

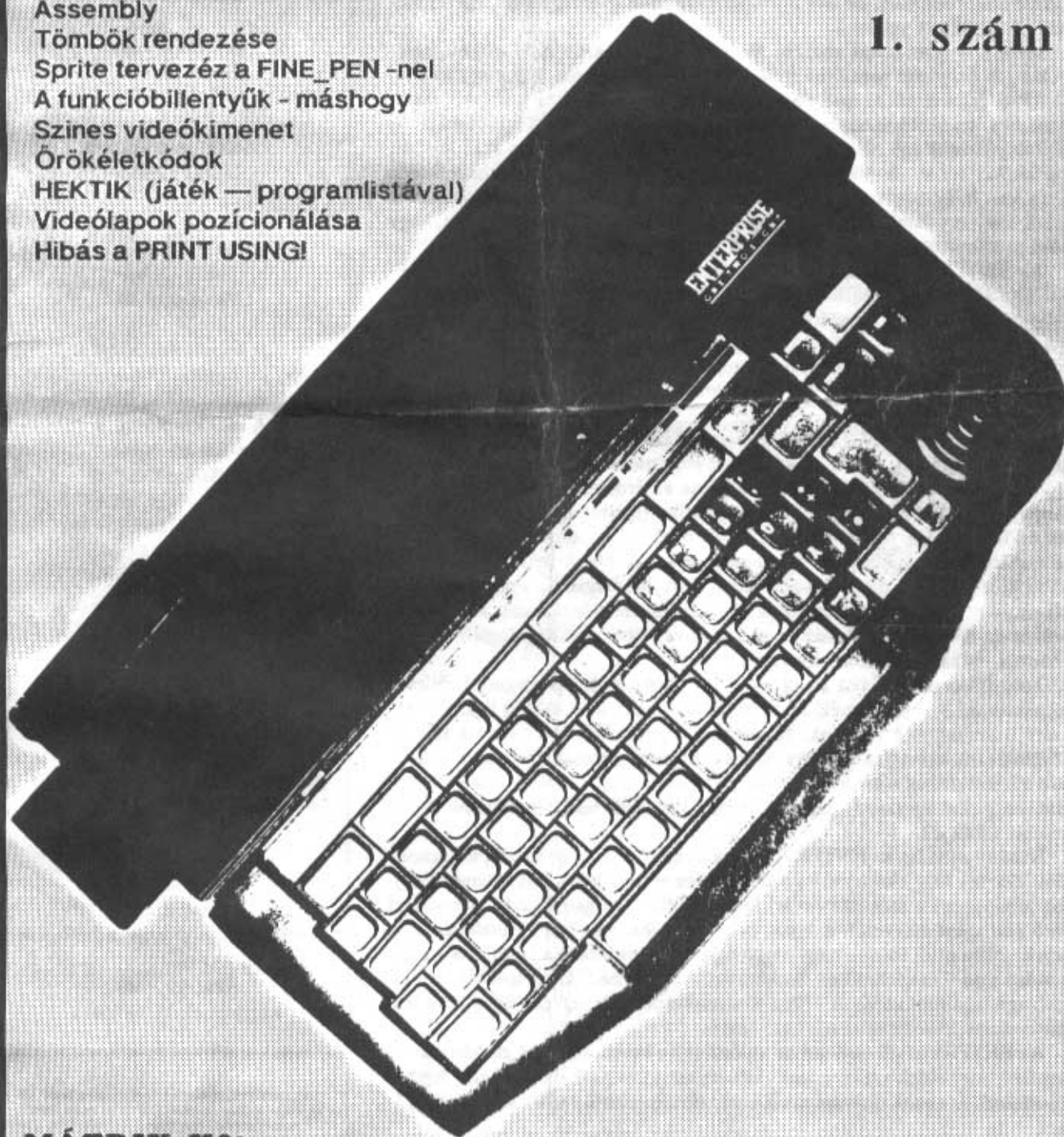
ENTERPRESS

IDŐSZAKOS KIADVÁNY AZ ENTERPRISE SZÁMÍTÓGÉPEK FELHASZNÁLÓINAK

A TARTALOMBÓL:

Assembly
Tömbök rendezése
Sprite tervezés a FINE_PEN -nel
A funkcióbillentyűk - máshogy
Színes videókimenet
Örökéletkódok
HEKTIK (játék — programlistával)
Videólapok pozícionálása
Hibás a PRINT USING!

I. évfolyam
1. szám



MÁTRIX Kft.

1990. SZEPTEMBER

Tisztelt Olvasó!

Az ENTERPRISE számítógép 1987 májusában tűnt fel a hazai áruházakban. Sokan döntöttek a gép megvétele mellett annak ellenére, hogy igazából nem tudtuk, mi rejtezik a fekete dobozban. Kevés volt a szoftver, a szakirodalom. Akkori hírek szerint az ENTERPRISE cég csődbement, a gép nyugaton csúfosan megbukott, és nekünk csak az olcsó "vas" jutott. A helyzet meglehetősen ellentmondásos és zavaros volt.

Aztán teltek-múltak a hónapok. Lassan megismertük gépünket, és rájöttünk, hogy az ENTERPRISE a maga kategóriájában kiváló számítógép. A hazai forgalmazó sok ígéretet tett, igyekezett ezeket teljesíteni, bennünket ellátni. Feltűntek a komoly, professzionális felhasználói és játékprogramok, igyekeztünk ezekhez hozzájutni (ki így, ki úgy). Megvásárolhattuk a régóta áhított szakirodalmat, leírásokat. A tehetősebbek lemezegységet, egeret, be-szedszintetizátort vehettek.

Mára a helyzet sokat tisztult. Nem igazolódtak azok a feltevések, amelyek a gép helyzetét negatívan ítélték meg. Természetesen igaz, hogy az ENTERPRISE lemaradt az iparszerű szoftver- és hardverfejlesztésről, de rendelkezünk a továbblépéshez szükséges alapszoftverekkel, információkkal. A géptulajdonosok egy része programokat készít, kisebb-nagyobb áramkörüi kiegészítőket tervez és illeszt a gépbe, sokfajta tapasztalattal rendelkezik. Nagy kár, hogy miután megszületik valami, az csak néhány közeli barátához jut el, vagy egyszerűen elsüllyed egy fiók mélyén. Pedig könnyen lehet, hogy néhány kilométerrel arrébb valakinek pontosan erre lenne szüksége.

Az ENTERPRESS kiadásával szeretnénk megteremteni az ENTERPRISE tulajdonosok közötti kapcsolatot. Tudomásunk szerint tizennyolcezer felhasználó dolgozik, játszik ENTERPRISE géppel, de ez a szám hamarosan eléri a húszezret. Egy ekkora tábor jogosan vár használható, értékes tudnivalókat. A hazai szakajtóban már jelentek meg vele kapcsolatos cikkek, de az összes ilyen lapra jellemző, hogy az ENTERPRISE-zal csak mellesleg foglalkoznak.

Célunk, hogy az ENTERPRESS-t igazi házi számítógépesekhez szóló kiadvánnyá formáljuk. Többféle sorozatot szeretnénk elkezdeni, ezekben egy-egy témával részletesen megismerkedhetne az érdeklődő. Például: Z80 Assembly és az EXOS, Pascal, dBase, a grafika (sprite-ok) és a hang kezelése, hardverelemek ismertetése. A Basic nyelvről kapcsolatos tanfolyamot nem szándékozunk közölni, hiszen erről már sok könyv jelent meg, mi inkább kész listákat, megoldásokat közölnénk több-kevesebb magyarázattal kiegészítve. Igyekezni fogunk a játékprogramokhoz leírásokat, térképeket, örökéletkódokat szerezni, bár ezzel a témával több kiadvány foglalkozik, és nehezen tudnánk velük konkurrálni.

Természetesen a lapot az olvasók beküldött írásai, programjai, ötletei, tapasztalatai is színesítik. Jó tollú szerzők akár sorozatot is indíthatnak, ezért honoráriumot fizetünk. Hosszabb programokról csak leírást fogunk közölni, ezek kazettára másolva lesznek megrendelhetők. Biztosan népszerű lesz a levelezési és a hirdetési rovat. (Már most felhívjuk a nepperek figyelmét, hogy ránk ne számítsanak!) A leendő szerzőkre vonatkozó tudnivalókat később közöljük.

Néhány mondat az ENTERPRESS helyzetéről: a lapot tavaly, karácsony előtt akartuk megjelentetni, de szerencsésen mindig közbejött valami. Az újság tulajdonosa a székesfehérvári Mátix Kft. Mint gazdálkodó, nyereséget váró gazdasági szervezet, a lapot valamelyes hasznot hozó vállalkozásként kezeli. (Nem ettől fognak nagy céggé fejlődni!) Manapság a lapkiadás – mint minden más – meglehetősen kockázatos. Az ENTERPRESS további léte ki-zárólag a fogadtatás sikerétől függ. Kedvező esetben jelentősen növeljük a példányszámot, kialakítva az elérhető árat is.

Az ENTERPRESS első száma valószínűleg kevesekhez jut el. Szeretnénk remélni, hogy előbb-utóbb minél több géptulajdonos szerez tudomást lapunk létezéséről, és annak elolvasása után már türelmetlenül várja a következő számot.

A szerkesztők

ENTERPRESS

időszakos kiadvány az
ENTERPRISE számítógépek
felhasználóinak

I. évfolyam 1. szám
1990. szeptember

Kiadja:
a MÁTRIX Számítástechnikai,
Kereskedelmi és Szolgáltató Kft.
Székesfehérvár

Felelős kiadó:
Annus István
ügyvezető

Felelős szerkesztő:
Ujlaki László

A lap szerkesztői:
Hajnal Csaba
Bozai Gábor
Varga Zoltán

Technikai szerkesztő:
Bartha István

A szerkesztőség és
a kiadó címe:
8000 Székesfehérvár
Dózsa György tér 10.
Telefon: (22) 16-520 / 141
Telefax: (22) 11-585

Levelezési cím:
Letét 334 ENTERPRESS
1399 Budapest, Pf. 701

Nyomda:
VIDEOTON Nyomda
Székesfehérvár
Felelős vezető: Csizmadia Ferenc

ISSN 0866 - 1820

Terjesztli:
a Magyar Posta
Előfizethető a megrendelőlapon

Ára: 49,- Ft
Megjelenik kéthavonta

Az ENTERPRESS-ben közreadott információk célja az, hogy segítse, tudnivalókkal lássa el az ENTERPRISE számítógépek felhasználóit. A közölt programokat, kapcsolási rajzokat és leírásokat mindenki szabadon felhasználhatja, de tilos azokat a kiadó írásbeli engedélye nélkül másolni, terjeszteni.

