT EM$;"                    ";128;CHR$(176)
370 INPUT PROMPT "Hangero (0-255)= ":VOL
380 !
390 DO
400   CALL PROG
410   INPUT PROMPT EM$&"Celpont  (2-40)= ":N
420   !
430   CALL TOROL
440   CALL ADAT
450   CALL MATRIX
460   CALL DEMO
470   CALL DIRECT
480 LOOP
490 END
500 !
510 DEF HALT
520   PING
530   DO
540   LOOP UNTIL INKEY$<>""
550   PING
560 END DEF
570 !
580 DEF TOROL
590   FOR CX=1 TO 2
600     SET #CX:PALETTE 0,255:SET #CX:PAPER 0:
          SET #CX:INK 1:CLEAR #CX
610   NEXT CX
620 END DEF
630 !
640 DEF GRAF
650   LET CX=CL:LET CL=CN:LET CN=CX
660   SET BORDER 2:SET STATUS OFF
670   SOUND PITCH 61,DURATION 32,LEFT VOL,
        RIGHT VOL,SOURCE 0
680   FOR SOR=1 TO VY
690     DISPLAY #CL:AT 1 FROM 1 TO SOR
700   NEXT SOR
710 END DEF
720 !
730 DEF PROG
740   SET STATUS ON
750   SET BORDER 0:SET #102:PALETTE 4,31,2,31:
        DISPLAY TEXT
760 END DEF
770 !
780 DEF ADAT
790   LET U$=""
800   RANDOMIZE
810   LET X1,Y1=32767
820   LET X2,Y2=0
830   FOR I=1 TO N
840     LET U$=U$&CHR$(I)
850     LET XA(I)=RND(32767):LET X1=MIN(XA(I),X1):
          LET X2=MAX(XA(I),X2)
860     LET YA(I)=RND(32767):LET Y1=MIN(YA(I),Y1):
          LET Y2=MAX(YA(I),Y2)
870   NEXT I
880   LET V$=U$
890   LET X2=INT((X2-X1)/(XM-64)+1)
900   LET Y2=INT((Y2-Y1)/(YM-72)+1)
910   FOR I=1 TO N
920     LET XA(I)=INT((XA(I)-X1)/X2+32)
930     LET YA(I)=INT((YA(I)-Y1)/Y2+36)
940   NEXT I
950   !
960   FOR I=1 TO N
970     LET VN$(I)=ESC$&"A"&WORD$(XA(I))&
          WORD$(YA(I))
980     LET PT$(I)=ESC$&"A"&WORD$(XA(I)-16)&
          WORD$(YA(I)+16)&"•"
990   NEXT I
1000 END DEF
1010 !
1020 DEF DEMO
1030   CALL GRAF:CALL PONT(CL)
1040   PRINT #CL:ESC$&"s"&VN$(1)&ESC$&"S";
1050   FOR I=2 TO N
1060     PRINT #CL:VN$(I);
1070     SOUND PITCH 63,DURATION 1,LEFT VOL,
           RIGHT VOL,SOURCE 0
1080   NEXT I
1090   PRINT #CL:VN$(1)&ESC$&"s"
1100   PLOT #CL:16,YM-16,:PRINT #CL:"Nyomjon
         meg egy gombot !"
1110   CALL HALT
1120 END DEF
1130 !
1140 DEF PONT(CX)
1150   CLEAR #CX
1160   FOR L=1 TO N
1170     PRINT #CX:PT$(L)
1180   NEXT L
1190 END DEF
1200 !
1210 DEF MATRIX
1220   CALL GRAF:CALL PONT(CL)
1230   FOR I=1 TO N
1240     SOUND PITCH 63,DURATION 2,LEFT VOL,
```

30 – 'a' STUDIO

```
          RIGHT VOL,SOURCE 0
1250      LET TAV(I,0),TAV(0,I)=0
1260      FOR J=I TO N
1270        SOUND PITCH 67,DURATION 1,LEFT VOL,
              RIGHT VOL,SOURCE 0
1280        LET Y=(YA(J)-YA(I))/9
1290        LET X=(XA(J)-XA(I))/9
1300        LET TAV(I,J),TAV(J,I)=INT(SQR(X*X+Y*Y))
1310        PRINT #CL:ESC$&"s"&VN$(I)&ESC$&"S"&VN$(J)
1320      NEXT J
1330      SOUND PITCH 65,DURATION 4,LEFT VOL,
            RIGHT VOL,SOURCE 0
1340    NEXT I
1350 END DEF
1360 !
1370 DEF TEST
1380    CALL HOSSZ
1390    IF T<TM THEN LET TM=T:LET U$=V$:CALL RAJZ(CN)
          CALL GRAF:PRINT EM$,TM
1400 END DEF
1410 !