TARTALOM :

Assembly 1.rész	3
Tömbök rendezése	6
Sprite tervezés a FINE_PEN-nel	7
Funkcióbillentyűk-máshogy	8
Nyomtató hiányában	9
Színes videókimenet	9
Gyorsabb directory	9
Röviden az editorcsatornáról	10
Üzenet a státusz sorban	10
Rendszerindítás a START programmal	11
Örökéletkódok	11
HEKTIK	12
Percenként hatvan keretszín	14
Látszólag több	14
Videólapok pozícionálása	14
Hibás a PRINT USING!	14
Megrendelőlap	15
Kérdőív	15-16

Assembly

1. rész

A most induló sorozatban az assembly programozás rejtelmeibe szeretném bevezetni az érdeklődőket. Megismerkedünk az alapfogalmakkal, a processzorral és utasításkészletével, a számítógép felépítésével, operációs rendszerével. Lehet, hogy sokan már e rövid bevezető után tovább lapoznak, mert néhány kifejezés jelentésével nincsenek tisztában. Ne tegyék, már kezdjük is!

Számrendszerek, bitek, byte-ok

Elsőként a kettes (bináris) számrendszerrel ismerkedünk meg. A kettes számrendszer csak két számjegyet használ: a nullát és az egyest. A kettes számrendszerben ugyanúgy lehet számokat ábrázolni, mint a jól megszokott tizesben (decimális). A kettes számrendszerbeli számjegyeket a számítástechnikában biteknek nevezik (BIT=BInary digiT, kettes számrendszerbeli szám). A bitek növekvő sorszámozása jobbról balra történik. Ábrázoljuk példaképpen a 147-es számot bináris számként!

Bitsorszám	7	6	5	4	3	2	1	0
Értéke	128	64	32	16	8	4	2	1
147=	1	0	0	1	0	0	1	1

A számjegyet követő "b" betű a bináris számrendszer jelzésére szolgál. Az adott bit decimális értékét úgy hatá-

rozzuk meg, hogy kettőt az adott hatványra emeljük. A hetedik bit értéke 128, mert $2^7=128$ stb. A 147-et a következőképpen kapjuk meg:

$$1*2^7+0*2^6+0*2^5+1*2^4+0*2^3+0*2^2+1*2^1+1*2^0=128+16+2+1=147.$$

(A ^ a hatványozás jele.)

Nem véletlenül ismertettem a bitek ilyen nyolcas felállítását. Ezt szokás ugyanis byte-nak nevezni. A byte-ban tehát nyolc bit helyezkedik el, így rajta 1111111b=255 és a 0000000b=0 tartomány ábrázolható. A biteket másképpen is lehet értelmezni, de erről majd később lesz szó.

Egyszerre két byte-on is ábrázolhatunk számokat, ezt szokás szónak (word, szó) nevezni. Egy szón 65535 és 0 közötti számok férnek el.

Írjuk fel gyakorlásképpen az 57269-et!

Bitsorszám	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
57269=	1	1	0	1	1	1	1	1	0	1	1	0	1	0	1	1

A kettes számrendszerben felírt számok sokjegyűek, nehezen megjegyezhetők. Éppen ezért célszerűbb a 16-os (hexadecimális) számrendszer használata. Ebben sincs semmi különös, adott helyiértéken 16-nak megfelelő hatványa szerepel. A hexadecimális számrendszerben 16 számjegy van: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

A 147 hexadecimális ábrázolásához bináris megfelelőjéből induljunk ki!

147- 1 0 0 1 0 0 1 1 b

Vizsgáljuk meg a felső és az alsó 4 bitet önmagában. A felső 4 bit: 1001b, értéke 9h (a "h" a hexadecimális számot jelöli), az alsó 4 bit: 0011b, értéke 3h. így tehát 147=93h.

Nézzük meg a 16 biten ábrázolt 57269-et hasonló szempontok alapján!

57269- 1 1 0 1 1 1 1 1 1 0 1 1 0 1 0 1 b

Osszuk fel ezt is 4 bites csoportokra! Az első: 1101b=Dh, a második: 1111b=Fh, a harmadik: 1011b=Bh, végül a negyedik: 0101b=5h. Ezek szerint 57269=DFB5h.

Ebben a formájukban a számok sokkal jobban kezelhetők, megjegyezhetők. Egy byte értéket kétjegyű, egy szó értékét négyjegyű hexadecimális számon ábrázolhatjuk. Szeretném hangsúlyozni, hogy a hexadecimális számszámrendszert használata elsősorban kényelmi célokat szolgál.

Vázlatosan a számítógépről

A számítógép "motorja" a mikroprocesszor (CPU=Central Processing Unit, központi feldolgozó egység). Az ENTERPRISE-ban a Zilog cég Z80A típusú CPU-ja dolgozik. E sorozat célja, hogy a Z80-as CPU programozását bemutassa, így erről még bőven lesz szó.

A számítógép tárában tárolódnak az adatok. A tára-knak két alapvető típusa van: az írható-olvasható tár (RAM=Random Access Memory, közvetlen hozzáférésű memória) és a csak olvasható tár (ROM=Read Only Memory, csak olvasható memória).

Az általunk írt programok a RAM-ban tárolódnak. A RAM-ok a tápfeszültség kikapcsolásakor elvesztik tartalmukat, ezért kell minden programunkat kazettára, lemezre menteni.

A ROM-okban tárolt információk a tápfeszültség megszűnése után is megőrződnek. Ilyen tára-kba nem tudunk adatokat beírni, csak a gyártók által beégetetteket lehet onnan kiolvasni. ROM tartalmazza a gép működéséhez szükséges alapprogramokat.

A tárat tárrekeszek sorozataként lehet elképzelni, a rekeszek egyenként 8 bitet tartalmaznak, tehát egy tárrekesz egy byte-nak felel meg.

A számítógép a perifériáin keresztül kommunikál a külvilággal. Ilyen periféria például a képernyő, a billentyűzet, a lemezegység stb. Az ENTERPRISE egyik erőssége éppen a periféria-kezelésben rejlik.

A CPU alapvetően két részre osztható: a vezérlőegység-re (Control Unit) és az aritmetikai-logikai egységre (ALU=Aritmetical-Logical Unit, aritmetikai-logikai egység). A vezérlőegység a memóriának egy adott helyétől kezdve adatokat olvas be. A beolvasott byte-ot, mint utasítást bitenként értelmezi, az utasításban minden egyes bitnek saját jelentése van. (Természetesen egy utasítás nemcsak egy, hanem több byte-ból is állhat.) Az utasítás-

tól függően a CPU sokféle műveletet végezhet. Például újabb adatokat (operandusokat) olvas be a tárból, hogy valamilyen műveletet végezzen velük az ALU segítségével, vagy valamilyen perifériához fordul stb. A vezérlőegység tehát az utasításnak megfelelően vezérli a többi egység (ALU, tár, perifériák stb.) működését. Miután végrehajtotta az utasítást, beolvassa és végrehajtja a következőt. A processzor ilyenkor egy gépi kódú programot hajt végre. A gépi kódú program tehát nem más, mint a memóriában tárolt bináris számok sorozata. A processzor a program utasításainak megfelelően értelmes feladatokat tud elvégezni.

Az első számítógépek az 1940-es évek második felében jelentek meg. Ezeket a gépeket eleinte csak közvetlenül gépi kódban lehetett programozni. Képzeliük csak el az akkori feladat nagyságát: programot kellett írni a processzor saját nyelvén, az embertől teljesen idegen nullák és egyek sorozataként! E tarthatatlan helyzetben segített az assembly (assembly=összeállítás) nyelv kialakulása.

Minden gépi kódú utasításnak van egy ún. mnemonikája. A mnemonikok használatával egyszerűbb a programozó dolga, mert emberközelibbakké válnak az utasítások. Nézzünk egy Z80-as példát, mindenféle egyéb magyarázatot nélkülözve! Feladatunk, hogy a processzor belsejében egy bitet 1-re állítsunk. A szükséges utasítás hossza 1 byte, a gépi kódja: 00110111b=37h. Ez így nem sokat mond! Ugyanennek az utasításnak az assembly megfelelője, a mnemonikája: SCF. Az SCF a "Set Carry Flag" rövidítése. Ez a három betű "nagyságrendekkel" egyszerűbbé teszi az utasítás megjegyzését.

Az assembly-ről tényszerűen

A sorozat címe szándékosan "Assembly", nem "Gépi kód" vagy valami hasonló. Nem gépi kódban fogunk programozni, hanem assembly-ben. Az assembly-ben megírt programot egy fordító programmal, az ún. assemblerrel fogjuk gépi kóddá lefordítani. Mielőtt azonban a programozásba belefognánk, néhány ténnyel tisztában kell lennünk.

Ritkán fordul elő, hogy egy kezdő azonnal assembly programokat ír. Ezt szinte mindig megelőzi egy másik, magasabb szintű programozási nyelv elsajátítása (Pascal, Basic). Előbb-utóbb azonban valami olyan feladatot kellene megoldani az adott nyelven, ami azon eleve lehetetlen, vagy ha ráerőszakolható is a nyelvre, akkor azt lassan végzi el a gép. Ekkor nyújthat segítséget az assembly.

A gépi kódú program minimális helyet foglal a tárból, és nincs nála gyorsabb kód. Szeretném mindjárt hangsúlyozni, hogy az előbbi kijelentéseket nem lehet általánosítani! Semmi akadálya nincs a hosszú, lassú és buta gépi kódú programok írásának.

Az assembly alacsony szintű programozási nyelv. A géphez a lehető legközelebb kerülünk, "lealacsonyodunk" hozzá, programunk veszi át a gép irányítását. Nincs közöttünk a Basic értelmező, vagy a Pascal fordító. Ennek persze megvannak a maga hátrányai is. Elveszítünk sokfajta kényelmes lehetőséget, amit a Basic vagy a Pascal bizto

sít. Az assembly programok elkészítési ideje lényegesen hosszabb, mint az az említett nyelveken lenne. Nem lehet az adott gép felépítésének alapos ismerete nélkül igazán kihasználni lehetőségeit.

Minden mikroprocesszor típusnak mások a gépi kódú utasításai. Az ENTERPRISE-on megírt programot az IBM AT nem tudja lefuttatni, bármilyen okos is. Még a Z80-ra épített ZX Spectrum sem fogja "megenni" a kódot, mert teljesen más a két gép szervezése. Továbbá az ENTERPRISE esetében néhány dologgal finoman kell bánni, mert esetleg az angol gépen "kihegyezett" programra a német gép furcsán reagál. (Ezek ún. kompatibilitási problémák. A kompatibilitás összeférhetőséget jelent. Ha két gép teljesen kompatibilis, akkor képesek egymás programjait fogadni, és azokat rendesen végrehajtani.)

Az assembly nem varázsszer! Mielőtt bármiféle assembly program írásába belefognánk, gondoljuk végig a következőket:

- Hányszor akarjuk futtatni a programot?
- Képes lesz-e számítógépünk a programmal az elvárásokat teljesíteni?
- Mennyivel lesz hatékonyabb a program gépi kód-ban?
- Nem lenne-e jobb, egyszerűbb másik nyelven megírni a programot?
- Csak egy rövid rutint vagy egy teljes programot szeretnénk készíteni?
- Ha rutinról van szó, mennyire illeszthető a főprogramhoz?
- Feltétlenül gyorsnak kell-e lennie a programnak?
- Mennyi időnk van az elkészítésre?
- Később esetleg tovább kell-e fejleszteni a programot?

A felmerülő kérdéseket még hosszan lehetne sorolni. Basic-ben és főleg Pascal-ban nagyon jó programokat lehet készíteni. Semmiféle problémát nem jelent Basic programunkat gépi kódú rutinokkal megtámogatni. További lehetőség, hogy a belőtt Basic programot lefordítjuk. A Hi-Soft Pascal több függvénnyel, eljárással támogatja a gépi kódú rutinok kezelését. Szeretném megemlíteni, hogy létezik egy programozási nyelv, amelyre a "magasszintű assembly" jelzőt szokták ragasztani. Ezt a programozási nyelvet C-nek hívják. A C programok képességeikben csak kismértékben maradnak el az assembly-ben írt programoktól, ugyanakkor nagymértékben kiküszöbölik azok fejlesztési hátrányait. Kifejezetten az ENTERPRISE-hoz illeszkedő C fordító még nem készült, de az IS-DOS-on háromféle is futtatható. (Az IS-DOS az ENTERPRISE bővített lemezes operációs rendszere.)

A gépre koncentrálvá

Mint már említettem az ENTERPRISE számítógépet a Zilog cég Z80A típusú CPU-ja vezérli, amelyet 4MHz-es órajel éltet. A 4MHz-es órajelnek köszönhetően a gépben minden gépi kódú utasítás néhány mikroszekundum alatt lefut. A Z80A-nak 16 bites címbusza van. Ez azt jelenti, hogy a CPU 2^{16} , azaz 65536 tárrekeszt tud közvetlenül elérni, megcímezni. A busz egyébként párhuzam-

os vezetékek halmazát jelenti, amelyekben adatok áramlanak. A busznak 3 fajtája van: címbusz, adatbusz, vezérlőbusz. A címbusz tehát 16 vezetékkel jelent, amelyek a CPU-t összekötik (egyebek között) a tárral.

A rekeszt a tárból elfoglalt sorszáma alapján lehet megcímezni. Az első rekesz a 0-ás, a legutolsó a tár tetején, a 65535-ös címen van. Hogy ne kelljen mindig nagy számokról beszélnünk, ismerkedjünk meg a kilobyte (KB) fogalmával. A kilobyte a vártól eltérően nem 1000 byte-ot, hanem 1024 byte-ot takar. Ebben a bináris számrendszer a bünös ($1024=2^{10}$).

A 65536 byte 64KB-ot jelent ($64 \cdot 1024=65536$), a CPU tehát 64KB nagyságú memóriát lát egyszerre. Az ENTERPRISE gépben van egy bonyolult memórialapozó áramkör, ezért képes a gép akár 4096 KB-os, azaz 4MB-os (MB=MegaByte, 1MB=1048580 byte=1024KB) tár kezelésére is.

A CPU által látott memóriatartományt 4, egyenként 16 KB-os részre, ún. lapra osztották. Egy lapra (page, lap) sorszámmal szokás hivatkozni. A lapok elhelyezkedése a következő:

lap	címtartomány
P0	0000h-3FFFh
P1	4000h-7FFFh
P2	8000h-BFFFh
P3	C000h-FFFFh

(A P0 a Page0, nullás lap jelölése stb.)

A tár "szeletelése" is ehhez a logikához illeszkedik. A memóriát 16KB-os darabokra, ún. szegmensekre tagolták. A memóriát tehát ilyen szegmensek sorozata alkotja. Minden szegmensnek saját száma van, ez alapján lehet azonosítani. Az ENTERPRISE-ban 256 szegmens fér el, ez képezi a 4096 KB-os maximális memóriát. A CPU valamelyik lapjára a sorszámmal azonosított szegmenst lapozhatjuk. A gép nevében a 128K az összes RAM memóriára utal. A 128KB-os RAM tehát 8 szegmenst jelent.

Az ENTERPRISE számára a legfontosabb program az EXOS. Az EXOS (EXOS=ENTERPRISE eXpandable Operating System, ENTERPRISE bővíthető operációs rendszer) a gép operációs rendszere. Az EXOS-ra az ENTERPRISE-nak mindig szüksége van (ez adja a gép "jellemét"), ezért magától értetődően az EXOS-t ROM-ba égették. A CPU az EXOS rutinjait végrehajtva vezérli az ENTERPRISE-t. Az EXOS-ról a sorozatban még bőven lesz szó. Azok a kártyák, amelyek a gép ROM-BAY csatlakozójába illeszthetők, ROM-okat tartalmaznak. Ezek szerint a Basic értelmező is egy ROM-ba égetett program.

A Z80 adatbuszának szélessége 8 bit, a Z80 ún. 8 bites mikroprocesszor. A CPU egyszerre 8 bitet, azaz egy byte-ot tud mozgatni (a tárból beolvasni vagy oda kiírni).

A vezérlőbusz 14 vezetékből áll. Ezekről egyelőre csak annyit, hogy közöttük van az órajellet szállító vezeték is.

(folytatjuk)

TÖMBÖK RENDEZÉSE

A számítógép – mint tudjuk – sok mindenre jó. Adatok, adatbázisok kezelésére különösen. Egy ilyen állományban sokkal gyorsabban érhetünk el információkat, ha a belsejében rend van. A karbantartó munkát valamilyen rendteremtő algoritmusra kell bízunk. Rengeteg ilyen létezik, mi ezek közül most a beszűrásos rendezéssel (Insertion-Sort) ismerkedünk meg.

Összekevert számsort rendezünk növekvő sorrendbe. A beszűrásos módszer a következőket teszi: mozgat egy mutatót, amely az éppen feldolgozás alatt álló adatra mutat. A mutató a második pozíciótól az utolsó felé halad. Összehasonlítja a mutatónál lévő adatot az előtte levő, eggyel kisebb sorszámú adattal. Amennyiben a kisebb sorszámú adat értéke kisebb vagy egyenlő, mint az aktuális, akkor nincs semmi tennivalója csupán a mutatót kell eggyel növelnie. Ellenkező esetben a mutatónál lévő adatot megjegyzi, és elkezd a mutató értékénél kisebb sorszámú adatokat egy hellyel "jobbra" eggyel nagyobb sorszámú pozícióra mozgatni, az aktuális mutatóértéktől lefelé a kisebb sorszámok felé haladva egészen addig, amíg a megjegyzett adatnál kisebbet vagy azzal egyenlőt nem talál. Ekkor a megüresedett helyre beírja a megjegyzett adatot, majd növeli a mutatót, és folytatja a rendezést.

A könnyebb érthetőség kedvéért nézzük a következő, félig már rendezett számsort:

Sorszám	1.	2.	3.	4.	5.	6.	7.
Adat	15	27	53	21	77	22	15

Legyen a mutató értéke éppen 4. Összehasonlítja a 21-et a 53-mal. A 21 nyilván rossz helyen van, megjegyzi a számot. Elindítja az adatok mozgatását "jobbra" a 3. pozíciótól kezdve a csökkenő sorszámok felé, egészen addig, amíg a 21-nél kisebbet vagy azzal egyenlőt nem talál. A mozgatás befejezésekor a 21-et beszűrja a megfelelő helyre. A módosult sor:

Sorszám	1.	2.	3.	4.	5.	6.	7.
Adat	15	21	27	53	77	22	15

Növeli a mutató értékét, így az az 5. adatra mutat. A 77 viszont nagyobb, mint az 53, tehát nincs semmilyen művelet, csak a mutatót kell növelnie. A rendezés véget ér, ha a mutató elérte a tömbhatárt, és az utolsó mozgatás is véget ért. Az algoritmust Basic-ben a *listán* látható program valósítja meg:

```

20 PROGRAM "INSORT.BAS"
21 DEF TOLT
22   FOR I=1 TO UBOUND(A)
23     LET A(I)=RND(5000)
24   NEXT
25   LET A(0)=-INF
26 END DEF
27 DEF REND
28   FOR I=2 TO UBOUND(A)
29     IF A(I)<A(I-1) THEN
30       LET M=A(I)
31       LET Z=I-1
32       DO
33         LET A(Z+1)=A(Z)
34         LET Z=Z-1
35       LOOP UNTIL A(Z)<=M
36       LET A(Z+1)=M
37 END DEF
38 END IF
39 NEXT
40 END DEF
41 REM *** FOPROGRAM ***
42 RANDOMIZE
43 NUMERIC A(0 TO 20)
44 CLEAR SCREEN
45 CALL TOLT
46 CALL KIIR
47 CALL REND
48 CALL KIIR
49 END

```

Az "A" tömb elemeit a "TOLT" inicializálja, a "KIIR" megjeleníti, a "REND" rendezi. Természetesen a "REND" eljárás másik programban is használható, a tömbnek új nevet, módosított hosszát is adhatunk. A legnagyobb tömbméretnek a gép memóriája és szabad időnk nagysága szab határt. A programot 200 elemmel próbáltam ki: a futási idő 4 perc 40 másodperc volt. Valószínű, hogy TURBO ENTERPRISE-on valamivel jobb eredmények születnének.