1420 DEF HOSSZ
1430    LET T=0
1440    FOR K=1 TO N-1
1450      LET T=T+TAV(ORD(V$(K)),ORD(V$(K+1)))
1460    NEXT K
1470    LET T=T+TAV(ORD(V$(K)),ORD(V$(1)))
1480 END DEF
1490 !
1500 DEF RAJZ(CX)
1510    SOUND PITCH 59,DURATION 4,LEFT VOL,
          RIGHT VOL,SOURCE 0
1520    CALL PONT(CX)
1530    PRINT #CX:ESC$&"s"&VN$(ORD(V$(1)))&ESC$&"S";
1540    FOR IR=2 TO N
1550      PRINT #CX:VN$(ORD(V$(IR)));
1560    NEXT IR
1570    PRINT #CX:VN$(ORD(V$(1)))&ESC$&"s"
1580    PLOT #CX:16,YM-16,:PRINT #CX:STR$(TM)
1590 END DEF
1600 !
1610 DEF DIRECT
1620    !
1630    PRINT :PRINT EM$&"utvonal hossz":PRINT EM$
1640    LET V$=U$:CALL HOSSZ:LET TM=T
1650    PRINT EM$,TM
1660    CALL RAJZ(CN):CALL GRAF
1670    PING
1680    ! ---loc.min---
1690    DO
1700      LET T1=TM
1710      CALL STRAT_3
1720      CALL STRAT_4
1730      CALL STRAT_5
1740    LOOP UNTIL TM=T1
1750    PRINT EM$
1760    !
1770    SET #CL:COLOUR 0,31
1780    SET #CL:COLOUR 1,0
1790    FOR J=1 TO 5
1800      FOR I=12 TO 24
1810        SOUND PITCH I+37,DURATION 1,LEFT VOL,
              RIGHT VOL,SOURCE 0
1820      NEXT I
1830    NEXT J
1840    DO
1850    LOOP UNTIL INKEY$<>""
1860    CALL PROG
1870 END DEF
1880 !
1890 DEF STRAT_3
1900    FOR I=1 TO N
1910      SOUND PITCH 63,DURATION 1,LEFT VOL,
            RIGHT VOL,SOURCE 0
1920      FOR H=2 TO N-1
1930        LET V$=""
1940        FOR J=H TO 1 STEP-1
1950          LET V$=V$&U$(J)
1960        NEXT J
1970        LET V$=V$&U$(H+1:):CALL TEST
1980      NEXT H
1990      LET U$=U$(2:)&U$(1)
2000    NEXT I
2010 END DEF
2020 !
2030 DEF STRAT_4
2040    FOR I=1 TO N-1
2050      SOUND PITCH 65,DURATION 1,LEFT VOL,
            RIGHT VOL,SOURCE 0
2060      FOR J=2 TO N
2070        LET V$=U$(J)&U$(2:J-1)&U$(1)&U$(J+1:):
              CALL TEST
2080      NEXT J
2090      LET U$=U$(2:)&U$(1)
2100    NEXT I
2110 END DEF
2120 !
2130 DEF STRAT_5
2140    FOR I=1 TO N
2150      SOUND PITCH 67,DURATION 1,LEFT VOL,
            RIGHT VOL,SOURCE 0
2160      FOR H=2 TO N-2
2170        FOR J=2 TO N-H
2180          LET V$=U$(H+1:H+J)&U$(:H)&U$(H+J+1:):
                CALL TEST
2190        NEXT J
2200      NEXT H
2210      LET U$=U$(2:)&U$(1)
2220    NEXT I
2230 END DEF
2240 !
2250 DATA ROUNDSMAN,-----------,Sorrend
         optimalizalas,(MONTE-CARLO modszer),
        ---,ENTERPRISE-128,
        IS-BASIC/ZZZIP Compiler
2260 DATA ----,Kota  Bela,----,** 1988 **,
       --------------------------
2270 !
```

Ennyi
fért bele első számunkba. Reméljük, nem volt teljesen haszontalan!
Terveink szerint a következő szám november elején jelenik meg.
Miután nem tervezzük lapunk postai forgalmazását, kérjük,
ha igényt tartotok a lapra, rendeljétek meg címünkön!

Címünk: 'a' STUDIO
Bp. Pf. 105. 1525.

Más.
Ha tetszettek az ENTER-MANÓK, igérjük, a következőekben is találkozni fogtok velük.
Novemberben majd felteszünk egy találós kérdést is, s aki jól válaszol,
ajándékba kap egy kis textilmanót!

Tehát, írjatok!