Ezek szerint nem elég hatékony a beszűrásos rendezés? Ebben az állapotban tényleg nem. De van megoldás: ZZZIP compiler! A rendezés tipikus példája a ZZZIP alkalmazásának. A program simán "lefordul", és 200 adat esetén 5 másodperc alatt lefut. Az idők önmagukért beszélnek, a javulás közel 60-szoros!

Természetesen 200 adat nem számít nagy mennyiségnek, komolyabb alkalmazásoknál több ezerrel kell számolni. Ezzel már a fordított Basic program is nehezen bírkozik meg. Hétezer adatnál negyedóra elteltével meguntam a várakozást. A rendezést úgy gyorsíthatjuk, ha más, hatékonyabb algoritmust állítunk munkába, és igazi, compiler típusú nyelven (Pascal) írjuk meg az adatrendező programot.

Ezekre még visszatérünk.

Sprite tervezés a FINE_PEN-nel

Amikor az ENTERPRISE "ellenségei" kifogásolták a gép adottságait, akkor leginkább a hardverből támogatott spritekezelés hiányát hangoztatták. Teljesen igazuk van, a gépben tényleg nincsenek beégetett sprite-ok úgy, mint mondjuk a C64-nél. Csakhogy...

Egy 128 KB-nyi RAM-mal rendelkező gépre olyan sprite kezelő készíthető, amilyen csak teszünk! Ha a gép ráadásul olyan ügyesen kezeli a bővítéscet, mint az ENTERPRISE, akkor tényleg semmi akadályja nincs a dolognak. Helyzetünket még könnyebbé teszi, hogy az "a" Studio programozói már elkészítették saját sprite-animátorukat, az "EnterSPRITE"-ot. A Basic bővítőként inicializálódó program a sprite-ok kezelését, irányítását rendkívül leegyszerűsíti.

Az animátor 8 sprite vezérlését végzi el a "háttérből". Ráadásul egy sprite 8 fázisra tagolódik, így aztán még látványosabb és főleg gyorsabb programokat írhatunk Basic-ben. Kár, hogy az animátor alatt megáll az óra, ami játékkprogramoknál igen csak gyakran használatos. A program az USER_ISR mutatón "lóg", így saját rutinunkat ez elé, korrekt módon kell beillesztenünk.

A kezelendő sprite-okat egy másik programmal, a FINE_PEN-nel kell elkészíteni. Eddig gondolom sokan el is jutottak, csak a FINE_PEN-nel voltak problémáik. Most ezen próbálunk segíteni.

A program bejelentkezése után a [H] billentyű lenyomásakor a help-lapokhoz juthatunk. A program szerény grafikus lehetőségei kiderülnek a leírásból.

A spritetervezéshez először üssük le az [ENTER]-t. Ekkor négy pontot látunk villogni, a pontok belseje a sprite méretét definiálja. Nyomjuk le a [STOP]-ot, így visszajutunk a rajzoló módba. A sprite méretének ismeretében rajzoljunk meg annyi fázist, ahányat a sprite-ban váltogatni szeretnénk! A sprite-ban maximum 8 fázis fér el.

A fázisok elkészülte után ismét nyomjuk meg az [ENTER]-t! A botkormány segítségével álljunk rá az első fázisra (a mozgás gyorsítható az [LSHIFT], azaz a bal oldali SHIFT gomb nyomásával), majd a [HOLD] (vagy más billentyűzeteken a [PAUSE]) lenyomásával a fázis bekerül a bal alsó sarokban levő munkaterületre. A munkaterületen az [O] és a [P] segítségével forgathatjuk, az [X]-el és a botkormánnyal tükrözhetjük a fázist.

A munkaterületen levő fázist az [LSHIFT]-el és a nyolc funkcióbillentyű valamelyikével tölthetjük a spriterészbe (8 fázis - 8 funkcióbillentyű). A leírt módon vegyük fel a fázisokat a grafikus lapról. Természetesen nem kötelező az összes fázist megadnunk.

A betöltött fázisokat a funkcióbillentyűk nyomogatásával előhívhatjuk, sorrendjüket ellenőrizhetjük. A kész sprite-ot az [S] leütésével, nevének megadása után elmenthetjük. A mentés a spriterészről történik, elsőként az [F1]-re töltöttet menti el a program. A teljes sprite pontosan 2KB. Elmentett sprite-ot az [L] lenyomása, és a sprite-név beírása után a sprite-területre tölthetünk.

Lehetőségünk van a sprite animálására is, ehhez az animálandó fázisokat a grafikus lap tetejére kell kiraknunk. A funkcióbillentyűk nyomogatásával a fázist, az [1]...[8] gombokkal a fázis helyét kereshetjük ki a képernyőn. Az animáció során a bal oldali fázis lesz az első, a jobb oldali az utolsó. A munkaterületen levő fázist lágyan a szóközzel, keményen a [LOCK]-kal tehetjük le. Lágy letevésnél a sprite rámásolódik a háttérre, nem írja felül azt.

Az animációt a [TAB]-bal indíthatjuk. A [STOP]-pal kiléphetünk a sprite módból, az animáció viszont a grafikus kurzornál tovább folytatódik. Az animáció sebessége a [K]-val lassítható, a [;]-vel gyorsítható. Az animáció az [ESC] leütéséig tart.

A FINE_PEN sajnos nem mentes a hibáktól. A fázisforgatás és a fázistükrözés funkciók értelmét kérdőjelezi meg az, hogy hiába van e két funkció valamelyikével módosított fázis a munkaterületen, csak az eredeti fázis vehető fel onnan!

A sprite-dump-nak gyakorlatilag semmi értelme nincs, a felhasználónak nem ad használható információt az adott fázisról.

A program kezelése rendkívül nehézkes, körülményes! A program helpje is nehezen érhető el. Az ENTERPRISE kiváló grafikai képességei messze nincsenek kihasználva. A program csak a magnót szereti, lemezzel nem indul el. Mentségül szolgáljon, hogy ez az egyik legősibb ENTERPRISE program, kiadási dátuma: 1985.

Szeretnénk felhívni a spritetervezés iránt érdeklődők figyelmét arra, hogy rövidesen ismertetünk egy teljesen új, EnterSPRITE kompatibilis spritetervezőt, a "SPRED"-et.

A FUNKCIÓBILLENTYŰK - MÁSHOGY

A funkcióbilleentyűk átprogramozása jelentősen megkönnyítheti a munkát. A billeentyűk eredeti értelme elsősorban a BASIC használatához van optimalizálva. Más rendszerek - ha jól vannak megcsinálva - átdefiniálják a funkcióbilleentyűket a saját ízlésük szerint, így jár el például a HiSoft PASCAL rendszer automatikusan, vagy a nálunk forgalmazott - bővített - IS-DOS operációs rendszer az FKEY parancs kiadásával.

Ha elsősorban nem BASIC-ben dolgozunk, szükségünk lehet egy hasonló megoldásra. Különösen a lemezegység-gel rendelkezők tudják rugalmasan kihasználni ezt a lehetőséget.

Az itt bemutatott program (1. lista) az EXDOS környezethez illeszkedő funkcióbilleentyűket definiál.

1. lista

```

100 PROGRAM "fkeys"
110 STRING CR$,N$
120 NUMERIC LBLUE,HBLUE
130 LET LBLUE=RGB(.2,.2,1):LET HBLUE=RGB(0,0,.3)
140 LET CR$=CHR$(13)
150 LET N$=CHR$(165)&CHR$(161)
160 CALL HELPLINE
170 CALL SETKEYS
180 DEF SETKEYS
190 SET FKEY 1 N$&"dir"&CR$
200 SET FKEY 2 N$&"type "
210 SET FKEY 3 N$&"load "
220 SET FKEY 4 N$&"copy "
230 SET FKEY 5 N$&"text 80"&CR$
240 SET FKEY 6 N$&"set speaker on"&CR$
250 SET FKEY 7 N$&"set key click on"&CR$
260 SET FKEY 8 N$&"wp"&CR$
270 SET FKEY 9 N$&"dir "
280 SET FKEY 10 N$&"cls"&CR$
290 SET FKEY 11 N$&"cd "
300 SET FKEY 12 N$&"copy e: to a:"&CR$
310 SET FKEY 13 N$&"text 40"&CR$
320 SET FKEY 14 N$&"set speaker off"&CR$
330 SET FKEY 15 N$&"set key click off"&CR$
340 SET FKEY 16 N$&"run ""clrkeys""&CR$
350 SET #133:PALETTE HBLUE,LBLUE,LBLUE,HBLUE
360 DISPLAY #133:AT 25 FROM 1 TO 3
370 END DEF
380 DEF HELPLINE
390 SET VIDEO X 40:SET VIDEO Y 4
400 SET VIDEO MODE 2:SET VIDEO COLOR 0
410 HANDLER DONTCARE
420 CONTINUE
430 END HANDLER
440 WHEN EXCEPTION USE DONTCARE
450 CLOSE #133
460 END WHEN
470 OPEN #133:"video:"
480 SET #133:INK 1
490 PRINT #133:" :DIR_ :CLS :CD E: to A:
TEXT 40 -off -off CLR FKEYS "
500 SET #133:INK 3
510 PRINT #133:" :DIR *.* :TYPE_ :LOAD_ :COPY_
TEXT 80 SPEAKER CLICK :WP "
520 SET #133:INK 1
530 PRINT #133:" F1 F2 F3 F4
F5 F6 F7 F8 "
540 END DEF

```

A billeentyűk kiosztása, ahol lehet, illeszkedik az IS-DOS alatt megszokott elrendezéssel. Hogy ne kelljen a billeentyűk hatását memorizálni, sem pedig a billeentyűsor fölötti emlékeztetőt csíkot cserélni, a program a képernyő senki által nem használt alsó három sorában létrehoz egy segítő menüt. Itt a kulcsszavak után álló aláhúzás karakter azt jelzi, hogy a billeentyű lenyomásával a parancs

kiadása nem fejeződik be, ahhoz egy vagy több paramétert kell megadni.

Az itt megadott billeentyűkiosztás természetesen nem kötelező előírás, azt mindenki a maga ízléséhez alakíthatja.

Nem lenne teljes a megoldás, ha az átdefiniált funkcióbilleentyűket nem lehetne egy mozdulattal visszadefiniálni. Ezt a feladatot az F16 funkcióbilleentyű látja el a CLRKEYS program (2. lista) indításával.

2. lista

```

100 PROGRAM "clrkeys"
110 STRING CR$,N$
120 NUMERIC LGREY,HGREY
130 LET LGREY=RGB(.6,.6,.6):LET HGREY=RGB(.3,.3,.3)
140 LET CR$=CHR$(13)
150 LET N$=CHR$(165)&CHR$(161)
160 CALL HELPLINE
170 CALL CLRKEYS
180 DEF CLRKEYS
190 CLEAR FKEYS
200 SET FKEY 16 N$&"run ""fkeys""&CR$
210 SET FKEY 4 N$&"load "
220 SET FKEY 12 N$&"save "
230 SET #133:PALETTE HGREY,LGREY,LGREY,HGREY
240 DISPLAY #133:AT 25 FROM 1 TO 3
250 END DEF
260 DEF HELPLINE
270 SET VIDEO X 40:SET VIDEO Y 4
280 SET VIDEO MODE 2:SET VIDEO COLOR 0
290 HANDLER DONTCARE
300 CONTINUE
310 END HANDLER
320 WHEN EXCEPTION USE DONTCARE
330 CLOSE #133
340 END WHEN
350 OPEN #133:"video:"
360 SET #133:INK 1
370 PRINT #133:" CONTINUE LLIST RENUMBER SAVE_
DISP TEXT DISP GR SPEAKER SET FKEYS "
380 SET #133:INK 3
390 PRINT #133:" START LIST AUTO LOAD_
TEXT GRAPHICS CLICK INFO "
400 SET #133:INK 1
410 PRINT #133:" F1 F2 F3 F4
F5 F6 F7 F8 "
420 END DEF

```

Ez néhány kivétellel visszaállítja a billeentyűk eredeti értelmét, és az előzőhöz hasonló segítő menüt ír ki. A menü eltérő színe is figyelmeztet a változásra. A lemezegység-gel dolgozó felhasználóknak nincs szükségük a REM1 és a REM2 funkcióbilleentyűre, így ezeket át lehet definiálni hasznosabbra. Megintcsak az F16 billeentyű adja a visszatérést a módosított billeentyűkhöz.

Ennek a megoldásnak a hátránya az, hogy a billeentyűk át- és visszadefiniálásakor elvész a memóriában esetleg ott lévő (és el nem mentett) program. Ilyankor használjuk időben a SAVE parancsot.

Ne feledkezzünk meg viszont arról a tényről, hogy lehetőségünk van egyszerre több programot a memóriában tárolni, és ezeket egymásból a CHAIN utasítással indítani. Egy kis átalakítással a "magnósok" is kialakíthatnak egy saját funkció billeentyű-kiosztást.

NYOMTATÓ HIÁNYÁBAN

A hiányos, felületesen elkészült dokumentációkból gyakran igen nehéz, vagy egyenesen képtelenség megérteni bizonyos dolgokat. Éppen ezért azok, akik komolyabb programok írásába kezdenek, kénytelenek a már elkészült munkákból elvesni trükköket, fogásokat.

Amíg valaki csak egy Basic listát tanulmányoz, addig nincs is különösebb problémája, hiszen az editor emlékszik azokra a sorokra, amelyek esetleg a képernyőből kúsztak. Emiatt a kíváncsiskodó könnyedén megnézheti, hogy pontosan hova ugrik a GOTO, vagy mit kezel egy adott HANDLER-blokk stb.

Assembly listák tanulmányozásakor már sokkal rosszabb a helyzet! Az ASMON csupán 16 sorban tudja a tárgy kód listáját megjeleníteni. Csak keserves munkával lehet egy teljes programot végigkövetni: a lista kifut a képből, szubrutinhívás miatt a rutint kell listázni, de közben nem jegyeztük fel a hívó címet, vagy elfelejtjük egy már megvizsgált rész funkcióját stb. Mennyivel jobb lenne, ha a listát szöveges állományként a WP-be tölthetnénk, és ott vizsgálhatnánk, megjegyzéseket fűzhetnénk hozzá!

Mindaz megoldható. Az EXOS csatornaalapú operációs rendszer, ezért furfangos dolgokat művelhetünk vele. Az ASMON "Printer options" funkciójával a listázást a PRINTER: eszköz helyett másra irányíthatjuk. Miután ide beléptünk (ennek módjáról a [H] (Help) billentyű lenyomásával kapunk tájékoztatást) definiálhatjuk a paramétereit. A dolgunk egyszerű, mert csak három helyen kell változtatnunk: a "Use:PRINTER:"-t írjuk felül a leendő állomány nevével, a "Lines"-zal állítsuk 255-re a sorok számát, végül a "Left margin"-t nullázva a sor elején lesz a bal margó.

Először a [P]-vel (Printer On/Off) megnyitjuk a csatornát, majd az [L] (Disassemble) billentyű lenyomásával, a cím beállításával az [ESC] billentyű lenyomásáig tart a listázás. Végül a [P]-vel zárjuk le a folyamatot, elkészült a lista.

A listafájlban már könnyedén mozoghatunk, megjegyzéseket tehetünk, sőt! Aki hajlandó egy kis kézi munkát végezni (a címeket törölni, címkéket megadni, a hexadecimális számokat törölni), az akár forrásszintű állományt is kreálhat!

A generált lista 16 KB-ig feldolgozható a WP-ben, de praktikusabb az ASMON saját szerkesztőjét alkalmazni, mert ide akár 46 KB-os szöveg is tölthető.

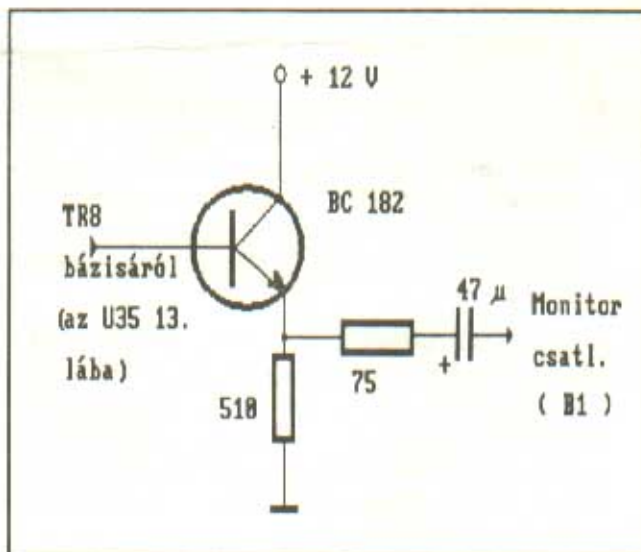
SZÍNES VIDEÓKIMENET

Tudvalevő, hogy számítógépünk "csak" professzionális kimenetekkel rendelkezik a monitorok számára: analóg RGB-jellel a színes, monokróm videójellel a fekete-fehér monitorok használatához. Ha televíziókat az antenna-csatlakozón keresztül nagyfrekvenciás jellel működtetjük, jelentősen romlik a felbontás, és kénytelenek vagyunk időnként utánhangolni, mert a számítógépbe épített RF modulátor nem a stabilitásáról híres.

Sok televízión van manapság videóbemenet, és sok monitor van, amelyik csak videójeleket fogad. (Nem beszélve a képmagnetofonokról.) Az ezekhez való csatlakozást teszi lehetővé az itt ismertetendő illesztőfokozat, ami beépíthető a gépbe, és egy üres csatlakozóponton át kivezethető a színes (PAL) videójel.

(Hasonló átalakítást vállalt el az egyik ártatlan ENTERPRISE-on egy "professzionális" szerviz, aminek eredménye a gépet átmenetileg használhatatlanná tevő kontakthiba és olyan szabálytalan videójel volt, amitől a kép oldalra gyűrődött a világosság függvényében.)

A színinformációt is tartalmazó jelet az eredeti kapcsolási rajzon TR8-cal jelölt (2N3904) tranzisztor bázisáról vesszük le. A bázisára kapcsolódó U34 (LM1886) és U35 (LM1889) integrált áramkörök állítják elő az összetett videójelet. A beépítendő egytranzisztoros fokozat az *ábrán* látható. A beépítés részleteit következő lap-számunkban olvashatják.



GYORSABB DIRECTORY

Mint köztudott, a lemez aktuális könyvtárának tartalmát a "DIR" parancs segítségével írathatjuk ki. Ez a kiírás általában valamilyen editorcsatornára irányul. Mivel ez a periféria meglehetősen intelligens, sok olyan műveletet is elvégez, ami számunkra ennél a funkcionál felesleges és időrabló is. Ezen a problémán segíthetünk, ha az alapértelmezésű csatornának egy videólapot jelölünk ki. Ezt megtehetjük úgy, hogy a 4. EXOS rendszerváltozóba (DEF_CHAN) beírjuk ennek a csatornának a számát. Basic-ben a megoldás a következő:

```
SET 4,102
SET SCROLL ON
```

A második sorra azért van szükség, hogy a videólap betelése esetén a rendszer felfelé tudja léptetni a karaktereket. Ha most kérjük a directory-t, a listázás gyorsabb lesz. A megoldás hátránya, hogy editorcsatornánk nem azt a képet látja, amit mi.

RÖVIDEN AZ EDITORCSATORNÁRÓL

Egyik laptársunk egy olvasói levél alapján – arra hivatkozva, hogy ez az EXOS leírásból hiányzik – ismereti a saját editorcsatorna megnyitásának módját, mégpedig pont az EXOS leírás szerint (lásd a 243. oldalon). Mivel azonban a lapban megjelent ismertetés hiányos, az EXOS leírás pedig köztudottan csapnivalóan rossz fordítás – és egyébként sincs meg mindenkinek –, most közhírtően, példával illusztrálva összefoglaljuk a teendőket, és kiegészítjük egy-két olyan tudnivalóval, ami sehol sem jelent meg. Nos, az eljárás a következő:

1. Nyissunk meg egy videócsatornát 40 vagy 80 oszlopos szöveges üzemmódban. A függőleges méret (VIDEO Y vagy Y_SIZ_VID) legalább 3 sornyi legyen, a vízszintes méret (VIDEO X vagy X_SIZ_VID) pedig legalább 4 karakternyi.
2. Most (esetleg később) jelenítsük meg a videócsatornát a képernyőn.
3. Az EDITOR VIDEO gépi opció (a VID_EDIT rendszerváltozó) értéke legyen az 1. pontban megnyitott videócsatorna száma. Ha semmit nem teszünk, itt az utoljára megnyitott editorcsatorna száma van; BASIC alatt ez 102, (a szabványos szöveges csatorna).
4. Az EDITOR KEY gépi opció (a KEY_EDIT rendszerváltozó) értéke legyen az editorcsatornához használni kívánt - nyitott - billentyűzetcsatorna száma. Ha semmit nem teszünk, itt az utoljára megnyitott billentyűzetcsatorna száma van; ez a Basic alatt 105 (a KEYBOARD: eszközre megnyitott szabványos csatorna); más programnyelvénél más lehet, illetve nekünk kell megnyitni billentyűzetcsatornát. (Itt más eszközre megnyitott csatornát is megadhatunk, ezzel érdekes trükköket lehet csinálni, de erről majd máskor.)
5. Az EDITOR BUFFER gépi opció (a BUF_EDIT rendszerváltozó) értéke legyen az editorcsatornához szükséges puffer mérete (256 bájt az alapegység). Ha semmit nem teszünk, itt az utoljára megnyitott editorcsatorna puffermérete van, ez Basic alatt 8 egység (2 Kbájt). Vigyázzunk: túlzott nagyvonalúság mellett hamar elfogy a memória!
6. Most megnyithatjuk az editorcsatornát (EDITOR: eszköznévvel). Ha kevés a videómémória, és nincs szükségünk a szabványos editorcsatornára, le is zárhatjuk (a hozzá tartozó videócsatornával együtt) a saját csatorna (vagy csatornák) megnyitása előtt. Ezzel azonban vigyázzunk: ha nem adunk meg alapértelmezés szerinti csatornát (a DEFAULT CHANNEL gépi opcióval, illetve a DEF_CHAN rendszerváltozóval), a rendszer nem tudja megjeleníteni az esetleges hibajelzéseket.

Ügyeljünk arra is, hogy a gépi, illetve videóopciók (rendszerváltozók) idejében visszakapják eredeti értéküket, különben furcsa jelenségeket tapasztalunk. A Basic például visszaállítja a 102-es csatornát az editor videó-

csatornájának, de nem változtatja meg a pufferméretet, és nem állítja vissza a billentyűzetcsatorna számát sem.

Egy példa Basic-ben, más programnyelven az eljárás gyakorlatilag ugyanez:

```

100 PROGRAM "Editorcsatorna"
110 I 1. lps:
120 SET VIDEO X 40
130 SET VIDEO Y 4
140 SET VIDEO MODE 0
150 SET VIDEO COLOR 0
160 OPEN #1:"video:"
170 I 2. lps
180 DISPLAY #1:AT 1 FROM 1 TO 4
190 I 3. lps
200 SET EDITOR VIDEO 1
210 I 4. lps
220 SET EDITOR KEY 105
230 I 5. lps
240 SET EDITOR BUFFER 1
250 I 6. lps
260 OPEN #2:"editor:"
270 WHEN EXCEPTION USE VISSZA
.
.
.
990 END WHEN
1000 HANDLER VISSZA
1010 SET EDITOR BUFFER 8
1020 DISPLAY TEXT
1030 END HANDLER

```

ÜZENET A STÁTUSZSORBAN

Mindenki tudja, hogy a SET 26,42 parancs hatására az állapotkijelző sorban az operációs rendszer készítőinek monogramjai jelennek meg. Ezt a névsort a gép a rendszerszegmensből olvassa ki, amit saját szövegeinkkel felülírhatunk. A RAM-terület a szegmensen belül a 12848. offseten kezdődik, hossza 34 bájt.

```

100 PROGRAM "STLINE.BAS"
110 LET A$="ENTERPRESS STATUS DEMO"
120 CALL MAIN(A$)
130 SET 26,42
140 DEF MAIN(P$)
150 LET P$=P$(1:34)
160 FOR I=1 TO LEN(P$)
170 SPOKE 255,12847+I,ORD(P$(I:I))
180 NEXT
190 FOR J=I TO 34
200 SPOKE 255,12847+J,32
210 NEXT
220 END DEF

```

Rendszerindítás a START programmal

A lemezegységgel rendelkező felhasználók számára igen kényelmes, ha csak be kell tenni a lemezt a meghajtóba, és az adott feladathoz tartozó szolgáltatások maguktól elindulnak. Erre az egyik megoldás az EX-DOS.INI file használata. Ennek korlátot szab az, hogy ez csak hidegindításkor hajtódik végre automatikusan, és hogy csak EXDOS parancsokat képes végrehajtani. Ugyanakkor sok problémára a legegyszerűbb megoldás - ha a gyorsaság nem követelmény - egy kis BASIC program írása.

Egy elfelejtett trükk a START billentyű felhasználása. Tegyük a gyakrabban használt lemezekre egy-egy START nevű programot, amely elvégzi a hiányzó beállításokat, és még sok minden más szükséges apró feladatot. Amikor leültem ENTERPRISE gépemhez ezt az írást elkészíteni, az újságcikkeket gyűjtő lemezen lévő START program elindította a karakterkészlet-módosító programot, amely a teljes magyar ékezetes karakterkészletet előállítja; innen léptem be a szövegszerkesztőbe.

Az itt közölt változat először megállapítja, hogy nem maradt-e el a bekapcsolás után a dátum beállítása, és szükség esetén pótolja a mulasztást. Ezután megkérdezi, hogy hány lapos RAMDISK-et hozzon létre. Ha itt a válasz 0, nem csinál semmit; ha más érték, létrehozza a virtuális lemezt (vigyázzunk, ha már van RAMDISK-ünk,

és van rajta valami). A virtuális lemezre, ha szükséges, átmásolja a 8. oldalon közölt funkcióbillentyű-átdefiniáló és -visszállító programot, majd el is indítja az el-sőt.

```

100 PROGRAM "Start"
110 IF DATE$="19800000" THEN
120   EXT "var 79 185"
130   DATE
140   TIME
150 END IF
160 INPUT PROMPT "Randisk size: ":N
170 IF N>0 THEN
180   EXT "randisk "&STR$(N)&"/d"
190   PRINT "Copy FKEYS and CLRKEYS to E: (Y/N)? ";
200   DO
210     LET X$=INKEY$
220     LOOP UNTIL X$<>""
230     PRINT CHR$(164)
240     LET X$=UCASE$(X$)
250     IF X$="Y" OR X$=CHR$(13) THEN
260       EXT "copy fkeys e:"
270       EXT "copy clrkeys e:"
280     END IF
290     EXT "e:"
300     PRINT "Logged drive is E:"
310   END IF
320 RUN "fkeys"

```

ÖRÖKÉLETKÓDOK

Az ENTERPRISE külön fejrészformát használ a gépi kódú játékprogramok elindításához. Ezeknél a betöltő részt is gépi kódban írták meg, így ezekbe utólag belenyúlni csak valamilyen assemblerrel lehetséges. A következőkben három, eredetileg Spectrumra írt szoftver átalakításához adunk tanácsokat, assemblerként az ASMON-t használjuk.

Egy adott fájl betöltése után az ASMON mindig kiírja az utolsó címet (Last address). Ha az Öné nem egyezik meg az itt közölttel, akkor az egy másik verzió, a további teendők nem érvényesek. Az új betöltőt egy másik kazettára rögzítse. Az örökéletes változat használata: indítsa el az átírt betöltőt, tegye be a régi kazettát, hogy a gép a hátralevő modulokat is beolvashassa. Persze sokkal kényelmesebb, ha valamilyen másolóval a fájlokat egy kazettára teszi.

A lenyomandó billentyűket szögletes zárójel közé tettük, természetesen a számokat, címeket is nekünk kell megadnunk. Az [R]-el beolvassuk, [M]-el módosítjuk, [S]-el a másik kazettára rögzítjük a programot.

DAN_DARE
- [R] 10F0 [ENTER] BFFF [ENTER] DAN_DARE [ENTER] 'Last address: 16F2' - [M] 1253 [ENTER] F5 3E C9 32 62 BA F1 C3 58 98 [ESC] - [S] 10F0 [ENTER] 16F2 [ENTER] DAN_DARE [ENTER]
JACK_THE_NIPPER
- [R] 10F0 [ENTER] BFFF [ENTER] JACK_THE_NIPPER [ENTER] 'Last address: 1269' - [M] 10F2 [ENTER] 83 [ESC] - [M] 11B5 [ENTER] 70 02 [ESC] - [M] 1270 [ENTER] E5 21 FC A9 36 3E 23 36 04 23 36 B7 23 36 00 E1 C3 00 40 [ESC] - [S] 10F0 [ENTER] 1282 [ENTER] JACK_THE_NIPPER [ENTER]
MONTY_ON
- [R] 10F0 [ENTER] BFFF [ENTER] MONTY_ON [ENTER] 'Last address: 17DB' - [M] 1280 [ENTER] 3E FF 32 7F 98 C3 63 02 [ESC] - [S] 10F0 [ENTER] 1287 [ENTER] MONTY_ON [ENTER]

HEKTIK

Az akadályok között elhelyezett összes értékes karaktert kell összeszednünk a játékban. Saját figuránk mozgása a beépített botkormányal történhet. A szóköz lenyomásával bármikor kiléphetünk a játéktérből. A végleges kilépés az ESC leütésével lehetséges.

A program közvetlenül a videó RAM-ba ír, így viszonylag gyorsan fut. Érdeemes áttanulmányozni a SEG, és az OFS függvényeket, mert ezek adják vissza a képernyőre kirakott videólapp szegmens-és ofszet címét.

A program a Zzzip-el lefordítható. A lassításhoz törölni kell az 1920, 1930-as sorok elejéről a felkiáltójeleket. Ha a 1920-as sorban szereplő 900-at csökkentjük (például 500-ra), akkor figuránk mozgása igencsak felgyorsul.

```

1000 PROGRAM "HEKTIK"
1010 NUMERIC DR(0 TO 4),MS(0 TO 1)
1020 LET DR(0)=1:LET DR(1)=-1
1030 LET DR(2)=40:LET DR(3)=-40:LET
      DR(4)=0
1040 LET MS(0)=0:LET MS(1)=5
1050 CLEAR FONT:CALL CHR$:SET 26,1:SET
      27,255
1060 SET 7,1:SET 10,1:SET 11,1
1070 SET 22,5:SET 23,1:SET 24,12:SET 25,1
1080 OPEN #1:"VIDEO:"
1090 SET #1:PALETTE 255,RED
1100 PRINT #1,AT 1,1:"HEKTIK"
1110 SET 22,0:SET 23,0:SET 24,40:SET 25,27
1120 OPEN #2:"VIDEO:"
1130 OPEN #3:"VIDEO:"
1140 PRINT #3:CHR$(27);"o"
1150 SET #2:PALETTE 255,148,132,31
1160 PRINT #2,AT
      8,11:CHR$(27);"oENTERPRESS
      PROGRAMS"
1170 DISPLAY #3:AT 1 FROM 1 TO 1
1180 LET S=SEG:LET F=OFS
1190 SET #3:PALETTE 255,36,219,73
1200 LET HI=2000:LET SC,DX=0
1210 IF SC>HI THEN LET HI=SC
1220 LET M=5
1230 PRINT #2,AT 15,16:"HIGH:";
1240 CALL RES(HI,2,15,21)
1250 PRINT #2,AT 17,15:"SCORE:";
1260 CALL RES(SC,2,17,21):LET SC=0
1270 PRINT #2,AT 22,7:"      PLEASE
      WAIT!      "
1280 DISPLAY #2:AT 1 FROM 1 TO 27
1290 DISPLAY #1:AT 12 FROM 1 TO 1
1300 CALL SCR(M):LET CT=5*M
1310 DO
1320 LET G=RND(1040)+40
1330 LOOP UNTIL SPEEK(S,F+G)=32
1340 PRINT #2,AT 22,7:"PRESS ANY KEY TO
      CONTINUE..."
1350 SET 5,3
1360 SPOKE 255,16042,0 ! BRD==>11141
1370 WHEN EXCEPTION USE SCON
1380 GET #105:KB$
1390 IF KB$="" THEN 1380

1400 END WHEN
1410 IF KB$=CHR$(27) THEN
1420 CLEAR FONT:SET 27,0
1430 EXT "BASIC"
1440 END IF
1450 SET 5,0
1460 DISPLAY #3:AT 1 FROM 1 TO 27
1470 LET C=0:LET K=4
1480 SPOKE 255,16042,0 ! BRD==>11141
1490 SELECT CASE JOY(0)
1500 CASE 1
1510 LET K=0
1520 CASE 2
1530 LET K=1
1540 CASE 4
1550 LET K=2
1560 CASE 8
1570 LET K=3
1580 CASE 16
1590 GOTO 1210
1600 CASE ELSE
1610 !
1620 END SELECT
1630 SELECT CASE SPEEK(S,F+G+DR(K))
1640 CASE 96,97
1650 LET SC=SC-5
1660 LET K=4
1670 CALL RES(SC,3,1,10)
1680 SOUND PITCH 0,DURATION 2
1690 CASE 146
1700 LET SC=SC-10
1710 LET K=4
1720 CALL RES(SC,3,1,10)
1730 SOUND PITCH 10,DURATION 1
1740 CASE 147
1750 LET SC=SC+10
1760 CALL RES(SC,3,1,10)
1770 LET CT=CT-1
1780 SOUND PITCH 70,DURATION 2
1790 CASE 148
1800 LET SC=SC+50
1810 CALL RES(SC,3,1,10)
1820 LET CT=CT-1
1830 SOUND PITCH 80,DURATION 2
1840 CASE 149
1850 LET SC=SC+100

```

```

1860 CALL RES(SC,3,1,10)
1870 LET CT=CT-1
1880 SOUND PITCH 90,DURATION 1
1890 CASE ELSE
1900 !
1910 END SELECT
1920 IFOR Y=0 TO 900 !ZZZIP
1930 !NEXT
1940 LET G=G+DR(K):LET DX=DX+1 BAND
1
1950 SOUND PITCH MS(DX),DURATION
5,LEFT 60,RIGHT 60,SOURCE 1
1960 SPOKE S,F+G,102
1970 IF G<>C THEN
1980 SPOKE S,F+C,32:LET C=G
1990 END IF
2000 IF CT<>0 AND SC=0 THEN 1480
2010 IF SC>0 THEN
2020 IF SC<=HI
2030 LET M=M+5
2040 CALL SCR(M):LET CT=5*M
2050 GOTO 1470
2060 ELSE
2070 GOTO 1210
2080 END IF
2090 END IF
2100 GOTO 1210
2110 DEF SEG
2120 LET
B=PEEK(30000+19140)+256*PEEK(30000
+19141)+21
2130 LET A=(PEEK(B) BAND
BIN(11000000))/64
2140 LET SEG=A BOR BIN(11111100)
2150 END DEF
2160 DEF OFS
2170 LET
B=PEEK(30000+19140)+256*PEEK(30000
+19141)+21
2180 LET A=PEEK(B) BAND BIN(111111)
2190 LET OFS=PEEK(B-1)+256*A
2200 END DEF
2210 DEF RES(R,X,YP,XP)
2220 LET A$="000000"
2230 LET X$=STR$(R)
2240 LET X$=A$(1:6-LEN(X$))&X$
2250 SET #X:INK 2
2260 PRINT #X,AT YP,XP:X$;
2270 SET #X:INK 1
2280 END DEF
2290 DEF SCR(N)
2300 CLEAR #3
2310 FOR T=0 TO 25
2320 SPOKE S,F+40+T*40,96:SPOKE
S,F+40+T*40+39,96
2330 NEXT
2340 FOR T=1 TO 38
2350 SPOKE S,F+40+T,97:SPOKE
S,F+40+1000+T,97
2360 NEXT
2370 SPOKE S,F+40,98:SPOKE S,F+79,99
2380 SPOKE S,F+1040,100:SPOKE
S,F+1079,101
2390 FOR T=1 TO 10*N
2400 DO
2410 LET P=RND(1040)+40
2420 LOOP UNTIL SPEEK(S,F+P)=32
2430 SPOKE S,F+P,146
2440 NEXT
2450 FOR T=1 TO 5*N
2460 DO
2470 LET P=RND(1040)+40
2480 LOOP UNTIL SPEEK(S,F+P)=32
AND SPEEK(S,F+P-1)=32 AND
SPEEK(S,F+P+1)=32
2490 SPOKE S,F+P,RND(3)+147
2500 NEXT
2510 PRINT #3,AT 1,4:"SCORE:000000
HEKTIK HIGH:";
2520 CALL RES(HI,3,1,33):CALL
RES(SC,3,1,10)
2530 END DEF
2540 DEF CHR$
2550 SET CHARACTER
96,24,24,24,24,24,24,24,24,24
2560 SET CHARACTER
97,0,0,0,255,255,0,0,0,0
2570 SET CHARACTER
98,0,0,0,7,15,28,24,24,24
2580 SET CHARACTER
99,0,0,0,224,240,56,24,24,24
2590 SET CHARACTER
100,24,24,28,15,7,0,0,0,0
2600 SET CHARACTER
101,24,24,56,240,224,0,0,0,0
2610 SET CHARACTER
146,255,129,153,189,189,189,153,129,255
2620 SET CHARACTER
147,0,60,102,66,90,66,255,36,66
2630 SET CHARACTER
148,36,126,153,165,102,24,60,66,129
2640 SET CHARACTER
149,0,60,126,129,165,129,153,66,60
2650 SET CHARACTER
102,60,126,255,255,255,255,126,60
2660 END DEF
2670 HANDLER SCON
2680 DISPLAY #3:AT 1 FROM 1 TO 27
2690 WAIT 1
2700 DISPLAY #2:AT 1 FROM 1 TO 27
2710 DISPLAY #1:AT 12 FROM 1 TO 1
2720 SET 5,3
2730 RETRY
2740 END HANDLER

```

PERCENKÉNT HATVAN KERETSZÍN

Az alábbi lista begépelése és elindítása után a keret minden másodpercben más színt vesz fel:

```

100 PROGRAM "FLASH.BAS"
110 ALLOCATE 64
120 REM -----
130 CODE R1=HEX$("DB,B2") ! In A,(0B2h)
140 CODE =HEX$("F5") ! Push AF
150 CODE =HEX$("AF") ! Xor A
160 CODE =HEX$("3D") ! Dec A
170 CODE =HEX$("D3,B2") ! Out (0B2h),A
180 CODE =HEX$("3A,F0,BF") ! Ld A,(0BFF0h)
190 CODE =HEX$("32,E0,BF") ! Ld (0BFEDh),A
200 CODE =HEX$("F1") ! Pop AF
210 CODE =HEX$("D3,B2") ! Out (0B2h),A
220 CODE =HEX$("C9") ! Ret
230 REM -----
240 CODE R2=HEX$("F3") ! Dí
250 CODE =HEX$("22,ED,BF") ! Ld (0BFEDh),HL
260 CODE =HEX$("FB") ! Eí
270 CODE =HEX$("C9") ! Ret
280 REM -----
290 CALL USR(R2,R1)

```

A magyarázatot a 240. sortól kezdem. Az "R2"-es gépi kódú rutin illeszti a gép megszakításrendszerébe a villogtatást végző "R1"-t. Először letiltja a megszakításokat, nehogy a címbeírás közben elinduljon rossz mutatóértékkel a felhasználói megszakítás kiszolgálása. A beállítás után ismét engedélyezzük a megszakításokat, és visszatérünk a Basic-be.

Az "R1" rutin először elmenti a 2-es lap tartalmát, majd belapozza ide a 255-ös számú rendszerszegmenst. Kiolvassa a SEC_COUNTER másodpereszámológó tartalmát, és közvetlenül beírja azt a BORD_VID címre. Ezután visszaállítja a 2-es lap eredeti állapotát, és befejezi munkáját. A villogtatás megállítható a CALL USR(R2,0) utasítással.

Jogosan merül fel a kérdés: mire jó ez az egész? A rutin olyan helyeken használható, ahol valamilyen lassú, "nem látható" folyamat közben (pl. keresés, tömbök inicializálása stb.) a gép előtt ülőt szeretnénk meggyőzni arról, hogy a gépe nem fagyott le, csak egy kicsit nehéz feladatot kapott, és most azzal bibelődik ilyen hosszán.

LÁTSZÓLAG TÖBB

Számítógépünk a rendszerszegmens 16291. ofszet címén működő, a következőn pedig a rossz szegmensek számát tárolja. Ha ide mindenféle hamis értékeket teszünk, akkor az INFO parancs a látszólagos memória méretét jeleníti meg. A

```

SPOKE 255,16291,255
SPOKE 255,16292,1

```

sorokat példaként megadva a gép látszólag 4 Mbajt memóriával rendelkezik. Legjobb, ha az értékeket a kísérletezés után eredetire visszaállítjuk (vagy kétszer megnyomjuk a reset gombot), mert különben az ENTERPRISE nem áll jól önmagáért.

VIDEÓLAPOK POZÍCIONÁLÁSA

Amikor egy videólapot a megnyitás után elhelyezünk a képernyőre, akkor azt az operációs rendszer automatikusan középre állítja.

A sorparamétertáblában bizonyos bájtok átírásával mi is pozícionálhatjuk az ablakot. Ezt teszi a következő Basic program:

```

100 PROGRAM "VPOS.BAS"
110 DEF VPOS(L,E,D)
120 LET Z=16372
130 LET A=SPEEK(255,Z)+256*SPEEK(255,Z+1)
140 LET A=A+L*16:LET LEM=A+2:LET RIM=LEM+1
150 FOR J=1 TO E
160 POKE LEM,PEEK(LEM)+D
170 POKE RIM,PEEK(RIM)+D
180 LET LEM=LEM+16:LET RIM=RIM+16
190 NEXT J
200 END DEF
210 SET VIDEO MODE 5
220 SET VIDEO COLOR 0
230 SET VIDEO X 10
240 SET VIDEO Y 8
250 OPEN #30:"VIDEO:"
260 SET #30:PALETTE 13,32
270 PLOT #30:150,150,ELLIPSE 100,100,
280 DISPLAY #30:AT 1 FROM 1 TO 8
290 CALL VPOS(1,8,-10)

```

A függvénynek meg kell adnunk az eltolandó első sort, az eltolás hosszát és azt, hogy a középállástól mennyivel térjen el az ablak. A 150-190-es sorokban beírjuk a lap margóit meghatározó memóriahelyekre a módosított értékeket.

A sorparamétertábla újraépülésekor (pl. a TEXT parancsra) a régi értékek fognak szerepelni.

HIBÁS A PRINT USING!

A Basic programnyelvben a kiírandó adatok formálásának eszköze a PRINT USING utasítás. Szomorú tény, hogy az IS-BASIC ennek az utasításnak a végrehajtását hibásan végzi. Sem a 12 jegyű, sem a 14 jegyű számokkal nincs problémája, 13 jegyű szám kiírása után viszont leáll a

```
'BASIC data has been corrupted'
```

hibaüzenettel. Hogy miért éppen a 13 jegyű számokat nem tudja kiírni? Végül is a felhasználót ez nem érdekli.

A másik, talán még nagyobb probléma sokkal veszélyesebb, mert hibajelzés nélkül, észrevétlenül követi el. Nagy pozitív és negatív számoknál nincs hiba, viszont az 1-nél kisebb és -1-nél nagyobb számok helyett a tízszerecsüket írja ki a program. Mivel az adatok nagyságrendjét a feladat határozza meg, a PRINT USING utasítást nem lehet teljes biztonsággal használni. Kár...

Hirdetésfelvétel

Az apróhirdetések ára: 1 Ft karakterenként. A szöveget és a befizetést igazoló nyugtát (rózsaszín postautalványon) az alábbi címre kérjük elküldeni:

MÁTRIX Kft.



ENTERPRESS
8000 Székesfehérvár
Dózsa György tér 10.

Bankszámlaszám: OTP 679-022096-9

Közületi hirdetőknak kívánságra hirdetési árajánlatot küldünk.

Válaszoljon, hogy jobb lehessen!

A kérdőív segítségével pontos képet szeretnénk kapni olvasóink kívánságairól, elvárásairól. Lehetőségeink határain belül minél több géptulajdonos igényének szeretnénk megfelelni.

A kérdésekre értelemszerűen válaszolva, a kérdőívet borítékba téve az alábbi címre küldje:



Letét 334
ENTERPRESS
1399 Budapest
Pf. 701.

MÁTRIX Kft.

MEGRENDELŐLAP

Megrendelem az ENTERPRESS című lapot

..... példányban az alábbi címre:

Megrendelő neve:

.....

Címe:

.....
(város, község)

.....
(utca, tér, lakótelep)

.....
(emelet, ajtó)

Irányítószám:

Előfizetési díj egy évre 294 Ft, félévre 147 Ft.

Az előfizetési díjat a részemre küldendő átutalási postautalványon egyenlitem ki.

.....
alírás

A megrendelőlapot borítékban bérmentesítve az alábbi címre kérjük feladni:

Hírlapelőfizetési és Lapellátási Iroda
Budapest
1900

ENTERPRESS KÉRDŐÍV

- Jónak találja a lap címét?
- Elégedett a lap tartalmával?
- Elégedett a lap külsejével?
- Jól olvashatók a programlisták?
- Szívesen látna a lap oldalain hosszabb programokat?
- Küldene írásokat, programokat a lapnak?
- Szívesen olvasna az ENTERPRISE helyzetével foglalkozó írásokat?
- Szívesen olvasna különböző hardver és szoftver teszteredményeket?
- Érdekelnék programleírások?
- Elfogadhatónak tartja a lap árát?
- Szokott nagyobb programokat írni?
- Tagja valamelyik ENTERPRISE klubnak?

Milyen konfigurációja van? (A gépen és a magnón kívül.)

- lemezegység
- nyomtató
- egér

ENTERPRESS HESKRENDÉLŐLAP

MÁTRIX

Számítástechnikai, Szolgáltató és Kereskedelmi Kft.

- Tanfolyamok szervezése
 - » számítógépes alapismeretek
 - » szövegszerkesztők
 - » DTP-k (kiadványszerkesztők)
 - » video tanfolyamok
 - » vállalkozói tanfolyamok
 - » nyelvtanfolyamok
 - » érettségi, felvételi felkészítők
- Számítógépes kiadványszerkesztés
 - » névjegykártyák, meghívók, szórólapok
 - » jegyzetek, könyvek
 - » plakátok, termékismertető
- Szoftverek fejlesztése
 - » raktárnyilvántartó, könyvelő, számlázó stb. programok PC-re
- Számítástechnikai, irodatechnikai, informatikai eszközök forgalmazása

MÁTRIX Kft.



Székesfehérvár
Dózsa Gy. tér 10.
Telefon: (22) 16-520/141
Telefax: (22) 11-585

ENTERPRISE belső memóriabővítés
320KB-ra, hibátlan RAM frissítéssel. Átkapcsolás 128KB/320KB üzemmód között.
Bozai Gábor 8000 Székesfehérvár,
Budai út 92. fsz. 5. Üzenet: (22) 20-857

beszédszintetizátor
Spectrum emulátor
egyéb:

Milyen programozási nyelveket ismer?

semmilyen sem
Basic
Pascal
Assembly
egyéb:

Mire használja leginkább számítógépét?

játék
programozás
feladatmegoldás
szövegszerkesztés
egyéb:

Miről olvasna szívesen sorozatot?

Milyen témájú rövidebb írások érdeklnek?

Mint ENTERPRISE tulajdonosnak, milyen a közérzete?

Egyéb megjegyzés: