

μ

mikro számítógép magazin

RAJZOLJUNK

SZÁMÍTÓGÉPPEL!

20 éves az
UNIX

Az ismeretlen
DOS-parancs

Mit tud a
WORD?

LOGIKUS!!



1990/1



A NEUMANN JÁNOS SZÁMÍTÓGÉPTUDOMÁNYI TÁRSASÁG LAPJA

A szerkesztőbizottság vezetője:
Kovács Győző

A szerkesztőség munkatársai:
Bakos Tamás
(programozástechnika)

Broczkó Péter
(hírek)
Kis János
Kovács Győző
(levelezés)
Nagy Imre
(tanuljunk együtt)
Petróczy Judit
(könyvek)
Pinke György
(Enterprise)

Soltészné Vizi Zsuzsa
(tervezőszerkesztő)
Szebenszki Sándor
Tamásné Lakó Erika
Terebessy Ákosné
Varga János
(olvasószerkesztő)

Címlapunk:
Velekey József Lajos
munkája

Felelős szerkesztő:
Könyves Tóth Pál

Szerkesztőség:
1027 Budapest, Fő u. 68.
Telefon: 115-4250

Levél cím:
1371 Budapest Pf. 433

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

Levél cím:
1368 Budapest 5. Pf. 240

Telefon: 132-9349

Felelős kiadó:
Tóth Istvánné ügyvezető
főtitkárhelyettes

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

Megjelenik havonta.
Egy szám ára 31,— Ft
Előfizetési díj:
egy évre 372,— Ft
fél évre 186,— Ft
Külföldön terjeszti
a Kultúra,
1389 Budapest, Pf. 149
és a Magyar Média
1932 Budapest, Pf. 279
88—1135

89 - 2681 - Révai Nyomda,
Budapest
Felelős vezető:
Horváth Józsefné dr.

INDEX: 25 629
ISSN 0236-6088

TARTALOM

- 3 Politizálunk . . .
- 10 Feladatok — megoldások
- 23 Blokk törlés, összevonás, szétválasztás
- 36 Az Amiga programozása assembly nyelven
- 41 Az ismeretlen DOS-parancs
- 42 Számítógép a kézben
- 43 A Solarsoft kínálatából
- 44 Adok — veszek — cserélek
- 44 Pályázati felhívás

TANULJUK EGYÜTT!

4

- 4 Prologia
- 5 A REALNET és a TVCNET hálózatok
- 7 Rajzoljunk számítógéppel!

CSIPEGETŐ

12

- 12 Hullámoztató program
- 12 $2 \times 2 = ?$
- 13 Rasztercsíkok
- 13 Grafikus rutinok C Plus/4-re
- 14 Elforgatott ellipszisek
- 14 Hogy a lista újra TOP legyen!
- 15 Szoftverszolgálat
- 15 Felhívás!
- 15 TOP-lista

PROGRAMOZÁSTECHNIKA

16

- 16 BASIC-bővítések Commodore 16-ra
- 17 20 éves a Unix
- 21 Semmi sem tökéletes ezen a világon
- 22 Programozási fogások és melléfogások

ENTERPRISE

24

- 24 Területek és funkciók
- 26 EXOS gyakorlatok
- 27 Az IBM és az Enterprise
- 27 Mi a manó?

PÉCÉZZÜNK!

28

- 28 Egy csokor batch-fájl
- 33 Microsoft Word 4.0
- 35 384 kb-ajtos alaplemez

KÖNYVEK — HÍREK — ÉRDEKESSÉGEK

45

AZ OLVASÓ ÍRJA

47



Politizálunk...

„Semmi sem emelheti fel anyaföldünket, csak agyvelünk 's kezeink; 's nem hiányos geographiai helyzetünk oka kereskedésünk nem létének.”

(Széchenyi István: Hitel)

Csodálatosan érdekes időkötünk, s ezt nemcsak abból lehet érezni, hogy az embert külföldön jártában-keltében egyszerűen csak a magyarsága miatt — teljesen érdemtelenül — szinte Európa vagy inkább a mévelt világ hőseként tisztelik, hanem abból is, hogy megszorodtak külföldi látogatóink, akik örömmel vállalkoznak, hogy akár egy konferenciára, de még egy nyúlfarknyi megbeszélésre is Magyarországra jöjnek. Jönnek és nemcsak nevezetes ünnepnapjainkon, szenciára éhesen — akkor különösen —, de „csendes” hétköznapjainkon is, és nem is kevésszer segítő szándékkal, ami mögött nyilvánvalóan a jövőbeni együttműködésekkel származó gazdasági haszonnak a lehetősége is ott lebeg.

Hallgatva az elmúlt hetek pártkongresszusait, figyelve régi és új politikusiainkat a tételre (első sorban a magyar csárdást járják, ami tudvalevően kettő balra, kettő jobbra), nem vagyok egészen biztos abban, hogy valóban itt van már a főt emlelt ideális állapot, amikor is a munka, pontosabban az eredmény határozza meg az ember helyzetét a társadalomban. (Csak zárójelben jegyzem meg egy sokszor hangzottatott elgondolást, hogy például május elsejét nem a munka ünnepe, hanem a teljesítmény ünnepe, a ki tudja még meddig meglevő érdemeket és érdemrendeket pedig nem Munka-érdemrendnek, hanem Teljesítményérdemrendnek kellene elnevezni. Nem tudom, hogy azt juttunk-e ide, ahol ma vagyunk, mert eddig az állam nem a teljesítményt, hanem a munkát kívánja meg; talán ezért volt az egyik legnagyobb civil elismerés a „Szocialista munka hőse” cím elnyerése, ahelyett, hogy a szocialista teljesítmény hőseként ünneplött volna a termelés legkiválóbbjait. Talán majd a hatalom átvevő új kurzus, de még az is lehet, hogy már a jelenlegi kormány is elgondolkozik azon, hogy nem volna-e érdemes a napi gyakorlatban is a munka ellenében a teljesítményt favorizálni, még a szóhasználatunkban is.)

Visszatérve a csárdás hasonlatra, egyelőre úgy tűnik, hogy az emberekben élenkel még az elmúlt negyven év beidegződése, hogy ti. a karrierhez nemcsak hogy fontos, de szükséges is a politikában való nagyon aktív részvétel. Számomra egyszerűen mulatságosnak tűnik, hogy azok a karriervadászok, akik társadalmi, de sokszor szakmai sikereiket, még a számítástechnikában is, nem kimondottan szakmai kiváltságaikkal, sokkal inkább politikai „rátermettségükkel” érték el, most nem is kicsit meg vannak zavarodva. Egyelőre nem lépnék be egyik pártba sem, hiszen nem tudják, hogy melyik párttagság valójában a karrier igazi útja. Még bele gondolni is borzasztó, hogy ne adj' Isten, visszalépnék az MSZP-be, és az SZDSZ vagy az MDF győz a választásokon. Arról nem is beszélve, hogy mi történik, ha jön egy visszarendeződés, és akkor talán mégiscsak a sorait újraszervező MSZMP vagy talán még egy ennél is balosabb formáció volna a legjobb választás. Szinte magam előtt látom ezeket a valójában szerencsétlen embereket, amint napról napra fűrészlik a politikai széljárás, hiszen még tévútvány is veszélyes számukra, mert előfordulhat, hogy valaki későn lép, és akkor a másik „vadászt” egyszerűen elcsípi előle a legjobb lehetőséget. Arra a kissoresre már szinte gondolni sem merek, hogy ezeknek az embereknek a végén még el kell kezdeniük dolgozni. Ez aztán valóban nagyon kínos lenne.

E kis politikai kitérő után visszatérek üzleti partnereimhez, akikről voltaképpen ebben a szerkesztő-

ségi cikkben írni akartam. Azt már elmondtam, hogy rendkívüli módon érdeklő őket minden, ami itt történik, de kíváncsiságuk egyelőre nagyobb, mint a bizodalom. Rettenően nagyra értékelik, hogy elútom az a vasfüggöny, amiről nagyon sokáig tagadtuk, hogy létezik, most meg ezt a nincset élelmes hazai állami és magánvállalkozóink árulják a bécsi jobb ajándéküzletekben.

Rendkívüli módon dűjják új demokratikus intézményeinket, a szabad szakszervezeteket, az egyházak önállóságát, a nem „felügyelt” sajtót, a nem csak gomba módra, de ostromdással is szaporodó pártjainkat. Mindent nagy érdeklődéssel figyelnek, mindenről van véleményük, csak egyetlen dolgot hajlanak nehezen: nagyon lassan döntenek el, hogy befektessenek-e Magyarországra vagy sem. Az egyik üzleti partneremnek megkérdeztem összehozni egy bécsi és egy budapesti székhelyű vegyesvállalatot. A bécsit megcsinálta, nem kellett hozzá két óra, míg a budapestin idestova több mint egy éve tollazkodik. Persze a dolog nem hagyott nyugodni, megpróbált nyomára jutni a titoknak. Azt ugyanis valamennyi partnerem ímszpongetti, hogy aki ma nem hoz létre egy „üzleti támaszpontot” Magyarországon, az holnap nem tud üzletet csinálni (a várhatóan) a piacgazdaság irányában elmozduló szocialista országokban. Szinte egyöntetű a vélemény, hogy Ausztria eddigi szerepét, azaz a fejlett ipari termékek szállítását a szocialista országokba, nagy valószínűséggel, igen rövid időn belül Magyarország veszi át. Osztrák barátaim sem ellenségeik önmaguknak, ezért ha nem akarnak a szocialista piacról kiszorulni, akkor Magyarország kell leányvállalatot vagy közös vállalatot, de legalább egy kicsi kis képviselést létrehozniuk. Ezt ott Ausztriában minden vállalkozó tudja, mégis vár.

A várakozás oka pedig az, hogy azok a partnerek, akik hajlandók lennének tőkeüket itt nálunk befektetni, egyszerűen nem bíznak a jelenlegi magyar gazdaságban. Az egyik kérdezőm például úgy fogalmazott, szerinte a bizalmatlanság oka az, hogy a jelenlegi kormánypolitika a külföldi beruházó számára kiszámíthatatlan.

Mi tárgyak szívesen fogalmazzunk úgy — lásd az ebben a tárgyban megjelent újságcikkeket —, hogy a pénzügyi kormányzat ismét kitalált valamit, amivel megint kiszűrhet velünk, szerencsétlen magyarokkal. Osztrák barátaim ezzel szemben azt mondták, hogy egyetértenek velem, valóban mindenfajta pénzügyi kiszűrés nagyon kellemetlen mindenki számára, de ennél sokkal nagyobb baj az, hogy az országunk rossz híre támadt, ti. egy olyan gazdaság, amely véletlenszerű gazdasági intézkedéseivel permanens bizonytalanságot tartja az állampolgárait, sokszorosan megbízhatatlannak tűnik a külföldi beruházók szemében. Az a néhány száz vagy talán néhány ezer potenciális nyugati partner, aki ma a magyarországi beruházások iránt érdeklődik, egyszerűen visszazretten attól, hogy a pénzét egy olyan országban fektesse be, ahol — idézek egy bécsi beszélgetőpartneremtől — „egy láthatóan ellenőrizhetetlen pénzügyi helyzetbe van a kormány, és azt arra félétekeri, amerre pillanattal érdeke dikálta”. (Ismétlem, beszélgetőpartnerem idefogalmazta így, pedig akkor még nem is lehetett tudni, hogy ugyanezek a pénzügyi „szakemberek” újabb gálán nagyvonalúsággal a magyar turisták részére, rendez évi szá-

badságunknak a nyugati világban való eltöltéséhez majdnem négy ébédnyi valutát utalnak ki évente. Nem hiszem, hogy tévedek, de szerintem még érvényben van az a másik zseniális rendelkezés is, hogy a kiutazó turista három napra való hideg élelmet vihet magával; még az a szerencse, hogy ennek ellenőrzésére az amúgy is túlterhelt vámokok már egyszerűen nem képesek.)

Még egyszerűbben fogalmazva, a nyugati beruházók a jelenlegi helyzetet gazdaságilag oly mértékben instabillnak tartják, és emiatt a beruházási kockázatot olyan nagyoknak, hogy egyszerűen megfogadják, érdemes-e nekik idehozni a pénzüket, amit majd a pénzügyi hatóságok egy könnyed „úri” gesztussal, a költségvetés túllöntni nagy döntéje miatt, például befagyasztanak. Azt ugyanis hiába mondja az ember, hogy a magyar kormány a külföldieknek az ilyen intézkedések ellen garanciát nyújt. Ezt a szerencsétlen külföldi nem érti, ugyanis náluk a garanciát elsősorban a saját állampolgáraiknak adja a kormány, vagyis őket, miután a saját hazájukról van szó, minden külföldinél több jog és garancia illeti meg! Ezzel az (eléggé el nem félhető) észjárással aztán nyilvánvalóan úgy vélik, hogy ha a magyar hatóságok a magyar állampolgárokkal kénykű-kedvűk szerint bánhatnak, akkor szinte el sem tudják képzelni: mi vár majd azokra a szegény külföldiekre, ha egyszer az államháztartás megint pénzügyi zavarba kerül. Még rá gondolni is rossz!

Osztrák barátaim azt is elmondták, hogy szerintük nálunk — ami az üzletet illeti — egy kétarcú kormány van, amit mégképp nem értenek. Ok úgy látják, hogy van egy rendkívül progresszív kereskedelmi politika, amellyel bárki szívesen dolgozik együtt. Vannak pozitív kereskedelmi törekvések, amelyeket a pénzügyi kormányzat rendre „überele” és rendkívül gyorsan megfűsít. Azt tartják, hogy a pillanatnyi helyzet mindig a pillanatnyi érvényesnyoktól függ; még azt is feltételezték, hogy talán a Minisztertanács a béke megőrzése érdekében, hol az egyik, hol a másik lobby javaslatát fogadja el, aminek ez a bizalomvesztés az eredménye.

A legérdekesebb számomra talán az volt, amikor rajtam kérték számon, hogy a kormány miért nem a saját költségeinek a radikális csökkentésével kezdi a hiányok eltüntetését, miért a lakosság zsebében söröl? Arról nem is beszélve, hogy a külföldi beruházók pontosan ismerik a központi devizagazdálkodás valamennyi buktatóját, és attól is félnek, hogy az általuk létrehozott vegyesvállalat egy váratlan rendelkezés következményeként úgy fog működni, hogy ők beleteszik a tőkejüket, ami majd becsorog az Állam bácsi nagy zsebébe, nekik pedig meglesz majd a joguk ahhoz, hogy alázatos kérelmükre a saját pénzükből néhány morzsát visszakapjanak.

A mi dolgunk, hogy sokszor meggyőződésünk ellenére is megpróbáljuk kompenzálni az állami bizalomvesztő intézkedéseket, mert ha ezt nem tesszük, akkor valószínűleg nem a mindenki által óhajtott európai házhoz, hanem sokkal inkább az európai szegényházhoz tudunk majd csatlakozni.



PROLOGIA

Tények, szabályok — megoldás

Napjainkban a programozás egyik legizgalmasabb kalandja a Prolog. Előzményei az 1970-es évek elejére nyúlnak vissza, amikor is a kutatás arra irányult, hogy a programozási munka nagyobb részét a számítógép végezze el. A Prolog ennek a sokéves kutatómunkának az eredménye. Az első hivatalos Prolog-verziót a marseille-i egyetemen fejlesztették ki Alain Colmerauer vezetésével az 1970-es évek elején. Ők adták a nevet is (PROgramming in LOGic). Ez sokkal eredményesebb és hatékonyabb lett, mint az addig ismert legtöbb programozási nyelv, köztük a Pascal és a BASIC. Például egy adott alkalmazás Prologban általában tízszer kevesebb programsort igényel, mint Pascalban. Olyan nyelv gondolatai körvonalazódtak, amely szabályok és tények alapján dolgozik, látszólag önállóan oldja meg a problémát. Ma a Prolog talán a legfontosabb eszköze a mesterséges intelligenciát alkalmazó programozásnak és a szakértői rendszereknek.

1983-ban Japánban hirdették meg azt az ambiciózus projektet, amely az ötödik generációs számítógép tervezését és kivitelezését tartalmazza, és amely számára a Prolog az alaprendszerrel (kiegészítve a hordozó berendezés assembly nyelvének használatával). Ma még nincs ötödik generációs gép, ezért a jelenlegi Prolog-implimentációk Neumann-elvű gépeken működnek. E sajátos helyzet miatt (nincs párhuzamos feldolgozás, viszonylag lassú, soros memóriák stb.) ezek az implementációk az eredeti elképzelésekhez képest lassúak, hiányosak, nem mindenben követik a Prolog filozófiáját.

Rutinból nem lehet

A Turbo-Prolog rendszer valószínűleg az IBM PC és vele kompatibilis gépeken futó legnépszerűbb Prolog-implimentáció. A keretrendszer a Borland szoftverház stílusjegyeit viseli, annak szokásos kényelmével. A fordító képes a memóriába és a lemezre is fordítani. A lefordított program futása elég gyors, hivatások felhasználók igényeit is képes kielégíteni. Kiválóak a program belsővéit segítő TRACE funkciók, a különböző rendszerablakok (EDITOR, TRACE, MESSAGE, DIALOG), a színek, felhasználói ablakok, a fájlkezelés stb.

A Turbo-Prolog program kezeléséről nem ejtünk szót, mert nem érdemes. Az olvasó szánjon rá egy kis időt, hogy a funkciókat könyvedén „elővarázsolja”, és ne ez okozzon problémákat.

Ha valaki Prologban szándékozik programozni, ajánlom, hogy ne megszokásból, kizárólag a jól bevált sémák szerint gondolkodjon, hanem a tiszta logika vezérelje a problémák megoldásában. Ez a nyelv sokkal inkább épít a kreatív gondolkodásra, mint a hagyományos nyelvek. Prologban rutinból programozni nehezen képzelhető el.

Az olvasóknak már találkozhattak e lap 1987/1. számának hasábjain a Prolog nyelv ismertetésével, Kilián Imre írásában. Elolvasását ajánlom mindenkinek, mivel az alapok megértésében, a fogalmak tisztázásában jó segítség.

Deduktív következtetéssel

A jól ismert Pascal, a BASIC és a többi hagyományos programozási nyelv procedurális: a programozónak kell lépésről lépésre leírnia az eljárásokat, amelyek megmondják a számítógépnek, hogyan kell a problémát megoldani.

Ezzel szemben a Prolog deklaratív nyelv, ami azt jelenti, hogy megadva a szükséges tényeket és szabályokat, képes deduktív következtetésekkel megoldani a programozott problémákat. A programozónak csak a probléma leírását (goal=cél) és a megoldáshoz vezető alapszabályokat kell felállítani, a Prolog rendszer fogja a megoldáshoz vezető utat meghatározni. A Prolog program tehát csak a problémát adja meg a számítógépnek, leírva azt a mikrokörnyezetet vagy axiómarendszert, ahol a tények és szabályok érvényesek, majd egy jól célzott kérdéssel megkéri a rendszert, ugyan keresse már meg a probléma (összes) lehetséges megoldását.

Ha tehát egyszer a programozó leírta, hogy MIT kell megoldani, a Prolog rendszer maga szervezi meg, hogy HOGYAN kell elérni az eredményt. A benne alkalmazott deklaratív (nem procedurális) megközelítés miatt a Prolog kikényszeríti, hogy a programozó a problémát alaposan elemezze, és azt strukturált módon tudja leírni.

Egy Prolog program tények és szabályok alapján keresi a megoldást. Ehhez ismernie kell a szabályok szerkezetét és a szerkezeti elemek minőségét. Az igazán teljes Prolog rendszerek ezt automatikusan elvégzik, bár így sok hiba rejtve maradhat. A Turbo-Prolog megköveteli az elemi objektumok és a szabályok formai deklarációját, ami azonban nem minden Prolog rendszerben kötelező. Formája:

domains (az elemi objektumok típusdeklarációja)

elemi_objektum_neve=már létező_típus
Ez a rész kötelező. Ha a szabályok leírásánál (lásd majd a predicates-nél) csak a rendszer által biztosított alaptípusokkal definiálunk, akkor elhagyható. Ellenkező esetben — és a „beszédés változónevek elve” betartásakor sem — nem kerülhetjük meg. Az elemi objektumok nevei **kisbetűkkel** kezdődnek. A név és egy egyetűsséggel után vagy

a: egy már létező (standard) változó típus nevét vagy

b: egy alaptípusból álló konstanst kell tenni. A standard típusok a következők:

Integer, real, char, symbol, string.
Példákat majd a programokban mutatunk.

A meglévő típusokkal újabb deklarációk építhetők fel. A következő szekcióban azokat a szabályokat kell bevezetnünk, amelyeket alkalmazni fogunk. Ezzel rögzítjük az alkalmazott szabályok neveit és a használható paraméterek típusait. Formalizmusa:

predicates (a szabályok a paraméterezés rögzítésével)

szabály(paraméterek)

Legálább egy ilyen szabálynak kell lennie, máskülönben nem is beszélhetünk Prolog programról. Néhány példa:
szabaly1(symbol)
szabaly2(integer,integer,symbol)
masik_szabaly(real,symbol)
uj_szabaly(adat,symbol,adat)

Cél=goal

A deklaratív foglalkozók után most a program célját leíró szekció következik. Formája:



goal (a program addig fut, amíg ez a cél nem teljesül, vagy meg nem állapítja, hogy sosem teljesülhet)

célkifejezés

A Turbo-Prolog 1.1-es verziójánál a célmeghatározás a program végén van. Ez a rész is elmaradhat, de léte vagy nemléte erősen befolyásolhatja a program működését. Ha van, akkor pontosan azt teszi, amire az ezt követő utasítások kényszerítik, és az első kielégítő eset után leáll (hacsak másra nem utasítjuk). Ha nincs, akkor a menüből választott RUN parancsra a DIALOG ablakban megjelenik a GOAL: kiírás, ami után az aktuális cél, más néven a kérdés begépelése következhet. Maga a program tehát csak kérdés megválaszolásáért hajlandó futni!

A Prolog program utolsó része a tények és szabályok felsorolása. Ennek a programrésznek a bevezető szava a **clauses**. A tények és szabályok tetszőleges sorrendben követhetik egymást, az azonban fontos, hogy az azonos nevűek egymást követően álljanak, az azon belüli sorrend pedig a végrehajtás sorrendjét fogja befolyásolni. Ennek az az oka, hogy számítógépeink a szabályokat nem egyszerre, hanem szép sorban egymás után dolgozzák fel. Ha majd olyan gépen fut a Prolog, ahol párhuzamos feldolgozás lesz, akkor a szabályok sorrendjére valószínűleg nem kell ilyen megkötést tenni.

A szabály olyan kifejezés, amely egy vagy

több kisebb célállítást logikai kapcsolatából áll. Deklaratíván megfogalmazva: a szabály bal oldalán a következmény, jobb oldalán pedig az ahhoz szükséges előfeltételek állnak. Procedurálisan gondolkodva a bal oldalán az eljárásfejet értjük, a jobb oldalán pedig az eljárás törzsét találjuk.

A célkifejezés argumentumlistáiban szerepelhet egy vagy több változó, és az azonos nevű változónevek egy szabályon belül mindig ugyanazt a helyettesítési értéket jelentik. Logikai nyelvről lévén szó, a Prolog-terminológiában logikai változókról beszélnek, ami azonban messze nem azonos például a Pascal boolean típusú változójával.

Kis- és nagybetűk

Fontos szintaktikai szabály, hogy a változónevek **nagybetűvel** kezdődnek, míg a konstans szimbólumok és a szabályok egyaránt kisbetűvel. Az aláhúzásjel kisbetűnek számít a megnevezésekben, önállóan pedig egy olyan argumentum helyét jelölheti egy részcel kifejezésében, amelynek értéke a szabály megoldásában nem változik.

Egy paraméterezett eljárásban nem teszünk formai megkülönböztetést a bemenő és a kimenő paraméterek között. Ha egy változónak van már értéke (foglalt), akkor az a szabály értelmezésekor automatikusan bemenő paraméter lesz, és ha nincs értéke, akkor a szabály

csak akkor teljesül, ha ebből a szabályból következően érték veszt fel, tehát így a határozatlan változó kimenő paraméterre válik. A határozott változók egy szabályon belül nem kaphatnak új értéket, azaz ha egy változó bemenő paraméter, akkor az értéke megmarad a szabály kielégítése után is.

A szabály teljesülése feltételektől függ. A feltételek az if kulcsszó vagy a vele ekvivalens — jel után állnak, és több szabály esetén logikai kapcsolatukat is meg kell jelölnünk. Ilyen logikai alapszavak és a velük ekvivalens írásjelek a következők:

and illetve , (vessző)
or illetve ; (pontosvessző)
not

Néhány Prolog-implementáció csak az egyik formát fogadja el, mások egy szabályon belül csak egyfajta jelölést engednek meg, a Turbo-Prolog viszont teljes szabadságot ad a program szerkesztőinek e kérdésben. Stilisztikai szempontból azonban jó, ha egységes jelölérendszerrel maradunk. Én például a kulcsszavas írásmóddhoz igyekszem ragaszkodni, mert a szavak kifejezőbbek és főleg kevésbé téveszthetőek, mint az írásjelek. Természetesen a szabályok végén minden esetben ott kell lennie a szabályt lezáró pontnak.

A tényállítás konstans argumentumú, feltétel nélküli szabály, a szabály nevével és az argumentumlistával, a végén egy pont.

Makány György

A REALNET és a TVCNET hálózatok

Eredetileg a REALNET a Commodore gépek soros buszáinak bővítésére készült. Minthogy az iskolai alkalmazásban is igen hasznosnak bizonyult, a fejlesztők célszerűnek látták a — kezdetben csakis hardveresen működő — eszköz továbbfejlesztését oktatóhálózatá. Ez némi hardveres változtatáson túl egy hálózati szoftver kifejlesztését is megkívánta.

A fejlesztés során messzemenően figyelembe vették az iskolai igényeket, lehetőségeket, sőt, még arra is volt gondjuk, hogy a Videoton TV-Computerre fejlesztendő — azóta már kifejlesztett — hálózat szoftverével a lehető legnagyobb mértékben egyezzen. Pedig az ilyen fokú kiszolgálástól, a kompatibilitás követelményétől az iskolákban már-már lemondunk.

A REALNET hardver-üzemmódja

A REALNET bővítőre — a változatnak megfelelően — legfeljebb 6, illetve 16 soros busszal rendelkező Commodore vagy ilyen csatlóval ellátott más típusú számítógép kapcsolható. A csatlakoztatás a mellékelt kábelekkel történik. A hálózat üzembe helyezése szakértelmet, szerszámokat nem igényel, a hibás összekapcsolásokat az alkalmazott csatlakozók csaknem teljesen kizárják. A mellékelt dokumentáció az üzembe helyezéshez és a használathoz egyaránt elegendő. A hálózatra — az egyedi gépeknél megszokott módon — lehet a perifériákat csatlakoztatni. Ezek használata is a megszokott módon történhet, bár a rendszer néhány tulajdonsága nyilvánvalóan megváltozik.



A hardveres üzem módhoz — a furcaságok megértéséhez — érdemes néhány dolgot tudni, átgondolni.

1. A gépek kiszolgálása (a perifériákhoz fordulás lekérdezése) ciklikusan történik. A gépek sorrendjét a csatlakozás helye határozza meg.

2. Ha a soros vonal foglalt, akkor a (többi) gép várakozik a kiszolgálásra. Bár a megoldás néhány esetben túlságosan hosszadalmasnak tűnik, a felhasználó számára ez látszik kényelmesebbnek.

3. A gép és a periféria operációs rendszere között működik a hálózat, így a periféria hibabüszete is csak a kiszolgálás során jön létre; ez persze egy kiszolgálásnak minősül.

4. Az átlagos várakozási idő a gépek számával nő, így — a soros vonalat gyakran használó programok esetén — a kiszolgált gépek száma és a várakozási idő között kompromisszumot kell kötni.

5. Lehetőség van arra is, hogy valamelyik gép helyére újabb REALNET bővítőt, ehhez pedig további gépeket kapcsoljunk. A gyártó erre az esetre nem garantálja a működést, de a tapasztalatok így is kedvezőek. Nyilvánvaló azonban, hogy két REALNET—6 összekapcsolása helyett célszerűbb egyetlen REALNET—16 alkalmazása, ez utóbbitól kétfoldó viszont a várakozási idők miatt nem érdemes összekapcsolni.

6. Természetesen perifériás egység(ek)et tartalmazó konfigurációk is csatlakoztathatók a hálózatra, ilyenkor azonban az egy-egy géppel elérhető perifériák egység számainak különbözniük kell. A közös perifériákat minden gép használhatja, de egymás önálló perifériáit nem érthetik el a gépek.

Ebben az üzemmódban számos felhasználói program futtatható, a perifériák használata a szokásos módon történhet. Azt persze figyelembe kell venni, hogy a perifériákhoz több gép is hozzáférhet.

Szoftver-üzemmód

A hálózati szoftver mintegy „rásimul” a BASIC rendszerre. Az utasítások formalizmusa illeszkedik a BASIC-hez, általában parancs- és programmódban egyaránt kiadhatók. A BASIC program és a változók kiküldésén és bekérésén kívül módot ad tárterület és képernyő áttöltésére, valamint üzenet küldésére. Lehetőse van a rendszer állapotának lekérdezésére és átállítására is, sőt, a tanári géppel még a billentyűzetet is „el lehet venni” a tanulói gépektől. Mindezzellett lehetőség van arra is, hogy a hálózatra kapcsolt gépek vegyesen működjenek hardver-, illetve szoftver-üzemmódban.

Meglepetések persze itt is adódhatnak, ám ezek a sajtóságok is könnyen érthetőek, ha a funkcióikat kicsit elemezzük.

— Amíg kivételnél van értelme több gép egyidejű feltöltésének, addig beolvasásnál ez értelmetlen.

— A vegyesen alkalmazott géptípusok esetén a tárterület mozgataa komplikációkat okozhat a különböző tárfelosztás miatt. Ilyen gond adódhat például a Plus/4 és a nem bővített C16 típusú gépek esetén.

— A már működő rendszerbe való belépést csak egy lekérdező ciklus után nyugtázza a tanári gép. Ez viszont nagyon elhúzódhat egy-egy adatmozgatás miatt.

— A tanulói gépek számának bekérése (itt a rendszer eléggé védtelen!) nem látszik elégságnak, mégis ez tűnik kényelmesebbnek: a logikai számok ügyes kiosztása egyszerűbb kiszolgálást tesz lehetővé (például ciklusban).

Tapasztalatok

Az elektronika megbízhatóan működik, csak igen kevés meghibásodásról hallani. Sok esetben egy-egy perifériahibát vélnék hálózati hibának.

Néha előfordul, hogy a periféria „nem szólal meg”. Ilyenkor ál-

talában segít, ha a bővítőt, illetve a perifériát kikapcsoljuk, majd ismét bekapcsoljuk. Ez a jelenség akkor tapasztalható, ha a hálózatra kikapcsolt gép is csatlakozik.

Bár a bővítő elektronikája nem mondható kényesnek, a csatlakozások szabályait érdemes betartani a gépek és a perifériák védelmében is. Néhány ellenkező értelmű tapasztalat alapján ne adjuk fel óvatosságunkat; azokat tekintsük a szerencse ajándékainak!

Egy-egy felhasználói program futtatása során előfordul, hogy például a lemezegység használata közben lemeredveik a rendszer. Ennek oka leggyakrabban az alacsony hálózati feszültségben vagy a hálózati zavarokban (feszültségingadozás, kapcsolási tranzienek) keresendő. A lemeredés a REALNET ki- és bekapcsolásával rendszerint megszűnik, mindenféle adatvesztés nélkül.

A hálózati szoftver rendkívül megbízhatóan működik. BASIC programok futtatása esetén kimonodottan kényelmes szolgáltatásokat nyújt; azzal azonban számolni kell, hogy az adatmozgatási műveletek egymást követően hajtódnak végre, és időbe telik, míg a tanári gép kezdeményezését a tanulói gép érzékeli, nyugtázza.

Vegyük figyelembe, hogy bármilyen gyors is, az átvitelnek látható jelei vannak a tanulói képernyőn. A tanári gép beavatkozási tehát nem lehetnek „titkosak”.

Ha a hálózatot szoftveres üzemmódban használjuk, de hardveres üzemmódot gépet is kiszolgál a rendszer, akkor a tanári gép és egy hardver-üzemmódot hálózatra csatlakozó gép egyidejűleg nem használhatja a soros vonalat. Erre fokozottan kell ügyelni.

A TVCNET felépítése és működése

A hálózat a gépek magnetofon-csatlakozóit használja, és csak szoftveres üzemmódban működik. A tanári géptől indulva sorosan (láncba) felfűzött gépeket speciális kábelek kötik össze, a tanári gépre csatlakoznak a perifériák, valamint a rendszer tartozékaként a hálózatvezérlő. A kábelek hibás csatlakoztatása esetén a hálózat nem működik; bizonyos funkciók nem használhatók.

A gyártó szállít elágazásos hálózatot is külön megrendelés — a hálózat szerkezetének és a kábelhosszak megadásánál — alapján. A típusban szereplő szám a hálózatra kapcsolható tanulói gépek számának felső határa.

A hálózati szoftver a tanári gépbe töltődik be először a lemezzel, és azonnal elindulva a tanulói gépekbe tölti a megfelelő programot. Ez is öndiindul, és a gép számának bekérése után lehetővé teszi a kommunikációt a tanári géppel.

Különleges szolgáltatás a katalógus oldalankénti listázása, ami — a periféria használat engedélyezése után — tanulói gépen is kérhető.

A hálózati utasítások a REALNET hálózatéhoz hasonlóak; az eltéréseket a gép sajátosságaihoz való alkalmazkodás indokolja.

A friss fejlesztés miatt még aligha lehet tapasztalatokról számot adni, de annyi már kiderült, hogy a magnót és lemezegységet egyaránt használó programok esetén (könnyen érthető okok miatt) hiba keletkezhet. Ennek elhárítása a kritikus rész átírásával történhet. Az eddigi tesztelt programokkal egyéb nehézségek nem adódtak.

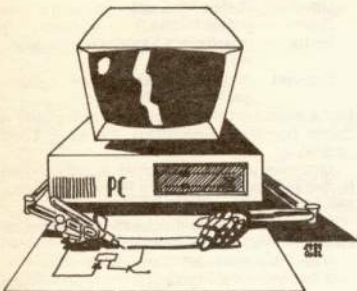
Mint ahogy a hálózati szoftver és a gép rendszere együtt működik, szokatlan eredmények adódhatnak. Ezek egy részét a gyártó már ismeri és vizsgálja; ígéri, hogy a készülő újabb verziókban az ilyen észrevételeket figyelembe veszi. Egyúttal kéri, hogy a felhasználók hozzák tudomására tapasztalataikat, és közöljék kívánságaikat.

Összegezve: a hálózat egyszerű telepíthetősége, a kényelmes szoftver és a két hálózat egyszerű használata kedvező előjel az iskolák számára.

Theisz György



Rajzoljunk számítógéppel!



A Mikromagazin 1989/10. számában örömmel fedeztem fel az áramköri kapcsolási rajzokat készítő program, az OrCAD/SDT bemutatásával foglalkozó cikket. Mint a Landler Jenő Műszaki Középiskola információj- és számítástechnika ágazatának 4. évfolyamos tanulója — és így a jövőben várhatóan betöltendő technikai munkakör várományosa — a nyári szakmai gyakorlaton jóformán dokumentáció és se-

gíttség nélkül, de mondhatni: izzadságos munkával már megismertem a program használatát — ha nem is teljesen, de sok mindenre kiterjedően.

A lapban megjelent leírás az adott terjedelmi keretek között jó, de kezdőknek a program megismerését és kezelésének elsajátítását — véleményem szerint — csak nagy nehézségek árán teszi lehetővé.

Azért, hogy további segítséget nyújtsak elsősorban azoknak, akik a számítógépes kapcsolási rajzok szerkesztésével csak ezután kívánnak megismerkedni, megpróbáltam egy olyan összeállítást készíteni, amely lépésről lépésre vezet be az OrCAD rejtelmeibe. Nemcsak úgy, hogy lexikonszerűen a parancsok és alparancsok definícióját és funkcióját adom meg, hanem úgy, hogy a cikket olvasók már az első lépésben el tudjanak készíteni egy egyszerű rajzot.

Irársóban természetesen nem lehetek szigorúan kiválasztott célra orientált (ez egy négybit-es gyűjtős számláló kapcsolási rajza), az egyes fő- és alparancsmenük ismertetésénél néha szükségesnek ítélem olyan dolgokra is kitérni, amelyek a feladat elvégzéséhez nem voltak közvetlenül szükségesek.

A rövid bevezetés után lássunk munkához! A jó hatások elérése céljából a tanulást — pontosabban: a kísérletezgetést — célszerű gép mellett végezni. Feltételezem, hogy az OrCAD rendszer könyvtárai és a fájlok a mervelemeken a megfelelő szervezésben rajta vannak.

Ha minden rendben, indítsuk el a DRAFT.EXE programot. A képernyőn ilyenkor jelentkező ecből az azt is láthatjuk, hogy a könyvtári fájlok betöltődnek. A betöltés után bejelentkezik az OrCAD, majd rajzunk fájlnevét kérdezi.

Ha név beírása nélkül nyomjuk le az ENTER billentyűt az OrCAD Unnamed Worksheet (névtelen munkalap) üzenetet ad. Ha új fájlnevet adunk meg, ennek kiterjesztése célszerűen SCH (schematic=vázlat) legyen. Így a SHEET alkönyvtárban, ahová az OrCAD elhelyezi szerkesztett rajzainkat, könnyebben tájékozódhatunk. Ha már létező fájl nevet adjuk meg, a rajz betöltődik és a képernyőn megjelenik.

Akár adunk meg fájlnevet, akár nem, a következő ernyőkép az üres vagy a betöltött lapész — pontosabban ennek bal felső része az OrCAD kurzorral. A munkalap alapmérete 11×8,5 inch, azaz kb. 28×21,6 cm („A” méret). Ha a méret nem felel meg, további öt fokozatban (B...F) növelhető a főparancsmenü SET elemének, majd a SET alparancsmenü WORKSHEET SIZE elemének meghívásával. Jelenleg ezt a lehetőséget nem kell használnunk, tervezendő rajzunk az „A” lapméreten is kényelmesen elhelyezhető lesz.

Ha most lenyomjuk az ENTER billentyűt, megjelenik a főparancsmenü. Ebből akár a pa-

rancs kezdőbetűjének megadásával, akár a kurzorbillentyűkkel és az ENTER-rel fővaló menüvezérlési módszerrel is választhatunk.

Mielőtt a részletekbe belemennénk, már most célszerű megemlíteni, hogyan is kell előírászerűen kilépni a programból. Ezzel megkímélhetjük az olvasót attól, hogy a programba „belekeveredve”, abból Alt—Ctrl—Del-lel lépjen ki. A kilépéshez válasszuk a főmenüből a QUIT parancsot, majd a QUIT menüből az ABANDON EDITS (felhagy a szerkesztéssel) alparancsot! Ennek eredménye a DOS-ba való visszatérés lesz. További praktikus tudnivaló, hogy a program futásának majd minden fázisában — a perifériaműveleteket kivéve — az ESC billentyű lenyomásával „visszafelel” lépkedhetünk.

Válasszuk elsők a főparancsmenüből a LIBRARY-! Ennek a választásnak az a célja, hogy megismerkedjünk a rendelkezésre álló alkatrészkezeléssel. A LIBRARY főparancs menüjének két alparancsa van: a DIRECTORY (címtár) és a BROWSE (böngészés).

A DIRECTORY-t választva az alkatrész-könyvtárak menüje jelenik meg a képernyőn a következő formában:

```
TTL.LIB
MEMORY.LIB
CONTR.LIB
```

stb.

(Természetesen attól függően, hogy a könyvtár mit tartalmaz.)

Az ebből való választás után (kurzormozgató+ENTER), a program a következő lehetőséget kínálja:

```
SCREEN
PRINTER
FILE
```

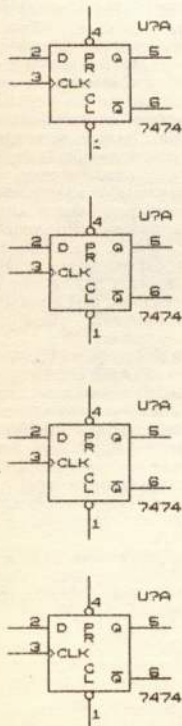
Az első két lehetőség a kiválasztott könyvtári fájl elemeinek azonosítóit (név, típusszám stb.) ki listázza a képernyőre, illetve a nyomtatóra. A képernyőre írás a DOS képernyőkezelésének megfelelő 80×25-ös formában történik. A FILE választásával az OrCAD megkérdezi az új fájl nevét, és kiírja a könyvtári listát a megadott meghajtónak és útnalnak megfelelő módon; külön előírás nélkül a default helyre — ez azonos a DRAFT.EXE fájl tartalmazza directoryval.

A BROWSE alparancsral lapozgathatunk az alkatrész-katalógusban; ekkor mind az azonosító, mind az alkatrész rajzi leve megjelenik a képernyőn. A BROWSE is két lehetőséget kínál:

```
ALL PARTS
SPECIFIC PARTS
```

Az ALL PARTS először kiírja a LIBRARY fájljok neveit (a DIRECTORY alparancsnál elmondottakhoz hasonlóan módon), majd a képernyőn megjeleníti a kiválasztott alkatrészfájl első elemét. Ekkor lehetőség van egy újabb menü:

1. ábra





FORWARD
BACKWARD
QUIT

segítségével az előre és visszafelé történő lapozásra, illetve a BROWSE elhagyására (QUIT). Ez utóbbi esetben a visszatérés a főparancsmenü „főlé” történik, azaz a menü az ENTER-rel hívható a képernyőre. Ez egyébként minden olyan esetben így van, amikor egy visszatérés — QUIT vagy ESC — után az ernyőn csak a rajz látható.

A SPECIFIC PARTS esetén a Part? kérdésre adott válasz után a könyvtárnak csak az általunk megjelölt elemét jeleníti meg az OrCAD, majd ezután újabb Part? kérdést tesz fel.

Az effektív rajzolási munkát a GET főparancsral célszerű kezdeni. Ezzel lehet ugyanis az alkatrészeket a könyvtárból elővenni és a munkalapra letenni. A GET választása után az OrCAD Get? üzenettel válaszra vár. Ekkor két lehetőség között választhatunk.

Ha nem tudjuk a keresett alkatrészt könyvtári helyét vagy azonosítóját, egyszerűen csak ENTER-rel válaszolunk a kérdésre. Ekkor megjelenik a képernyőn a .LIB fájlok listája. (Ezt a LIBRARY-DIRECTORY parancskombinációból már ismerjük.) A kívánt könyvtár kiválasztása után a könyvtár első tíz elemének azonosító-

ját láthatjuk a képernyőn; ha az elemek száma 10-nél kevesebb, természetesen csak a létezőket. Ha a kívánt alkatrészt nem találjuk, a kurzorvezérlő billentyűkkel folytathatjuk a listázást. A választás a listából kívánt elemre állás után ENTER-rel történik. Ha a típusazonosítóhoz több konkrét típus tartozik — ez példánkban így is lesz —, ezek listáját kapjuk a képernyőre.

Ha a keresett alkatrészt név (azonosítóját, pontos típusszámát) ismerjük, ezt a Get? kérdésre adjuk meg. Ezzel a módszerrel a keresésre fordított idő lényegesen rövidíthető.

Ha választottunk, az OrCAD az alkatrészt keretbe foglalva megjeleníti a képernyőn (1. ábra). Az alkatrészpozíció most a kurzorvezérlő billentyűkkel változtatható. Erre a szerkesztés későbbi fázisaiban is — bár bonyolultabb módon — lehetőség van.

Az alkatrészt képernyőre hozásakor az ernyő tetején (a prompt sorában) egy menü jelenik meg, az alkatrészek típusától függően a következő elemekből összeállítva:

| | |
|--------|---------------------------------------|
| Place | (elhelyezés) |
| Rotate | (forgatás balra 90 fokkal) |
| Normal | (az eredeti helyzet visszaállítására) |

| | |
|---------|---|
| Up | (felfelé fordítás) |
| Down | (lefelé fordítás) |
| Mirror | (tükörfés a függőleges tengelyre) |
| Convert | (logikai áramköröknél a DeMorgan-megfelelő) |

Bár a menüben nem szerepel, de hatásos a Zoom (kicsinyítés-nagyítás) és az Escape funkció is.

Mozgassuk az alkatrészt az általunk kívánt helyre és helyzetbe, majd a Place paranccsal helyezzük el a munkalapra.

Kitűzött feladatunk megkezdéséhez végezzük el most a következőket:

a GET kijelölése a főparancsmenüből a Get?-re a 74 válasz
Ekkor a konkrét alkatrésztlista jelenik meg a képernyőn:

74LS74
74S74
74ALS74

7474

Ebből válasszuk a 7474 típust! Mozgassuk az elemet a megfelelő helyre, ehhez hívjuk segítségül a kész rajzot tartalmazó 2. ábrát, majd a Place funkcióval helyezzük le!

Az eddig elvégzetteket ismételjük meg még háromszor a 7474-gyel! Egyelőre nem kell azzal foglalkozni, hogy a lábak számozása nem megfelelő. Ez egyébként annak a következménye, hogy 4 darab, egyenként két flip-flopot tartalmazó tokot választottunk, és az OrCAD mindig a tok első („A” jelű) tárolóját hívta elő. Ezt a hibát a későbbiekben fogjuk korrigálni.

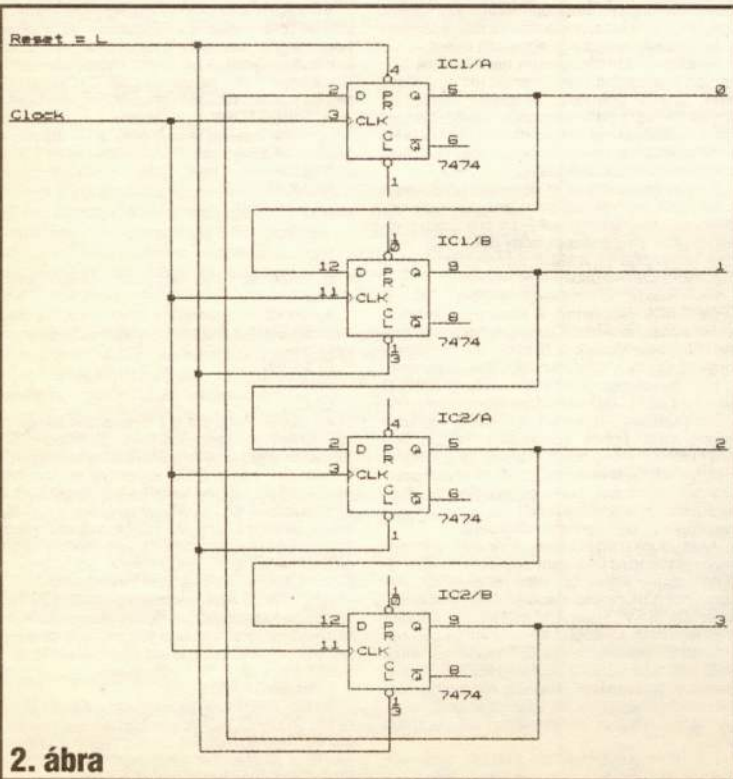
Hogy rajzunk megfelelő alkatrészt-elrendezéséről meggyőződjünk, ha nem vagyunk benne biztosak, a GET parancsmenüből a Zoomot választva kicsinyíthetjük rajzunkat olyan mértékben, hogy a munkalap teljes egésze látható legyen a képernyőn. Kicsinyítéskor a lábszámok eltűnnek, és a feliratok vastag vonalaként jelennek meg a rajzon. Ha ez utóbbiak nem zavaróak, az egész elhelyezési folyamat elvégezhetjük a „zoomolt” munkalapon is. A Zoom egyébként nemcsak a GET menüjéből, hanem a program más helyeiről — például a főparancsmenüből — is hívható.

A rajzi feladatunk elvégzéséhez szükséges következő főparancs a PLACE. Ezzel a szóval, mint a GET egyik alparancsával már dolgoztunk, de a PLACE funkciója most más lesz. Válasszuk ki a főmenüből, majd az almenüből térjünk rá a WIRE (vezeték) alparancsra. Ennek segítségével fogjuk az eddigi munkával elhelyezett elemek között az áramköri összeköttetéseket létrehozni. A későbbiekben felhasználjuk még a JUNCTION (csomópont) és a LABEL (címké) alparancsot is.

A WIRE kijelölése után a képernyő felső sorában egy újabb parancsmenü jelenik meg. Az egyes menüelemek és magyarázataik (csak azoké, amelyek lényegesek):

| | |
|--------|------------------------|
| Begin | (kezdőpont) |
| Find | |
| Jump | |
| Zoom | (kicsinyítés-nagyítás) |
| Escape | (ESC) |

Egy húzál „lerakásához” vigyük a kurzort a vezeték tervezett kezdőpontjához, majd válas-



2. ábra



szuk a Begin. Ekkor a menü a következő módon változik:

- Begin
- End
- New (új vezeték)
- Find
- Jump
- Zoom
- Escape

Ezután a kurzor mozgásával „húzhatjuk” a vezetéket. A végpont elérésekor két lehetőségünk van. Az End befejezi a vezetékezés folyamatát és az OrCAD a főmenühöz tér vissza. A New hatására a vezeték végpontja szintén a kurzor pillanatnyi pozíciója lesz, de a menüben maradvra a Beginnel új vezeték megrajzolását kezdhetsz el. Több vezeték megrajzolásához tehát a következő sorozatot célszerű alkalmazni:

B(begin) N(new) B N B N B ... E(end)

Az OrCAD sajátossága, hogy a vezetékek-összekötéseket egy ponttal (Junction) meg is kell jelölni; egymást keresztező vezetékek találkozásai pontja tehát nem jelent fizikai összekapcsolást is.

Váltsunk tehát a PLACE menüből a JUNCTION alparancsot! Vigyük a kurzort az összekötés helyére és a Place-szel helyezzük le az összekapcsolást jelentő pontot! Az összes pont lerakása után rajzunk szerkezetileg tulajdonképpen már készen is van. Ezt az állapotot szemlélteti a 3. ábra.

Hátra van még az alkatrészek melletti és egyéb feliratok elkészítése, illetve a meglévők kívánság szerinti módosítása. Most fogjuk kijavítani a már említett helytelen lábszámozást is. A rajz készítésének ezt a fázisát célszerű az „1” (1:1) méretarányban elvégezni; ezt a Zoom funkcióval érhetjük el.

A munkához válasszuk az EDIT főparancsot! Ekkor a képernyő felső sorában az Edit Find Jump Zoom Escape menü jelenik meg. Vigyük a kurzort a helytelen lábszámozással ellátott flip-flop „belsejébe”, majd hívjuk az Editet. Ekkor újabb — Edit parts — menühöz jutunk:

- REFERENCE
- PART VALUE
- ORIENTATION
- WHICH DEVICE

Váltsunk a WHICH DEVICE parancsot! Most lehetőség nyílik a token belüli egység számának (1 vagy 2) megadására. A kérdésre válaszoljunk 2-vel! Ismételjük meg ugyanezt az eljárást a rajzon legalul lévő flip-floppal is! Az OrCAD elvégzi a lábak átszámozását.

Ha nem felel meg az eddig a rajzon látható U... pozíciójelölés, ezt is átírhatjuk. Ehhez az Edit alparancs menüjéből hívjuk a REFERENCE elemet, majd a NAME-et! A képernyő felső sorában megjelenő Reference? kérdésre az előző törlése után új nevet írhatunk be válaszként. Legyen ez IC1/. Ismételjük meg ezt a többi flip-floppal is sorban, IC1/, IC2/, IC2/ válaszokat adva. Az új pozíciójelölések rendre IC1/A, IC1/B, IC2/A és IC2/B lesznek.

Hátra van még a bemeneti és kimeneti pontok feliratokkal való megjelölése. Ehhez a PLACE főparancs menüjéből a LABEL-t választjuk!

A Label? kérdésre adjuk meg a kívánt szöveget. Ez legyen most a törlő bemenet megjelölése: Reset=L! A szöveg beírása után nyomjunk ENTER-t, és a megjelenő menüből válasszuk a Commentet. Ekkor a képernyő felső sorában a

- Place
- Orientation
- Value
- Type
- Find
- Jump

menü jelenik meg. Mozgassuk a feliratot a kívánt helyre (például a 3. ábrán látható elhelyezés szerint), és a Place funkcióval tegyük le! Az újabb Label? kérdésre válaszoljunk „Clock”-ot, majd az előbbiekhöz hasonlóan ezt is helyezzük el a munkalapon!

Hasonló módon láthatjuk el feliratokkal a kimeneteket is (0, 1, 2, 3). Ha ezzel készen vagyunk, ESC-pel lépünk ki a címkézési eljárásból! A megfelelő „zoomozással” — ha ezt kívánjuk — rajzunk szerkezetét és tartalmát áttekinthetjük. Ha nem találunk hibát, szerkesztési munkánk gyakorlatilag véget is ért. Ha a rajzon javítani vagy módosítani kívánunk, az eddigiekben elmondottak alapján ezt nem tudjuk el-

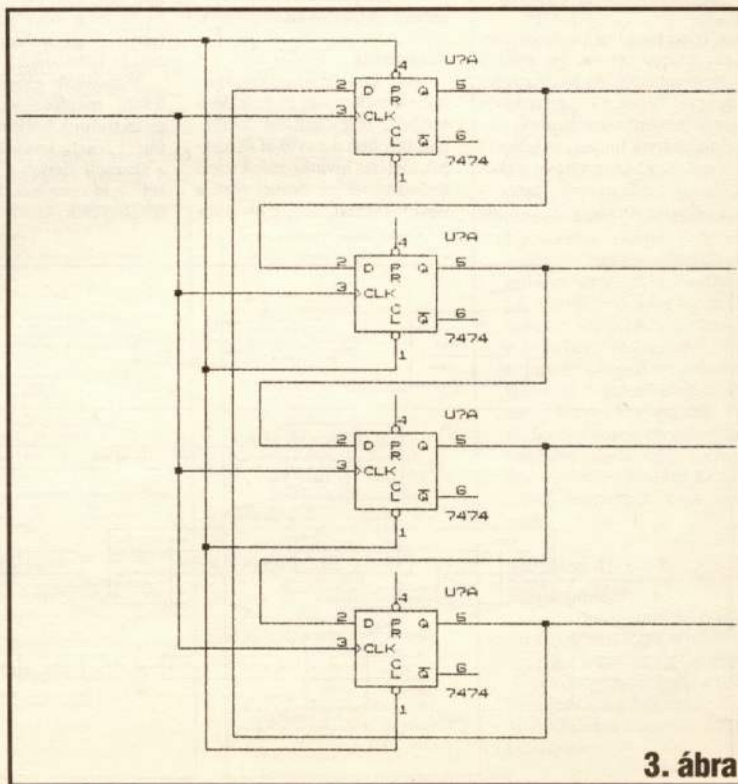
végezni. Ha valakinek van vállalkozókedve, próbálkozzon meg a főmenü BLOCK és DELETE parancsával, ha nincs, készítse el a rajzot még egyszer; ez gyakorlásnak sem rossz!

Ha teljességre törekednénk, rajzunkat címezővel is illesk el látnunk, ezzel a kérdéssel azonban most ne foglalkozunk! Minősítsük rajzunkat késznek, és mentjük lemezés fájlba, illetve nyomtassuk ki!

A fájlba való kiíráshoz hívjuk a főparancs-menüt, és válasszuk annak QUIT parancsát, majd a megjelenő almenüből a WRITE TO FILE alparancsot! Ha a kérdésre megadjuk a fájlnévet, az OrCAD munkalapunkat a SHEET al-könyvtárba helyezi el az általunk beírt fájlnévvel.

A nyomtatás a főparancs-menü HARDCOPY parancsával végezhető. A HARDCOPY menüből válasszuk a Print Mode elemet. Most lehetőség nyílik a keskeny (Compress) vagy a normál (Scale) nyomtatásra. Válasszuk a Scale-t! A HARDCOPY almenühöz visszajutva módosítsuk a papír szélességét (Width of Paper) 13 inchre! Ismét a HARDCOPY almenühöz jutunk vissza. Ekkor a Hardcopy alparancs-al indíthatjuk a nyomtatást.

Mayer Tamás



3. ábra

FELADATOK

— MEGOLDÁSOK

Sorozatunkat elsősorban középiskolásoknak szánjuk, de reméljük, hogy minden olvasónknak tanulási lehetőséget és szórakozást nyújt.

A feladatok a Nemes Tihamér országos számítástechnikai verseny színvonalának felelnek meg. Minden esetben olyat választunk, amely röviden, gyorsan megoldható, de a megoldáshoz ötletre van szükség. A megoldást mindig a következő számban közöljük.

Mivel változatosságra törekszünk, különböző programozási nyelveket használunk. Az is előfordul, hogy egy feladatra több programnyelven is közlünk megoldást, ezzel is elősegítve az ismeretszerzést.

A szerkesztőség várja az olvasók, a versenyzők leveleit. A legötletesebb program beküldőjét könyvtalvánnyal jutalmazzuk. Ne feledjenek azonban a programhoz leírást is mellékelni!

19. feladat:

Keresztreferencia-készítés

Írjon programot, amely beolvas egy assembly nyelvű programlistát, és keresztreferenciát készít róla, azaz kiírja az összes felhasznált címke definiálásának helyét, és felsorol minden címkére való hivatkozást. Ügyeljen arra, hogy a program gyors legyen, és nagy

számú címke esetén is működjön, a rendelkezésre álló memóriát hatékonyan használja.

Megoldás

Mielőtt a program írásához hozzáfekszünk, el kell döntenünk, hogy milyen formában tároljuk a neveket és a rájuk történő hivatkozások sorát a memóriában. Mivel sem a nevek számát, sem a hivatko-

zások számát nem ismerjük, a memória hatékony kihasználása csak dinamikus tárkezeléssel biztosítható.

A memóriába már felvett szimbólumok között gyakran kell majd keresni — minden új név esetén —, ezért a bináris fa szerkezet alkalmazható a legcélszerűbben.

A fa „ágait” az elemek alkotják. Minden előforduló névnek megfelel egy elem, amely tárolja a nevet, a definiálás sorát, két pointert, amelyek a fa további ágaira mutatnak, és a hivatkozások sorának láncára mutató pointert, amelyek a fa „levelei”. A fa ágaira mutató pointerek közül mindig az ábécé sorrendben név előtt álló, jobb pedig az az után következő elemre mutat. Egy néhány elemből álló fa látható az 1. ábrán.

Vizsgáljuk először a levelektől megfosztott fát! Hogyan tudunk keresni egy ilyen fán? Először hasonlítsuk össze a keresett elemet a fa „gyökerét” adó elemmel. Ha a kettő megegyezik, akkor készen va-

gyunk. Ha a név az ábécérendben korábban áll (kisebb), akkor a fa bal oldali ágán haladunk tovább, egyébként pedig a jobb oldali ágon. Végezzük el az összehasonlítást ezzel az elemmel. Most is bal vagy jobb felé kell folytatnunk a keresést, ha a két név nem azonos, attól függően, hogy melyik áll előbb az ábécében. A keresést mindaddig folytatjuk, amíg a keresett elemet megtaláljuk, vagy valahol NULL-t nem találunk. Az utóbbi eset azt jelzi, hogy a táblázatban a keresett elem nem található meg. Ekkor, ha az elemet fel akarjuk venni a táblázatba, a NULL-nak a helyére kell azt beláncolnunk.

A fa leveleit alkotó hivatkozási sort nem szükséges bináris fa struktúrában tárolni, mivel azok között nem kell keresni.

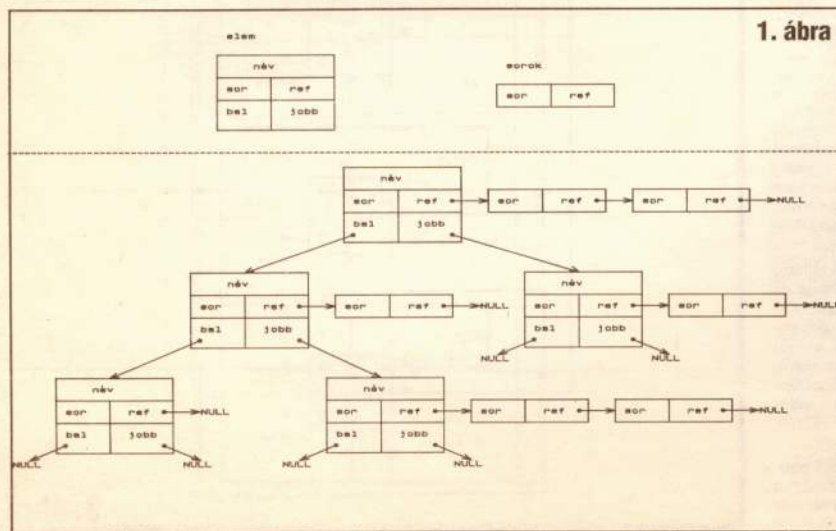
Tekintsük most át a közölt, Turbo-C 2.0-ban IBM PC-re megírt programot! Kezdjük a szokásostól eltérően a főprogrammal!

A főprogram a parancssorban a paraméterként megadott fájlokat próbálja megnyitni először. Ha ez nem sikerül, hibáüzenettel leáll. Ha a fájlok megnyitása sikerült, a bemeneti fájlt soronként beolvassa a sztringbe. A sorszámot sorszámváltozó számolja. A vége változó a fájl legutolsó sorának beolvasásakor vált egyre.

A sor feldolgozásának első lépéseként a beolvasott sorban meg kell keresni az első nevet. Ezt a keres eljárást végzi el. Az eljárás a név kezdetére mutató pointert adja vissza és h-ban a név hosszát. Ha a sorban nincs név, akkor NULL-t ad vissza, ezzel jelezve a sor feldolgozásának végét.

Ha a név a sor legelején kezdődik, akkor az címke. Ez a címke változó jelzi. Tábla-eljárás a memóriában tárolt elemek között keresi az st által mutatót nevet. Ha nem talál ilyet, akkor létrehozza új

1. ábra



elemként, minden esetben az elemre mutató pointerrel tér vissza. Ez kerül az-akt változóba.

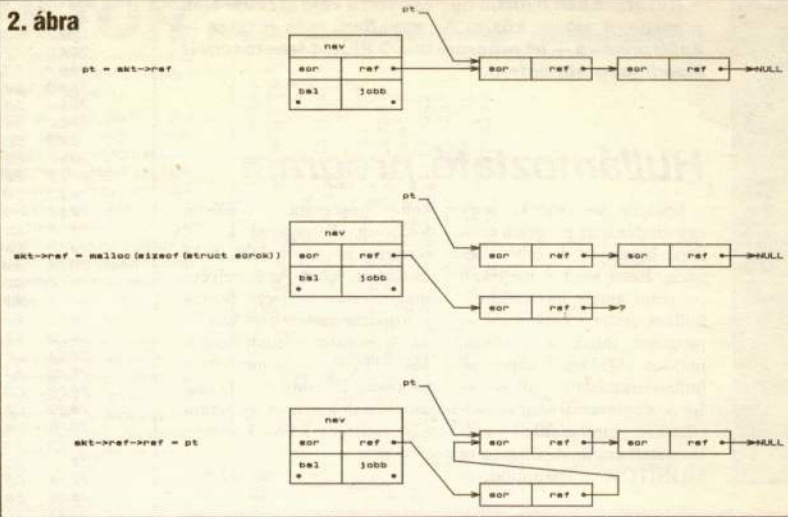
Ha a név a sor elején kezdődött, vagyis ha címke, akkor ebben a sorban definiált, és akt elembe a definiálás sora beírható. Ha a név nem a sor elején kezdődött, akkor a hivatkozási sorba be kell láncolni. Ezt a 2. ábra rajzban mutatja.

Egy név feldolgozása után a sorban álló további nevek keresése következik. Ha a sor végére értünk, akkor a következő sor, ha pedig a fájl végére, akkor már csak a táblázat kiírása van hátra. Ezt a kereset eljárás végzi. A bemeneti és kimeneti fájl lezárásával fejeződik be a főprogram.

Ezek után nézzük, hogyan látják el feladatukat a szubrutinok! A legegyszerűbb a kereset eljárás működése. Ez a sorban a név kezdetét keresi. Néven betűvel kezdődő, betű- és számkaraktereket tartalmazó karaktersorozatot értünk. Az eljárás feladata az idézőjelek közé tett sztringek és a megjegyzések figyelmen kívül hagyása is. Az eljárás végén a név hosszának meghatározása történik.

A táblázatban való keresést

2. ábra



a kereset eljárás végzi. Először összehasonlítja a keresett elemet a fa gyökerét adó első elemmel. Ha a kettő megegyezik, akkor visszatér, ha a név az ábcérendben korábban áll, akkor a fa bal oldali ágán halad tovább, egyébként pedig a jobb oldali ágban. A keresés mindaddig folytatódik, amíg a keresett elemet megtalálja, vagy NULL-ra nem akad. Ez utóbbi esetben létrehozza az új elemet, megfelelő kezdőértékekkel feltölti, és a

NULL pointer helyére beilleszti.

A keresztreferencia-táblázat kiírása a kereszt rekurzív eljárás feladata. A rutin önma-

gát hívja meg az ee elemtől balra eső elemek kiírásához, majd kiírja ee-t, és ismét önmagát hívja meg az ee elemtől jobbra eső elemek kiírásához. Ha valahol NULL-t talál, akkor visszatér az előző szintre.

Figyeljük meg a táblázat gyökerét adó első elem szerepét! Ez egy fiktív elem, egy nem létező névhez tartozik. Egyedüli feladata a kereset egyszerűbbé tenni. A kereszt eljárás ezt nem fogja kiírni, mivel a 0. sorban „definiált”.

A programban használatnál hatékonyabb memóriafelhasználás is elérhető. Az elemek neve lehetne változó hosszúságú sztring. Ha megfigyeljük a hivatkozási sor sorok elemeinek felépítését, láthatjuk, hogy a helynek legalább a felét a láncolat végző ref pontereket teszik ki. Valószínűleg átlagosan jobb eredményt érhetünk el, ha egy sorok elemben egy-nél több hivatkozást tárolunk. Így kevesebb pointer kell, viszont maradnak üres kelesztsek.

20. feladat:

Disassembler

Írjon disassembláló programot egy tetszőleges processzorra, tetszőleges nyelven! Törekedjen arra, hogy a program más assembly nyelv disassemblálására könnyen átirható legyen.

```
#include <stdio.h>
#include <alloc.h>
#include <ctype.h>
#include <dos.h>
#include <stdlib.h>

#define NHOVSZ 32

/* Az a sorban megjelenő az első név kezdetét. Visszaad egy név stringre mutató pointer-t ill. NULL-t, ha a sorban címke név, hosszában visszaszája a név hosszát. */
char kereset(char *s, int *hossz);
char *;

*hossz = 0;
/* Elejét keresi. */
while (*s && !isalpha(*s));
/* String kiírás. */
while (*s && (*s == ' ')) {
    s++;
}
while (*s && (*s != '\0')) {
    *hossz++;
}
/* Megjegyzés kiírása. */
if (*s == '/') return (NULL);
s++;
/* Ha nem talált. */
if (!*s) return (NULL);
/* Végét keresi. */
while (*s && !isalnum(*s)) {
    s++;
}
return (s);
/* Hivatkozások sora. */
struct sokor {
    int sor;
    struct sokor *ref;
};
/* Egy elem. */
struct elem {
    /* Az elem sorszám. */
    struct elem *sz1;
    /* Az elem neve. */
    char nev[NHOVSZ];
    /* Ebben a sorban definiált. */
    int sor;
    /* Itt az a sorokban hivatkozott. */
    struct sokor *ref;
};
/* A szűz elem. */
struct elem első = {NULL, NULL, "", 0, NULL};
/* input és output file. */
FILE *infile, *outfile;
/* Keresi a szűz elemet. Visszatér ha nincs ilyen létrehozva. Visszatér az elemre mutató pointerrel. */
struct elem tábla(char *s) {
    struct elem *t, *te;
    int c;
```

```
t = &első;
while (!t)
    if (!t = strcmp(n, t->nev))
        return (t);
    te = t;
    if (c < 0) t = t->bal;
    else t = t->jobb;
/* Uj elem létrehozása. */
if (!t)
    if (!t = malloc(sizeof(struct elem)))
        printf("Kétség a memórián.");
    abort();
strcpy(t->nev, n);
t->jobb = NULL;
t->bal = NULL;
t->sz1 = 0;
t->ref = NULL;
/* Az új elem beillesztése. */
if (c < 0) te->bal = t;
else te->jobb = t;
return (t);
/* Kírja a keresztreferenciát
és-től kezdve. */
void kereset(struct elem *ee);
static struct sokor *s;
if (ee) {
    /* ee szűz álló elemek kiírása. */
    kereset(ee->sz1);
    /* Ha definiált a szűz elem. */
    printf(outfile, "%d", ee->sz1);
    /* Hivatkozások kiírása. */
    while (s)
        if (s == ee->ref)
            while (!s)
                printf(outfile, "%d", s->sz1);
                s = s->ref;
            printf(outfile, "%n");
        /* ee után álló elemek kiírása. */
        kereset(ee->jobb);
}
if (!infile = fopen(argv[1], "rt"))
    printf("Nincs filem.");
abort();
if (!outfile = fopen(argv[2], "wt"))
    printf("Nincs filem.");
abort();
}
/* Egy sor beolvasása. */
sor = s;
for (i = 0; i < getc(infile); i++)
    if (i == EOP)
        break;
if (i == '\n') break;
return (i);
*st = '\0';
sorszám = 0;
/* A sor feldolgozása. */
st = s;
while (st = kereset(st, &h))
    címke = (st == s);
sth = *(st-h);
*(sth) = '\0';
akt = tábla(st);
if (!címke)
    /* Ebben a sorban definiált. */
    akt->sz1 = sorszám;
else
    /* Az új hivatkozások sor tárolása */
    sorszám = akt->sz1;
if (i < akt->ref)
    if (!akt->ref = malloc(sizeof(struct sokor)))
        printf("Kétség a memórián.");
    abort();
akt->ref->sz1 = sorszám;
akt->ref->ref = pt;
st = st + h;
*st = sth;
while (!feof);
/* Az input file lezárása. */
fclose(infile);
/* A keresztreferencia kiírása. */
kereset(sorszám);
/* A kereset lezárása. */
fclose(outfile);
```

Rovatunkban ezúttal egy csokorra való C Plus/4-es programot adunk közre. Az egyetlen, más típusra — Enterprise-ra — írt program is a C Plus/4 lehetőségeit kísérel meg átmenteni.

Hullámoztató program

Sokszor szeretnénk, hogy egy általunk írt program címkepe látványos és ötletes legyen. Ezen segít a mellékelt program, amely, ha elindítjuk, hullámoztatója a képernyőt. A program mind a grafikus, mind a karakteres képernyő hullámoztatására alkalmas. Így a programmal akár egy általunk készített grafikát is hullámoztathatunk. A programot MONITOR üzemmódban

kell begépelni \$7000-tól \$7034-ig. Kimenteni a S" a program neve", X, 7000, 7034 utasítással lehet. Az X helyére magnó esetén 1-et, floppy használata esetén 8-at kell írni. A program visszatérése is MONITOR üzemmódban történik. Elindítani a G7000 utasítással lehet. A program egy esetleges RESET esetén sem vész el.

Bácsi Péter

| | | | | | |
|---|------|----|----|-----|------------|
| - | 7000 | A9 | 08 | LDA | \$908 |
| - | 7002 | BD | 07 | FF | STA \$FF07 |
| - | 7005 | A2 | 00 | LDX | \$900 |
| - | 7007 | EB | | INX | |
| - | 7008 | E0 | 40 | CPX | \$940 |
| - | 700A | D0 | FB | BNE | \$7007 |
| - | 700C | A0 | 00 | LDY | \$900 |
| - | 700E | CB | | INY | |
| - | 700F | C0 | 01 | CPY | \$901 |
| - | 7011 | D0 | F4 | BNE | \$7007 |
| - | 7013 | 18 | | CLC | |
| - | 7014 | 69 | 01 | ADC | \$901 |
| - | 7016 | C9 | 10 | CHP | \$910 |
| - | 7018 | D0 | EB | BNE | \$7002 |
| - | 701A | A9 | 0F | LDA | \$90F |
| - | 701C | BD | 07 | FF | STA \$FF07 |
| - | 701F | A2 | 00 | LDX | \$900 |
| - | 7021 | EB | | INX | |
| - | 7022 | E0 | 40 | CPX | \$940 |
| - | 7024 | D0 | FB | BNE | \$7021 |
| - | 7026 | A0 | 00 | LDY | \$900 |
| - | 7028 | CB | | INY | |
| - | 7029 | C0 | 01 | CPY | \$901 |
| - | 702B | D0 | F4 | BNE | \$7021 |
| - | 702D | 38 | | SEC | |
| - | 702E | E9 | 01 | SBC | \$901 |
| - | 7030 | C9 | 07 | CHP | \$907 |
| - | 7032 | D0 | EB | BNE | \$701C |
| - | 7034 | 4C | 00 | 70 | JMP \$7000 |

```

100 PRINT"ELLENORZES INDLU"
110 H=0
120 FOR I=0 TO 33:S=0
130 FOR J=0 TO 7
140 READ X$
150 S=S+DEC(X$)
160 NEXT J
170 READ S$:OS=OS+DEC(S$)
180 IF S<>DEC(S$) THEN PRINT"ADATHIBA AZ
"1000+I". SORBAN I":H=H+1
190 NEXT I
200 IF OS<>33510 THEN PRINT"HIBA A KONT
ROLLOSSZEKEKBEN !!!"
210 IF H<>0 THEN PRINT"PRINT H" HIBAT TA
LALAM I":END
220 PRINT"ELLENORZES KESZ. A PROGRAM H
IBATLAN..."
230 KC=1818
240 PRINT"INDUL A BEIRAS I"
250 RESTORE
260 FOR I=0 TO 33:FOR J=0 TO 7
270 READ X$:POKE KC+I*8+J,DEC(X$)
280 NEXT J:READ S$:NEXT I
290 PRINT"BEIRAS KESZ..."
300 PRINT"INDITAS:"
310 PRINT"SYS"KC"OSZLOP,SOR,"SZOVEG"
I,INVERZ]"

```

```

1000 DATA 20, BF, C7, 20, D8, 90, E0, 27, 442
1001 DATA 80, 09, 86, D0, 20, 18, 9D, E0, 484
1002 DATA 18, 90, 03, 4C, 1C, 99, 86, D1, 303
1003 DATA 20, 91, 94, 20, 49, F0, F3, 42C
1004 DATA 85, D2, 96, 48, 8A, 48, 20, A5, 3CE
1005 DATA C3, 8A, 6A, 66, D3, 68, 85, 22, 3FF
1006 DATA 68, 85, 23, A0, 80, 84, D7, A9, 3B4
1007 DATA 00, 85, D6, 20, 0B, 04, A2, 25, 25F
1008 DATA D6, 8A, 0A, 26, D5, 85, D4, A5, 3E4
1009 DATA D6, D6, E4, 02, 85, D5, 38, A5, 450
1010 DATA D4, E9, 01, 85, D4, B0, 02, C6, 48F
1011 DATA D5, 20, AF, 03, E6, D0, E6, D0, 513
1012 DATA A6, D0, E9, 27, 30, 9F, A2, 00, 35E
1013 DATA 86, D0, E6, D1, E5, D1, A5, D1, 63B
1014 DATA E0, 18, 90, 01, 60, A4, D7, CB, 42C
1015 DATA C4, D2, D8, B9, 60, A0, 09, A9, 4D0
1016 DATA 00, 85, D9, 85, D9, B1, D4, 49, 499
1017 DATA 29, 8F, A2, 04, 4A, 90, 0A, E9, 20A
1018 DATA 18, D0, 36, 04, 65, D9, 85, D9, 3AB
1019 DATA 68, CA, D0, F0, 68, 29, F0, 6A, 400
1020 DATA 6A, 6A, 6A, A2, 04, 4A, 90, 0A, 2CB
1021 DATA 48, 18, 8D, 36, 04, 65, D8, 85, 319
1022 DATA D8, 69, CA, D0, F0, A5, D8, 99, 5E0
1023 DATA CB, 02, A5, D9, 99, D3, 02, 88, 441
1024 DATA D0, BD, A6, D1, A4, D0, 20, 91, 529
1025 DATA C2, A0, 09, A9, C0, 20, 1D, 04, 31B
1026 DATA A9, D4, 29, 1D, 94, 18, A9, 40, 2BF
1027 DATA 65, 8C, 85, 8C, A9, 01, 65, 8D, 39E
1028 DATA 85, BD, A0, 00, A9, D0, 20, 1D, 368
1029 DATA 04, A9, D8, 8D, 23, 04, A2, 00, 2D0
1030 DATA BD, D8, 02, D4, D3, 10, 02, 49, 2E9
1031 DATA FF, 91, 8C, C8, 91, 8C, C8, E8, 581
1032 DATA E0, 04, D0, EC, 60, C0, 30, 0C, 3FC
1033 DATA 03, 00, 00, 00, 00, 00, 00, 00, 003

```

2x2=?

Az iskolákban egyre jobban elterjedő Plus/4-es gép hátránya, hogy a képernyőre kiírt szövegek — méretük miatt — távolról nem láthatók jól, ez pedig erősen gátolja iskolai alkalmazásukat. Ezért határoztam el, hogy olyan programot írjak, amely: 2x2-es méretben ír, könnyen kezelhető, „keverhető” a nagy felbontású grafikával és gyors.

A következő, gépi kódú program a CHAR utasításhoz hasonlóan valósítja meg a nagyított kirajzolást. A program hívásakor a bal felső karaktermegyéd koordinátáit kell megadni (x=0—38, y=0—23). Ha a szöveg nem fér ki egy sorban, a kiírás a kettővel lejjebb levő sor legelső oszlopán folytatódik. A program a színmemóriába nem ír, így a képernyő színei nem változnak meg a szöveg kiírásával. A karakterkészlet kezdőcímét a \$02e4 (740) címen állíthatjuk, alapértelmezése \$d0 (208), azaz a nagybetűk/grafikus

jelek készletét jeleníthetjük meg, de \$d4 (212) beírásával a kis- és nagybetűk is kírathatók.

A program BASIC betöltője először végigellenőrzi az összes adatot és az ellenőrző összegeket, s csak ha mindent rendben talál, akkor ír bele a memóriába. Az 1. listán szereplő program a \$0332 (818)-as címen kezdődik, és nem helyezhető át más címre, csak a 2. listában lévő módosítások begépelésével. A program használja a zérólapon a \$d0—\$d9 és a \$02cc—\$02db címeket.

Indítás: SYS kezdőcím, X, Y, <kiírandó szöveg> [,inverz?]

Hibalehetőségek: a koordináták nem esnek a 0—38, illetve 0—23 intervallumba, a szöveg üres sztring, vagy nincs grafikus terület.

A program jól használható képűságokban, oktató- és más, nagyított kirajzolást igénylő programokban.

Kálmán Tibor

2. lista

```

230 INPUT"K KEZDOCI" :KC
283 FOR I=1 TO 7:READ A,B :X$=HEX$(B+KC)
284 POKE KC+A,DEC(LEFT$(X$,2))
285 POKE KC+A+1,DEC(LEFT$(X$,2))
286 NEXT I
2000 DATA 90,125,146,260,171,260,206
2001 DATA 235,211,235,231,235,236,241

```

1. lista

KORREKTOR

Sok felesleges bosszúságtól kímélt meg mindenkit az a program, amelynek magyarított változatát most készítjük el C Plus/4-es, C16-os és C116-os gépre.

Gépeljük be a programot, majd futtassuk. Javítsuk ki a gépelési hibákat (ezekre a program külön felhívja a figyelmünket). Ha a program hibátlan, akkor megkérdezi, ki akarjuk-e menteni. Ennek természetesen csak az első alkalommal van értelme. Ha megtörtént a kimentés, vagy a kérdésre nem az „I” gomb lenyomásával válaszoltunk, akkor a program átkódolja magát, majd megjelenik a READY felirat. Most indítsuk el a programot (RUN és RETURN), majd indítsuk el még egyszer (ismét RUN és RETURN). Most a képernyő alsó sora inverzbe vált. Ha ezután bármilyen sort begépelünk, megjelenik annak sorszáma és utána az ellenőrző szám. Ezt kell feljegyezni.

Ha nyomtatjuk az általunk írt programot, írjuk a megfelelő sorok után a megfelelő számot, pontosvesszővel elválasztva. Így a korrekttal elkészített programot igen kényelmesen begépelhetjük, csak be kell töltenünk a korrekort, el kell indítanunk a fentebb említett módon, és elkezdhetjük begépelni a korrekttal készített programot. Begépelés közben rögtön ellenőrizhetjük a sorokat. A gépelés akkor hibátlan, ha a listán lévő és a korrekort által megjelenített szám egyezik.

Figyelem! Tilos a korrekort programba beleírni, hozzáírni vagy törölni belőle, mert ez hibás futást eredményez!

Terveink szerint következő számunkban már a korrekort C64-es változatát is olvashatjuk.

Bácsi Péter

```
10 TRAP$20=RESTORE
20 DATA# 16, C1, 07, 0E, 26, 34, 31, 33, 38
30 DATA# 00, 00, 00, AD, E5, 07, C9, 18, D6, 03
40 DATABE, E5, 07, AD, E5, 07, C9, 18, D6, 03
50 DATABE, E5, 07, AD, 00, 8D, F1, 87, 4C, 0E
60 DATABE, A9, 30, 8D, 02, 03, AD, 16, 0D, 03
70 DATABE, A9, 11, 0E, 2C, 00, A2, 18, 0E, 15
80 DATABE, A2, FF, 05, 3A, 28, 5A, 00, 06, 3B
90 DATABE, 3C, AD, C8, 0F, C9, A9, F9, 18, A2
100 DATA# 1D, 0C, 0F, 9D, CE, 0F, A9, 0D
110 DATABE, 0F, CA, 18, F2, 29, 73, 0A, A9, F8
120 DATABE, 08, 03, AC, 25, 07, 28, 35, 0E, 28
130 DATABE, 05, 0A, 08, A9, 00, 04, 00, 0A, 45
140 DATABE, 46, 0A, 09, 09, C6, 28, A5, 0A, A4
150 DATABE, 38, A5, 00, 38, 0C, 1E, 20, F9, F1
160 DATABE, 3A, F8, 0C, E9, 0F, F9, 2D, E9, 22
170 DATABE, 02, 49, 00, 09, 01, 85, 00, 0A, A4
180 DATABE, F8, 05, 0A, 09, 0A, C8, FA, 18
190 DATABE, 45, 05, A5, 00, 82, 05, 05, 00
200 DATABE, 09, 29, 07, 05, 09, 10, C1, 4E, 00
210 DATABE, 17, A8, FF, A2, 1A, 28, D8, 1C, C0
220 DATABE, 8D, 99, C8, 0F, AC, 25, 07, 28, 35, 0E, 28
230 DATABE, 8B, 99, C8, 0F, AC, 25, 07, 28, 35, 0E, 28
240 DATABE, 00, 85, 03, 85, 01, 85, 62, AC, 20
250 DATABE, 28, CE, A2, 28, F1, A4, A4, 0B, A4
260 DATABE, 8D, 00, 81, F8, 00, 00, 9B, C0
270 DATABE, E5, D6, F2, 69, 00, 00, 00
280 PRINT "MINDEN GÖT 300:1=300"
290 M=N+1:READA:A+A+DEC(A):I=LEN(A)+2 THEN 290
300 PRINT "HIBAS ADAT: "A+POKE(1240,PEEK(03))+POKE(1250,PEEK(04))+POKE(1263,1)
310 A=PEEK(65)+PEEK(66)+256-LEN(A):I=POKE(1265,A) AND 255:POKE(1270,A)+256:HELP=END
320 IF E=38 THEN PRINT "A PROGRAM FUTA HEGEZKARÁ A STOP GOMB LEUTÉSE MIATT!"
330 IF E=14 THEN RESUME 300
340 IF E<13 THEN PRINT "FIGYELEM! HIBA AZ ADATOKÁRNI! HELP=END
350 IF E<258 OR AC>27332 THEN PRINT "FIGYELEM, HIBA AZ ADATOKÁRNI! "END
360 PRINT "ELLENÖRZÉS KEZDI!"
370 PRINT "MINDEN ELLENŐRZÉS MEGTARTOTT" GETKEY A#
380 IF A#="" THEN GOT 390:1=300
390 PRINT "MINDEN HIBÁZT VAGY HIBÁZTÁRRA MENTKÉRT? GETKEY A#
400 IF A#="" THEN USAVE"KORREKTOR":GOTO340
410 IF A#="" THEN SAVE"KORREKTOR":GOTO340
420 GOTO 300
430 RESTORE:FORA=400770A35:READA:POKE,DEC(A):NEXT
440 POKEA5,3:POKEA6,17:CLR
```

Turbo

Grafikus rutinok C Plus/4-re

A Commodore+4, —16 finomgrafikus üzemmódját használva gyakran joggal érezzük, hogy szép-szép, de lassú. Az alábbi programok ezen igyekeznek segíteni. A programok \$1100—\$1320 között helyezkednek el. Ez a területet a BASIC interpreter használja, ha a grafikus képernyő nincs bekapcsolva, ezért a programok begépelése előtt adjuk ki a GRAPHIC 1,1 : GRAPHIC 0 parancsot. Begépelés és elmentés után, minden egyes betöltés előtt és esetleges RESET után adjuk ki ezt a parancsot, így a BASIC ezekre a területekre nem fog írni.

A pontrajzoló a \$1100 kezdőcímen kezdődik. A \$02—03 címen az X, a \$01 címen pedig az Y koordinátát adhatjuk meg. Ha az X koordináta nagyobb, mint 255, írjuk a \$03-as címre 01-et, egyébként pedig 00-át. A koordináták megadása után a rutint gépi kódból a JSR \$1100, BASIC-ből a SYSDEC("S1100") utasítással aktivizálhatjuk. Ennek hatására a program a koordináták által meghatározott helyre egy pontot tesz a \$2000 — \$4000-ig terjedő finomgrafikus képernyőre.

A ponttörölő hasonló elven működik, mint a pontrajzoló. A törölendő pont koordinátáit ugyanarra a címre kell megadni. Hívása gépi kódból: JSR \$1150, BASIC-ből SYSDEC ("1150").

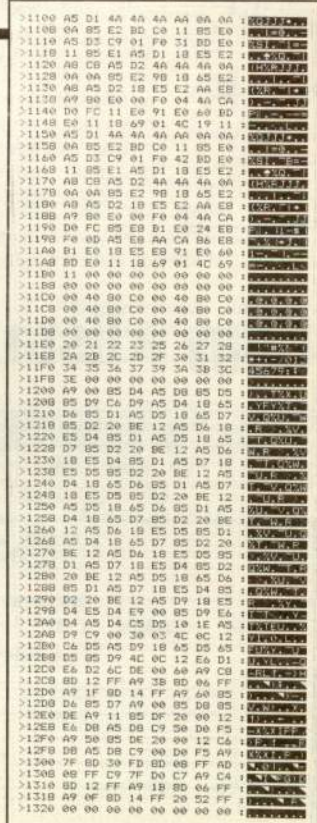
A kórrajzoló a pontrajzolóra épül. A

kör középpontjának X koordinátáját a \$D6-os címen, Y koordinátáját a \$D7-es címen kell megadni. A kör sugarát a \$D8 címen kell megadni, de az értékek természetesen csak 0—255-ig terjedhetnek, mivel a sugarat a program egy bájtól ábrázolja. A kétbájtos számokkal való számolás nagyobb és lassabb programot igényel. A \$DE—DF címenek állíthatjuk be, hogy rajzolni vagy törölni kívánjuk a kört. Ha rajzolni, 00 11-et helyezzünk ide, ha törölni akarunk, akkor pedig 50 és 11-et. Törlés esetén innen hívható meg a ponttörölő program. A program hívása gépi kódból: JSR \$1200, BASIC-ből SYSDEC ("1200").

Az összehasonlítás kedvéért írjuk be a BASIC nyelvű kórrajzoló, indítsuk el, és figyeljük egy darabig. A program 100 kört rajzol a képernyőre 80 másodperc alatt. A gépi kódú listában szerepel még egy kis demo is. Hívása MONITOR-ból a G12C6, BASIC-ből SYSDEC ("12C6"). Ez a rutin is 100 kört rajzol a képernyőre, de csak 4 másodperc alatt. A különbség szembeszökő a BASIC és a gépi kódú között.

Szabolcsi Szabolcs

```
10 GRAPHIC1,1:K=961:Y=961:R=0
20 CIRCLE1,X,Y,R:R=R+1:IFR<=20THEN20:ELSEEND
```



Elforgatott ellipszisek

Örülhetnek a Plus/4-tulajdonosok, mert a CIRCLE utasítással egyszerűen rajzolhatnak elforgatott ellipsziseket is. De sajnos azt Enterprise IS-BASIC ELLIPSE utasítása ezt nem teszi lehetővé. Az 1. listán látható eljárás ezt a hiányosságot pótolja. Az elforgatott ellipszis rajzolására tehát a következő módszer használható:

CALLELLIPSE (cha, x, y, xs, ys, n)
ahol:

- cha = a csatornaszám, ahová rajzolunk
- x = az ellipszis középpontja (X koordináta)
- y = az ellipszis középpontja (Y koordináta)
- xs = az ellipszis vízszintes sugara
- ys = az ellipszis függőleges sugara
- n = a negatív elforgatás fokában

Megjegyzés: a 9989-es sor végén az 50-es konstans csökkentésével az ellipszis közelítése javítható. Ennek különösen kis sugarú ellipsziseknél van jelentősége. A 2. lista egy demo program, amely az ELLIPSE eljárással együtt használható.

Kovács Gábor

```

9984 DEF ELLIPSE(CHA,X,Y,XS,YS,N)
9985 SET #CHA:BEAM OFF
9986 NUMERIC T,XP,YP,SI,CO,ST,X1,Y1
9987 LET ST=RAD(REM(N,180))
9988 LET SI=SIN(ST):LET CO=COS(ST)
9989 LET ST=PI*2/((XS+YS)*PI/50)
9990 FOR T=0 TO PI*2+ST STEP ST
9991 LET XP=SIN(T)*XS+X
9992 LET YP=COS(T)*YS+Y
9993 LET X1=(XP-X)*CO+(YP-Y)*SI+X
9994 LET Y1=(YP-Y)*CO-(XP-X)*SI+Y
9995 PLOT #CHA:X1,Y1;
9996 NEXT
9997 SET #CHA:BEAM OFF
9998 PLOT #CHA:X,Y,
9999 END DEF

```

1. lista

2. lista

```

100 GRAPHICS HIRES 2
110 SET PALETTE 0,255
120 FOR T=0 TO 179 STEP 30
130 CALL ELLIPSE(101,640,360,200,340,T)
140 NEXT
150 PLOT PAINT

```

Hogy a lista újra TOP legyen!

Új év, új lista. Mostantól az itt látható formában, szavazócédulával együtt kívánjuk megjelentetni TOP-listánkat. Ez a verzió az ol-

vasók észrevételei, javaslatai alapján készült el. Sokan és sokszor kifogásolták, hogy egy listán szerepelnek a különböző, eltérő

teljesítményű géptípusok programjai. Mi ezzel teljesen egyetértünk, így most igyekeztünk egy új, némileg realisabb képet nyújtó

TOP-listát összeállítani. Nemcsak egy-egy gép képességeit nézzük, hanem az adott típusra írt szoftverek mennyiségét és a gép elterjedtségét is hazánkban.

A nagy nyugati számítógépes újságok TOP-listáival természetesen nem kívánunk konkurálni, hiszen ott minden program előbb jelenik meg a nagyközönségnek, mint itt. Eddig is kaptunk észrevételeket, és valószínűleg ezután is kapunk mindkét irányból, vagyis a nagyközönségtől és a szűkebb, jobb szoftverimporttal rendelkező baráti köröktől. Az olvasók nagy része szerint ismeretlen programokkal tömjük meg a listánkat; a baráti kör pedig azt mondja, hogy ezek a programok már szakállasak. Igyekszünk megfelelni az igényeknek, bármilyen kacifántosak is legyenek.

Kérjük a kedves olvasókat, hogy minél többen szavazzanak a mellékelt szavazócédulákon, hogy a lista újra TOP legyen!

Szavazócédula

Szavazatom az alábbi géptípusra szól:

IBM-PC Amiga C-128 C-64 C-4 Spectrum TVC Enterprise

(A megfelelőt kérjük megjelölni)

| | | | |
|---|-------|----|-------|
| 1 | _____ | 6 | _____ |
| 2 | _____ | 7 | _____ |
| 3 | _____ | 8 | _____ |
| 4 | _____ | 9 | _____ |
| 5 | _____ | 10 | _____ |

Diákszerkesztőség

Szoftverszolgálat

A nagy év eleji megújulások közepette merült fel az ötlet, hogy más számítástechnikai lapokhoz hasonlóan a Mikroszámítógép Magazin diákrovatának is legyen programküldő szolgálata. A szolgáltatás lényege, hogy a rovatban megjelent programokat az olvasó által küldött adathordozóra felvéve, az önköltség felszámításával elküldjük, megkímélve ezzel az olvasókat a hosszadalmas, fáradtságos gépelgetéstől.

Jelenleg C128-as, C64-es és C Plus/4-es gépekre korlátozódik a lehetőségünk, e géptípusokra tudjuk küldeni a programokat, de az igényeknek megfelelően bővíthet a szolgáltatási kör.

A költség a program hosszának megfelelően 10 és 40 forint között alakul. Az adathordozó lehet szalag vagy lemez egyaránt, amit lehetőleg minél jobban becsomagolva, pontos feladót feltüntetve küldjenek el. Kérjük, minél többet írják meg véleményüket, észrevételeiket. A szolgáltatást természetesen csak kellő érdeklődés esetén tudjuk beindítani.

Felhívás!

A Mikroszámítógép Magazin Diákszerkesztősége várja az ország különböző területein élő számítógép-kedvelők jelentkezését szerkesztőségünk tagnak. Mivel pillanatnyilag a Diákszerkesztőség csak budapesti tagokból áll, szeretnénk, ha a vidéki olvasók, számítógéppelbarátok is bekapcsolódnának a rovat készítésébe. (Természetesen budapestiek jelentkezését is várjuk!)

A felhívásra levélben vagy személyesen jelentkezőknek először bizonyítaniuk kell rátermettségüket, hogy a Diákszerkesztőség teljes jogú tagjaivá váljanak. Ezt egy három hónapos próbaidővel kívánjuk felmérni, ami alatt a jelentkezőkkel személyesen vagy levélben tartanánk kapcsolatot. A jelentkezők írják meg életkorukat, iskolájukat, az őket érdeklő szakterületeket,

valamint azt, hogy milyen géppel foglalkoznak jelenleg és milyen szinten. A próbaidő alatt igyekezzenek minél több cikket, programot, érdekeséget küldeni, és alkalmanként látogassanak el a Mikroszámítógép Magazin szerkesztőségébe, a diákszerkesztőség összejöveteleire, megbeszéléseire.

Ezeket az összejöveteleket felvethetitek ötleteiteket a rovat újítására, témák megválasztására. Az összejöveteleket általában hetente péntek délutánonként tartjuk, de természetesen telefonon bármikor lehet érdeklődni a szerkesztőségben. Szeretettel várjuk jelentkezéseket!

A Mikroszámítógép Magazin
Diákszerkesztősége

| TOP LISTA | | | | |
|-------------------------|-----------------------|----------|--------------------------|----------|
| Osztály, géptípus | Játék programok | géptípus | Felhasználói programok | géptípus |
| IBM AMIGA | Mileneum | Amiga | Ventura Professional 2.0 | IBM |
| | Sentinel | IBM | PC Tools 5.5 | IBM |
| | Kick Off | Amiga | GEM Artline | IBM |
| C-128 C-64 C+4 | Pool of Radiance | C-64 | GEOS 2.1 | C-128 |
| | Bard's Tale III. | C-64 | GeoPublish | C-64 |
| | Neuromancer | C-64 | Giga Paint | C-64 |
| Enterp Spectr TVC | How to be complete b. | Spect. | --- | |
| | Bard's Tale III. | Spect. | --- | |
| | Knightmare | Spect. | --- | |

bitkiller

Listánkat felhasználói, illetve játékiprogramokból állítottuk össze. A legjobbakat, legerdekesebbeket a beküldött javaslatok alapján rangsoroljuk. Ehhez kérjük az olvasók közreműködését. C64-re, ZX-Spectrumra, Enterprisé-re, TVC-re, Atarira és IBM-re készült programrangsorokat várunk havonta.

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

BASIC-bővítések Commodore 16-ra

4. rész

A sorozatot egy újabb parancs ismeretével folytatom. Ez pedig a WINDOW.

A szó eredetileg ablakot jelent, és ez pontosan takarja a parancs tartalmát. A Commodore 16-nak van egy olyan lehetősége, hogy a felhasználó határozhatja meg, hogy a 25x40-es képernyő mely területét kívánja használni (ezt a területet nevezzük ablaknak).

Az <ESC>+„T” az ablak bal felső sarkát, az <ESC>+„B” a jobb alsó sarkát jelöli ki.

Ezeknek az ablakoknak a definiálása nagyon hosszadalmas, időrabló, hiszen a kurzort először az egyik, majd a másik sarokba kell vinni, és közben az „ESC” parancsokat kiadni. Továbbá, ha nem mindig egyforma az ablakok helye vagy mérete, akkor ezek definiálása — még programban is — bonyolult.

Ezen segít a WINDOW parancs, amelynek szintaxisa: WINDOW X1, Y1, X2, Y2, ahol

- X1, Y1 a bal felső sarok,
- X2, Y2 a jobb alsó sarok koordinátái.

A parancs kiadása után ez az ablak fog „működni”, és a kurzort a HOME pozícióban, azaz X1, Y1-ben kapjuk vissza.

Az X és Y értékénél be kell tartani azt az intervallumot, amit a CHAR parancs használata során már megszoktunk, hogy: $0 < X < 39$ és $0 < Y < 24$. Ezek az értékek nemcsak konstansok (állandók), hanem változók, sőt kifejezések is lehetnek!

Például a WINDOW 1,1,12,12
 a B=1:C=12:WINDOW
 B,B,C,C
 a B=1:WINDOW B+1, B+1, B+12, B+12

teljesen egyenértékű.
 Egy ablak addig létezik, amíg a <HOME> <HOME> vagy a CHAR vagy a WINDOW parancsot ki nem adjuk.

A bővítés listáját az ábrán láthatjuk. A működése egyszerű. Először beolvassa az értékeket, majd kiértékeli. Ha a megengedett intervallumon belül van, akkor ezt tárolja. Ezután beállítja a kurzor új helyét, kiszámolja az új címet. Ha a beolvasáskor hibát talál, akkor a hiba jellegének megfelelő hibaüzenetet, ha a kiértékeléskor hibát, akkor az ?ILLEGAL QUANTITY hibát írja ki.

```

asz -16: pass | ?
2000                                .opt p
2000                                #= $2000
2000                                tok = $0300
2000                                detok = $030e
2000                                vegre = $0318
2000                                alsosor = $02e5
2000                                felsosor = $02e6
2000                                chrset = $0473
2000                                balosz = $02e7
2000                                jobbosz = $02e8
2000                                xbe = $9d84
2000                                vesszo = $9491
2000                                plot = $fff0
2000                                be1 = $0b
2000                                be2 = $43
2000                                be3 = $23
2000 a3 1f                          lda #tok1
2002 8d 0c 03                       sta tok
2005 a5 20                          lda #>tok1
2007 8d 0d 03                       sta tok1
200a a5 33                          lda #<detok1
200c 8d 0e 03                       sta oetok
200f a5 20                          lda #>detok1
2011 8d 0f 03                       sta detok1
2014 a3 3f                          lda #<vegre1
2016 8d 10 03                       sta vegre
2019 a5 20                          lda #>vegre1
201b 8d 11 03                       sta vegre+1
201e 80                              rts
;
;
;
201f 48                              tok1 pha
2020 a5 2d                          lda #>tabla
2022 a0 1f                          ldy #<tabla
2024 20 27 8a                       jar $8a07
2027 68                              pia
2028 90 06                           boc vegre
202a a5 0b                          lda be1
202c 48                              pha
202d 4c d6 83                       jmp $89d6
2030 4c 6c 83                       jmp $896c
;
;
;
2033 aa                              detok1 tax
2034 24 13                          sty be2
2036 a0 20                          ldy #>tabla
2038 84 23                          sty be3
203a a0 1f                          ldy #<tabla
203c 1c 9c 8b                       jmp $8b9c
;
;
;
203f 38                              vegre1 sec
2040 e9 80                          sbe ##80
2042 8a                              asi a
2043 a8                              tay
2044 b5 57 20                       lda innen1,y
2047 48                              pha
2048 b5 56 20                       lda innen,y
;
;
;
204b 48                              pha
204c 4c 73 84                       jmp $0473
;
;
;
204f 57 19 4e                       lda #<tabla
2055 80                              .asc "window"
;
;
;
2056 57                              innen .byt <megy-1
2057 20                              .byt >megy
    
```

```

;
;
;
2058 ad e6 07 megy                  lda felsosor
205b 18                              pha
205c ad e5 07                       lda alsosor
205f 18                              pha
2060 ad e7 07                       lda balosz
2063 18                              pha
2064 ad e8 07                       lda jobbosz
2067 48                              pha
2068 70 81 9d                       jar xbe
206b 70 b3 70                       jar e11
206e b2 38                           bos hiba
2070 8d a7 07                       sta balosz
2072 70 91 94                       jar vesszo
2075 20 81 9d                       jar xbe
2079 20 be 70                       jar e11?
207c b2 22                           bos hiba
207e 8d e6 07                       sta felsosor
2081 20 91 94                       jar vesszo
2084 70 84 9d                       jar xbe
2087 70 b3 20                       jar e11
208a b0 11                           bos hiba
208c 8d e8 07                       sta jobbosz
208f 70 91 94                       jar vesszo
2092 20 81 9d                       jar xbe
2095 20 b3 20                       jar e12
2098 b0 06                           bos hiba
209a 8d e5 07                       sta alsosor
209d 1c c7 70                       jmp vegre1
20a0 68                              hiba pia
20a1 8d e8 07                       sta jobbosz
20a4 68                              pia
20a5 8d e7 07                       sta balosz
20a8 68                              pia
20aa 8d e5 07                       sta alsosor
20ac 68                              pia
20ad 8d e6 07                       sta felsosor
20b0 1c 1c 99                       jmp $991c
20b3 8a                              e111 txa
20b4 70 06                           bi hiba1
20b6 c5 28                           omp #40
20b8 b0 02                           bos hiba1
;
;
;
20ba 12                              clo
20bb 60                              rts
20bc 38                              h10a1 sec
20bd 60                              rts
20be 2a                              e112 txa
20bf 30 fb                           bi hiba1
20c1 c5 19                           omp #25
20c3 b0 f7                           bos hiba1
;
;
;
20c5 18                              clo
20c6 60                              rts
20c7 68                              vegre1 pia
20c8 68                              pia
20c9 68                              pia
20ca 68                              pia
;
;
;
20cb ae e6 07                       lax felsosor
20cc 86 cd                           stx $c4
20cd 86 c4                           stx $c4
20ce a4 ea                           ldy balosz
20d2 ac e7 07                       lda oalosz
20d5 84 ca                           sty $ca
20d7 84 c5                           sty $c5
20d9 18                              clo
20da 60                              rts
;
;
;
2000-20db
    
```

Kádár Sándor

20 éves a

UNIX

Már 20 éves? No, akkor már elég öreg lehet, ha a számítástechnika fél évszázados korszakához és rohanó élettelpójához viszonyítjuk. Mégis COCOM-listán van! Szoftverfejlesztő rendszerként 20 évesen is „High Technology Product”-nak számít! Jó, tudom. A 20 év alatt a Unixszal is történt azért valami. Kezdeti gyors fejlődése azonban hamar alábbhagyott. Generális megújítását ma éppen legjobban dicsért tulajdonsága, a hordozhatóság gátolja. A hordozhatóság és a szabványosság kedvéért, nekem úgy tűnik, elvtelenül gúzsba kötötték. Nem véletlen, hogy a szakma nagyjai (az IBM, a DEC stb., szóval a „hetek”) nem bírták tovább nézni ezt a vergődést, és az Open Software Foundation (OSF) keretében új fejlesztési irányt próbálnak indítani.

Megemlékezőm rendhagyó lesz, amennyiben nem állok be a Unix elvakult hozsannázói közé, hanem kicsit csapongok a téma körül. A cikk keretében először visszatekintek néhány olyan történelmi körülményre, amelyek a jubiláló Unix jellemzőit kialakították. Ezután a legnagyobb számítógépes aréna, a mikropiac kapcsán próbálom meg elemezni a Unixnak azokat a gyenge pontjait, amelyek miatt máig sem arathatott mindent elsőpró győzelmet. Végül megpróbálom válasz adni arra a kérdésre, hogy feljött-e végre a Unix csillaga, vagy pedig a társadalmi szférához hasonlóan a számítástechnikában sem a diktatórikus monopóliumoké, hanem a pluralizmusé a jövő.

A Unix és a COCOM

Mindenekelőtt visszatérek egy pillanatra a COCOM-ügyre. Nos, az a véleményem, hogy a Nyugat további hszontalan dollármilliárdokat fog ránk kidobálni, ha nem ad nekünk olyan fejlett technológiai eszközöket, amelyek gazdaságszervezési feladatainkban gyökeres változást tudnak hozni. A Unix kapcsán keményen le sze-

retném szögezni, hogy nem a 386-os, 33 MHz-es hardverek hiányoznak nekünk a legjobban, hanem a nagy teljesítményű szoftverfejlesztő rendszerek (továbbá a hálózati eszközök, de ezen a téren már lát-szik javulás). A COCOM pedig sajnos éppen a szoftverfejlesztő rendszerekkel kapcsolatban viselkedik a legcsökönnyösebben. A mi problémáink gyökere: 1945 óta, ismert módon kialakult eltérő gazdálkodási rendszerünk, továbbá szerencsétlen ábécénk miatt az amerikai, polcrol leemelhető szoftverek javarésze nálunk hajítófát sem ér. A hasonló szoftvereket kemény munkával magunknak kell kiizzadnunk. Ebben a Nyugat nekünk érdemben csak fejlesztő-rendszerekkel tud segíteni.

Ha nem lehet beindítani egy tisztességes felszerszámozott nagyipari szoftvergyártást, akkor tízezerrel helyen fog folyni valami minőségben aluli bütykölgetés a drága hardver pazarló használatával. Helyi zsenik próbálnak korszerűtlen, kétes helyről beszerzett szoftverrel megküzdeni az információs rendszerék feltámasztásával. A várható eredmény az, ami manapság a nyelvtanulás terén tapasztalható. A kezdő tanfolyamok tevé vannak, de a nyelvvisz-gákra igen kevesen jutnak el. (És vajon beszélni is tudnak a nyelvviszsgán dokumen-

tált nyelvtudással? Jó kérdés!) Szóval egy kis anyagnyilvántartása, egy kis bérszámfejtése stb. mindenkinek van vagy lesz, de az egész nem áll össze egy rendszerre sehogysém. Az információk úgy monopoli-zálódnak, hogy lokális érdekeket szolgál-nak. A szervezeti egységek nem tudnak, sőt nem is akarnak szót érteni egymással. Ezt még lokális hálózatban is el lehet ér-ni(!), a LAN tehát nem feltétlenül oldja meg a problémákat. A vállalat ügyfelei annak minden részlegétől más és valószínű-leg rossz információkat kapnak. Erre nem lehet piacot alapítani. Marad az össznépi zűrzavar (sok [k]kakas — sok szemét-domb).

Ezért szerintem a Nyugat ma a CO-COM szoftverfejlesztő rendszerekre kirótt normáinak enyhítésével tudna értünk a legtöbbet tenni, ha akarna. Továbbá ki merem jelenteni, hogy ha Amerika félti netán a nekünk kölcsön adott pénzt, akkor ez a lépés a jól felfogott érdekében is áll.

A magyar ábécé

Ha már a nyelvemre vettem a magyar ábécénket, akkor hadd ne menjek el mellette túl röviden. Egyben ostrom alá veszem nyelvtudásainkat. Ha csak egy icipici ékezetnyi engedményt tennének, akkor hozzájuthatnánk a világ lézernyomatatói-nak óriási tömegű betűfontjához. A szük-séges engedmény a következő lenne: a kétvésszős ékezetünk helyett — alternatív-e, divatos szóval — lehessen a tilde (hul-lámvonallal) ékezetet használni. Ezzel a nemzetközi készletből a hosszú ő betű köz-vetlenül kiürítható, az ő betű pedig az ő és a tilde ékezetből kikombinálható, ami a szo-kásos lézernyomatókon nem akadály.

Ezt a megoldást kísérletképpen kipró-báltam egy társszerzős könyvemben, ami-re az kényszerített, hogy az ábrákon a lé-zernyomatató beépített, az IBM PC grafi-kus karaktereit is tartalmazó karakterkész-letét szerettem volna használni. Kinek

még nem mutattam, próbáljon megtalálni egy pár ilyen ő és ő betűt a könyvben (Bakos-Zsadányi: Operációs rendszerek I-II. Bp., SZÁMALK, 1988-1989). Nem lesz túl könnyű, mert nagyon nem feltűnő a csel.

Persze, rajzoltak már az országban ő és ő betűket a szoftverfontokhoz, amiket aztán méregdrágán árulnak, minthogy míves betűkészletek csak magas színvonalú művészi munkával készíthetők, és azt illik jól megfizetni. (A mátrixnyomatok betűkészletének átszervezése többnyire jóval egyszerűbben, és főleg betűnként megoldható, így az átalakítás elég olcsó.) A DTP-kiadványokban a méregdrága fontok alkalmazása elkerülhetetlenül szükséges, de a nem DTP minőségű szövegszerkesztőkkel dolgozókat miért zárjuk ki a közvetlen, ezért gyors és olcsó kiadványelőállítás lehetőségéből? Az iskolában persze azért még a régi ékezetet kell tanítani! (Elnézést kérek, hogy a Unix ürügyen ilyen rögzítmét próbálok eladni, de ha már beletenyéreltem...)

A Unix „égövi jegyei”

Ideje volna a tárgyra térni. Azt azért nem ígérem meg, hogy többé nem fogok elkalandozni, de most nézzük azokat a történeti körülményeket, amelyek a Unix jellemét kialakították. Mindenekelőtt leszögezhetjük, hogy talán a Unix „apukája”, Ken Thompson sem tudná megmondani, hogy 1969-ben pontosan mikor is született gyermeke, így a tulajdonságjegyek megállapításában a csillagizálás aligha segít. A Unix 1969-ben „folyton született” (az angolul tudók talán felismerik a past continuous igeidőt). Tény viszont, hogy szülés közben apukája folytonosan konzultált a későbbi potenciális alkalmazókkal, a Bell Laboratórium szoftveres tojásfejeivel. Ezután nem meglepő, hogy az újszülöttről olyan csodagyerek lett, akit a Bell Lab tudósai kapásból megszerettek. A szó szoros értelmében úgy terjedt, mint a nátha. A kutatók „egymástól kapták el”. Ez az akkor még hiányzó dokumentáció miatt teljesen logikus volt. Ennél fontosabb ok, hogy a Unixot olyan kristálytisza logikával szerelték fel mindjárt az elején, aminek köszönhetően megtanulása szinte gyerekjáték.

Hibákat és gyengeségeket azért mindenben lehet találni, de sajnos az első Unix-változat még sok ilyenfélét szenvedett. Ezek közül egyeseket máig sem hvert ki a csodagyerek. Ilyen kelemetlen tulajdon-

sága, hogy kifejezetten tökéletes hardvert tételez fel maga alatt, rossz hardveren örültségekre hajlamos. Példaként idézünk egy SCO XENIX dokumentációból: „A XENIX-hez csak kiváló minőségű floppy lemezeket használjunk!” Ezt a figyelmeztetést véresen komolyan kell venni. (Szóba sem jöhet 1,2 Mbájtra formázott, eredetileg 360 kbájtig alkalmas lemez!) Más. Egy TPA 11/14-es gépen addig kellett szórakozni a winchester-partíciók tologatásával, míg a root-partíció végére olyan lemezterületre nem került, amelyik teljesen hibátlan volt. Az olyan partíciókban, amelyekben akármilyen kis hiba volt, a swapping rutin rapszodikusan időpontokban meghalt. A hiba okát másfél évig üldözték.

Tojásfej, a rendszergazda

A Unix eme gyenge pontja elengedhetetlenül teszi máig is, hogy környezetében legyen egy „superuser”, azaz egy tojásfej rendszergazda, aki a házard helyzetekből ki tud lábalni a Unixszal. Az ezzel kapcsolatos tevékenységeket azóta több-kevesbé automatizálták, de a Unix ezek mellett is tud olyan helyzeteket teremteni, amelyből elég nehéz kimászni. További jellegzetessége az, amit „anyasági komplexusnak” neveznek: szereti, ha dajkálják. Ezt egyébként meg is hálálja! A superusernek tehát nemcsak baj esetén kell helytállnia, hanem folyton oda kell figyelnie az installációra. Egy kimentés és visszatöltés drámai sebességjavulást okozhat az „elgubancolódott” állományrendszer „kiszimulása” folytatán.

Az eddig mondottak, azt hiszem, jól megvilágítják azt a szlogent, amit a Unix kapcsán örökké emlegetnek: a Unixot programozók írták programozóknak, és ezt azóta is folyton érezteti a környezetével. Fontosabb, hogy az említett jegyek meghatározták terjedésének irányát. Bár a Bell Labon kívül először csak a 6. verziója jutott el, de a célpont tudománytörténeti érdekességű. A Bell Lab az egyetemeknek adta oda először. Érezte, hogy a fertőzést az egyetemi hallgatókon kell kezdeni. A hatás frenetikus volt. Egy sereg ifjú, innovatív elmét lehetett befogni a Unix szekeérébe. Minthogy a Unixot forráskóddal együtt adták oda (ez a jó szokás legalább a kernel driverek szintjéig máig is megvan), az egyetemeken egy sereg új Unix-elem született, amit a Bell Lab aztán felvett a későbbi változatokba.

Kérem, ne rójon meg az olvasó, ha kicsit

lőugrásban haladok, de most akarom elővenni az első Unix-verzió második legújosabb gyengeségét, a hordozhatóság hiányát. Történt ahogy történt, de Ken Thompson a Unix első verzióját egy kis konfigurációjú, mára már éppen nem használt és tudtommal 6 bites, karakteres architektúrájú PDP-7-es miniszámítógépen szülte meg. Akkoriban rendszerszoftvert csak assembly nyelven írtak. Az indok persze elfogadható: hardveroptimumot csak „bitfaragással” lehet elérni.

Nem csoda, hogy a Unix is assembly nyelven készült, de ez a húzás bizony gyorsan visszaütött. Alig két éven belül megjelent a DEC cég 8 bites architektúrájú PDP-11-es gépsorozata, amit nyilvánvalóan az IBM 8 bites bájtiszervezésű gépeinek térhódítása provokált ki. (A 8 bites bájtok megjelenése szerintem Neumann János bináris számrendszere óta a legnagyobb ötlet a számítástechnika történetében.) Elkerülhetetlenné vált a Unix áttele az új hardverre. Még kétévés korát sem érte meg a bébi, és máris halálra volt ítélve. A teljes újjírás réme fenyegetett, hiszen a 6 és a 8 bites architektúra közötti különbség átugrása assembly szinten nyilvánvalóan áthidalhatatlan nehézségeket támaszt.

Ritchi segít: B-ből C

Ken Thompson fentiek miatt érzett lelki bánatán a korábbi ötletadóik egyike, Dennis Ritchi próbált segíteni. Csakhogy ő meg éppen a BCPL-ből kifejlesztett B nyelvbe volt fülüg szerelmes, így érthetően nem lelkesedett az assembly programozásért. Inkább a B módosításait szerette volna kipróbálni valami jó bonyolult példán. Nem tudni pontosan, mint jötték arra az ötletre, hogy a Unix kódjából amit csak lehet, átírnak B-be, de úgy lett. Végül alig maradt valami assemblyben frott rész. A dolgot azonban a B nyelv is megbánta, mert menet közben ki kellett egészíteni „bitfaragó” eszközökkel, hogy hatékonyan tudja pótolni a kegyvesztett assemblyt. Új nevet is adtak neki, az ábcében következő betűt, a C-t. Az átírásból ezért egyszerre két új gyermek született: a portábilis Unix és ikertestvére, a C nyelv. A Unix tehát második gyermekbetegségét olyan tökéletesen kiheverte, hogy máig is ez sikerének legerősebb mozgatórugója. A sikerből semmit sem von le, hogy mint a fentiekből látható, a portábilítás elég botcsinálta módon született meg, és nem valami zseniális elmélet eredménye.

A C nyelv, mint ismeretes, a Unixot

függetlenül is nagy sikert arat a hordozható rendszerszoftverek terén. Sajnos, az erre a célra szánt Ada nyelv ugyanis nem tud elterjedni, mert pénzadó gazdája, az USA Nemzetvédelmi Minisztériuma (DOD) nagyon félti még a nyugati partnerektől is, és monopolizálja a használatát. Ez szerintem olyan elszigetelődés okozhatja, ami gyakorlatilag a halállal fenyegeti. Ráadásul az Ada meglehetősen bonyolult nyelv, ami nem teszi túl vonzóvá az üzleti szoftverben, míg a C ma már olyan jól fel van fegyverezve, hogy univerzálisan használható szinte akármilyen feladathoz. Fordítóprogramjainak hatékonysága olyan jó szokott lenni, hogy maximum 10%-ot lehet veszíteni velük az assemblyhez képest. Ezért tényleg csak akkor éri meg assemblyben programozni, ha annyira szorít a cipő, mint például egy húszmilliószoros ciklus magjában. Jellemző, hogy ma már a hardvert észvesztően kizsigerelni akaró játékszoftver-gyártásban is használják C nyelvet, persze kritikus pontokon kombinálva a hardveroptimumot biztosító assemblyvel. A szoftverfejlesztőknek ez talán elég tanulságul szolgál. Mindenesetre a Unix alatt a természetes programnyelv a C.

A Unix jelenét és jövőjét tehát a hordozhatósága alapozta meg. Új hardverre nagyon könnyű átvinni. Mit kell ezen érteni? Azt, hogy lényegében „csak” egy assemblert kell megírni (amit amúgy is kellene), és át kell cipelni a C-fordítót. Erre nagyjából két út van.

Csizmahúzó és keresztfejlesztő

Az egyik szerint, amelyet csizmahúzó (bootstrap) módszernek nevezünk, fokozatosan közelítjük meg a problémát. Az új gépen az assembler működéséhez mindenesetre átmenetileg kell egy olyan monitor, amely biztosítja az assembler működését. Ezzel az assemblerrel már csinálni lehet egy primitív C-fordítót, amivel viszont csinálni lehet egy okosabb C-fordítót stb., mígnem elérünk a teljes C nyelvet ismerő fordítót. Ezzel már le is lehet fordítani a Unix kernel C nyelvű moduljait. A kernel assembly moduljait át kell írni az új gép assembly nyelvén, figyelembe véve az új hardver architektúráját. Ennek lefordítása után össze lehet állítani egy Unix-kernel, amelynek felügyelete alatt le lehet fordítani a Unix alkalmazói rutinjait (utilityk). Hú! Ezt mondják nagyon könnyűnek. Mihez képest? Természetesen ahhoz képest,

mintha az egészet alapjaiban előlről tervezve kellene megírni assemblyben.

Nézzük a másik módszert. Az átvitelhez felhasználhatjuk magát a Unixot. Ez a keresztfejlesztő módszer. A Unixot születése óta ilyen célokra használják, miért ne tudna segíteni a saját átvitelében is. Egy már működő Unix programfejlesztő környezetben írunk egy olyan assemblyt, ami az új hardver gépi kódjára tud fordítani. Mint hogy a Unix C-fordító általában úgy dolgozik, hogy assembly kódot generálnak, az átvitelhez ki kell cserélni a C-fordító kódgenerátorát úgy, hogy a kereszt-assembler nyelvére kódolja a lefordított C programokat. Most már csak meg kell írni az új gép architektúrájának megfelelő modulokat az új keresztassembly nyelven, és minden Unix-elemet le lehet fordítani, össze lehet szerkeszteni. Ezután megfelelő adathordozón vagy egy számítógép csatornáján átvisszük az új programmodulokat az új gépbe, és fokozatosan felkészítjük a Unixot. Így egy lépésben jutunk el a megoldásig, de láthatóan ez sem sokkal könnyebb, mint az előző.

Akármelyik utat választják, mindegyik olcsóbb, mint vadonatúj operációs rendszer frátit. Ezért a 80-as évek elején az üzleti szoftver is elkezdett beleszeretni a Unixba. Ez a szerelem azonban nagyon anyagi alapokon nyugodott, és nem vett tekintetbe más olyan Unix-jellemzőket, amelyek az ügyviteli környezetben nem bizonyultak eléggé „szobatisztának”. Ha csak a fent elemzett hardverérzékenységet nézzük, már akkor is baj van. Egy kutatókörnyezetben elviselhető, ha néhány adat vagy állomány elvész, vagy meghibásodik. Ha azonban a bérszámfejtés ilyen okok miatt nem készül el határidőre, akkor örül nagy botrány van. A Unixok készítői sem kevésbé hibáztathatók azért, hogy termékeiket olyan környezetre ajánlották, amelyre még nem, vagy talán sohasem lesznek alkalmasak. A Unixok — történetileg kialakult magas IQ-igényük (IQ = intelligenciahányados) miatt — leterspektivikusabb területe máig is a kutasfejlesztés.

Előnyök és hátrányok

E témakör befejezéséért igyekszem tömören összefoglalni a Unix főbb születési jeit:

- az alap-operációsrendszer hordozhatósága jó;
- a Unix-alkalmazások hordozhatósága kiváló;

- kristálytisza, könnyen érthető alapelvek;

- a fejlesztési munkák támogatása kiváló (más kutatási ágakban is, nemcsak a számítástechnikában);

- rendkívül rugalmas, interaktív alkalmazói interfész;

- bonyolult feladatok is megoldható tisztán parancsnyelvi szinten a beépített alkalmazói programozásra (200–4001, verziófüggő), valamint a programközi kommunikációt biztosító csővonalakra (pipeline) támaszkodva (az MS-DOS 2.0 átvette),

- az előző pontban említettek miatt igen kevés hozzáfejlesztési igény van, bár egy konkrét alkalmazási környezetben a Unixot „be kell lakni”, azaz eleinte sok, majd egyre csökkenő számú új szerszámot kell magunknak készíteni, szerencsére többségében a rendelkezésre álló elemekből;

- univerzális, átirányítható az adataramokon alapuló adatkezelés, ami a programok írásánál maximális periféria-függelenséget nyújt (az MS-DOS 2.0 ezt is átvette), kivéve a perifériákat különlegesen kezelő programokat (képernyős editor);

- az alkalmazók és az alkalmazások adatainak szétválasztására egyaránt alkalmas, hierarchikus katalógusokon alapuló, védelmet is biztosító állománykezelő rendszer (az MS-DOS 2.0 ezt is átvette, de a védelmi információkat csak ballasztként cipeli, nem használja semmire);

- az individuális és a csoportmunka támogatása (a komoly fejlesztési feladatok manapság nem egymélyesek, de a csoportmunka-kapcsolat biztosítása mellett, amely főleg állományvédelmi csoportok kialakítását és elektronikus levelezést jelent, szükség van az egyén zavartalan munkájának támogatására is, amelynek eszközei a katalógusok és az állományvédelmi rendszer);

- már a korai időkben rendelkezésre álló dokumentumkezelő eszközök szöveges információk kezelésére (a DTP korzakot körülbelül 15 évvel megelőzve);

- a 6. verziótól kezdve on-line kézikönyv (a washingtoni egyetemen készült, nem a Bell Labban!), amelyet a kezdő és a profi felhasználók egyaránt jól tudtak hasznosítani (a mai help elődje, de ki is lehetett nyomtatni alkalmazókat az egészet nyomdai szedőgépén!).

Ezekkel az alábbi hátrányok állíthatók szembe:

- rossz hibátűrőképesség (javult, de máig is kritikus);

— intenzív gondozási igény (superuser — rendszergazda);

— magas IQ-igény (ragyogó trükkök vannak arra, hogy ne kelljen hagyományos programnyelveken programozni, de ezekhez sok intuíció kell, a millió variációs lehetőség miatt csak kevés szájbarágott típusmegoldást lehet dokumentálni);

— két évtizeddel elmaradt, sororientált interaktív alkalmazói interfész, amelyet 6-8 éve sikerrel támadnak a menüorientált interfészprogramok (a szabványosítás mintha az M.I.T. X-WINDOWS-ának irányába haladna, bár a Microsoft sem tétlenkedik: Presentation ManagerX; az OSF pedig szintén hozott valami újat: Motif);

— a szövegkezelő rendszer és a beépített editorok igen régi technológiai szintet képviselnek, bár kétségtelen, hogy az ed és a vi editorokkal egy Unixot ismerő alkalmazó bármely Unix-verzióban vagy installációban szinte azonnal boldogulni tud, ami a korszerű videoeditorokról nem mindig mondható el (a leg súlyosabb probléma a WYSIWIG (What You See Is What You Get = azt látod a képen, amit nyomtatásban kapsz) hiánya, de ez az elv jóval később is született);

— bár a Unix vezette be a központosít-

tott pufferkezelést (buffer pool — egy program bemehet egy pufferért a „boltba”, ha szüksége van rá), ami rendkívül növelte a hatékonyságot, és amit csak egy évtized múlva követett a hardvergyorstár (cache memory) technológia, viszont mostanság a Unix lemaradóban van az értékes erőforrások mohóbb használatának irányzataitól (például rezidens állománykatalógus a NOVELL serverben, de akár a kis ügyes FASTOPEN az MS-DOS-ban), emiatt hatékonyságban egyre jobban alulmarad;

— a Unix állományrendszere több ok miatt sérülékeny és kedvezőtlen hatásokkal működik (a problémáért megint a 20 évvel ezelőtti „nyomorult” háttértári körülmények hibáztathatók, de ez a lényegen nem változtat, sőt!);

— a Unix állományvédelmi rendszere nem elég erős, amit sok szoftver úgy véd ki, hogy megpróbálja eltakarni a Unixot a felhasználó elől, nehogy illegális akciókba kezdhessen;

— a Unix eredeti adatkezelése rendkívül primitív, amin az alkalmazói rendszereknek kell javítaniuk, de azóta kiváló C-könyvtárak (CISAM) állnak rendelkezésre erre a célra;

— a Unix eredeti változatában a prog-

ramok nem tudnak osztott tármezőket használni, pedig a Unix a virtuális tárkezelési technikát ósídók óta alkalmazza, ami feltétlen alkalmas ilyen adatkezelés megszervezésére (bezzeg az OS/2-ben!).

Nem biztos, hogy az összes lényeges előnyt és hátrányt sikerült felsorolni, és nem biztos, hogy az itt vázoltakkal minden unixos egyetért. Továbbá a nem Bell Laboratóriák, valamint az újabb fejlesztések éppen a gyenge pontokon igyekeznek segíteni, több-kevesebb sikerrel. Mindenesetre, akik a cikket olyan szándékkal olvassák, hogy belőle éppen közelgő Unixszal kapcsolatos döntéshez szeretnének segítséget kapni, azok már ennyiből is nagyon sok következtetést le tudnak vonni. A Unix azonban mára a kezdeti egy-két emberévről kapacitással készített szoftverből talán sok százezer emberévről munkát magába sűrítő terméké fejlődött át. A történetileg kialakult jellemzők megismerése ezért fontos, de önmagában nem feltétlenül elégséges egy alapos döntéshez. Nem mondom tehát az olvasónak, hogy itt akár abba is hagyhatja a cikk olvasását, mert a későbbiekben még tartogatok meglepetéseket, hiszen ott szerepel a valószínűsíthető ellenpólusok elemzése.

Zsadányi Pál

Új szolgáltatásokkal
új helyen várja kedves ügyfeleit az

ISKOLASZÁMÍTÓGÉP SZERVIZ

IBM és Commodore számítógépek javítása
közútlevek és magánszemélyek részére

Éves átalánydíjas szerződések
rendkívül kedvező feltételekkel.

Egyéb szolgáltatások:

- C16 bővítése 64 kb-igra,
- magyar ékezetes karakterkészlet beépítése
- játékprogramok eladása és vétele
- Tectronix oszcilloszkóp előnyös áron

A javítás ideje alatt
szükség szerint cseregépet biztosítunk.

Címünk: 1088 Budapest, Rákóczi út 25.

Tel.: 1381-121



RÁVISZ
Fény- és
Elektrotechnikai
Szakcsoport

**SZÜNETMENTES
ÁRAMFORRÁSOK**
ORSZÁGOS SZERVIZE

1076 Budapest, Thököly út 30.

Telefon: 12-21-673, 12-24-275
Telex: 22-65-29

Semmi sem tökéletes ezen a világon

Sajnálatos módon az 1989/9. számban megjelent **Vírusirtó** című cikkem 1. programjába hiba csúszott. Alulról a 7. sorban „R.DX:=R.DX OR 1F;” helyett R.CX:=R.CX OR \$1F; kell! Hibámra mentségül én is csak másokra tudok mutogatni. A Turbo-Pascal című könyvben éppen ez a DOS-funkció (és még egy-kettő) hibás.

Saját kárán tanul igazán az ember. Én is levontam a megfelelő tanulságot: amit egy könyv ír, az még nem szentírás. Egy szent és sérthetetlen van: a SZÁMÍTÓGÉP. Ennek működése a mérvadó.

„Csak az nem hibázik, aki nem is dolgozik” jelszóval a könyvkiadók „indokoltan” hibáznak. Hibáik viszont jelentősen megkeseríthetik a programozók (és más, könyvből tanuló, dolgozó egyének) életét. Sorstársaim segítségére ezért most közreadom néhány könyv hibajegyzékét és a hibák valószínű javítását.

**R. Baumgartner—S. Hansjakob—W. Praxl:
Turbo-Pascal. Elmélet és gyakorlat. Novotrade**

67. oldal. \$30-as DOS funkció:
A verziót és az alverziót felcserélték. Helyesen:
verzio:=Lo(r.ax);
alverzio:=Hi(r.ax);
68. oldal. \$36-os DOS funkció:
r.dx:=meghajtó;
71. oldal. \$3C-s DOS funkció:
Az ismertetés végéről mintha valami lemaradt volna:
handle:=r.ax;
71. oldal. \$3E-s DOS funkció:
Az első sor nem „r.ax:=\$3D00 or típus”, hanem:
r.ax:=\$3e00;
72. oldal. \$3F-es DOS funkció:
A trükk nélküli megoldásba hiba csúszott:
Reset(Filvar,1);
Kiegészítés a \$40-es DOS funkcióhoz:
Ha r.cx=0, akkor a DOS a jelenlegi fájlmutató értékére állítja be a fájl hosszát.
73. oldal. \$42-es DOS funkció:
Az „_MsDos(r);” sor után kimaradt a feltétel (talán takarékoságból): if r.flags and 1=1 then
Még egy kis hozzáfűznievaló is volna e funkcióhoz:
a DOS-hívás után r.dx:=\$1000+r.ax tartalmazza a fájlmutató értékét.
76. oldal. \$4A-s DOS funkció:
A trükk nélküli megoldást összekeverték egy másik funkcióéval. Ennek a funkciónak nincs igazi megfelelője.
79. oldal. \$4F-es DOS funkció:
Biztos, ami biztos, tanolja meg a programozó, hogy elsőre semmi sem megy, ezért itt is van egy apró, de kellemetlen hiba: ut:=+ # 1; helyett jobb, ha a következőt használjuk: ut:=ut+ # 0;
80. oldal. \$57-es DOS funkció:
Itt a vége a DOS ismertetésének, gondolhatta a nyomdász vagy az író vagy a fordító (ki tudja most már eldönteni, hogy ki a felelős), ideje lazítani, lehet valami érdekesebb hibát ejteni: az egész funkcióismertetés során a DX és a CX regisztert felcserélték. Ennek kijavítását, gondolom, már az olvasókra bízta. Hát most itt az alkalom, tegyük meg!
88. oldal. INSERT eljárás:
Az első két paramétert felcserélték. Biztos azt gondolták, hogy a programozó néhány perc (esetleg óra) alatt rájön a turpisságra.

**Grochmann—Eichler: A 8086/88-as mikroprocesszor
Technika és programozás. Data Becker — Novotrade**

- Az apróbb nyomdahibáktól, mint például: MNI az NMI helyett (19. oldal) vagy CU a CPU helyett (27. oldal) akár el is tekint-hetünk.
211. oldal teteje:
A LOOPcc utasítás az SI regisztert egyáltalán nem változtatja meg.
- Ugyanaz az oldal, alulról a 4. bekezdés, miszerint AX-be valamilyen adat kerül, sületlenség (az egész bekezdést ki kell húzni). Az ugrási tartomány pedig a LOOP utasítás utáni bajttól számítva előre 127, vissza 128 bajt.
- Ugyanígy a 191. oldalon: rövid JMP-pal —127 és +127 bajt az ugrási táv. Itt még valami olyasmit is írnak, hogy 16 bites eltolással csak —32666 és 32667 tartományban lehet ugrálni. Ezt is ki lehet húzni, ugyanis a szegmensben belüli bárhova ugorhatunk.
50. oldal. A PUSHA, POPA utasítások az AX, CX, DX, BX, SP, BP, SI, DI regisztereket ebben a sorrendben kezelik.
- A könyv több helyen is írja: „A D irányjelző bitet a SED és a CLD utasításokkal meg lehet változtatni.” Az STD utasítást valószínűleg elírták SED-re (na de 7 helyen!): 162., 181. és 210. oldal alja, 211., 230., 257. oldal, 270. oldal alulról a 4. bekezdés.
- 231., 237. oldal: a POP és PUSH utasítások operandusa 8088-as mikroprocesszornál is lehet effektív cím. Nem tudom, honnan vették, hogy a 8088-as ilyen rokkant.
- A 244. oldal ábráján a jobb oldali ROL helyett ROR utasítás kell. Logikus: jobbra jobb (Right) kell!

**Pethő Ádám: IBM PC/XT.
A ROM BIOS és ami mögötte van. SZÁMALK**

Ez a könyv nem a hagyományos szedési technikával készült, ennek köszönhetően a könyvben valóban az van, amit a szerző leírt. Örömmel üdvözlöm az ilyen könyveket. Ezekben már csak azok a hibák lehetnek, amiket a szerző ejtett. Bár még ez is távol van a tökéletességtől, a hibák lehetősége mégis csökken. Pethő Ádám és munkatársai tiszteletre legyen mondva, hogy a magyar átlaghoz képest könyvsorozatuk szinte hibátlan (na de azért semmi sem tökéletes ezen a világon). Nagyon sajnálom hogy az ilyen embereknek sem adják meg a lehetőséget, hogy könyvüket gép mellett írják, így óhatatlanul kerül hiba a műbe. Ennek ellenére e könyvben mindössze egy komolyabb hibát véltém eddig felfedezni:

A 240. oldalon a 61H port bitkiosztásában a 3., 2. bit felcserélődött (3. bit= Multiplex p. 62H). A magnó motorját kapcsoló 2. bitet pedig az XT nem használja a leírások szerint. Valójában a turbo XT turbo-állapota közötti kapcsolásban játszik szerepet, de hogy hogyan, azt nem sikerült kiderítenem.

A TECHHELP szerint a 61H port 5. bitje nem a DRAM, hanem az I/O ellenőrzését vezérlő (hogy melyiknek van igaza, nem tudom eldönteni).

Azt, hogy a könyvek többségében sok hiba van, tényként el kell fogadni. Mit tehetünk mi, mit tehet a könyvkiadó? Mi maximum minden könyvből vett információt ellenőrizhetünk (ha tudjuk), de a helyes megoldást nem biztos, hogy mindenki ki tudja találni. Ezért azt ajánlanám, hogy a kiadók és a szerzők vegyék a fáradságot, teszteljék a már kész kiadványt, és a javításokat tudassák a felhasználókkal. Ha máshogy nem megy, esetleg e magazin hasábjain.

Hornák Zoltán

Programozási fogások és

melléfogások



Az 1989/6. számban azt a babonát próbáltam eloszlatni, mely szerint a GOTO és GOSUB utasításoknál „a számítógép a megadott sorszámot mindig a program elejétől kezdve keresi”. Úgy látszik, a mondanivalóm félreérthető volt, mert két hasonló tartalmú hozzászólás érkezett a szerkesztőséghez. A komlói Szalma Sándor levelében fetteje ki enyhemtől eltérő nézeteit. Vele egy levelváltás során sikerült a félreértések egy részét tisztázni. A másik — kétoldalnyi, írógéppel készült — írás inkább vitacikknek látszik, de szerzőjének kitérőből semmit sem sikerült megtudnom, ugyanis nevét elfelejtette feltüntetni írásán. (Az ismeretlen szerzőre a továbbiakban N. N. jelöléssel hivatkozom.)

A kifogásolt cikkben csupán a fentebb említett állítást kívántam cáfolni. Mindkét hozzászóló azt olvasta ki belőle, hogy a szubrutinokat nem érdemes a program elejére tenni. Ilyesmit egyáltalán nem írtam le, csupán a szóban forgó babona szellemében megvalósított *létszámológos gyorsítás* gyakori előfordulását kifogásoltam. Változtatlan állított, hogy egy programot csak úgy, öncélúan GOTO-val kezdeni, még a strukturált programozás apostolai szerint sem esztétikus. Négy magyar nyelvű, számítástechnikával foglalkozó lapban összesen 28 olyan programot néztem át, melynek elején GOTO-val átugrott szubrutin volt (esetenként több is), de egyetlen olyat sem találtam, melynél ez a módszer valódi időnyereséti szolgált volna. Egy olyan esetet találtokzom, melyben az említett módszer „kötelező”, a *Mikroszámítógép Magazin* 1985/1. számában közölt *Overlay-technika* című cikkben.

Mint említettem, a szóban forgó tévhitelt a *PC Mikrovilág* 1986. májusi számában találtokzom először. Bizonyára nem véletlen, hogy a *Mikrovilág* és a *Commodore újság* hasábjain csak ennél később megjelent számokban találtam annak szellemében készült programot, tíz-tíz darabot. A *BIT-LET*-ben talált három és a *Mikroszámítógép Magazin*ban közölt öt program többsége az említett *PC Mikrovilágnál* korábban jelent meg. A *Magazin* 1986. májusi számában, a *Strukturált programtervezés* című cikk keretében napvilágot látott programban egyértelműen gyorsítási céllal került a program elejére a GOTO-val átléptetett szubrutin, míg az októberi számban közölt *Tártartalom a képernyőn* szerzője a kísérő szövegben jelzi gyorsítási szándékát. Mindkettőre egy későbbi alkalommal kívánok visszatérni.

Írásában N. N. felvilágosít, hogy „... nem általános a Commodore gépek eljárása. A Spectrum és a TVC (mint az a ROM listából kiderül) mindig a program elején kezd a keresést”. Közösönm a tájékoztatást. Bevallom, hogy a Spectrum ROM listáján nem sikerült kismernem magam, de a próbatúras N. N. állítását igazolta, éppúgy, mint a TVC ROM listája. Az utóbbira vonatkozóan egy ellentétes — szóbeli — in-

formációt is kaptam, melyet csak egy próbatúrtatás tudna igazolni, amire sajnos egyelőre nincs lehetőségem. A HT ROM leírásából az derül ki, hogy ott is a Commodore gépekéhez hasonló a keresés. Ezt a tényt N. N. elhallgatja, nem tudom, véletlenül-e.

A cikkhez tartozó program is félreértést okozott. N. N. ezt írta róla: „A programban két, baki' is található: egyrészt a 180-as sor nem a program elején van. Ha ugyanis a hívott rutint tényleg a program elejére tennék, például a 110-es sorba, akkor a végrehajtás ideje kb. 250 TI-egységgel csökkenne, s máris kevesebb lenne, mint a 260-as sor hívásakor. A másik *hiba*, ami talán még fontosabb: a 260-as sor majdnem közvetlenül a szubrutint hívó 220-as sor mögött van, így az interpreter *gyorsan megtalálja*”. Ez nem baki, éppen ez volt a program célja: kézzelfoghatóan bizonyítani, hogy a keresés *nem mindig* a program elején kezdődik. Az N. N. érvel között szereplő Spectrumon a magasabb sorszámú sor eléréséhez *mindig* több idő kell. Abban viszont egyetérték mindkét hozzászólóval, hogy a program elejére tett szubrutinokat könnyebb *bárhonnan* gyorsabban elérhetővé tenni.

Szalma Sándor első leveléből idézek: „... a most végére helyezett szubrutint át kellene lépni a folytatáshoz, ami a GOTO-kat ugyancsak felszaporítaná! Aztán ha ezt a szubrutint meg (a jelenleg nem létező) 500 és 1500-as sorból is meg kell hívni, már kénytelen előlőrl végigkeresni a programost a gép”. Próbatúrtatást is végzett programom néhány sorral megtoldott változatával. Közölte a mérési adatokat a 300-as sorból kiadott GOSUB utasítások esetére is. Íme: a 180-as sor hívásával 3657 TI-egység adódott, a 260-asával 3893. Figyelmen kívül hagyta azt, hogy a jelenleg a 260-as sorban lévő szubrutint egy nagyobb számú sorba kellett volna tenni, például az 1510-esbe.

Ísmét N. N. írásából: „A szerző apró inkorrektségeket nevez, hogy a (...) kötet általa citált cikk programjában a program elején lévő sorszám kétféle, így a gép gyorsabban kiértékeli, mint a végén található háromjegyű sorszámot, ezért gyorsabban megtalálja a keresett sort. Véleményem szerint ez nem inkorrektség, hanem az interpreter egy tulajdonságának öltetes felhasználása. Az ebből fakadó előny még nagyobb is lehet, ha a program elején egyjegyű sorszámokat használunk a programok vé-

gén előforduló négy-ötjegyű sorszámok helyett”. Ugyanez a téma Szalma Sándor második levelében is előkerült: „... a sorszámkonverzió tovább tart nagyobb számok esetén. (...) Én tényleg sok szubrutint használok, egy 50–60 blokkos programnál csak a magas sorszámok eltávolításának helyigénye sem elhanyagolható”. Így igaz. Az 1. listán látható program 1652 TI-egység alatt futott le, az 1-es sorszámot 10 000-re változtatva ez 1993 TI-re nőtt. Ez több, mint 20 százalékos eltérés, ami nem elhanyagolható.

Ennek ellenére továbbra is inkorrektnek tartom az idézett könyv programjának fogását. Részint, mert ott sorszámkonverzióról szó sem volt, részint mert a mérések azonos feltételek mellett kell elvégezni.

Csak a cikk megjelenése után fedeztem fel a programomban egy bakit, amely arra figyelmeztet, hogy a rossz példa ragadós. A mérés szempontjából az értékdő utasítások feleslegesek, sőt erősen torzítják a mérési eredményt. Korábbi rögzítéseimben a szubrutinokban csak RETURN utasítások voltak.

Befejezésül néhány példa az indokolatlanul előre tett szubrutinokra.

Az októberi számban idéztem a *Mikrovilág* tavalyi 14. számából a *Mikromágia* program részleteit. Az elején hét megjegyzés soron át jutunk a GOTO 100 utasításig, mely két szubrutint ugrik át. Ezek közül az első kiír a képernyőre egy üzenetet, azután egy testzőleges billentyű leütésére vár. Fontos, hogy elől legyen?

A 2. listán a *PC Mikrovilág* 1987/23—24. számában megjelent *Síolimpia* első sorait láthatjuk. Az 1-es sorban itt is egy GOTO-val átugrott szubrutint látunk. Hívása előtt a C paraméter értéket kap, ennek megfelelő számú PRINT utasítást hajt végre. Két PRINT között *lassítás céljából* egy 50 lépéses *üres ciklus* fut le. Hogy van ez? Gyorsítunk, hogy lassítsassunk?

N. N. így fejezi be írását: „Mindezeket összefoglalva egyetértek a kritizált mondattal: a *szubrutinokat* a program elejére tegyük (még az említett Commodore gépeken is)!” Így, egyszerűen, felszólító módban. De az imént bemutatott két, és a lapokban látott további számos példa láttán hadd kérdezzem meg: Ezért? Minek?

Barna László

```

0 goto 60000
1 return
60000 ti$="000000"
60001 for=i:1to10000
60002 gosubl
60003 next
60004 print ti

```

1. lista

```

0 FORT=1T03:R(T)=40:NEXT:GOTO10
1 FORT=1T0C:PRINT:FORI=1T050:NEXTI, T:RETURN
10 ...

```

2. lista

Blokk törlés, összevonás, szétválasztás

BASIC programsorok nagyobb csoportját egyszerre törölni gépi kódú rutinnal szokás. A művelet azonban közvetlenül BASIC sorokkal is elvégezhető. Mivel sok más géppel ellentétben a miénk egy programsor hossza 256-nál nagyobb lehet, átírhatjuk a kezdő sor hosszát (a programterületen ezt meghatározó címek tartalmát) úgy, hogy az az egész tömb hosszának feleljen meg. Ehhez a következő két sort kell beszúrunk a tömb elé, illetve után:

```
xx LET q=PEEK 23637+256*PEEK 23638-57 GO TO VAL
"00yy"
yy LET w=PEEK 23637+256*PEEK 23638:LET x=INT
((w-q)/256):
POKE q-2, w-q-256*x: POKE q-1,x: STOP
```

Ha például a 90–220-as sorokat akarjuk törölni, az xx sort 89-es, ha yy sort 221-es sorszámmal írhatjuk be. Fontos, hogy az xx sor hossza nem változhat, ezért példánkban a GO TO VAL "0221" kifejezéssel kell végződnie. Ide tehát az ugrás sorszámát mindig négykarakteres formában, a szükséges vezető nullákkal együtt kell beírni. Végül a GO TO 89 parancs kiadása után a 89 szám beadásával végezhetjük el a törlést.

A két beírt utasítás a NEXTLINE rendszerváltozóból olvassa ki a számítható szükséges címeket. A törlés a beírt sorokra is kiterjed, így ha ismételt törlést végzünk, ezeket félretéve célszerű tartalékolni.

Érdemes megnézni, hogy a törlés beadása előtti listázás esetén a mozgató programkurzor átugorja a tömbnek az xx sort követő sorait. Ha pedig az xx sort lehijük (EDIT), majd visszaküldjük, a tömb az xx sor kivételével törlődik.

Programszerkesztés közben gyakran jó lenne, ha tudnánk programsorokat egyesíteni és bontani a BETA BASIC (JOIN és SPLIT) betöltése nélkül is. Nos, ezek a műveletek BASIC sorokkal is elvégezhetők, bár kissé körülményesebben. Az alábbi programsorokat MERGE-dzsel célszerű a programunkhoz tölteni:

9919 REM ÖSSZEVONÓ

```
9921 LET q=PEEK 23637+256*PEEK 23638:LET w=PEEK
(q+2)+256*PEEK (q+3): LET x=PEEK
(q+w+6)+256*PEEK
(q+w+7): LET y=INT ((w+x+4)/256): POKE q+2,
w+x+4-256*y: POKE q+3, y: FOR z=3 TO 7: POKE
q+w+z, 58: NEXT z: STOP
```

9929 REM SZÉTBONTÓ

```
9931 LET q=PEEK 23637+256*PEEK 23638: GO SUB
9941: STOP
```

```
9941 LET w=PEEK (q+2)+256*PEEK (q+3): LET
y=q+w+2:
FOR z=q+4 TO y: IF PEEK z=226 THEN LET x=z: LET
z=y
```

```
9942 NEXT z: LET z=x-q-4: LET w=w-z-4: LET y=INT
(z/256): POKE q+2, z-256*y
```

```
9943 POKE q+3, y: LET y=INT (w/256): POKE x+2,
w-256*y:
POKE x+3, y: POKE x-1,13
```

```
9944 POKE x, PEEK q+(PEEK (q+1)=255): POKE x+1,
(PEEK (q+1)+1)*(PEEK (q+1) <> 255): RETURN
```

A furcsa sorszámozás azt a célt szolgálja, hogy a betöltött sorok elférjenek az eredetiek mellett, de ezt célszerű előzetesen ellenőrizni.

Az összevonáshoz a 9921-es sort át kell helyezni az első összevonandó sor elé, majd lefuttatni. Ekkor a beszűrt sort követő két sor eggyé válik. A kapcsolódás helyén négy felesleges kettőspont marad, ezek azonban a működést nem zavarják. A beszűrt sort azonban feltétlenül törölni kell.

Most is a blokk törléshez már közölt elvet alkalmazzuk, és a NEXTLINE rendszerváltozót olvassuk ki. Kiolvassuk a programterület megfelelő címeiről az egyesítendő sorok hosszát is, és betöltjük a kiszámított közös hosszértéket. A feleslegessé váló sorvégeket (13) és a második sor számát és hosszát tartalmazó részeket pedig 58-cal (kettőspont) írjuk felül.

A szétválasztáshoz először a STOP kulcszót kell beírni a kettévágás helyére, és utána három kettőspontot (STOP ::), majd a sor elé át kell helyezni a 9931-es programsort; természetesen a hívott helyen kell lennie a 9941–9944-es sorszámú szubrutinnak is. A beszűrt sor futtatása után a bontásból kapott új sor sorszáma eggyel lesz nagyobb az eredetinél. Most sem szabad elfeledkezni a beszűrt sor törléséről.

A kezdőcím kiolvasása után (NEXTLINE) egy ciklus megkeresi a STOP kulcszó címét is. Kiolvasva a sor eredeti hosszát, meghatározható és betölthető a két új hosszérték, az új sor kezdőcíméből számított címekre pedig betölthető a sorvégeket és az új sor hosszértéke és sorszáma. A beírásuk ezek számára biztosítottunk helyet.

Baracs Tibor

```
0000 STOP : REM BLOKKTÖRLŐ
0001 LET q=PEEK 23637+256*PEEK 2
0002 LET w=PEEK (q+2)+256*PEEK
0003 LET x=PEEK (q+w+6)+256*PEEK
0004 LET y=INT ((w-q)/256): POKE
0005 q-2, w-q-256*x: POKE q-1,x: STO
```

```
0006 REM ÖSSZEVONÓ
0007 LET q=PEEK 23637+256*PEEK 2
0008 LET w=PEEK (q+2)+256*PEEK
0009 LET x=PEEK (q+w+6)+256*PEEK
0010 LET y=INT ((w+x+4)/256): POKE
0011 q+2, w+x+4-256*y: POKE
0012 q+3, y: FOR z=3 TO 7: POKE
0013 q+w+z, 58: NEXT z: STOP
```

```
0014 REM SZÉTBONTÓ
0015 LET q=PEEK 23637+256*PEEK 2
0016 GO SUB 9941: STOP
```

```
0017 LET w=PEEK (q+2)+256*PEEK (
0018 q+3): LET y=q+w+2: FOR z=q+4 TO
0019 y: IF PEEK z=226 THEN LET x=z: L
0020 ET z=y
```

```
0021 NEXT z: LET z=x-q-4: LET w=
0022 w-z-4: LET y=INT (z/256): POKE
0023 q+2, z-256*y
```

```
0024 POKE q+3, y: LET y=INT (w/25
0025 6): POKE x+2, w-256*y: POKE
0026 x+3, y
0027 POKE x-1,13
```

```
0028 POKE x, PEEK q+(PEEK (q+1)=2
0029 55): POKE x+1, (PEEK (q+1)+1)+(PE
0030 EK (q+1) <> 255): RETURN
```

Területek és funkciók

Az itt közölt alapképzésű, kétnyelvű Enterprise 128K-ra vonatkozóan. A fel nem tüntetett címek funkcióit még nem ismerjük.

A BFFF-ABD1 területek konstans címeket tartalmaznak, így ezek angol EP-k esetén is helytállóak. A RAM- és a bővítő ROM-szegmensekre vonatkozó táblázat a konfigurációtól függően változó méretű. Ezt követi a bővítő ROM-ok által igényelt terület — ha van ilyen —, majd a beépített eszközök táblázata, valamint a csatorna-RAM.

A 24 bites távoli cím a következő adatstruktúrát jelenti: a legkisebb című bájton a Z80-cím alacsony helyiértékű bájta (LSB), a következő bájton a magas helyiértékű bájta (MSB) található. A harmadik, legmagasabb című bájta a szegmensszámot tartalmazza, melyen a Z80-címnek kell lennie.

A csatornaleíró címén egy olyan első lapú, távoli címért értünk, mely a megcímzett leíró legmagasabb című bájtnálal egyvel nagyobb címre mutat.

A címek csökkenő címsorrendben vannak, mivel az EXOS is így tölti fel a rendszerszegmenst.

A változóterületet nem konstans címekekkel jelezzük, hanem a ROM-bővítők által elfoglalt memóriaterület legalsó bájtnak címéhez képesti eltolással, mivel a bővítő ROM is így fér hozzá. A fentebb említett cím a most ismertetésre kerülő rendszerben A706, ezért a +04B2 cím itt A706+04B2-1, azaz ABB8-at jelent.

- BFFF-ED:** Az EXOS általános rendszerváltozói (lásd EXOS 2.1 leírás, 44. o.)
- BFEC-C5:** Az EXOS rendszerhívással kezelhető EXOS rendszerváltozók a 27h — 00 kódsorrendben. A BFC6 című bájta üres, ui. a hozzá tartozó FLAG_SOFT _IRQ változó tényleges címe BFF2!
- BFC3-C4:** A REM1 és REM2 rendszerváltozó állítások használt segédrutin címét tartalmazza az 1. szegmens értelmezve. A magnókezelő inicializáló rutinja E9C1-re állítja be. Az itt lévő rutin a megfelelő rendszerváltozó aktuális értékét beírja a PORTB5 rendszerváltozóba és a /WRO (0B5h) portra
- BF0C-C3:** A RAM-bővítő lánc első elemének 24 bites első lapú címe. A pointer a belépési pontra mutat
- BFBD-BF:** Az eszközeleíró lánc első elemének 24 bites címe. A pointer a leíró DD-TYPE mezőjére mutat
- BFBA-BC:** A csatornaleíró lánc első elemének 24 bites címe
- BFB6-B9:** ?? (mind 00)
- BFB4-B5:** A csatorna-RAM területének kezdőcíme (egy eltolási érték a BFBD címhez képest)
- BFB2-B3:** ?? (0000)
- BFA5-B1:** A három legutóbb használt csatorna kódja és leírójuk 24 bites címe, valamint egy lezáró 00 a BFB1 címen. A leg-alacsonyabb bájton a csatornakód egyvel megnövelt értéke áll, majd a hosszú cím. A csatornakezelő funkciók esetén az EXOS a BFA5 címtől kezdve végignézi ezt a táblát, hogy szerepel-e benne a keresett csatornakód. A tábla végét 00 csatornakód jelzi. Az EXOS így próbálja lerövidíteni a keresést a láncban
- BF9E-A4:** A rendszerállapotot leíró bájtok, ahogy azt a 14h kódú EXOS-funkció adja vissza
- BF9C-9D:** A rendszerben lévő ROM-bővítők listájának kezdőcíme
- BF9A-9B:** A rendszerben lévő RAM-szegmensek listájának kezdőcíme
- BF99:** A RAM-bővítőknek kiutalt szegmensek listájának első eleme
- BF97-98:** A ROM-bővítők RAM-területei után következő első bájta címe
- BF95-96:** A kernel által lefoglalt területre következő bájta címe
- BF93-94:** Az eszközpuffer területének végcíme
- BF91-92:** Az EXOS-határ értéke

- BF8F-90:** A felhasználói határ értéke
- BF84-8E:** A csatornapuffer-kijelölési funkció területe:
- BF84:** Az igényelt puffer típusa: =01 — video-, =F0 — eszköz-puffer
- BF85:** A puffert igénylő csatorna száma +1
- BF86/8:** Az igénylő eszköz periférialeírójának 24 bites címe
- BF89:** A pufferigénylés jelzője: OFFH = nem volt még igénylés, 00 = érvénytelen hívás (van már puffer)
- BFBA:** ? a legkisebb működő videosegmens száma
- BF8B/C:** átmeneti adattároló, a bővítő-lefoglalás is használja
- BFBD/E:** A videocsatornák által elfoglalt RAM utolsó bájtnak Nick címe
- BF81-84:** A csatornaláncban történő kereséskor használt puffer. A periféria-alprogramokat hívó szubrutin itt várja a csatornaleíró 24 bites első lapú címét
- BF80:** A csatornaszám, ahonnan a modult be kell tölteni. (Nincs egyvel megnövelve)
- BF7F:** A relokálható betöltés futási lapja (gyakorlatilag a cím-MSB)
- BF7E:** ? ROM-program típusszám, /kontroll-summa. Értéke A7h. Képzése: Ach XOR (01:FFD8-tól 19h bájta össze-XORozva)
- BF7C-7D:** A kernel itt tárolja a HL regiszterpár belépés előtti értékét
- BF7A-7B:** A kernel itt tárolja az SP belépés előtti értékét
- BF79:** Kerneljelzők: b0=1 — a ROM-bővítők már jelöltek ki RAM-ot, b7=1 — a kernel futás közben van (innen állapítja meg, ha rekurzívvan fut)
- BF78:** Kerneljelzők: b0=1 — a ROM-bővítők már inicializálódtak
- BF72-77:** Az EXOS által kezelt óra
- BF54-71:** Fájlnev-puffer: eszköz-/fájlnev megadásakor itt képződik a kernel által kezelhető fájlnev
- BF36-53:** Eszköznev-puffer: eszköz-/fájlnev megadásakor itt képződik a kernel által kezelhető eszköznev a határoló „:” és egységszám nélkül
- BF19-35:** Az alapértelmezésbeli eszköznev a határoló „:” és egységszám nélkül
- BF18:** Az alapértelmezésbeli egységszám
- BF08-17:** A betöltés alatt álló modul fejresze
- BEB8-DF:** A státuszsor memóriabeli képe
- BD16-B7:** Az angol billentyűzet-meghajtó munkaterülete:
- BD16-F:** A legutóljára leolvasott billentyűmátrix törölt shiftkekkel
- BD20-9:** Az előző billentyűmátrixból képzett segédadatok
- BD2A-EA9:** A funkcióbillentyű sztringek 24 — 24 bájton
- BEAA:** Ha < > 0 — BEAE tartalma érvényes (van éppen lenyomott billentyű)
- BEAB:** Ha = OFFH — HOLD állapot van érvényben
- BEAC:** A shift-billentyűk állapota. b1=SHIFT, b2=CTRL, b3=ALT
- BEAD:** Ha = OFFH — van nyitott billentyűzetcsatorna
- BEAE:** Az éppen lenyomott billentyű kódja
- BEAF:** Ha = OFFH — LOCK-állapot van
- BEB0/1:** A billentyűzetpuffer legfelső elemének címe
- BEB2:** A pufferben tárolt billentyűkódok száma
- BEB3:** Számláló a billentyűkésletelés megvalósításához
- BEB4:** Számláló az automatikus ismétlés megvalósításához
- BEB5/6:** Annak a billentyűsornak a címe, amelyben megváltozott állapotú billentyű van
- BEB7:** A megváltozott állapotú mátrixsor képe
- BC80-97:** A soros/hálózati meghajtó munkaterülete:
- BC80/1:** A CRC-regiszter
- BC82:** A soros pufferben a következő bájtra mutató ofsztet

| | | | |
|------------|---|------------|---|
| BC83: | A soros pufferben lévő bajtók száma | +483: | Az éppen lenyomott billentyű kódja |
| BC84/5: | Várakozásciklus-számláló a megfelelő baudértékhez | +484: | =FF — LOCK-állapot van érvényben |
| BC86/7: | ?? | +485/6: | A billentyűzetpuffer legfelső elemének címe (lefelé bővül!) |
| BC88-97: | A soros/hálózati meghajtó főpuffere | +487: | A pufferben tárolt kódok száma |
| BBA1-BC: | A videovezérlő munkaterülete | +488/9: | A billentyűzet-késleltetés és ismétlés funkció számlálója |
| B900-B1F: | Az alapértelmezésbelli LPT | +48A/B: | Annak a billentyűzetmátrix sort reprezentáló bajtnak a címe, amelyben megváltozott állapotú billentyű van |
| B480-8FF: | A CH128 módú karaktergenerátor | +48C: | A fenti rendszerválasztó által megcímezett bajt másolata |
| B380: | A hangkezelő escape-szekvenciáinak beolvasásánál használt jelző =00 — szabad karakter olvasódik be, > 00 — maga az ESC karakter olvasódik be, < 00 — a még beolvasandó escape-argumentum bajtjai számának kettes komplementese | +48D-96: | Az aktuális billentyűzetmátrix |
| B37E-7F: | A hangkezelő escape-szekvencia argumentum következő bajtljának címe | +497-A0: | Az előző mátrixállapotból képzett segédválasztók |
| B37C-7D: | A hangkezelő escape-szekvenciát lekezelő rutin címe | +A1: | ?? Ez a BCh kódú EXOS-rendszerválasztó |
| B2F4: | =00 — törölt burkoló | +4A2: | =00 — a billentyűzetkódok változtatás nélkül kerülnek kivételre; =C1 — a következő kód ABB3-ban tárolt című táblázatból képződik; =C2 — ugyanaz, mint az előző, csak a táblázat pointerre ABB5 |
| B2F3: | A hangkezelő használatához az első lapon szükséges szegmens | +4A3: | A szövegszerkesztő program kapcsolója: =00, a német nyelvű, < > 00, az angol nyelvű aktivizálódik „WP” parancs hatására |
| B2E3-F2: | Az A0-AF portok aktuális értékei | +4A4: | ?? FF |
| B28D-E2: | A magnókezelő munkaterülete | +4A5/6: | Shift nélküli billentyűdekódoló táblázat címe |
| B28D: | A 0038 cím eredeti tartalma | +4A7/8: | Shiftes billentyűdekódoló táblázat címe |
| B28E: | Az aktuálisan működtetett távvezérlő | +4A9/A: | CTRL-es billentyűdekódoló táblázat címe |
| B28F: | Olvasásra megnyitott fájl csatornakódja +1 vagy 00, ha nincs | +4AB/C: | ALT-os billentyűdekódoló táblázat címe |
| B290: | Írásra megnyitott fájl csatornakódja +1 vagy 00, ha nincs | +4AD/E: | Ha ABA8 tartalma C1, akkor a vezérlő az e címen tárolt táblázatban megkeresi a leolvasott billentyűkódot, és ha megtalálta, a rákövetkező bajtot adja ki billentyűkódként |
| B291: | Az input fájl távvezérlője: 00 — REM1, más — REM2 | +4AF/B0: | Ugyanaz, mint az előző, csak a C2 kiterjesztőkódra |
| B292-AE: | Az input fájl neve | +4B1: | A shiftes billentyűdekódoláshoz az angol betűbillentyűkön kívül betűként kezelendő billentyűintervallum alsó határa. Itt a megfelelő kisbetű kódját kell elhelyezni, vagy 00-t, ha nincs rá igény. (BRD módban itt 7Bh, azaz „ä” szerepel |
| B2AF: | Az output fájl távvezérlője: 00 — REM1, más — REM2 | +4B2: | Ugyanaz, mint az előző, csak a felső határra. Ha az előző 00, akkor tartalma érdektelen. (BRD módban itt 7Eh, azaz „ü” van) |
| B2B0-CC: | Az output fájl neve | +43E-47E: | A német billentyűzetvezérlő puffere és munkaterülete |
| B2CD/E: | A pufferben lévő bajtók száma | +2FF-+33D: | A VSAVE bővítőparancs munkaterülete |
| B2CF/D0: | A fej-szelet mérete kimentésnél | +2C0-+2FE: | A VLOAD bővítőparancs munkaterülete |
| B2D1/2: | A puffer legfelső elemének címe | +007-+2BF: | A VDUMP bővítőparancs munkaterülete |
| B2D3/4: | ?? | +007: | A nyomtatandó videocsatorna X-irányú mérete karakterekben |
| B2D5: | EOF-jelző: < > 00 — a fájlból nem lehet olvasni | +008: | Az Y-irányú méret karakterekben |
| B2D6: | Az olvasott szelet típusa: 00 — adat, FF — fej | +009: | A színmód |
| B2D7: | Adat a csatornalezárásnak: 00 — van még adat a pufferben | +00A: | A videomód |
| B2D8: | A védelmi bajt értéke: 00 — védelem bekapcsolva | +00B/C: | A nyomtatandó logikai képpontsorok száma (A logikai sor 4 tényleges pixelsornak felel meg, ui. a rutin ennyit kódol egyszerre a 8 tús bitkép-mód miatt) |
| B2D9: | 00 — nem volt hiba, FF — CRC-hiba | +00D: | Az előző adat maradék része: azt határozza meg, hogy az utolsó, tört logikai sor hány tényleges sorból áll (0-3) |
| B2DA/B: | Az SP regiszterpár átmeneti tárolója | +00E/F: | Az aktuálisan dekódolandó sorok 1. lapú címe a videome-móriában |
| B2DC: | A kimentés jelszintje (a 0A8h portra kikerülő érték) | +010: | A nyomtatócsatorna száma |
| B2DD-E0: | Konstansok a megfelelő felvételi sebességhez (a 0A0h portra) | +011: | A videolap csatornaszáma |
| B2E1: | A státuszsor eredeti COL2 palettaszíne | +012/3: | Egy logikai sor dekódolása után annak mérete bajtokban |
| B2E2: | A státuszsor eredeti COL3 palettaszíne | +014/5: | Tartalma mindig 1Bh, 4Bh, azaz ESC, „K” — Epson-kompatibilis nyomtatón 8 tús normál sűrűségű bitkép-módot állít be |
| B230-51: | A különleges státuszüzenet (ez látszik, ha az ST-FLAG tartalma 2Ah) | +016/7: | A ténylegesen kinyomtatásra kerülő grafikus adatbajtok száma |
| B217-2F: | A kernel segédrutinjai: ezeken keresztül hívja meg azokat a rutinokat, melyek 3. lapú címekkel más szegmensen futnak (például rendszerbővítők, az 1. szegmensben lévő EXOS-rutinok), valamint itt van az a rutin, amelyen keresztül a vezérlés elhagyja a kernelt | +018/2BF: | A pixelsorok nyomtatódattára történő konvertálására használt terület. Itt helyezkednek el a kinyomtatandó grafikus bitkép-bajtok a dekódolás után |
| ACE7-B216: | Az EXOS saját rendszerverme | +006: | ?? 00 |
| ABD1-D2: | A csatornapuffer-kijelölés EXOS-funkció ide menti el az SP-1 | +005: | ?? FF |
| ABCA-D0: | A rendszerben lévő RAM-szegmensek listája (a nulláslap szegmens kivételével) | +001-04: | A 90h kódú rendszerválasztó kikapcsolásakor a 00B8-00BB területre átmásolandó kódreszlet. Ez helyezi üzemen kívül a német nyelvű üzeneteket megvalósító rutint. (Alapértelmezésbeli értéke: 3E 01 CB 25, azaz LD A,01:SLA L) |
| ABB9-C9: | A ROM-bővítők leíró listája. Minden bővítőre a következő: a bővítő RAM-területének 24 bites 2. lapú címe, valamint a bővítő szegmense. A táblázatot 4 db 00 bajt zárja | +000: | A 90h kódú EXOS-rendszerválasztó: =00 — német nyelvű üzenetek |
| +47F-B2: | A kétnyelvű gépek billentyűzetvezérlőjének rendszerválasztói: | -0AA/-001: | Az eszközeleírók lánc táblázat formájában |
| +47F: | =FF — ABB9-ben tárolt érték érvényes | -0AB-től: | A csatornaleírók és -pufferek területe |
| +480: | =FF — HOLD-állapot van érvényben | | |
| +481: | A shift-billentyűk állapotai, b3 — ALT, b2 — CTRL, b1 — SHIFT | | |
| +482: | =FF — van megnyitott billentyűzetcsatorna | | |

EXOS gyakorlatok

Szaniszló Zoltán cikkét úgyszólván egy kicsit pihentettük, mert információk szerint az EXOS használata lassan terjed az Enterprise-osok táborában. Több érdeklődő levél után mégis úgy döntöttünk, hogy az EXOS használatára folyamatosan visszatérünk.

Az első feladat: grafikus lapok görgetése jobbra. Ha az Enterprise-on általunk definiált videolap kezdőcíme vagyunk kíváncsiak, az első pillanatban nehézkés tűnő feladat előtt állunk. Az EP ugyanis minden csatornapuffer létrehozásával, illetve megszüntetésével járó utasítás után mozgatja a RAM megfelelő területeit. Rendelkezésünkre áll egy gépi kódban alkalmazható ún. EXOS-funkcióhívás (1. lista), de ezt csak gyakorlati programozóknak ajánlatos alkalmazniuk. Én a kezdők számára is könnyen érthető BASIC programot mutatok be (2. lista).

A program megértését néhány mondatos magyarázattal szeretném elősegíteni. Az EP 255. lapján helyezkedik el a rendszerszegmens. Itt található az LP_POINTER nevű változó a 49 140–49 141-es címen. Ez itt kiolvasható érték a Video Sorparaméter Tábla Z80-as kezdőcíme. Mi az a sorparaméter-tábla? A képernyő minden egyes sorához tartozik egy-egy 16 bájthosszú adattábla, amely az adott sor videojelmezőre vonatkozóan tartalmaz adatokat (például videomód, video színmód, a margók értékei, palettaszínek stb.). Közöttük szerepel az adott sor bal felső bájtiának kezdőcíme is. Ezt az értéket a 16-os egységen belül az 5–6. bájttal adja meg. Figyelem! A 16-os egységek közül az első a STATUS sor paramétereit tartalmazza. Tehát az LP_POINTER által megadott értékhez 20-t hozzáadva találjuk meg a legelső sor kezdő bájtiának alsó, és 21-et hozzáadva a felső helyértékű cím értékét (lásd az ábrát).

Külön magyarázatra szorul a program 250-es sora. Az EP FC-től FF-ig terjedő RAM-szegmensét VIDEORAM-nak nevezzük. Ezt a Nick chip közvetlenül címezi, függetlenül attól, hogy be vannak-e lapozva a Z80-as lappszegmensek valamelyikére. Mivel az előző sorban kapott „CIM” nevű változó a 255. RAM-szegmensre mutat, de ez a második

Z80 lappszegmensre van belapozva, értékből egy szegmensnyit, azaz 16 384-et le kell vonnunk. Ekkor kapjuk a tényleges Z80-as címet.

Futtassuk le a begépel programot! Ha figyelmesen szemléljük, észrevehetjük, hogy nem pixelcscroll látunk, hanem bájtonkénti léptetéssel van dolgunk. Ennek magyarázatát adja az, hogy az egy bájttal megjeleníthető négy képernyőpixel színe a bájton belül kevert sorrendben van kódolva, így bitenkénti léptetésnél gördülő szinkavakádban lenne részünk.

Most javítsuk át a 110-es sorban található 36h értéket 77h-ra. Futtatásakor azt tapasztal-

juk, hogy az ablak jobb oldalán kilépő karakterek a bal oldalra újra megjelennek.

Az EP ATTRIBUTUM-videolapja hasonlít leginkább a Spectrum képernyőkezelésének elvéhez. Itt is külön tárolódik az ún. bittérkép és külön az egyes bájtkhoz tartozó PAPER és IKN színérték. A 3. lista tartalmazza az ATTR-videolap görgetését. Itt már lehetőség nyílt a bitenkénti léptetésre. Figyelem! ATTR-videolapok esetén a sorparaméter-tábla 16-os egységeiben a bittérkép kezdőcíme a 7–8. bájttal. Ezért a megváltozott érték a program 220-as sorában. Az eltolási érték 22, illetve 23 lett. Ennél a videomódnál is lehetőségünk van egy picit változtatással más eredmény elérésére.

Irájuk át ennek a programnak is a 110-es sorában két bájttal értéket. Lengyen F0h helyett 08h, CB, 1Eh helyett CB, 33h. Ezzel elérjük, hogy futtatásakor karakterhelyen belül gördül ki a képernyőtartalom.

A két gépi kódú rutint bármilyen méretű videolapra alkalmazhatjuk a listákon jelzett változók átírása után. A program német billentyűzetű gépen készült.

Szaniszló Zoltán

2. lista

```

100 ALLOCATE 36
110 GDE scroll=HEX/"11,0b,0e,19,
    e9,e1,06,3c,c5,c9,06,2d,45,e1,
    7b,c5,e1,3b,0f,1e,ad,b0,23,3e,
    0f,2b,2b,1b,c1,1f,ed,d1,c1,1f,
    e9,c9"/
120 SET VIDEO MODE 1
130 SET VIDEO COLOR 1
140 SET VIDEO X 3F
150 SET VIDEO Y 5
160 OPEN #1:"VIDEO:"
170 SET #1:PALETTE 7,8,7,4
180 DISPLAY #1:AT 1 FROM 1 TO 5
190 PRINT #1:18,179,
200 PRINT #1:"Enterprise scroll rutin"
210 PLOT #1:18,179,
220 PRINT #1:"INTROGILLIAN software 1988"
230 LET spt=HEX/49140/*HEX/49141/*256
240 LET CIM=HEX/spt+3C/*HEX/spt+21/*256
250 LET CIM=CIM-16384
260 WAIT 2
270 CALL USR/scroll,CIM/
280 END
    
```

A fenti program Assembler listája:

```

11 08 #A LD DE,2699 ; 10h*Y-1
19 ADD HL,DE
85 PUSH HL
D1 POP DE
#6 3C LD B,6F ; 20k
C5 PUSH BC
D5 PUSH DE
#6 2D LD B,45 ; 90k
D5 PUSH DE
K1 POP HL
2B DEC HL
C5 PUSH BC
#1 3B 0F LD B0,59 ; 2k*1
1A LD A,/DE/
8D 06 LDH
23 INC HL
36 0F LD /HL,/E ;Est szerdó 77h-ra
2B DEC HL
C1 POP BC
1B DEC DE
13 POP BC
2F ED DWG 237
D1 POP DE
C1 POP BC
1F E5 DWG 229
C9 RET
    
```

1. lista

```

62 01 LD A,1 ; videolap ent.szám
#6 03 LD B,3 ; funkciószám
F7 0F EXOS 11 ; funkcióhívás
        eredetny 30 registerpárban
    
```

3. lista

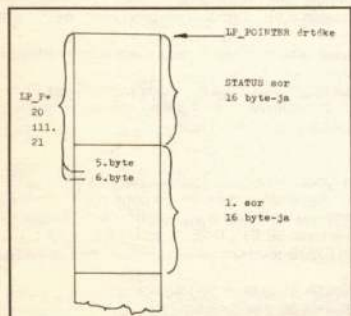
```

100 ALLOCATE 26
110 GDE scroll=HEX/"e5,d1,06,0f,
    c5,66,6b,06,2d,c5,37,3c,06,1e,
    eb,1e,23,1f,fb,c1,1f,c3,c1,1f,
    eb,c9"/
120 SET VIDEO MODE 15
130 SET VIDEO X 3F
140 SET VIDEO Y 5
150 OPEN #1:"VIDEO:"
160 SET #1:PALETTE 7,8,7,4
170 DISPLAY #1:AT 1 FROM 1 TO 5
180 PRINT #1:AT 1,1:"ENTERPRISE ATTR SCROLL"
190 PRINT #1:AT 5,1:"INTROGILLIAN SOFTWARE 1988"
200 LET spt=HEX/49140/*HEX/49141/*256
210 LET CIM=HEX/spt+22/*HEX/spt+23/*256
220 LET CIM=CIM-16384
230 WAIT 2
240 CALL USR/scroll,CIM/
250 END
    
```

A program Assembler listája:

```

85 PUSH HL
D1 POP DE
#6 0F LD B,24F ;0kX /ide /sr #6 00 -at/
C5 PUSH BC
62 LD H,D
6B LD L,E
#6 2D LD B,45 ;90k
C5 PUSH BC
37 SCF
3F CDF
#6 1E LD B,3F ; 1 Y
CB 1E RR/HL ; ide /srj CB 3E -t
23 INC/HL/
1F FB DWG 201
C1 POP BC
1F F3 DWG 243
C1 POP BC
1F EB DWG 235
C9 RET
    
```



Az IBM és az Enterprise

Hosszas finomítgatások után elkészült egy anyagnyilvántartó program Enterprise-ra, melyet már a Magazin is közölt. Ez a program arra volt hivatva, hogy lehetővé tegye számunkra a festékanyagok mennyiségi nyilvántartását. Csofáló érzés volt birtokolni egy olyan saját programot, amely mintegy 2000 adatot tárol, kezel.

Be kell persze vallani, hogy egyelőre csak kb. 800 adat nyilvántartása volt a cél. A program elkészülte után hatalmas elánnal neki is láttam, hogy a memóriát a jól bevált módon begépeléssel feltöltsem a szükséges adatokkal. Körülbelül 100 adat begépelése után — természetesen bizony sok rongtort adatot is begépeltem — kezdtem nem úgy lelkesedni, mint az elején. A türelmem teljesen elfogyott, és arra gondoltam, hogy ez az őskor.

Persze az élethez szerencse is kell, s bezabonyosodott, hogy az néha rám is mosolyog. Mint a mesében, egy szép napon a kezembe került egy hajlékonylemez, amely a vállalatomnál üzemben lévő IBM-kiön számlaprogramjához szükséges új termékek adatait tartalmazta. Eléggé el nem ítélhető módon kitétem ezt a lemezt egy orvkísérletnek és hazavittém egy napra, azzal a nem titkolt szándékkal, hogy megpróbálom az Enterprise-on kilistáztatni.

Igy is volt. Otthon behelyezve a lemezt a meghajtóba, kiadtam a :dir utasítást, és nagy örömmel olvastam a képernyőről:

```
„FESTEKEK.TXT stb...”
```

Kíváncsi voltam arra, hogy ki tudom-e olvasatni BASIC-ben a fájl tartalmát. Ismét szerencsém volt. Futtatam az alábbi programocskát:

```
10 OPEN 106:"FESTEK.TXT" ACCESS INPUT
20 LINE INPUT 106:As
30 PRINT As
40 CLOSE 106
```

s megjelent a képernyőn a következő:

```
"517703MODELL GIPSZ 1/1 DB"
```

Még láttam utána számokat is felvilanni, de azok eltűntek. A PRINT LEN As utasítással meggyőződtem arról, hogy az As ténylegesen hosszabb, mint a képernyőn megjelenített szöveg. Ez azonban nem keserített el, mert amire szükségem volt, az a rendelkezésemre látszott állni.

A lemezt természetesen visszajuttattam a számítástechnikai osztályunknak, de egyben meg is kértem őket, elmesélve tapasztalataimat, hogy az árutörzsből listázzák ki egy lemezen, kísérleti célból, a festékanyagokat. Dicséretükre legyen mondvá, másnap már birtokomban volt a kívánt anyag. Azt is meglelté érdekemben a számítástechnikai osztály, hogy a lemezre csak a vállalati kódot és az anyagszervezést mentette ki.

Otthon azonnal a géphez rohantam, és behelyeztem a lemezt a meghajtóba, s az alábbiak szerint elkezdtem a lemeztartalom, azaz a fájlartalom kilistázását.

```
10 OPEN 106:"FESTEK.TXT" ACCESS INPUT
20 LINE INPUT 106:As
30 PRINT As
40 GOTO 20
```

A CLOSE 106 lezárásra azért nem volt szükség, mert az adatok elfogytával a gép hibaüzenettel úgyis leáll. Nem kellett sokat listáznom ahhoz, hogy lássam, olyan adatok is vannak a törzsadatok között, amelyekre nincs szükségem. Ennek figyelembevételével írtam meg az alábbi kis programot, melynek lényege, hogy feltölti a szabad szegmenseket az általam kiválasztott anyagok karakterenkénti kódszámával. Így persze sztringből is feltölthető volna.

Végül egy tanulság: kétheti unalmas munkától mentett meg, hogy bedőltém egy pletykának.

Papp Miklós

```
100 TEXT
110 OPEN #106:"FESTEKEK.txt" ACCESS INPUT
120 LET A=250:LET B=1
130 LINE INPUT #106:A#
140 FOR C=1 TO 31
150 SPOKE A,B,ORD(A#(C))
160 LET B=B+1
170 IF B>16368 THEN LET B=1:LET A=A+1
180 NEXT C
190 PRINT AT 10,10:B
200 GOTO 130
210 CLOSE #106
```

Mi a manó?

Kivételesen nem információkat és tanácsokat közlünk ebben a rovatocskában, hanem apró, trükkös programokat. A listákat begépelve érdekes dolgokat tapasztalhatnak a kezdő Enterprise-osok, a profik pedig összehasonlíthatják a közölt programokat saját megoldásaikkal. Kérjük olvasóinkat, hogy fejtsek meg e programokat, és küldjenek hasonlókat szerkesztőségünkbe.

```
10 GRAPHICS ATTRIBUTE
12 SET #101:VIDEO Y 40
15 SET BEAM OFF
16 PLOT 0,710;
21 FOR II=0 TO 255
31 SET #101:INK II
32 SET #101:PAPER 0
40 PRINT #101:"Mi a manó?"
50 NEXT II
```

```
90 !*****
100 ! Villogó STATUS megszakításban
105 !*****
110 FOR I=48897 TO 48905
120 READ A
130 POKE I,A
140 NEXT
150 SPOKE 255,49134,191
160 SPOKE 255,49133,0
170 DATA 237,95,50,9,185,50,11,185,201
```

```
1100 !*****
110 ! Villogó sorok megszakításban
120 !*****
130 FOR I=48897 TO 48905
140 READ A
150 POKE I,A
160 NEXT
170 SPOKE 255,49134,191
180 SPOKE 255,49133,0
190 DATA 237,95,50,24,185,50,136,186,2
```

01

```
90 ! STATUS szinek
100 TEXT
110 INPUT AT 11,7,PROMPT "STATUS szine
? (0-255)":A
120 TEXT
130 SPOKE 255,14601,A:SPOKE 255,14603,
A
140 LET A#="Nyomd le a SPACE billentyu
t"
150 FOR I=1 TO LEN(A#)
160 SPOKE 255,16061+I,ORD(A#(I)):SPO
KE 255,16062+I,142
165 SOUND PITCH 56,DURATION 1
170 NEXT
180 SPOKE 255,16062+LEN(A#),32
190 GET A#
200 IF A#<>" " THEN 190
```

SZOFTVER

Egy csokor batch-fájl

Batch-fájlokat szövegszerkesztőkkel írhatunk, vagy közvetlenül DOS-utasításokként, begépelve a COPY CON név.BAT utasítás kiadása után, ahol a név.BAT a DOS szabályainak betartásával szabadon megválasztható. Ha a COPY utasítást használjuk, akkor a bevitel végét egy fájlvége karakterrel (Ctrl/Z) jelezni kell, ami a Z billentyű lenyomását jelenti, miközben a Ctrl billentyűt is lenyomva tartjuk (helyette használható az F6 billentyű).

Kapcsolat a felhasználóval

A batch-fájlok legnagyobb hibája általában, hogy szinte semmilyen kapcsolat nincs a felhasználóval. A PAUSE utasítás az egyetlen, amellyel a batch-fájl megállítható, és billentyűzetről történő bevitelre várakozik — amire aztán a batch-fájl folytatódni fog, függetlenül attól, hogy melyik gombot nyomtuk meg. Ezt a parancsot gyakran használjuk a batchből való kilépésre, a PAUSE parancs el után a Ctrl/C beadásával (1. lista).

A batch-fájl gombnyomásra várakozik, mielőtt a FORMAT utasításra lépne. De a Ctrl/C használata esetén a batch-fájlból kilépünk, és visszatérünk a DOS-ba.

A Ctrl/C használata nem túl szerencsés, és talán jobb, ha a felhasználó közvetlenül a „Terminate batch job (Y/N)?” üzenetet kapja, és a Ctrl/C-t a bemeneti pufferbe magától a batch-fájltól veszi át. Ilyen módon a kilépés az „Y” billentyű megnyomásával történik a CTRL/C helyett. Az egész nem annyira bonyolult, mint az első pillanatban látszik. A fenti példát a 2. listán látható formában módosíthatjuk.

A második ECHO parancsot használjuk arra, hogy a Ctrl/C karaktert a TIME parancsba vigye, amelyik viszont kiküldi a kimenetet a NUL-ba (e fogalmakat később részletesebben érintjük). A NUL egy fura dolog, mert úgy viselkedik, mint a COM1 vagy LPT1 kimenet, de az odaküldött kimenettel nem csinál semmit. A TIME egyike annak a kevés belső DOS-programnak, amelyik a billentyűzetről kéri a bemenetet.

Mikor a Ctrl/C-t megkapja, a TIME utasítás törölődik, és a DOS „Terminate batch job Y/N?” üzenetet küld: vajon kilépünk a teljes batch-fájlból, vagy folytatjuk a következő parancsot? A Ctrl/C-t úgy írhatjuk be az ECHO utasításba, hogy az Alt billentyű megnyomása alatt a 3-as számot gépeljük be (ez a Ctrl/C ASCII kódja).

Sok programozó számára a batch-fájlok használata aránylag világos, és nem okoz gondot. Ha mégis úgy érzi valaki, hogy egy, az útkijelölésnél, a directory-váltásnál és a programindításnál bonyolultabb feladatot kellene a batch-fájlokkal megoldani, bizonyára örömmel fogad néhány ötletes példát. Sok olyan lehetőséget bemutatunk, amelyek közvetlenül nem olvashatók ki a DOS-kézikönyvekből, és sok olyat is, amelyek csak használat közben gyűjtögethetők össze.

Menükészítés

A batch-fájlok használatokor szükség van egy olyan segédprogramra (utility), amelyek lehetővé teszi, hogy a felhasználó a billentyűzeten keresztül oldhasson meg különböző feladatokat. Számos ilyen programot találunk a Public Domain szoftverek között, melyek különböző neveken ismeretesek, mint például ASK, REPLY, CHOICE.

Ezek mindegyike érvényes kódot vár a billentyűzetről, és a segédprogramba küld egy kilépési kódot. A tényleges kód attól függ, hogy melyik billentyűt nyomtuk meg. Ez a kilépési kód aztán tesztelhető a batch nyelv IF ERRORLEVEL parancsának felhasználásával.

Azoknak, akiknek nem áll rendelkezésre ilyen program, bemutatunk egy nagyon egyszerűt, amelyet a DOS-programokhoz adott DEBUG-gal hozhatunk létre (3. lista).

A DEBUG-gal írt KEY.COM program a BIOS billentyű funkcióját használja fel. Mikor egy billentyűt lenyűnk, a program leolvassa azt (INT 16H), az ERRORLEVEL névű batch-fájlból vizsgálható belső DOS-változóban tárolja, és a futást befejezi.

A 4. listán látható példa egy egyszerű menükialakítási lehetőséget mutat be. Ebben a példában az ECHO parancsok a menü megjelenítésére szolgálnak, az ECHO parancs után egy teljes üres sort kapunk a képernyőn.

Az ECHO ON használata azt eredményezi, hogy a DOS kijelzi a képernyőn az ECHO aktuális állapotát (ON vagy OFF). Az ECHO OFF parancs letiltja, hogy a batch-utasítások megjelenjenek a képernyőn, és csak az első ECHO OFF utasítás lesz látható. (Azok a felhasználók, akiknek a gépe DOS 3.3 vagy magasabb verziójú operációs rendszer alatt

fut, az ECHO utasítás első egy @ karaktert téve, annak kiírását is megakadályozhatják.)

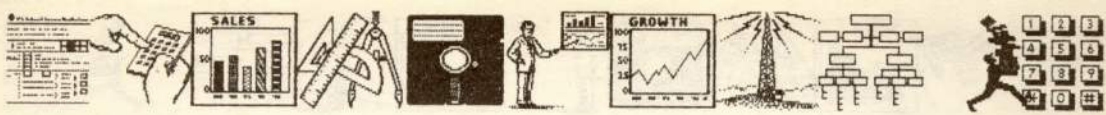
Ha elérjük a KEY parancsot, a program billentyűnyomásra várakozik, de csak az „1” és „2” és „ESC” kisbetűket fogadja el, és az ERRORLEVEL belső DOS-változóknak 49, 50 vagy 27 értéket adva, a batch-fájl végrehajtása folytatódik. A következő sorban az ERRORLEVEL-értékek tesztelése történik, és a GOTO parancs segítségével a batch-fájl egy másik részére ugorhatunk.

A batch-programozást segítő programok közül igen jól használható a Microsoft Macro Assembler WHAT.EXE utility programja. Ez a kis győngyszem sok egyéb információt is nyújt a lemezterületről, a fájl méretéről, a printer státuszáról, a memóriáról, a koprocészor vagy az ANSI meghajtó meglétéről. Teszi ezt a DOS rendszerváltozók tárolásával a fenti néven (WHAT). Innen már egyszerű a dolog: a batch-fájlon belül tesztelhetjük ezeket a változókat, és elvégeztethetjük a szükséges műveleteket.

A WHAT utility különböző lehetőségei a HELP fájlban található; ezek az információk kinyomathatók, ha a WHAT-ot paraméterek nélkül hívjuk meg. Ez a segédprogram a fentiekkel sokkal bonyolultabb menüprogram kialakítására alkalmas, ahogy ezt az 5. listán látható példa kapcsán bemutatjuk.

A batch-fájlok minden egyes sort elolvassanak a lemezről, aminek következtében lassú lesz a végrehajtás. Sok szöveg esetén — hogy a kiírás minél gyorsabb legyen — okosabb a TYPE parancsot használni a menü megjelenítésére a képernyőn. Egy tipikus menüfájl (MENU.TXT), amelyet a fenti példában láttunk, ilyen formájú lehet:

- A) Word processor (Szövegfeldolgozó)
- B) Spreadsheet (Számolótableta)
- C) PC Tools



D) Print text file (Szövegfájl-kirratás)
 E) Exit menu (Kilépés a menüből)

A % karakterrel zárójelezett DOS-rendszerváltozókat a DOS behelyettesíti az adott változó — a batch-fájl által meghatározott — aktuális értékével. Ez egyszerűen szemléltethető: ha az ECHO OFF utasítást kitöröljük a fenti példából, megfigyelhető az IF kifejezések kiértékelésének módja.

Az egyenlőségvizsgálatot követő parancsot végrehajtjuk, ha a kiértékelés eredménye igaz. Ha a parancs egy programnév, akkor azt a programot futtatjuk le, majd, ha befejeződött, a batch-fájl folytatódik.

A példában a menü ismételt kijelzésre kerül, mert ebben a példában a GOTO ciklust így helyeztük el. Ha az utasítás egy másik batch-fájl neve, a vezérlés átkerül erre a batch-fájltra a BASIC CHAIN parancshoz hasonlóan. Ha az eredeti batch-fájlról van szükségünk az eljárás folytatására, a második batch-fájl utolsó utasításaként be kell írni az első batch-fájl nevét. Egy másfajta megoldásban a második batch-fájlt szubrutinként is kezeljük a CALL utasítás segítségével:

IF "%WHAT%"=="A" CALL WP

Ezek után, ha a WP batch-fájl végrehajtása befejeződött, a Menu-batch-fájl folytatódik a következő IF sorral. Sajnos a 3.3-nál korábbi verziózámú DOS-ban a CALL parancs nem szerepel, és az azonos funkció megvalósításához szükség van arra, hogy az IF parancsba helyezett COMMAND parancsot hívjuk meg:

IF "%WHAT%"=="A" COMMAND/C WP

A /C azt jelenti, hogy a COMMAND eljárás megszűnik, és az előző szintre tér vissza a WP végrehajtása után. Valójában itt egy másik program fut (COMMAND.COM), és nem egy batch-fájl, ezért az eredeti batch-fájl akkor folytatódik, ha a WP futása befejeződött.

Természetesen, ennek ellenére, ha nem állnak rendelkezésre ezek a segédprogramok, mégis készíthetünk a fentiekhez hasonló menüt, és a batch-fájl visszatérhet a DOS-ba, amikor is egy betűt és Enter billentyűt leütve egy betűcsoportot ad ki, mint például: A.BAT. A menükijelző utasítást az A.BAT végén is be kell írunk, hogy ismét kiírja a menüt, ha már lefutott a batch-fájl.

DOS-változók használata

A DOS változói nagyon hatékonyak lehetnek. Például kell egy batch-fájl, amely betölt egy meghajtó vagy „Terminate and Stay Resident” programot (TSR); mielőtt valamely más programot töltené be, fontos lehet annak biztosítása, hogy csak egyszer legyen betöltve. Az indító batchbe a következő kódot illeszthetjük be:

```
IF "%LOADED%"=="YES" GOTO LABEL1
...beállító kód...
SET LOADED = YES
:LABEL1
...az indítási kód többi része...
```

A többi DOS-változóknak is hasonlóan jó hasznát vehetjük. Gyakran használjuk a batch-fájlokat a kiválasztott könyvtárak váltására, és

```
ECHO OFF
CLS
ECHO ** Az 'A' meghajtóban levő lemez formázása **
ECHO Ctrl-C esetén kilépés, vagy...
PAUSE
FORMAT A:
```

1. lista

```
ECHO OFF
CLS
ECHO ** Az 'A' meghajtóban levő lemez formázása **
ECHO ^C : TIME >NUL
FORMAT A:
```

2. lista

| | |
|--|--|
| <pre>A>DEBUG -A XXXX:0100 MOV AH,0 XXXX:0102 INT 16H XXXX:0104 MOV AH,4C XXXX:0106 INT 21H XXXX:0108 -RCX CX 0000 :B -NKEY.COM -W Writing 000B bytes -Q A></pre> | <p>A vastágbetűs részeket kell begépelni. A begépelte szövegeket az ENTER billentyűvel zárjuk.</p> |
|--|--|

3. lista

```
ECHO OFF
CLS
ECHO EGYSZERŰ MENÜ
ECHO *****
ECHO.
ECHO 1 > DOS VERZIO
ECHO 2 > KÖTET NEV
ECHO ESC > KILÉPES
ECHO.
:GETKEY
KEY
IF ERRORLEVEL 49 IF NOT ERRORLEVEL 50 GOTO TIPUS
IF ERRORLEVEL 50 IF NOT ERRORLEVEL 51 GOTO VOLUME
IF ERRORLEVEL 27 IF NOT ERRORLEVEL 28 GOTO EXIT
GOTO GETKEY
:TIPUS
VER
GOTO EXIT
:VOLUME
VOL
:EXIT
```

4. lista

a programbetöltés előtt a megfelelő keresési útvonal beállítására. De a programból kilépünk, lehet, hogy az útvonal és a könyvtár rosszul van beállítva. Bár be tudnánk szűrni két sort a batch-fájlból — hogy állítsa át a keresési útvonalat és a könyvtárat —, de az lenne a kellemes, ha az eredeti könyvtár és útvonal visszaállítódna, sőt bármire átváltható lenne. Az útvonal átválítása nem túl nehéz. A mi példánkban az alábbi sort kell beszűrni:

SET OLDPATH = %PATH%
 mielőtt az útvonalváltás történik, és a SET PATH = %OLDPATH% sort, mielőtt a batch éppen befejeződné.

Az aktuális könyvtár tárolása már nem annyira egyszerű feladat, mivel ezeket a DOS nem mint változókat tárolja. Természetesen használhatjuk a WHAT segédprogramot, a következő sor beszűrésével:

WHAT Y



```

@ECHO OFF
:START
CLS
TYPE MENU.TXT
ECHO.
SET WHAT=
WHAT C"Your selection=>" ABCDE
IF "%WHAT%"=="A" WP
IF "%WHAT%"=="B" SPREAD
IF "%WHAT%"=="C" PCTOOLS
IF "%WHAT%"=="D" GOTO PRINT
IF "%WHAT%"=="E" GOTO EXIT
GOTO START
:PRINT
WHAT P
IF "%WHAT%"=="1" GOTO OK
ECHO
ECHO ERROR - check printer
PAUSE GOTO START
:OK
ECHO.
SET WHAT=
WHAT S "Name or file to print?"
IF NOT "%WHAT%"==" " TYPE %WHAT% > PRN
GOTO START
:EXIT
ECHO
  
```

5. lista

```

@ECHO OFF
SET OLDPATH = %PATH%
WHAT YE
SET DRIVE = %WHAT%
WHAT Y
SET DIRECTORY = %WHAT%
.. a batch fájl többi része..
SET PATH = %OLDPATH%
%DRIVE%
CD%DIRECTORY%
  
```

a mi batch-fájlunkban a következő sorral együtt:

```

CD %WHAT%
  
```

mely megőrzi a könyvtárunkat egy későbbi időpontig.

Ha a WHAT-ot már más célra használtuk a batch-fájlon belül, a visszahozott sztringet egy másik változóba kell átmásolni. Például az útvonal, a meghajtott és könyvtár tárolása a 6. listán látható módon lehetséges.

Ha nem áll rendelkezésünkre a WHAT.EXE, egy kicsivel több ravaszág kell az aktuá-

6. lista

```

@ECHO OFF
COPY PREFIX.CD OLDDIR.BAT
REM ITT FÜZZÜK HOZZÁ AZ AKTUALIS DIRECTORY-T
CD >> OLDDIR.BAT
...a batch fájl többi része...
CALL OLD-DIR.BAT
  
```

7. lista

```

@ECHO OFF
IF "%1" == "" GOTO ERROR
SET FILESPEC = %2
IF "%2" == ""
SET FILESPEC = *.*
FOR %A IN (%FILESPEC%) DO FIND %1 %A
GOTO EXIT
:ERROR
ECHO ^GUsage: %0 "string" [fájlspec:default = *.*]
:EXIT
  
```

8. lista

lis directory tárolására. Először is a következő kis fájlt kell létrehozni:

```

COPY CON PREFIX.CD
CD < Ctrl-Z >
  
```

Az aláhúzás jelenti az üres karaktert, azaz a léptető billentyűt (Space-bar) nyomjuk meg, nem pedig az aláhúzás karaktert. Nagyon fontos, hogy a CD_ után ne nyomjuk meg az Entert, ezzel szemben nyomjuk meg a Ctrl-Z kombinációt, a fájl vége karaktert. A fentieket a mi esetünkben a 7. lista szerint hasznosíthatjuk.

Ahogy korábban említettük, ha valaki a 3.3-nál alacsonyabb verziósámú DOS-t használ, akkor a CALL-t a COMMAND/C-vel kell majd helyettesíteni. Ha ez a sor a batch-fájl végén van, elég csak az OLDDIR-t használni, a CALL vagy COMMAND nélkül fűzzük a fájlhoz. Ezenfelül gondoskodni kell arról, hogy az OLDDIR.BAT-nak szükséges korrekt könyvtárat megtaláljuk, ha visszatérünk oda — általában minden ilyen fájl egy backslash (\) karakterrel kell kezdeni, hogy visszatérjünk a gyökérbe (root directory), ahonnan aztán hívhatók.

Milyen lehet egy kis batch-fájl, amely az aktuális keresési útvonalat kiterjeszti? Az ADDPATH.BAT nevű fájl így nézhet ki:

```

IF "%1"=="GOTO ERROR
SET PATH=%PATH%;%1 GOTO EXIT
:ERROR ECHO ^GUsage: %0 directory
:EXIT
  
```

Az "ADDPATH C:\UTILS"-t begépelve, a C:\UTILS könyvtárat (melyet a %1 állít vissza a harmadik sorban) hozzáfűzi az aktuális útvonalhoz. Ha parancst nem adunk meg, a batch-fájl elágazik az ERROR címkézett sorra, és aztán az ECHO parancs végrehajtására tér rá. A "C" jel az ECHO parancsban egy kontroll karakter, egy hangjelzést ad (a Ctrl-G-vel írható be a batch-fájlab), a %0 pedig a visszaállítható paraméter, amely a batch-fájl nevét őrzi meg. Így még akkor is, ha később átnevezi a fájlt, mindig a helyes név kerül kijelzésre. Annak ellenére, hogy ez egy egyszerű batch-fájl, mennyi bonyolalmat okoz, ha egy directory bővítésére van szükség egy bonyolultabb keresési útvonal esetén!

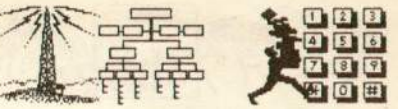
A DOS-környezet kiterjesztése

Ha túl sok DOS-változót használunk, vagy nagyon hosszú útvonalat jelölünk ki, azt veszünk észre, hogy a DOS „Out of environment space” (kívül került a környezet területéről) üzenettel válaszol. Ez akkor következik be, ha túllépjük a DOS számára engedélyezett 127 bájtos határt. Bár a DOS környezet kiterjeszt-

9. lista

```

@ECHO OFF
IF "%1" == "" GOTO ERROR
CHKDSK /V : FIND "%1"
GOTO EXIT
:ERROR
ECHO ^GUsage: %0 filename
:EXIT
  
```



hető, a TSR program jelenléte ezt megakadályozza. (Emlékeztetőül: a Mode, a Print és a Graphics is TSR típusú programok.) E probléma megoldására a CONFIG.SYS fájlba beírjuk az alábbi sort:

```
SHELL=C:\COMMAND.COM/E:256
```

Ez azt közli a DOS-szal, hogy a felső szintű parancsfeldolgozó a COMMAND.COM, és hogy 256 bájt van lefoglalva a rendszer számára. Természetesen ezt a számot tetszőlegesen megválaszthatjuk, de maximálisan 32 768 bájt lehet — azonban ne felejtjük el, hogy ez a felhasználó által felhasználható memóriaterületet csökkenti.

Segédprogramok (utilities)

Természetesen más eszközöket is igénybe vehetünk, hogy fájlunkat még használhatóbbá tegyük. A 8. listán közölt SEARCH.BAT program megkeres egy bizonyos sztringet a fájlokból. A FILESPEC változó tartalmazza a fájl specifikációt — a kereséshez használjuk —, és ez a SEARCH második paramétere. Ha határozatlan, a FILESPEC értéke *.* lesz, és az összes fájlban keresi a kulcs-sztringet. Bár a FIND utility képtelen elfogadni határozatlan fájlnevet — ugyanígy többszörös fájlokat sem képes keresni —, a FOR ciklus azonban képes erre, mert a FIND utasítás folyamatosan ismétlődik, de mindig a megfelelő FILESPEC-cel, azaz sorban az összes fájlspecifikációt megadva.

Talán a FOR a legnehezebben érthető batch-utasítás. Lényegében az a specifikált változó (itt: %A) van folyamatosan kijelölve, amelyek a zárójelek között foglal helyet. Ez lehet valamely adat, utasítás, fájlnev vagy paraméterlista. Ha egy nem teljesen definiált változót használunk, ez az utasítás automatikusan felveszi az összes lehetséges értéket, amely megfelel a fájlspecifikációnak. A változók minden kiértékelése után az utasítás folytatódik, és a DO által meghatározott eljárás végrehajtásával, kiértékelt paraméterekkel.

Felfűzés és átirányítás (piping and redirection)

Egy másik használható batch-fájl a 9. listán közölt WHERE.BAT, amelyik megkeres egy teljes lemezen lévő fájlt vagy fájlnevet, amelyet egy sztringben specifikálunk. Itt a CHKDSK-t a /V paraméterrel használjuk a lemezen található összes fájl kilistázására. Ezt kapcsoljuk a FIND utilityhez (pipe művelet), ahol is csak azokat a fájlokat listázzuk ki, amelyek megfelelnek a megadott paramétereknek (%1). Például a WHERE RESULT.DOC kiírja a katalógus teljes fájl-specifikációját, és a WHERE.BAT kiírja az összes katalógust és a batch-fájlokat nevet.

Jegyezzük meg, hogy a FIND utilityben nem használható metakarakter, és a keresés karakterérzékeny (azaz a keresendő sztring nagybetűvel kezdődik és a „*„ „?” karakterek nem használhatók). A függőleges vonal (|) karakter jelöli az összefűzést (pipe), ami végül is az

utasítások egyszerű kapcsolata olyan módon, hogy az egyik kimenete a következő utasítás bemenete is egyben. Ha a kimenetet sorba rendezve akarjuk, akkor hozzá kell fűznünk a kimenetet a SORT utilityhez.

CHKDSK /V | FIND "%1" | SORT

```
@ ECHO OFF
FOR %ZA IN (n e s NE S) DO IF "%1" == "%ZA" GOTO %I
ECHO Usage: %0 followed by N or E or S [/R] ^B
ECHO to sort by name, extension or size
        (sorba rendezés név, kiterjesztés vagy méret szerint)
ECHO /R reverses sort (option)
GOTO EXIT
:n
DIR | FIND "--" | FIND /V "<DIR>" | SORT %2 | MORE
GOTO EXIT
:e
DIR | FIND "--" | FIND /V "<DIR>" | SORT +10 %2 | MORE
GOTO EXIT
:s
DIR | FIND "--" | FIND /V "<DIR>" | SORT +16 %2 | MORE
EXIT
```

10. lista

```
@ ECHO OFF
IF EXIST LASTUSED.TXT TYPE LASTUSED.TXT
DATE < CRLF > TODAY
EDLN TODAY < EDITDATE > NUL
COPY MESSAGE LASTUSED.TXT > NUL
TYPE TODAY >> LASTUSED.TXT
DEL TODAY
```

11. lista

```
@ ECHO OFF
IF "%1" == "" GOTO ERROR
FOR %ZA IN (%1 %2 %3 %4 %5 %6 %7 %8) DO ATTRIB +R %ZA
ECHO Files about to be detected
ECHO.
ATTRIB *.* | FIND /V "R" | MORE
ECHO.
PAUSE ECHO Y | DEL *.* > NUL
ATTRIB -R *.*
ECHO Done GOTO EXIT
:error
ECHO ^GUsage: %0 filespec1 filespec2 .....filespec8
EXIT
```

12. lista

```
@ ECHO OFF
REM HP DeskJet beállító szekvencia - 132 karakteres módba
REM a nyomtató törlése
ECHO ^[C >PRN
REM az NLQ mód beállítása (Courier) @ 20 karakter
        per inch és 6 pontos méret
ECHO ^[s2q3t6v20H >PRN
REM a bal margin = 0 kijelölése
ECHO ^[&a0L >PRN
REM a felső margin = 2 sor, és 8 sor per inch
ECHO ^[&l8D2e >PRN
REM fájl nyomtatása és formázás kiküldése
TYPE %1 > PRN
ECHO ^[&l0H >PRN
```

13. lista



CHKDSK /V | FIND "%1" | SORT |MORE

Ha a „WHERE DIR” sort begépéljük, egy tartalomjegyzék-listát kapunk. Biztosítani kell persze a DOS utility helyes keresési útvonalát.

Mielőtt a láncolás (pipe) témáról továbblépnénk, a 10. listán bemutatunk egy másik példát, melyet DIRSORT.BAT-nak neveznek. Ez a batch-fájl sorba rendezi a könyvtárakat név, kiterjesztés vagy méret szerint, és a FOR ciklus egy érdekes felhasználását is bemutatja.

Itt a FOR ciklust arra használjuk, hogy ellenőrizzük: az első paraméter (%1) érvényes bemenetnek tekinthető-e. Ha nem találtunk megfelelő, akkor a folyamat folytatódik a következő sor végrehajtásával, amely kijelzi a használati utasítást. Ha azonban találtunk megfelelő, akkor a vezérlést a megfelelő címre adja át. Érdekes, hogy a GOTO címke nem érzékeny a kis- vagy nagybetűre, így aztán használható, bármi legyen is az első karakter (itt: %1).

Minden opció igényel valamilyen „szűrést”. Az első szűrés a FIND „-”, amely csak azokat a sorokat keres, amelyek dátumokat tartalmaznak; a fájlbejegyzéseket és más redundáns információt eldobja. A második szűrő, a FIND /V ” <DIR >”, olyan sorokat keres, amelyek nem tartalmaznak <DIR > bejegyzést. Végül, az így keletkező kimenet a megfelelő részekre tagolódik, amit a MORE utility kijelöl — egy-egy képernyőnyi információt egyszerre. Ha a parancsorbán a /R paramétert használjuk második kimenet, a rendezési sorrend fordított lesz.

Két lehetőség van az átirányításra: a kimenet átirányítása (>) és a bemenet átirányítása (<). A kimenetet vagy fájlba, vagy valamilyen eszköze irányíthatjuk, illetve a bemenet lehet egy fájl vagy egyéb készülék; a kijelölés hiányában a bemenetet a billentyűzetről várjuk, illetve a kimenetet a képernyőre küldjük ki. Mind a billentyűzet, mind a képernyő a Con (Console) nevet viseli. A bevitel/kivitel átirányítást egy másik példán keresztül mutatjuk be, ahol is egy kis program az utolsó bekapcsolás idejét jelzi ki. Először is csináljunk egy kis fájlt a következőképpen:

```
COPY CON CRLF <Enter> <Ctrl-Z>
```

Ez a fájl mindössze egy kicskivisza-soremelés karaktert tartalmaz (inocen a neve is). Ezután két újabb fájl kell létrehozni: EDITDATE és MESSAGE néven.

```
COPY CON EDITDATE
2DIRCurrent date is c
<Ctrl-Z>
COPY CON MESSAGE
```

```
Computer last used on... <Ctrl-Z>
(a számítógépet utoljára használták)
```

A 11. listán látható LASTUSED.BAT nevű fájl a fenti három kis fájlt használja az aktuális adatok tárolására a LASTUSED.TXT nevű fájlban.

A DATE parancs tulajdonképpen egy Entert olvas be a CRLF fájlból, de a kimenetét a TODAY fájlba irányítottuk át. Ezt a fájlt editáljuk (az EDLIN DOS-szerkesztővel használva), de ahelyett, hogy a billentyűzetről várnánk utasításokat, a parancs az EDITDATE fájlból jön. Az EDITDATE tartalmazza azokat a parancsokat, amelyek törlik a kimenet második sorát, törli a „Current data is” (aktuális adat) kiírást, és befejezi az editálást. A kimenetet átirányítottuk a NUL-ba, hogy az editálás folyamatát ne lássuk a képernyőn. A szükséges adat

még a TODAY fájlban van. Az üzenet szövegét, amely a MESSAGE fájlban van, átmásoljuk a LASTUSED.TXT nevű új fájlba, és ezt is átirányítjuk a NUL-ba, hogy itt is elkerüljük az „I file copied” felírat megjelenését a képernyőn. Végül a TODAY-ban lévő adatot hozzáfűzzük a LASTUSED.TXT-hez. Itt a >> jelet használjuk, ami „átirányítás hozzáfűzéssel”-t jelent.

Fájlok törlése

A FOR ciklusa egy másik példa a 12. listán közölt DELBUT.BAT program. Ez törli az összes fájlt, kivéve azokat, amelyeket, mint paramétereket megadtunk. Ha ezt a utilityt használjuk, beírjuk a DELBUT parancsot, amelyet azoknak a fájl-specifikációknak kell követniük, amelyekkel rendelkező fájlokat nem akarunk törölni.

Például:

```
DELBUT *.COM *.EXE *.C *.H
```

A fenti példában minden olyan fájl törölni fog, amelynek kiterjesztése nem COM, EXE, C vagy H. Maximálisan nyolc fájl-specifikáció adható meg.

Az ATTRIB utilityt használjuk, hogy a megadott fájl-specifikációk alapján a megfelelő fájlokat a „csak olvasni” állapotba állítsa. Ugyanis azok a fájlok nem törölhetők, amelyek csak olvasásra vannak kijelölve. Az ATTRIB-et később is felhasználjuk a FIND és MORE kapcsán, hogy azokat a fájlokat kijelezzük, amelyeknek a csak olvasni kijelölése nem érvényes (azaz nincs bebillentve) — így ezeket jelöltük ki törlésre.

A fájlokat a DEL *.* parancssal töröljük, de a bemenetét az Y-on keresztül egy ECHO parancsból kapja, azért, hogy elkerüljük a további billentyűzetről történő bevitelt. A törlés után az ATTRIB parancsot még egyszer kiadjuk: segítségével visszatöltjük a fájlokat az eredeti feltételekkel.

Nyomatóvezérlés

Az ECHO parancsot és az átirányítást felhasználva ki lehet küldeni egy „escape” szek-

enciát és beállító sztringet a printernek. Gyakran használható a nyomtatók sürített nyomtatással. A 13. listán szereplő DUMP.BAT nevű program a HP DeskJet típusú nyomtatót programozza fel.

Emlékeztetünk arra, hogy az Escape karakter ^\j bevitelle CTRL-V\j-vel lehetséges, amely mint ^V\j lesz látható. Figyeljünk arra, hogy a hátulról a negyedik és az utolsó sorban a & jel után egy-egy „kis l” betű van.

A DOS-t kiterjesztő programok közül a Norton Utilities 4.5 programcsomagban is van a batch-programozást segítő program. Ez a BE (Batch Enhancer=batch-kiterjesztő). A program a BE parancs [paraméterek] vagy Be fájlnevé módon aktivizálható, ahol a fájlnevé nevű fájl tartalmazza a végrehajtható parancsok sorozatát. A parancsok és funkcióik:

- ASK — Egy szöveg kiírása után billentyűnyomásra vár (WHAT-hoz hasonló)
- BEEP — Hangjelzessorozat generálása
- BOX — A képernyőn keret megjelenítése
- CLS — Képernyő törlése
- DELAY — Adott időtartamú várakozás
- PRINT-CHAR — Karakterkiírás a képernyőre
- ROWCOL — Kurzor pozicionálása a képernyő adott sorára és oszlopára
- SA — A képernyő színeinek beállítás
- WINDOW — Ablakos megjelenítés

A program — mint a parancsok felsorolásából látható — elsősorban a batch-programok „felhasználóbarát” kialakítását segíti, amivel a batch-programjainkat csillogó-villogóvá varázsolhatjuk.

Az itt bemutatott néhány batch-fájl mindegyike lemezre meghajtott igényel, és igen lassú a kétfloppys XT rendszeren. A példák azonban — úgy véljük — jól szemléltették a batch-fájlok alkalmazásának néhány érdekesebb területét, és alapot adhatnak a saját batch-fájlok készítéséhez. Ehhez sok sikert kívánunk!

Tóth István

Megváltozott néven, új profillal!

A PC—HOMELAB KLUB

minden kedden 18 órától 21 óráig

tartja összejöveteleit a Belvárosi Művelődési és Ifjúsági Házban (Budapest V., Molnár u. 9. Telefon: 117-5928).

Új nevének megfelelően a klub elsősorban a PC-felhasználók köréből várja az érdeklődőket.



SZABAD SZOFTVER



Microsoft Word 4.0

A Microsoft szoftverház Word nevű szövegszerkesztő programja IBM PC-n vagy vele kompatibilis számítógépen futtatható. Használatához minimálisan 256 kb-át memória szükséges. Monokróm vagy színes grafikus, illetve EGA adapterrel is működik. A program támogatja az egér használatát.

A Word tíz darab 5 1/4 inches lemezen kerül forgalomba. Természetesen a munkához szükséges verzióknak nem kell az összes, lemezeken meglevő fájl tartalmaznia, a konkrét hardverkiegészések (kijelző, printer, egér) megfelelő verzió létrehozásában az egyik lemezen megtalálható SETUP program segít. A programrendszer floppyn és winchesteren is létrehozható.

Az installált rendszer a WORD parancs begépelésével indítható. A parancssorban opcionálisan megadható a szerkesztendő fájl neve és néhány kapcsoló is. Az utóbbiak segítségével kiválasztható, hogy szöveges (character) módban (/c) vagy grafikus módban (/g) akarjuk-e használni a programot. Az utóbbi használatánál — grafikus képességekkel rendelkező kijelző esetén — a karakterformátó utasítások (vastag betű, dőlt betű, aláhúzás stb.) hatása a képernyőn is nyomon követhető, az előbbi módban viszont gyorsabb működés érhető el. A két mód közötti váltás a program futása alatt is megtehető. Hercules vagy EGA kijelző esetén használható a /h kapcsoló is, ami például Hercules kértánál 43 soros, soronként 90 karakteres megjelenítést tesz lehetővé.

Segíti a program működésének elsajátítását a Learning Word tanító program, amely különböző szintű, menüvezérelt leckéken keresztül mutatja be az egyes funkciók használatát. A programhoz adott dokumentáció több részből áll. A Using Microsoft Word a használatot mutatja be, a Reference to Microsoft Word pedig kézikönyvként használható. Kiegészítő információk találhatók a Printer Information for Microsoft Word című kötetben. A dokumentáció összesen mintegy 800 oldal terjedelmű.

A Word program meghívásakor két lehetőség közül választhatunk: ha új dokumentumot kezdünk szerkeszteni, akkor egy üres ablakot látunk a képernyőn, ha pedig egy már meglévő szöveget akarunk javítani, és annak nevét megadtuk a parancssorban, akkor azt tölti be a kezdő ablakba a program.

Szöveget beírni a szokott módon lehet: egyszerűen bebenteltyűzzük. A hibák javítása a BackSpace billentyűvel történik. Az írási helyzetünket az ún. highlight mutatja (inverz karakterpozíció). Ezt a kurzormozgató és a Home, End, PgUp, PgDn, illetve Control+PgUp, Control+PgDn billen-

tűkkel mozgathatjuk, így a szövegbe tetszés szerinti helyre beszurhatunk új szöveget. A pozíció kijelölését egérrrel is végezhetjük.

A szövegszerkesztő word-wrap üzemmódban működik, vagyis sor végére érve, a következő szó automatikusan új sorba kerül. Ha új bekezdést (paragraph) akarunk kezdeni, az Enter leütésével iktathatunk be „kemény” soremelést a szövegbe.

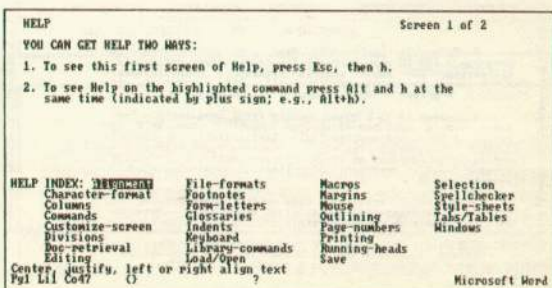
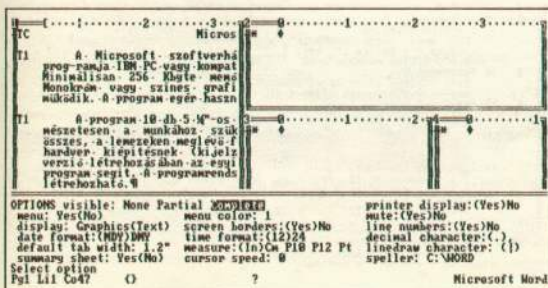
A program alapértelmezés szerinti üzemmódja a beszurási (insert) üzemmód. A felülírás (overtyp) üzemmód az F5 funkcióbillentyű megnyomásával választható.

Válassz ki, és hajtsd végre!

A program alapvető filozófiájához tartozik a „válassz ki és hajtsd végre” elv, vagyis mielőtt valamilyen változtatást akarunk a szövegben végezni, meg kell mondanunk, hogy azt a szöveg mely részén kell megtenni. A kiválasztást (selecting) a billentyűzetről és egérről is megtehetjük. A kiválasztható elemek a karakter, szó, mondat, bekezdés, sor, illetve ezek többszörösei. Lehetőség van a szövegben oszlopok szerinti kiválasztásra is, vagy a teljes dokumentáció is kiválasztható. A kiválasztott szöveget a highlight jelzi.

Miután megtörtént a kiválasztás, meg kell mondanunk, hogy mit akarunk tenni a kiválasztott szöveggel. A parancsok közötti választást többszintű menürendszer segíti, amelybe az Esc billentyű megnyomása után térhetünk át. A parancsok közötti választás a kurzormozgató billentyűk és a space segítségével történik. A parancs az Enter leütésével végrehajtható. A parancsok közötti választás és a végrehajtás egérről is elvégezhető.

A program futása közben — mint említettük — a szerkesztés alatt levő szöveg a képernyőn egy ablakban jelenik meg. A képernyő alján látható a parancs-menü — amely opcionálisan eltüntethető —, valamint egy státuszsor, amely pillanatnyi pozíciókról ad felvilágosítást a kurrens lap, sor és oszlop kijelzésével. Ugyanitt található az ún. scrap (cédula), ahol a





legutóljára másolt vagy törölt szöveg (vagy legalább annak eleje és vége) látható. A szerkesztés alatt levő szövegen kívül megjeleníthetők olyan — nem nyomtatandó — információk is, mint a szóközlők, a bekezdést, az új sort, a tabulátort stb. jelképező karakterek. Ezek az opcionálisan megjeleníthető információk segíthetnek a szöveg rendezésében. Az ablak felső sorában megjeleníthető egy vonalzó (ruler), amely a margókat, tabulátörzsi pozíciókat stb. mutatja.

A Word program széles körű segítő információs rendszerrel (help) van ellátva. Egyszerű választhatjuk a Help parancsot, ahol témakörök szerint csoportosítva kaphatunk segítséget, vagy az Alt-H billentyűkombinációval (illetve a státuszsorban látható kérdőjele a egérről rámutatva) az éppen végrehajtás alatt levő parancsról kapunk segítő információt (online help). Arra is van lehetőség, hogy szerkesztés közben használjuk a leckezerület felépített Tutorial Helpet.

A szerkesztés alatt levő szövegen a legalapvetőbb műveletek a törlés (Delete), a másolás (Copy) és a beszúrás (Insert). Ezeket a műveleteket a már említett scrap használatával végezhetjük. Ha a kiválasztott szöveget töröljük vagy másoljuk, akkor az megjelenik a scrapben, ahonnan a szöveg másik helyére beszúrhatjuk, akár többször is. Hasznos lehetőség, hogy a legutolsó műveletet egy billentyű leütésével (F4) megismételhetjük vagy érvényteleníthetjük (Undo parancs).

vagy paragrafus-formátumokat kereshetünk és cserélhetünk ki más formátumra. A keresési és kicserélési műveletek — mint bármely más művelet is — ismételhetők az F4, illetve a Shift-F4 billentyűkkel.

Tablázatok szerkesztésénél lehet segítségünkre a tabulátor. A tabulátorpozíciók beállíthatók, törölhetők, áthelyezhetők. Egyszerűen vissza lehet térni az eredeti beállításra is. Azt is megadhatjuk az egyes tabulátorpozíciókhoz, hogy ott a beirt szöveg balra, jobbra, középre vagy decimálisan legyen-e igazítva (az utóbbi például számoknál hasznos). Jól használható a táblázatok szerkesztésénél az oszlopok szerinti kiválasztás lehetősége. Ezután a kiválasztott szövegrész az egyéb szövegekhez hasonlóan törölhető, másolható, mozgatható stb.

A szerkesztett dokumentum nyomtatásakor is számos opcióval élhetünk. A Print Pageinate parancsral automatikus vagy manuálisan elvégezhetjük a lapokra tördelést. A Print Option parancsral megadható a printer típusa, a nyomtatandó szöveg terjedelme (ami lehet a kiválasztott szövegrész, adott lapok vagy az egész dokumentum), valamint a példányok száma. Mindezek után a Print parancsral nyomtathatunk printerre vagy akár fájlba is, ha gépünkhoz nem csatlakozik printer. A queued nyomtatást is választathatjuk, ilyenkor folytathatjuk a munkát, és a program a „háttérben” nyomtat.

A lemezkezelő parancsok a Transfer menüpontonál érhetőek el. Lehetőségünk van a dokumentum mentésére (Transfer Save), opcionálisan a formázási információk nélküli is (így ASCII fájlként kapunk, amely minden más szövegszerkesztő formátumával kompatibilis), új dokumentum betöltésére (Transfer Load) vagy lemezen levő fájl törlését is elvégezhetjük (Transfer Delete).

Balra, jobbra, középre, kizárva

A szöveg formázásának két legalapvetőbb szintje a karakter és a bekezdés formázása. Az előbbinél (Format Character) a kiválasztott szövegrészben szereplő karakterek milyenségét írhatjuk elő (vastag betű: bold, dőlt betű: italic, aláhúzott: underlined, kétszer aláhúzott: double underlined, nagybetűs: uppercase, alsó-, illetve felső indexű: subscript/superscript stb.), és megadhatjuk a karakter típusát és méretét is. A bekezdés formázásánál (Format Paragraph) a szöveg elrendezését írhatjuk elő a kiválasztott bekezdés(ek)re vonatkozóan. Ilyen lehet az, hogy a szöveget balra, jobbra vagy középre igazítva, esetleg sorközarással akarjuk elrendezni, valamint itt állíthatjuk be a margókat, a sortávolságot stb. A lényegesebb formázó utasítások megadhatók egyetlen, Alt+ valamilyen billentyű kombinációval is (speed formatting). Lehetőség van arra is, hogy a paragrafusokat egymás mellé helyezzük el (side by side).

A formázás következő szintje a fejezet (division) formázás. Ebben a parancskörben beállíthatjuk az egész fejezetre érvényes margókat és a lap méretét (Format Division Margins), előírhatjuk az automatikus lap-számozás helyzetét és formáját (Format Division Page-numbers). Megadhatjuk azt is, hogy a dokumentumunk több hasámban jelenjen meg a nyomtatásban — mint egy újságnál — (Format Division Layout), és előírhatjuk a sorok megszámozását is (Format Division Line-numbers). Ez utóbbi két tulajdonság csak a nyomtatott dokumentumban jelenik meg, a képernyőn nem látszik.

Definiálhatunk ún. running headet is, amely egy olyan szövegrész, amely a fejezeten belül minden nyomtatott lapon megjelenik majd a lap tetején vagy alján (Format Running-head). Előírható az is, hogy a páros és a páratlan oldalakon különböző running headek jelenjenek meg.

Gyakran előforduló feladat szövegszerkesztésnél adott szövegrész keresése vagy kicserélése. A Search parancsral kereshetünk, a Replace parancsral pedig kicserélhetünk szövegrészeket. Az utóbbinál előírhatunk automatikus vagy megerősítendő cserét is. Érdekes lehetőség a Format sEarch, illetve a Format replAce parancs, amellyel karakter-

Nyolc ablak

Mint már említettük, a szerkesztendő szöveg a képernyőn egy ablakban jelenik meg. A Window menüpont segítségével nyithatunk új ablak(ok)at a képernyőn. Egyszerre maximum nyolc ablak nyitható meg, és ezek mindegyike tartalmazhat különböző dokumentumokat vagy akár egy dokumentum különböző részeit. Így egyszerűen megvalósítható például két szöveg egyes részeinek cseréje, másolása stb. Az ablakok elhelyezkedhetnek egymás alatt vagy egymás mellett is. A már megnyitott ablakok mérete utólag is megváltoztatható, akár egérrrel is. Külön ablak nyithatunk a lábjegyzeteknek is (footnote). Jól használható lehetőség az éppen aktuális ablak „zoomolása”, vagyis ilyenkor a teljes képernyőt az aktuális ablak fogja kitölteni.

Az eddigiekben ismertetett tulajdonságok is érzékeltetik, hogy a Word szövegszerkesztő program igen sokoldalúan használható. A következőkben azokat a lehetőségeket ismertetjük, amelyek a program fejlettebb használatát teszik lehetővé.

Minden egyes dokumentumhoz készíthetünk egy úgynevezett summary sheetet, vagyis egy olyan, néhány információt tartalmazó összefoglalót (cim, szerző, dátum stb.), amelynek segítségével később könnyebben eligazodhatunk a lemezen tárolt fájlok között. Erre a célra egy külön menürendszer szolgál, a Library Document-retrieval.

A már említett lábjegyzetek nyomtatásakor a lapok alján vagy az egyes fejezetek végén összegyűjtve. A lábjegyzetek jelzésére használhatunk csillagot, keresztet, vagy választhatjuk az automatikus sorszámozást is. Természetesen a lábjegyzetekre is érvényesek a formázási lehetőségek.

Körlevek készítésénél jól használható az a lehetőség, hogy a min-





den levélnél azonos szöveget csak egyszer kell megírunk (main document), és egy másik dokumentum tartalmazza az egyedi információkat (data document). A Print Merge parancssal elkészíthetjük az egyedi nyomtatványokat, ahol a változó információk bekerülnek a fő dokumentum megfelelő helyeire.

A Delate, Copy és Insert parancsokról már volt szó. Eddig mindig a scraper keresztül történtek ezek a műveletek. A Glossary (szógyűjtemény) fogalmánál bevezetésével megtehetjük azt, hogy egy kiválasztott szövegrészt (például egy gyakran ismétlődő bonyolult szót) ne a scrabe, hanem egy általunk elnevezett glossyba másoljuk be. Ezután a szó leírása helyett a glossary (rövid) nevét kell csak beírunk és az F3 billentyű megnyomásakor a program az eredeti szót iktatja be a szövegbe.

A tárgymutató (Index) elkészítésénél is segítségül hívhatjuk a Word szövegszerkesztő programot. Azokat a szavakat, amelyeket szerepeltetni kívánunk az indexben, meg kell jelölnünk a szövegben (ez a megjelölés a nyomtatásban persze nem fog megjelenni). A Library Index parancssal automatikusan elkészíthető az index, ahol a megfelelő szavak mellett megjelennek azok az oldalszámok, ahol a megjelölt szavak szerepelnek.

A tárgymutatóhoz hasonlóan a tartalomjegyzék is automatikusan elkészíthető.

Hasznos lehetőség, hogy a program megengedi a táblázatrajzoló grafikus karakterek használatát. Ha átterünk a Line Drawing üzemmódba (Control-F5), akkor a kurzorozgató billentyűkkel „rajzolhatunk” a szövegben. Még egyszerűbb a helyzet, ha például egy paragrafust be akarunk keretezni, mert ehhez csak a Format Border parancsot kell aktiválnunk.

Mint minden korszerű programnál, a Wordnél is élhetünk a makrók használatával. A gyakran ismétlődő hosszabb parancssorozatokat makrókat lehet képezni, melyek később aktiválhatók. A makrókban használhatunk utasításokat is, mint IF..ELSE..ENDIF, MESSAGE, REPEAT..ENDREPEAT stb. Definálhatunk autoexec makrókat is, amely automatikusan végrehajtódik, mikor indítjuk a programot.

Heading és body text

Nagyon jól használható az Outlining tulajdonság. A Shift+F2 billentyűkombinációval átkapcsolhatunk egy olyan megjelenítési módra, ahol a már meglévő szövegünk szerkezetét tanulmányozhatjuk: ilyenkor az egyes szövegrészeknek csak a címét (heading) fogjuk látni. Így gyorsabban tájékozódhatunk, mozoghatunk a dokumentumban, sőt, mivel egy cím áthelyezése az „alatta” levő szövegrész (body text) áthelyezésével is jár, könnyen átrendezhetjük a szöveget.

A program lehetőséget ad egy kiválasztott szövegrész rendezésére is. Így például egy felsorolást automatikusan ábécé-sorrendbe rendezhetünk.

Ha a dokumentumunkat sorkizárással formáztuk, akkor célszerű az automatikus elválasztás (hyphenation) lehetőségével élnünk. Így elkerülhető, hogy a szavak között túl sok köz legyen, és a dokumentum megjelenése sokkal szebb lesz. Mivel a program felkínál egy elválasztási pozíciót, és ezt megváltoztathatjuk, ezért ez a funkció nagyon nyelvi szövegeknél is jól használható (az elválasztási pozíció felkínálása természetesen az angol helyesírás szabályai szerint történik).

Két tulajdonság sajnos csak angol nyelvű szövegeknél használható: a helyesírás-ellenőrzés (spelling) és a szinonimaszótár (thesaurus).

Igen értékes tulajdonsága a programnak az F2 billentyűvel aktiválható kalkulátor (Calculate) üzemmód. Ilyenkor a kijelölt számsorozaton a szereplő műveletek végrehajtódnak, és az eredmény a scrabe kerül. Nagyon jól használható táblázatos adatok összesítésére.

Utoljára maradt talán az egyik legjobban használható funkció: a Style Sheet (formátum lap) rövid ismertetése. Ha összegeyítjük az általunk általában használt formázási mintákat, akkor sokkal gyorsabban és automatikusan formázhatjuk szövegeinket. Másrészt ezzel elérhető, hogy különböző időpontokban írt (vagy akár különböző személyek által írt) dokumentumok kitalálják egységes legyen. A Style Sheetek létesítésére, módosítására egy menürendszer, a Gallery szolgál.

Ebből a rövid ismertetésből is kitűnik, hogy a Word szövegszerkesztő program igen változatos lehetőségeivel kiválóan alkalmas professzionális minőségű dokumentumok létrehozására.

? AJÁNDEK



384 kbájtos RAM-lemez

Sokan vásároltak külföldön olyan, 4,77/10 MHz-es XT turbó alapelemezeket, amelyekbe 1 Mbájttal összkapacitású — $4 \times 9 = 36$ darab —, 256 kbités RAM-ok dughatók. Az XT-ben levő 8088-as mikroprocesszor által megcímezhető 1 Mbájtos memóriatartományból a DOS csupán 640 kbájtot tart fenn a programok számára (az operatív tár: 00000H-9FFFFH), a többi a képernyő-memória, a ROM-BIOS és egyéb kiegészítő részek használják. Ily módon az alaplapon levő 1 Mbájtos memóriák fennmaradó 384 kbájtos részét csak speciális módon, ún. RAM-lemezként használhatjuk. Szokásos még a virtuális lemez (disk) elnevezés is.

Mi is ez a RAM-lemez? A számítógépen futó programok — és maga a DOS is — információkat írnak és olvasnak, amelyek a lemezek levő állományokban vannak. Óriási sebességnövekedést jelentene, ha az állományok nem a viszonylag lassú, nagy elérési idejű lemezekben, hanem a számítógép tárában lennének. Nos, a RAM-lemez a számítógép memóriájának egy erre a célra fenntartott része, amely pontosan úgy viselkedik, mint egy lemezegység: írhatunk bele, olvashatunk belőle. Természetesen ehhez szükség van egy speciális, a lemezegységet szimuláló programra. A különféle DOS-változatok tartalmaznak ilyen programokat (VDISK, RAM-DISK), de ezek vagy a PC/AT kibővített memóriáját, vagy a 640 kbájtos operatív tár egy részét használják.

A fent említett 1 Mbájtos alapelemek fennmaradó 384 kbájtnyi memóriája is pontosan ilyen RAM-lemezként használható, azonban használatához egy, az alaplapon áramköri kialakítását figyelembe vevő program szükséges. A programot a kereskedőnek az alaplappal együtt lemezen kellene adnia, de a tapasztalatok szerint ez nem történik meg. Rovatunkban ezúttal ezt a RDISK.SYS elnevezésű programot adjuk közre. A program csak a MULTI TURBO MB (ON BOARD MEMORY ABOVE 640 KB) típusú alaplaphoz használható. A CONFIG.SYS állományba elhelyezzük a DEVICE=RDISK.SYS sort, magát a programot pedig a rendszerlemezre vesszük fel. Ezek után a rendszert újraindítva a RAM-lemez már használható.

FIGYELEM!

A PÉCÉZZÜNK rovatban megjelent cikkek szövege szövegfájlok formájában, valamint az „Ajándék” szabad szoftver 360 kbájtos DS-DD lemezen, utánvétellel, önköltségi (lemezár, lemezmásolás, postázás) 300 forintos áron megrendelhető.

Cím: Kocncz Edit, Budapest, Kunigunda u. 44. 1037

Grafika Az Amiga programozása assembly nyelven II.

Mivel az Amiga grafikai képességei messze felülmúlják a korábbi Commodore gépek adottságait, mindenekeelőtt egy sor új fogalommal kell megismerkednünk. Az „amigás alapműveltség” megkívánja, hogy tisztában legyünk az olyan gyakran előforduló kifejezésekkel, mint Interlace, Overscan, HAM üzemmód, Dual-playfield display, Bitplane és még sorolhatnám.

Az európai tv-szabvány szerint egy teljes tv-kép — legyen az PAL vagy SECAM rendszerű — 625 sorból áll. Ebből körülbelül 580 sort látunk a képernyőn, a többi nem látható az elektronsugár függőleges és vízszintes visszafutása miatt. A 625 sorból álló teljes kép viszont két félképből tevődik össze. Mindkét félkép felrajzolása során az elektronsugár soronként végigpásztázza a teljes képernyőt. A különbség csak az, hogy az egyik félkép idejében a páratlan, a másik félkép idejében pedig a páros számú sorokat rajzolja fel az elektronsugár a képernyőre. A képek ismétlődésének frekvenciája, vagyis a képfrekvencia Európában 25 Hz, a félképfrekvencia pedig természetesen ennek a kétszerese, 50 Hz.

Biztosan mindenkinek feltűnt, hogy a programok többsége nem használja ki a képernyő minden sorát, hanem annak kb. csak a felső négyötödét. Ennek a tendenciának az európaiól eltérő amerikai (NTSC) televíziós rendszer az oka. Mivel az NTSC-kép csak 525 sorból áll, ahhoz, hogy a program mind PAL, mind NTSC rendszerű Amigán használható legyen, nem szabad az összes sort felhasználni. Egy PAL rendszerű rajzolóprogram tehát annyiban előnyösebb számunkra egy NTSC rendszerűnél, hogy az előbbivel a teljes képernyőt telerajzolhatjuk.

Emlékezzünk vissza a C64 grafikájára: a függőleges felbontás 200 sor volt. Ebből adódik, hogy egy C64-es kép kényelmesen elfér egy félképben, hiszen egy félkép $625/2=312,5$ sorból áll. A C64 tehát mindkét félkép alatt ugyanazt az információt juttatta el a képernyőre, vagyis a páros és páratlan tv-sorok tartalma megegyezett. Egy PAL rendszerű — tehát európai — Amiga viszont 512 sort tud megjeleníteni, és ez már nem fér bele egy félképbe.

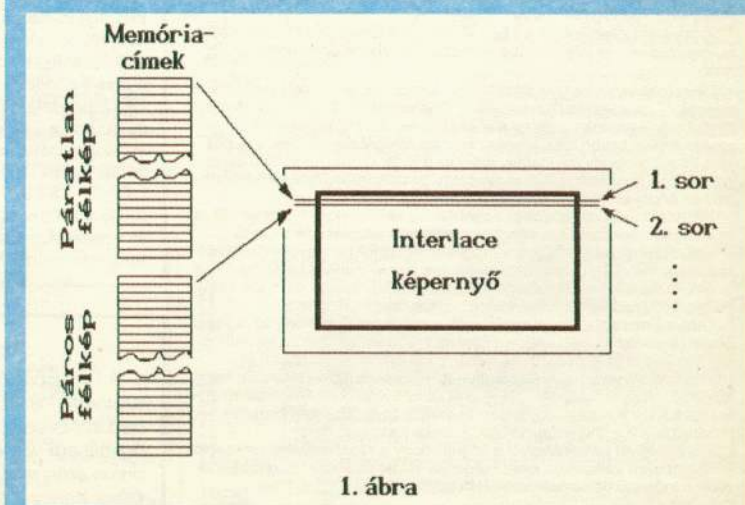
Ezt a nagy függőleges felbontású üzemmódot Interlace üzemmódnak nevezik. Interlace módban a két félkép tartalma nem egyezik meg egymással, tehát a páros és páratlan tv-sorok különböző rajzolatot tartalmaznak. Ebben az üzemmódban válik láthatóvá az a kis szépséghiba, amelyért a hozzá nem értők az Amigát okolják. Az Interlace módban megjelenített kép ugyanis a képtartalomtól függő mértékben remeg, vibrál. Ennek oka viszont nem a számítógépben keresendő, hanem a tv-szabványban, amely a képfrekvenciát a szemmel már érzékelhető 25 Hz-es értékben határozza meg.

Más személyi számítógépeknek a problémát úgy oldották meg, hogy feladták a televíziókkal való kompatibilitást, és nagyobbra választották a képfrekvenciát. Nagyobb képfrekvencián azonban már csak speciális — és korántsem olcsó — grafikus monitorok használhatók. A Commodore cég ezzel szemben nem adta fel annak a lehetőségét, hogy az Amiga hasz-

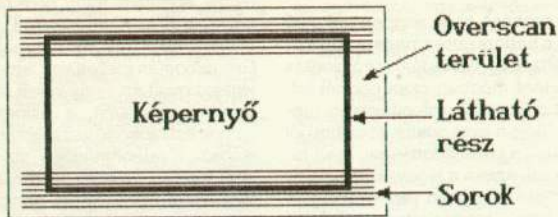
nálható legyen a háztartásokban amúgy is meglévő televíziókkal. Az elfogadható képminőség érdekében egyébként ebben a kategóriában is beszerezhető a különféle professzionális monitorok.

A nagy függőleges felbontású üzemmód neve tehát Interlace vagy másképpen Lace (512 sor), a normál függőleges felbontású üzemmód pedig Non-Interlace vagy másképpen No-Lace (256 sor). Amígán mindkettő lehetséges. Egy Interlace képernyő pontosan két Non-Interlace képernyőnyi helyet foglal el a memóriában úgy, ahogy az az 1. ábrán látható. Vízszintes irányban is kétféle felbontás létezik, nevéük Hi-Res (640 oszlop) és Lo-Res (320 oszlop). Hangsúlyoznom kell azonban, hogy a vízszintes és a függőleges felbontás megválasztásával csak az elemi képpontok (pixelek) méretét állítjuk be. A koordinátangelyeket mindkét irányban tetszőleges hosszúságúnak definiálhatjuk. Ezek szerint olyan képernyőt is létrehozhatunk, amelynek alsó és jobb széle egya-

1. ábra



1. ábra



2. ábra

ránt „kilóg a képből”. Miután az Amigán szabadon megadhatjuk az aktív kijelzési terület bal felső sarkának pozícióját, lehetőségünk nyílik az egész képernyőt betöltő (tehát keret nélküli) ún. Overscan screen létrehozására. Ezt illusztrálja a 2. ábra. Az Overscan tehát szintén új fogalom és azt a grafikai üzemmódot jelenti, amikor a screen szélei kívül esnek a látható mezőn, vagyis nincs keret.

Mint tudjuk, az aktuális paletta színeit 4096 színárnyalatból választhatjuk ki. Additív színekveréssel a vörös (Red, jele: R), a zöld (Green, jele: G) és a kék (Blue, jele: B) színekből jó közelítéssel az összes látható szín előállítható. Amigán 16-féle intenzitású vörös, zöld és kék szín eredője képezhető, ez pontosan $16 \times 16 \times 16 =$

$=4096$ -féle szín. Egy közönséges screen egyidejűleg annyi szint tartalmazhat, amekkora a hozzá tartozó paletta színkészlete. A paletta színeit a 4096-ból válogathatjuk össze, tetszés szerint. A színeket három hexadecimális jegyből álló számokkal adhatjuk meg. Például:

- \$000=fekete
- \$fff=fehér
- \$888=középszürke
- \$0ff=türkizkék (cián)
- \$ff0=sárga
- \$90f=indigókék
- \$f90=narancssárga

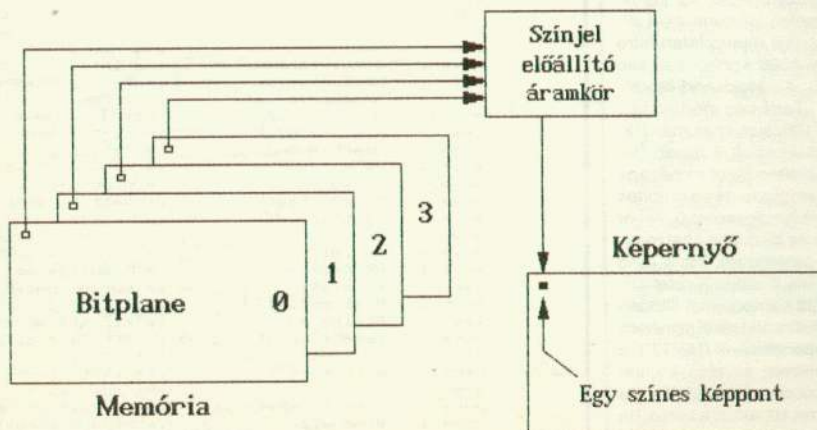
A kódban a számjegyek RGB sorrendben állnak. Egy színes screen elhelyezkedése a memóriában korántsem olyan

agyafűrt, mint az a C64-nél volt a multicolor üzemmódban. A paletta elemeinek száma 2 hatványai szerint alakulhat. 2, 4, 8, 16 és 32 elemű paletta használatára nyílik lehetőség.

Felhívom a figyelmet arra, hogy a 32 elemű paletta csak normál felbontású (Lo-Res) képernyőnél alkalmazható. Kételelemű paletta esetén tulajdonképpen monokrom (egyszínű) képernyőről van szó. Ilyenkor, ahol a képernyő-memóriában 1-es bit áll, ott a képernyőn a 0-ás palettaszín jelenik meg, ahol pedig 0-ás bit van, ott az 1-es palettaszín látható. Azt a memóriaterületet, amely egy monokrom screen megjelenítéséhez szükséges, Bitplane-nek nevezzük.

Tegyük fel, hogy egy 16 színű képernyővel van dolgunk (3. ábra). A memóriában ekkor egymás után 4 darab Bitplane-t találunk, mert $2^4=16$. A Bitplane-ek akkorák, hogy külön-külön lefedik a teljes képernyőt. Ha meg szeretnénk tudni, hogy egy pixel milyen színű, akkor mindegyik Bitplane-ből „ki kell vadászniunk” azt a bitet, amelyik a kérdéses pixelhez tartozik. Ha ezt a 4 bitet leírjuk egymás mellé, akkor az így kapott bináris szám a pixelhez tartozó palettaszín száma. A Bitplane-ek számát mélységnek (depth) nevezik a szakirodalomban. Az alábbi táblázatban a színekkel kapcsolatos tudnivalókat foglaltam össze.

3. ábra



Bitplane-ek száma

| Színek (mélység) | Szám | Lehetséges üzemmódok |
|------------------|------|-------------------------------|
| 2 | 1 | Hi-Res, Lo-Res, Lace, No-Lace |
| 4 | 2 | Hi-Res, Lo-Res, Lace, No-Lace |
| 8 | 3 | Hi-Res, Lo-Res, Lace, No-Lace |
| 16 | 4 | Hi-Res, Lo-Res, Lace, No-Lace |
| 32 | 5 | Lo-Res, Lace, No-Lace |

Azért nem kell elképedni, van lehetőség arra, hogy több mint 32 színt — sőt mind a 4096-t — egyszerre használhassuk a képernyőn! Ezek a következők:

1. Extra-Halfbrite (EHB) üzemmód beállítása
2. Hold-And-Modify (HAM) üzemmód beállítása
3. A Copper használata

Az Extra-Halfbrite egy olyan új tulajdonsága a video chipnek, amely az Amiga 1000 idejében még nem létezett. EHB módban a paletta 64 színből áll, ennek megfelelően 6 darab Bitplane alkot egy screen-t. Az „Extra-Halfbrite” egy torzított angol elnevezés, amely „Extra-Félfényességű” lenne magyarul. A név azt takarja, hogy a második 32 palettaszín az első 32 palettaszín félfényességű változata. Tehát csak az első 32 palettaszínt (0-tól 31-ig) választhatjuk meg szabadon, a második 32-t (32-től 63-ig) a video-áramkör automatikusan képezi az első 32-ből. Ennek a trükknek az ad létjogosultságot, hogy a grafikus chipnek mindössze 32 színre-gisztere van.

Hold-And-Modify (HAM) üzemmódban mind a 4096 szín egyidejűleg jelen lehet a képernyőn. Akkor értjük csak meg a HAM kijelzési mód nagyszerűségét, ha kiszámoljuk, hogy egyszerű Bitplane-eket alkalmazva milyen óriási memóriaterületre lenne szükség egy 4096 színből álló kép megjelenítéséhez. A Hold-And-Modify magyar jelentése: „Tartás és módosítás”. Egy HAM display 6 Bitplane-t használ. Palettája 16 színt tartalmaz. A 6 darab Bitplane miatt tehát minden pixel színét egy hat bites bináris szám írja le. Ha a hat bites szám 4-es és 5-ös biteje egyaránt 0, akkor az illető pixel színe az első 4 bit által meghatározott számú palettaszín. Ha a 4—5 bitpár tartalma bináris 0, akkor a pixel színe a vörös és a zöld komponensnél illetően megegyezik a tőle balra álló pixel színének vörös és zöld komponensével (Hold!!!), a kék komponens viszont az első 4 biten megadott értékre módosul (Modify!!!). Ha a 4—5 bitpár tartalma 10, akkor a vörös, ha 11, akkor a zöld komponens módosul az első 4 bit által meghatározott értékűre. A másik két színkomponens a bal oldali

szomszédos pixelhez képest nem változik meg.

Fontos szabály, hogy minden sor első képpontja a háttérszínnel megegyező színű (0. palettaszín). Kissé fájó megszorítás az, hogy HAM módban csak normál felbontású (Lo-Res) megjelenítésre van lehetőség. Mégis a gyakorlati igényekkel jól egybevágnak a HAM adottságai, mert hiszen ilyen sok színre a leggyakrabban digitalizált képek esetében van szükség. A digitalizált képek természetes, valóságos

témákat ábrázolnak, a valóságos fotók pedig az árnyékok miatt nem tartalmaznak ugrásszerű színátmeneteket. Arra a korlátozásra, hogy a HAM üzemmódra csak Lo-Res felbontás mellett van lehetőség, némiképp vigaszt nyújt az, hogy ennél nagyobb vízszintes felbontást a színes összetett videójelel működő készülékek (színes televíziók, videomagnók, videokamerák) sem tudnak nyújtani. A felbontási korlát egyébként közönséges színes televízió szinté észrevehetetlen.

PÉLDAPROGRAM

```

ExecBase:      equ      4
OpenLibrary:   equ      -$228
CloseLibrary:  equ      -$19e
OpenScreen:    equ      -$c6
CloseScreen:   equ      -$42
OpenWindow:    equ      -$cc
CloseWindow:   equ      -$48
Move:          equ      -$f0
Text:          equ      -$3c
LACE:          equ      $0004
EXTRA_HALFBRITE: equ    $0080
DUAL_PLAYFIELD: equ    $0400
HAM:           equ      $0800
HIRES:         equ      $8000
CUSTOMSCREEN:  equ      $01
IDCMP_Flags:   equ      0
WINDOWSIZING: equ      $0001
WINDOWDRAG:   equ      $0002
ACTIVATE:     equ      $1000
Flags:         equ      WINDOWDRAG+ACTIVATE

Start:  move.l  ExecBase,a6          ;az Exec báziscíme
        clr.l   d0                  ;verziószám: 0
        lea    IntuitionName,a1     ;mutató az Intuition nevére
        jsr    OpenLibrary(a6)      ;az Intuition megnyitása
        move.l d0,IntuitionBase     ;a báziscím elmentése
        beq    NoIntuition          ;ha nem sikerült, kilép
        clr.l   d0                  ;verziószám: 0
        lea    GfxName,a1           ;mutató a Graphics nevére
        jsr    OpenLibrary(a6)      ;a Graphics megnyitása
        move.l d0,GfxBase           ;a báziscím elmentése
        beq    NoGfx                ;ha nem sikerült, kilép
        move.l IntuitionBase,a6     ;az Intuition báziscíme
        lea    NewScreen,a0         ;mutató a NewScreen-re
        jsr    OpenScreen(a6)       ;a screen létrehozása
        move.l d0,NewScreen         ;beírja a NewWindow-ba
        lea    NewWindow,a0        ;mutató a NewWindow-ra
        jsr    OpenWindow(a6)       ;az ablak kinyitása
        move.l d0,Window            ;a Window struktúra mutatója
        move.l d0,a0                ;előkészítés
        move.l 50(a0),RastPort      ;mutató a RastPort-ra
        move.l RastPort,a1         ;elmentés
        move.l #90,d0               ;a kurzor x koordinátája
        move.l #40,d1               ;a kurzor y koordinátája
        move.l GfxBase,a6           ;a Graphics báziscíme
        jsr    Move(a6)              ;a kurzor pozícionálása
        move.l RastPort,a1          ;a RastPort mutatója
        lea    String,a0            ;a kiíratandó string
        move.l #EndOfStr-String,d0 ;a string hossza
        jsr    Text(a6)              ;kiíratás
Wait:   cmpi.b  #116,$bfec01        ;le lett nyomva az ESC gomb?
        bne.   Wait                ;ha nem, akkor várakozik
        move.l IntuitionBase,a6     ;az Intuition bázisa
        move.l Window,a0            ;a Window struktúra
        jsr    CloseWindow(a6)      ;az ablak bezárása
        move.l NewScreen,a0         ;a Screen struktúra

```

A Copper nevű társprocesszor az MC68000-tól függetlenül hajtja végre a saját programjait írott programot, az ún. Copper-listát. A képernyőt felrajzoló elektronsugár tetszőleges, előre meghatározott pozíciója esetén képes módosítani a számítógép bármelyik hardverregiszterét. A Copper segítségével így tetszőleges képpont-pozíciónál megváltoztathatjuk a paletta tartalmát, és a grafikus hardvert „átverve” a teljes színskálát a képernyőre varázsolhatjuk.

A Coppernek köszönhető egyébként az is, hogy egyidejűleg többféle grafikus üzemmódban levő screen is megjeleníthetünk a képernyőn. Gondoljunk csak arra az esetre, amikor egy rajzolóprogramban Lo-Res, Interlace, HAM módban vagyunk, és az egész screen lehúzzuk az egér segítségével. Ekkor a képernyő tetején megjelenik a Workbench screenje a maga Hi-Res, Non-Interlace üzemmódjában. Mind-

ez a Copper munkájának eredménye. Ha ez nem lenne, egy jelentős erőforrás hiányozna az Amigából.

Hogy még bonyolultabb legyen a helyzet, beszélhetünk Single-Playfield és Dual-Playfield képernyőről is. Mirovár mind-egy szó volt, az volt a Single-Playfield display. Dual-Playfield megjelenítés esetén két screen másolódik egymásra a Blitter nevű grafikus chip jóvoltából. A két screen közül a magasabb prioritású takarja a „mögötte levő” alacsonyabb prioritású. A mögöttes (alacsonyabb rangú) screen csak azokon a helyeken látjuk, ahol a magasabb prioritású screen háttér-színt tartalmaz. A két Playfield egymástól függetlenül kezelhető, semmilyen hatással nincs egymásra. Dual-Playfield megjelenítéshez legfeljebb 8 színű (3-as mélységű) screenek használhatók fel. A két screen más színregisztereket használ palettaszíneinek tárolására. Az alábbi táblá-

zatban a Dual-Playfield esetén lehetséges screen-kombinációkat foglaltam össze.

| Mélység (Playfield 1) | Használt színregiszterek | Mélység (Playfield 2) | Használt színregiszterek |
|-----------------------|--------------------------|-----------------------|--------------------------|
| 1 | 0,1 | 1 | 8,9 |
| 2 | 0-3 | 1 | 8,9 |
| 2 | 0-3 | 2 | 8-11 |
| 3 | 0-7 | 2 | 8-11 |
| 3 | 0-7 | 3 | 8-15 |

Ehhez az üzemmódhoz hasonló egyébként a GENLOCK áramkör által nyújtott kép, azzal a különbséggel, hogy ott külső videojel-forrásból származó kép látható a háttérben.

Tisztázván a legalapvetőbb fogalmakat, ráterhetünk a konkrét programozási kérdésekre.

Az előző részben ismertettem a könyvtárkezelés szabályait az Amigán. A könyvtárakról ezért itt csak annyit, hogy a grafika programozásánál leggyakrabban a Graphics és az Intuition nevű könyvtárakat kell használnunk. A legelső lépésben meg kell ismernünk, hogyan lehet egy screen és egy ablakot kinyitni, illetve bezárni.

Screeneket az Intuition könyvtár OpenScreen nevű rutinjával hozhatunk életre, és a CloseScreennel szüntethetünk meg. Az OpenScreen a -\$C6, a CloseScreen pedig a -\$42 opszetre „hallgat”. Tehát az assembly program elején az alábbi deklarációknak kell szerepelniük:

```
OpenScreen equ -$C6
CloseScreen equ -$42
```

Az OpenScreen szubrutin bemenetelként az A0 címregiszterbe az ún. NewScreen struktúra kezdőcímet kell betölteni. A rutinból való visszatérés után a D0 regiszter az ún. Screen-struktúra kezdőcímet fogja tartalmazni. (Ehhez hasonló struktúrák tömegére támaszkodik az Amiga operációs rendszere, ezért nem árt, ha idejében megbarátkozunk az adatstruktúrák kezelésével.) A NewScreen-struktúra egy általunk létrehozandó tömb, amelyet az alábbi módon kell felépíteni:

```
NewScreen:
dc.w LeftEdge, TopEdge ; a bal felső sarok
; koordinátái
dc.w Width, Height ; szélesség, magasság
dc.w Depth ; Bitplane-ek száma
dc.b DetailPen, BlockPen ; a fejjéc színei
dc.w ViewModes ; grafikus üzemmód
dc.w Type ; a screen típusa
dc.l Font ; Font-struktúra
dc.l DefaultTitle ; a screen neve
dc.l Gadgets ; Gadget-struktúra
```

```
jsr CloseScreen(a6) ;a screen megszüntetése
move.l ExecBase,a6 ;az Exec báziscíme
move.l GfxBase,a1 ;a Graphics báziscíme
jsr CloseLibrary(a6) ;bezárja a Graphics-ot
NoGfx: move.l IntuitionBase,a1 ;az Intuition bázisa
jsr CloseLibrary(a6) ;bezárja az Intuition-t
NoIntuition: ;
illegal ;kilépés exception-nel

IntuitionName: dc.b "intuition.library",0
GfxName: dc.b "graphics.library",0
even
IntuitionBase: dc.l 0
GfxBase: dc.l 0
Window: dc.l 0
RastPort: dc.l 0
String: dc.b "Hello World !!!"
EndOfStr:
even
NewScreen: dc.w 0,0
dc.w 640,256
dc.w 2
dc.b 2,3
dc.w HIRES
dc.w CUSTOMSCREEN
dc.l 0
dc.l ScreenTitle
dc.l 0
dc.l 0
ScreenTitle: dc.b "The Screen",0
even
NewWindow: dc.w 150,50
dc.w 300,100
dc.b 2,1
dc.l IDCMP_Flags
dc.l Flags
dc.l 0
dc.l 0
dc.l WindowTitle
NWScreen: dc.l 0
dc.l 0
dc.w 50,30
dc.w 350,160
dc.w 40F
WindowTitle: dc.b "The Window",0
```

dc.l CustomBitMap : CustomBitMap-struktúra

A DetailPen és BlockPen azt mondja meg az Intuitionnek, hogy milyen színekkel jelenítse meg a screenhez tartozó szimbólumokat, az ún. Gadgeteket (például a fejléceket, a háttérbe és előtérbe helyező szimbólumokat stb.). A Struktúra ViewModes nevű elemével az alábbi értékek VAGY kapcsolatát képezve lehet beállítani a screen grafikus üzemmódját:

LACE equ \$0004
EXTRA_HALFBRITE equ \$0080
DUAL_PLAYFIELD equ \$0400
HAM equ \$0800
HIRES equ \$8000

Például Hi-Res és Interlace screen esetén a ViewModesba \$8004-et kell írni. Ha már kész screen szeretnénk megjeleníteni, ezt a Type nevű elemet keresztül jelezhetjük, és ekkor a CustomBitMap nevű elemnek a kész screen ún. BitMap-struktúrájára kell mutatnia. Más esetben a CustomBitMap elemnek 0-t kell tartalmaznia. A Default-Title egy mutató, amely a screen nevének kezdőcíme mutat. A Screen nevének (mint minden stringnek) 0-ával kell lezárni! Ha nem elégszünk meg azokkal a szimbólumokkal, amelyeket a ROM kínál nekünk, akkor saját Gadget-struktúrát építhetünk fel, saját grafikai kellékekkel. Ha erre nincs szükségünk, a Gadgets nevű elembe egyszerűen 0-t írunk.

A screen megszüntetése a megnyitáshoz képest pofon egyszerű feladat. AZ A0 címregiszterbe kell betölteni azt a mutatót (a Screen-struktúra bázismutatóját), amelyet a DO-ban kapunk meg az OpenScreen rutintól.

Egy ablak (window) kinyitása a legegyszerűbben az Intuition könyvtár OpenWindow szubrutinjával történhet. Az ablakot bezáró funkció neve természetesen: CloseWindow. E két funkcióhoz az alábbi ofszetek tartoznak:

OpenWindow equ -\$cc
CloseWindow equ -\$48

Mielőtt egy „jsr OpenWindow(a6)” utasítással meghívánk az ablakot kinyitó rutint, létre kell hoznunk egy ún. NewWindow-struktúrát, és ennek a kezdőcímét az A0 címregiszterben kell átadnunk. Ebben az adattömbben írjuk le azt, hogy milyen ablakot szeretnénk. A NewWindow-struktúra felépítése a következő:

NewWindow:

dc.w LeftEdge, TopEdge : a bal felső sarok koordinátái

dc.w Width, Height : szélesség, magasság

dc.w DetailPen, BlockPen : a Gadgets színei

dc.l IDCMP_Flags : IDCMP jelzőbitek
dc.l Flags : Üzemmodi-beállítás
dc.l FirstGadget : a Gadget-struktúra eleje
dc.l CheckMark : menükezeléshez kell
dc.l Title : a window neve
dc.l Screen : a Screen-struktúra
dc.l BitMap : a BitMap-struktúra
dc.w MinWidth, MinHeight : minimális szélesség és magasság
dc.w Type : a window típusa

AZ IDCMP az Intuition Direct Communications Message Port kifejezés rövidítése. Ahhoz, hogy meg tudjuk érteni, mit takar ez az elnevezés, meg kell ismernünk az Input Device (magyarul: bemeneti eszköz) feladatát. Egyszerűen fogalmazva az Input Device az Amiga ROM programjának az a része, amely a felhasználó „cselekedeteit” (egy billentyű leütését, az egér elmozdítását stb.) észleli és feldolgozza. A rendszerszoftver felépítését az Amiga-sorozat első részében egy ábra formájában ismertetem. Az ábrán jól látszik, hogy az Input Device három dologgal áll kapcsolatban: a Console Device-szal, a Keyboard és Gameport Device-szal, valamint az Intuitionnel. Ez utóbbival az IDCMP porton keresztül tart kapcsolatot. Figyelmet érdemel, hogy ez nem egy fizikailag létező port — mint mondjuk a Parallel Port —, hanem egy jelzőbiteket magába foglaló, a RAM-ban lefoglalt terület. Az ilyen szoftverszintű portokra többek között a multitasking miatt van szükség. Az IDCMP bitjein keresztül szerezhet tudomást a futó program arról, hogy az ablakot a felhasználó be akarja csukni, el akarja mozgatni, vagy például a háttérbe akarja lapozni az egérről. Az IDCMP jelzőbitjeinek használatáról a sorozat következő részében lesz szó.

A NewWindow-struktúra Flags nevű elemének megadásával tulajdonságokat kölcsönözhetünk a leendő ablaknak:

WINDOWSIZEING equ \$0001 : az ablak mérete változtatható
WINDOWDRAG equ \$0002 : az ablak elmozgatható
WINDOWDEPTH equ \$0004 : háttérbe/előtérbe lapozható
WINDOWCLOSE equ \$0008 : bezárható
SIZEBRIGHT equ \$0010 : a méretváltóztató Gadget az ablak jobb szélén legyen
SIZEBOTTOM equ \$0020 : a méretváltóztató Gadget az ablak alján legyen

SMART_REFRESH equ \$0000 : az Intuition frissít
SIMPLE_REFRESH equ \$0040 : a felhasználó frissít
SUPER_BITMAP equ \$0080 : nincs szükség frissítésre
BACKDROP equ \$0100 : ún. háttérablak legyen
REPORTMOUSE equ \$0200 : információk érkeznek az egér helyzetéről és állapotáról
GIMMEZEROZERO equ \$0400 : az origó a window belső sarkában legyen
BORDERLESS equ \$0800 : nincs keret
ACTIVATE equ \$1000 : ez legyen az aktív ablak
RMBTRAP equ \$00010000 : a Right Mouse Button figyelése
NOCAREREFRESH equ \$00020000 : teljes frissítést az Intuition végzi

A NewWindow-struktúra Screen elnevezésű tagja mondja meg a rendszernek, hogy melyik screenen akarjuk megnyitni az ablakot. Ide a kérdéses screen Screen-struktúrájának kezdőcímét kell betölteni (azt, amit az OpenScreen-től a DO regiszterben visszakaptunk).

Ugyanúgy, ahogy a NewScreennél látuk, a BitMap változó segítségével itt is megadhatjuk egy kész window BitMap-struktúrájának kezdőcímét. Ha új ablakról van szó, akkor ezen a helyen 0-nak kell állnia.

A Min/MaxWidth/Height változók azt a határt szabják meg, amekkorára lecsökkenthető, illetve megnövelhető az egérről az ablak mérete.

Végezetül lássunk egy konkrét példát. Ez a rövidke program kinyit egy screen, azon pedig egy ablakot. Az ablakba egy rövid üzenetet ír ki a Graphics könyvtár Text funkciójának meghívásával. Az IDCMP flagek használatáról csak a következő részben lesz szó, ezért ez a program még nem használja az ablak kikapcsoló szimbólumát a befejezéshez, hanem egyszerűen az ESC billentyű megnyomására vár. Ha az utolsó, ILLEGAL utasítást a CLR.L DO RTS

utasításokra cseréljük ki, akkor programunk CLI szinten is futtathatóvá válik.

Tóth Zoltán

DOS -parancs

Köztudott, hogy a Commodore mikrogepek soros vonalra illeszkedő perifériái — a nyomtató, a rajzológép és a lemezegység — maguk is szinte teljes számítógépek — RAM, ROM, processzor(ok) együtt van ezekben az eszközökben. A lemez meghajtó operációs rendszere, a DOS, lehetővé teszi a hardverközeli programozást. A normál funkcióknak — lemezformázás, törlés, másolás — túl közvetlenül hozzáférhetünk a floppy-RAM-hoz, írhatjuk, olvashatjuk, és a floppy processzorának vezérlését saját programunkra irányíthatjuk; így az akár az alapgép koprocesszora is lehet.

Igaz ez a „legkisebb fivér”, az 1541-es meghajtó esetében is. Ennek a programozásáról már több könyv, folyóirat írt. Fehér foltja azért maradt még. Ezek egyikét szeretném most eltüntetni. Ússuk fel például a Data Becker sorozat „Nagy floppy könyvét”! A ROM visszafejtett listájában a 233. oldalon az utasításszavak táblázatában ott látjuk az & jelet. Az ehhez tartozó utasítást azonban sehol nem ismertetik. Az 1571-es meghajtó gépkönyve öt sorban intézi el — ráadásul hiányosan és hibásan. De mielőtt lehullna róla a lepel, tekintsük át a floppy közvetlen programozását szolgáló utasításokat!

A floppy-RAM írható az M-W (MEMORY WRITE) segítségével, bájtönként. Az így összeállított programunkat az M-E (MEMORY EXECUTE) utasítással futtathatjuk. Az esetleges eredmények az M-R (MEMORY READ) segítségével olvashatók ki. Ez a legalacsonyabb szint, „deszkamodell”-programjainkhoz.

A működő programok a B-W (BLOCK WRITE) utasítással a lemezre írhatók, onnan a B-R (BLOCK READ) olvassa vissza a floppy RAM-jába. A B-E (BLOCK EXECUTE) nemcsak visszatölti, hanem mindjárt át is adja a vezérlést. Az így lemezre vitt programok viszont a lemezen csak körülményesen kezelhetők: a blokkot foglaltatni kell tenni a felülírás elkerülésére; a VALIDATE még ilyenkor is törli, hiszen nem tartozik a tartalomjegyzékben egyik bejegyzéshez sem, és jó, ha megjegyezzük a blokk sávszektor-számát, mert ez a betöltéshez szükséges. Programunk a fenti ok miatt fájlmásolóval sem másolható.

Az előbbi hátrányokat szünteti meg az (eddig) rejtélyes & utasítás, amellyel a tartalomjegyzékben szereplő fájlokat tölthetünk be a floppy (!) memóriájába, rögtön futtatva is azo-

kat. Így lényegében a fájl neve a floppy utasításkészletének bővítésévé válik, hasonlóan az MS-DOS-hoz. Itt azonban az így kijelölt programok magában a lemez meghajtó rendszerében futnak.

Minél összetettebb a feladat, annál érdemesebb ezt a módszert használni, hiszen a floppy rendszerében futó programok a központi egység egyetlen bájttát és egyetlen mikrosekundum idejét sem rabolják el! Az előbbieket ismeretében néhány programunk máris elavultnak tűnik: az összes turbóprogramot lemezről töltjük be (ráadásul még turbó nélkül, azaz lassan), majd azt az alapgépben elindítva, egy részét — a floppy-programot — bájtonként visszairjuk a floppy-RAM-ba. Mennyivel egyszerűbb lenne ez az egyetlen utasítás: &:TURBO — s ezzel megtakarítanánk az oda-vissza adatátvitelt is. A programok kezdő- és végcímének megállapításához eddig be kellett tölteni vagy a lemezen végigolvasni azokat. Ehelyett egyszerűen átadhadjuk a program nevét a floppy-programnak, majd kiolvassuk a kérdéses címeket.

De essen szó most az & utasítás szintaktikájáról is! A lemezparancs-fájlok nevének első betűje tehát az & jel kell hogy legyen, utána a kettőspont nem kötelező, de ha a fájl nevében benne van, akkor a parancsban is meg kell adni! Egy parancsfájl felépítése a következő:

- 1—2. bájttal: betöltési cím alacsony/magas bájttal sorrendben
 - 3. bájttal: a programbájtok száma
 - 4—x. bájttal: a program
 - x+1. bájttal: ellenőrző összeg
- 256 bájtnál hosszabb program esetén itt ismét az 1. bájttól folytatódik a felírás. Az ellenőrző összeget a következőképpen kell kiszámítani: össze kell adni a startcím két bájttjának értékét, a program összes bájttjának értékét, a kapitány életkorát, az összeget alacsony/ma-

gas bájttra bontani, a magas bájt átvitelét az alacsony bájtához hozzá kell adni, ez utóbbi az összeg. Kétféle hibaüzenetet kaphatunk betöltés közben: OVERFLOW IN RECORD jelenik meg, ha a programbájtok tényleges száma nem egyezik a megadottal, és RECORD NOT PRESENT, ha az ellenőrző összeg hibás. Ilyenkor természetesen a program sem fut le.

Puding próbája az evés: gépeljük be a programlistát! Ez lefuttatva lemezre ír egy átszámozó floppy-programot. Ezentúl, ha kilences egység számú meghajtóra van szükségünk, be kell tenni ezt a lemezt, és kiadni az

```
"&DRIVE9" parancsot:
OPEN 1,8,15
PRINT # 1,"DRIVE9"
CLOSE 1
```

Akik rendelkeznek a DOS 5.1 vagy hasonló bitvitessel:

```
& &DRIVE9
```

A példa érthetőnek tűnhet, de szemléltetésre és tanulmányozásra jó. A legszebb program, amit ismerek, az NSZK-beli 64'er által közzétett Sub Dir System: az MS-DOS-ból ismert altartalomjegyzékeket — vagyis a strukturált lemezkatalógust — valósítja meg az 1541-esen, minden hardver- vagy programkiegészítés nélkül!

Most ismét az 1571-es meghajtóról. Az 1571 módban a DOS 3.0 hibája folytán a szintaxis kissé módosult, például:

```
& &:DRIVE9
```

Ugyanez igaz az 1551 és 1581 meghajtókra is!

Végül egy kérdés: miért nem olvashattunk erről a parancsról idehaza? A profik félve őrzik titkaikat, ez rendben. De mi köze a Commodore mikrogepeknek a profizmushoz? Én — amatőrként — mindenesetre szívesen állok minden érdeklődő rendelkezésére.

Zoltai Péter

```
10 OPEN 1, 8, 2, "&DRIVE9, U, W"
20 FOR I=1 TO 27: READ A
30 PRINT #1, CHR$(A) ; ; S=S+A
40 NEXT I: CLOSE 1
50 IFS > 2283 THEN PRINT "HIBA A DATA-SOROKBAN!"
60 DATA 000, 003, 023, 169, 041, 133, 119, 169
70 DATA 073, 133, 120, 096, 032, 077, 080, 083
80 DATA 032, 083, 079, 070, 084, 087, 065, 082
90 DATA 069, 032, 249
```

Számítógép a kézben

A számítástechnika és informatika fejlődése azt is jelenti, hogy egyre személyesebbé válnak a hozzájuk tartozó berendezések. A számítástechnikai eszközök személyes használatára könnyen kezelhetők, kis méretű és súlyú, kis fogyasztású berendezések kialakítását igényli.

Egy ilyen eszköz az angol Psion cég által gyártott és forgalmazott Psion Organiser II XP elnevezésű eszköz, amely — ahogy a nevéből is látni (organiser = szervező) — használója életének szervezését kívánja segíteni. A berendezés lényegében egy nagy teljesítményű zsebszámítógép — egy bővíthető rendszer alapgépe —, amelynek teljesítménye nagyobb, mint egy fizikai méreteiben nagyobb mikroszámítógépé. Magában foglal egy BASIC-hez hasonló nyelven programozható számítógépet, egy telefonkönyvet, kalkulátort, előjegyzési naptárt és egy órát. Igen lényeges, hogy a gépbe betáplált információ egy másik hasonló gépbe vagy egy IBM PC-be is átvihető.

RÁNEKÉZRE: KALKULÁTOR

Megjelenési formája egy szokványos kalkulátoréhoz hasonló. Az információ egy 8×6-os mátrix formában elrendezett többfunkciós billentyűzeten keresztül vihető be a gépbe, a megjelenítés kétsoros, soronként 16 karaktert tartalmazó folyadékrkristály kijelzővel történik.

Az Organiser valójában egy 8 bites mikroprocesszoron (típusa: HD6303X) alapuló mikroszámítógép, amelyet a teljes táplálás érdekében kis fogyasztású CMOS áramkörökkel alakítottak ki. A gép 32 kb-ot ROM-jai tartalmazza a működtető programrendszert, információ tárolására 32 kb-ot nem felejtő RAM szolgál. A gépbe helyezhető két ún. DATAPACK (memóriamodul), amely szintén információt képes tárolni. Két kialakításra kapható: RAM, illetve EPROM memóriát tartalmazó modul.

Az EPROM-os modul kapacitása 8 kb-át és 128 kb-át között lehet, és két módon használható. A gépbe helyezhető ún. DATAPACK információval feltölthető, majd kiolvására után törölhető (ezt formázásnak nevezik, és külön eszközt igényel, a neve: PSION FORMATTER), és újra használható. A modul másik felhasználási lehetősége a gép lehetőségeit kibővítő programok tárolása. A RAM-DATAPACK kapacitása 32 kb-át, és mivel törlése külön eszközt nem igényel, ezért elsődlegesen adattárolási célokra használható.

MENÜRENDSZER

Az eszköz használhatóságát alapvetően meghatározza a benne lévő programrendszer. Bekapcsoláskor az LCD kijelzőn egy menü jelenik meg, ebből választhatunk. A következő opciók önálló rendelkezésre:

FIND. A memóriában tárolt adatrekordok keresésére szolgál. A keresés a rekord bármelyik része alapján lehetséges.

SAVE. A beírt információk tárolhatók a belső RAM-ban vagy DATAPACK-ban.

DIARY. Ez egy határidőnapló funkcióját valósítja meg. Ha egy bejegyzett időpont aktuális, akkor figyelemreztető hangjelzést ad.

CALC. Egy matematikai függvényekkel rendelkező zsebkalkulátor funkciót valósítja meg. A műveletekhez tíz adattároló áll rendelkezésre.

PROG. A BASIC-hez hasonló OPL (Organiser Programming Language) programozási nyelven programot lehet írni és futtatni.

ERASE. A belső RAM-ban vagy DATAPACK-ban lévő rekordok törölhetőek a paranccsal.

ALARM. Maximum 8 figyelemreztető hangjelzés programozható be úgy, hogy egyszer, óránként, naponta vagy hetenként szólaljon meg.

TIME. A belső, állandóan működő óra állapotának (idő, dátum) megjelenítésére és beállítására szolgál.

INFO. Információt jelenít meg arról, hogy mennyi helyet foglalnak el a határidőnapló bejegyzései, mennyi hely van a belső RAM-ban és a csatlakoztatott DATAPACK-okban.

COPY. Információ másolása a DATAPACK-ok és a RAM memória között.

RESET. A rendszer "hüdeginđítása". A belső RAM-ban lévő minden információ törlődik.

OFF. Ez a funkció kikapcsolja a készüléket.

A gép használhatóságát alapvetően ezek a beintegrált funkciók határozzák meg. Fontos megjegyezni, hogy a menüpontok bármelyike (az OFF kivételével) törölhető, illetve más funkciókkal bővíthető. A menü bármelyik pontja vagy a funkció közébtűjével, vagy a kurzorral történő mozgatással, majd az EXE billentyű megnyomásával aktivizálható.

BŐVÍTETHŐ FUNKCIÓK

A gép más funkciókkal való felruházása három módon történhet:

- a DATAPACK-ban vásárolható programok segítségével;
 - a HD6303X mikroprocesszor gépi nyelven történő programozással,
 - az OPL nyelven írt programokkal.
- Ez utóbbira illusztrációként nézzünk egy példát. Írjunk egyt, az aktuális időt megjelenítő programot OPL nyelven! (Bár a TIME funkció pont erre szolgál...). A programot billentyűzeten be a gépbe a következő sorrendben:

1. A gép bekapcsolása után válasszuk ki a PROG menüpontot.
2. A megjelenő almenüből a NEW menüpontot.
3. A megjelenő A: után írjuk be a program nevét: NOW, majd nyomjuk meg az EXE billentyűt.
4. Ezek után megjelenik a program neve NOW-formában, és elkezdhetjük a begépelést. (Zárjelben a magyarázat, nem kell begépelni. Az új sorba az EXE billentyűvel léphetünk.)
- CLS (kijelző törlés)
- PRINT "PONTOS IDŐ", HOUR; ":", MINUTE (megjelenítés)
- GET (billentyűnyomásra várakozás)
5. A begépelés végén nyomjuk meg a MODE billentyűt, és a megjelenő menüből válasszuk ki a TRAN (fordítás) menüpontot. Ennek hatására a program gépi kódra fordíródik, és utána a megjelenő SAVE A: végrehajtásával (EXE gomb megnyomása) a gép memóriájában tárolódik.
6. A gép menüjéhez hozzáfűzhető és ezután bármikor végrehajtható, ha a főmenü megjelenésekor a MODE billentyűt nyomjuk meg, és a megjelenő INSERT ITEM szövegére választul a NOW sorozatot gépjeljük.

KIEGÉSZÍTŐK

A géphez számos kiegészítő hardvereszköz és felhasználható program kapható. Ezek közül felsorolunk néhány fontosabbat.

FORMATTER. A berendezés lényegében egy UV-fényvel működő EPROM-törölő készülék. Az EPROM-DATAPACK-ok törlésére szolgál. A behelyezett darabot kb. 30 perc alatt törli, és utána újra használható.

PSION PRINTER II. Hordozható, teljes, hőnyomatós mátrixnyomató. Nyomatási sebessége 1 sor/s. A karakterek száma egy sorban: 80. RS232 soros illesztés az Organiserhez.

COMMS LINK. Az Organiser felső végén lévő csatlakozó nyílásba dugható soros illesztőegység. Segítségével az Organiser letszöveges soros perifériával rendelkező egységgel: nyomtatóval, számítógéppel, modellem stb. összekapcsolható. A COMMS LINK menüvezérelt programmal működtethető. Az adatátvitelt egyszerűbb megoldása érdekében lemezen kapható az IBM PC-n, illetve a Macin-

toshon futó kommunikációt megvalósító program. Fájlok és OPL programok mindkét irányú átvitelét lehetővé teszi. Terminálemulációval elektronikus levelezés és telexkezelés is megvalósítható. Az adatátviteli sebesség 50 és 9600 baud között állítható.

FORMULATOR. Szimbolikus formában megadható képletek, összefüggések számíthatók a program segítségével. Az összefüggések szerkeszthetők és tárolhatók. A program több mint 250 beépített függvényt tartalmaz az üzleti élet, a tudomány és a műszaki élet területéről.

MATHS PACK. Matematikai programcsomag. **DEVELOPER.** Ez egy IBM PC-n futó, OPL program fejlesztésére szolgáló programcsomag. Az IBM PC-n lehetséges az OPL programok írása, szerkesztése, futtatása, hibakeresése és tárolása. A hibátlan program a COMMS LINK-en keresztül az Organiserbe átvihető, és ott a már hibátlan program futtatható.

Az Organiser szabadban és irodákban történő alkalmazására számos perifériát fejlesztettek ki (Corporate products), amelyek elsősorban a hordozható terminálnál való alkalmazását segítik. Ezek közül néhány:

- Vonalkódolvasó, EAN, UPC, CODE39, ITF, CODABAR, TELEPEN, PLESSEY, MODIFIED PLESSEY és CODE 128 szabványú kódok olvasására képes.
- Mágneskártya-olvasó. Sankyo MCM-300R típusú kártyaolvasó illesztése az Organiserhez.
- PSION Mark III másoló. Többfunkciójú készülék. Leggyorsabb alkalmazása: egy DATAPACK tartalmának átmásolása maximum nyolc DATAPACK-ba. A készülék sorosan vagy párhuzamosan kapcsolódhat az IBM PC-hez, és így lehetséges az adatátvitel a PC és a DATAPACK-ok között, mindkét irányban.

MŰSZAKI ÉLET ÉS KERESKEDELEM

Általában a műszaki és kereskedelmi élet területére jellemző, hogy az információk nem egy helyre koncentrálnak jelennek meg, hanem több forrásból származnak, és elközösítve hálózatok kialakításánál és a felismerés döntő szerepet játszik. Azonban a hálózatok kialakítása és működtetése költséges megoldás. Néhány alkalmazási lehetőség:

- villanyóra- és vízára-leolvasás eredményei az Organiserben gyűjtethők, a feltöltött DATAPACK IBM PC-vel kiolvasható és feldolgozható;
 - erdőkben lévő fák adatainak gyűjtése és feldolgozása;
 - raktárkészlet-nyilvántartás;
 - adatgyűjtés több telephelyre kiterjedően;
 - bolti eladásoknál pénztárgép-funkció.
- A felsoroltakon kívül természetesen még számos alkalmazási lehetőség van.

ÖSSZEFOGLALÁS

A fentiekben összefoglaltuk az Organiser legfontosabb jellemzőit. A fejlődés azonban nem állt meg. Az ismertett XP modell továbbfejlesztéseinként már megjelennek a még többet tudó LZ és LZ64 változatok. A legfontosabb, már bekapcsoláskor is szembetűnő változás, hogy a kijelző négy soros, soronként 20 karaktert tartalmaz. Mindkét új modellnél a ROM az XP-ének duplája: 64 kb-át. Ez azt jelenti, hogy a rendszer-program kétszeresére nőtt! A képernyő-kezelés is korszerűbb lett.

Új funkcióként megjelent a NOTE (jegyzetfüzet) funkció, amely lényegében egy szövegszerkesztő. A WORLD funkcióval a világ bármely pontján érvényes időpont kérdezhető le. Az XFLES funkcióval más gyártók adataiból is kezelhetők. A UTILS funkcióban számos alfunkció található: fájlokban keresés, fájlok másolása stb.

Az LZ-132 kb-át, az LZ64-et 64 kb-át RAM-mal szállítja.

Az LZ-132 kb-át, az LZ64-et 64 kb-át RAM-mal szállítja. A fentiekhez hasonló berendezések gyártásában a Psion mellett más cégek is fantáziáltak látnak. Kapható a Sharp cég IQ7000-es típusú gépe, és a piacon van a Casio is az SF-4000-es, az SF7000-as, az SF7500-as és az SF8000-es típusú jeltermékeivel.

A Solarsoft kínálatából

Mint arról már 1989/11-es számunkban hírt adtunk, új szoftverkategória jelentkezett a magyar piacon is: a shareware. A Cédrus Kiszövetkezet Magyarországon elsőként vállalkozott arra, hogy a vékony pénztárcájú, PC-hez, PC-kompatibilis géphez csak alkalmászerűen hozzájutó, tanulni, illetve ötletet meríteni kívánó felhasználók számára is

megfelelő árú szoftvertermékeket forgalmazzon. Úgy véljük, olvasóink között is sokan akadnak, akik csak ezt a szoftvert tudják megfizetni, ezért határoztuk el, hogy rendszeresen közzétesszük a választást, eligazodást megkönnyítő Solarsoft-katalógus információt.

Név: COLORADO ENTERPRISES C TUTOR v. 2.12

Szerző:
Gordon Dodrill, New Mexico, 1987.

Leírás:
A C nyelv tanulásához ad segítséget a közel 80 példaprogramból (*C) és rövid magyarázatokból (CHAP*.TXT) álló gyűjtemény.

A fontosabb témakörök:

- programok felépítése
- vezérlőszervezetek
- függvények, függvényhívások
- definíciók, makrók
- sztringek, tömbök
- mutatók (pointerek)
- I/O műveletek
- fájl I/O-műveletek
- struktúrák
- memóriaterület-igénylés
- karakteres és bitműveletek
- egyéb példák
- Visual Calculator

C-ben írt program, amellyel szinte programsoronként beírt, illetve tárolt adatokon számítások végezhetők. Saját kis HELP-je van, a VC.DOC pedig rövid oktató leírás.

Alkalmazható függvényei:

ABS(), SQRT(), EXP(), LOG(), SIN(), COS(), ATAN(), FACT().

Dokumentáció:

A COMPILER.DOC fájl a C-verziók installálását, az 1 vagy 2 floppy, illetve a hard diszk használatát, az első program írását és indítását mutatja be az alábbi C nyelvi fordítók kapcsán:

- Borland Turbo C
- Computer Innovations C 86 2.3
- Ecosoft C 3.14
- Lattice C 3.0
- Mark Williams C 3.02
- Microsoft C 3.0, 4.0, 5.0, Quick C 1.0
- MIX C 1.00

Melléklet: printelőprogram, amellyel a C-programokat és a számszöveges magyarázószöveget lehet kiírni.

Konfiguráció: —

Név: MYED v. 2.0

Szerző:
Ford Software

Leírás:
Microsoft QuickBasic-ben írt egyszerű szövegszerkesztő. A programokban egyszerű HELP is van (Alt-H). A forrásvátozat kiemelezhető, javítható. Erre szükség is lehet, mivel a program néhány billentyű féltreállításával „kiakasztható”.

Csak 500 sort tud kezelni (64k basic szegmens); a számos szövegszerkesztő között ezért nem túl hatékony. Mágneslemezen a rezidens mód nagyon lassú, ezért az egyszerű módú használata javasolt.

Elsősorban a BASIC nyelv tanulásához és a szövegszerkesztők működési elvének megértéséhez nyújthat segítséget.

Dokumentáció:

A MYED.DOC tartalmazza a használatához szükséges információkat (hardverigények, Alt-betű parancsok leírását, az installálást, az egyszerű és a rezidens mód, a szín és a hang beállítása stb.).

Konfiguráció:

Használatához hard diszk és legalább DOS 3.0 javasolt. EMS kiterjesztett memóriával rendelkezőknek: rezidens felhasználás esetén a program nem foglal helyet a RAM-ból.

Név: EGA UTILITY

Szerző:
Több cég, több személy, 1985–87.

Leírás:

- Segédprogramok EGA-monitorhoz.
- CLS+színek beállítása
 - színpaletta bővítése, cseréje
 - a 43, illetve 25 soros mód átállítása
 - kibővített ANSYS.SYS
 - NANSY.SYS — egy új konzolmeghajtó
 - help-screen
 - új font(ok)
 - EGACAM képernyőmásolás
 - Wordstar 3.3 — 43 soros módban
 - stb.

Dokumentáció:

Rövid leírások programonként.

Konfiguráció:

EGA-kártya és -monitor szükséges.

Név: QBWARE/1 v. 1.1

The QuickBasic Interface Libraries v. 1.1

Szerző:

AJM Software Co., Arrada, Marcel Madonna, 1987.

Leírás:

Kiegészítés a BASIC használatához.

A 3 fő fájlból:

QBWARE.EXE a QBASIC 2.0–3.0 RUN-jához hiányzik!
QBWARE.QLB a QBASIC 4.0 RUN-jához megvan!
QBWARE.LIB az .EXE fájl előállításához hiányzik!

Használata QB 4.0 esetén:

- A QBWARE.QLB és .LIB állományainak a QBASIC könyvtárba való bemásolása után
- QB /L QBWARE-rel, illetve
 - QB programnév /L QBWARE-rel indítható.
- QB 2.0–3.0 esetén:
- A QBWARE .EXE és .LIB állományok bemásolása után
 - QB /L QBWARE.EXE, illetve
 - QB programnév /L QBWARE.EXE.

Dokumentáció:

A kiegészítés részletes leírását a QBWARE.DOC-ban, a teljes felsorolást a QBWARE.TXT-ben találhatjuk meg.

Konfiguráció: —

ADOK

Amiga programok nagy választékban eladóak Erdéskőúti lehel: Keresztész Gábor. Tel.: Lakó köz 11. 1142. Tel.: 1643-452.

AMIGA !!! THE WINNER IN HUNGARY !!! Legjobb programok minden nyelvségi nyelven. Szófia Laci, Budapest, Tavirózsa 5. 1161. Tel.: 1848-471.

Amiga programok adok-veszek-cserélek. Választókérdést listát küldök. Helyes Amiga számítógéppel. Gyálszék Ákos, Budapest, Erdőrt. u. 42/70. 1026

Amiga programok olcsó eladók (30 Ft/lemez). Kérésre listát küldök. Ókó István, Budapest, Veres Pálné u. 13. 1053. Tel.: 1573-193.

ATARI-kés figyelem! A legjobb programok minden hónapban, minden anyagi segítségben. Szófia Laci, Budapest, Tavirózsa 5. 1161. Tel.: 1848-471.

C Plus/A magról, joystickkel, szakirodalmi, programok eladó. Irányár 12 000 Ft. Kisajzáslás, Tancsics u. 13. 5318

C Plus/A lemezelt eladom. Tel.: 1285-917

Commodore Plus/A magról, 1 db joystickkel, könyvekkel és programokkal eladó. Zalaapáti, Ády u. 6. 8741. Erdéskőúti dűlőn 3 óra után lehel.

C Plus/A + magról + 700 program + joystick + fénycsura + szaknyelvi eladók! Kérésre listát, Hajdúszoboszló, Hegyes u. 29. 4200

HI! C64-re a legfrissebb programok eladóak lemez, vagy más lehetőségek között-in (50 Ft/lemez) Istenföldgyógy György. Nyiregyháza, Egreskert u. 56. 1919. 4400. Nagyon friss programok esetén cserélni lehetőségek!

GRETTINGS TO VIP, DAO, TGS, D-PS, SILVIO/RI, GENIUS, BOSS

C64 és színfalas programok eladók (10 Ft/db), cserélek is. Baksa! Tamás, Harsány, Klauzál u. 6. 7813

C64 és Amiga programok eladók. Úgyanitt 5,25-os lemezek 55 Ft/db. Kanza Viktor, Siofok, Fenyves s. 11. 8600

C64 programokat adok kezettől (3 Ft/db). 1500 programot listát küldök. Tóth Károly, Nagykálló, Ády út 28. 4320

C64-hez tizenháromezer (1) TURBOBIK és TURBOAPE számítógépes eladók. Programok, cserélek is eladó, csomagolajom 2000 program de 250 forint. Szabó Csaba, Ócsa, Úllósi út 28. 2364

Commodore 64/II - 1530-an magról + jó játékok + joystickok eladók (22 000 Ft). Cím: Nagykőrös, Entősen K. u. 12. 2750. Tel.: (20) 51-805

C64-re új programokat adok kezettől (200 Ft/kazetta) és lemezen (100 Ft/lemez), kezettől vagy lemezt kérek. Cím: Körtör Isre, Sopron, Bözna Gy. u. 28. 9400

C64 programok figyelem! Defonar of the crown, Grand prix circuit, The last ninja 1-2 szalagon eladó! (Szusza egyenes körű 10 per) darabja 100 forint! Széplény Károly, Dunajváros, Martinyos út 17. IX/1. 2400

C64, C Plus/A, E Enterprise és más géptípusokra hardver-, szoftverfejlesztés, oktatás, EP-PRG-10 Ft/db óra, programok konvertálása géptípusok között. Cartridge-ko 700 Ft-tól. Mi minden változik! Küldjön választókérdést! 1519, Pf. 376

Enterprise programok eladók (10-50 Ft). Választókérdést listát küldök. Erdéskőúti kizárólag levezelt lemezzel, Budapest, Delej u. 51. W. Ith. IV.25. 1089

Enterprise programokat adok, veszek, cserélek. Szuper játékok és felhasználói programok + teremtőjelesek. Olcsó ár: 10-20 Ft. Választókérdést listát Lakatos Balázs, Keszthely, Fodor Isre u. 14. IV/14. 8360

Enterprise programok eladó lemezzel és kezettől. Listát választókérdésekkel küldök. Gorbe Tamás, Hémezvárosbány, Őrmező E. u. 2. 1/1. 6890

Enterprise 72 floppy + illusztr + Epsilon + beágyazott, szakirodalmi, lemezen és kezettől kasszi 400 program, külön-külön is eladó. Nyitrai, Szolnok, Rágó A. u. 16. 1.6. 5000

Enterprise programok eladók, 10-50 Ft. Választókérdést listát küldök. Zemen László, Budapest, Kacs u. 14/1. fikt. 9. 1180

Enterprise számítógép magról + EXOS eladók MM 1800/900, MM MF 6400, MM MF 3200, Olivetti AI 090-es lemezmagjátok. Nyáry János, Győr, Tóko u. 2/A. 9025

PRIMO A32-es személyi számítógép olcsó eladó. Erdéskőúti Kovács József, Heves, Betyén G. u. 22. 3360

TCV-er SEMOL, ROL, CIRCUS, karaktergrafika ... Funkciókat segítő prg-on eladó (200 Ft kezettől). Választókérdést tájékoztatást küldök. Kims István, Mosonmagyaróvár, Széplé u. 2. 4275

TVC-tulajdonosok figyelem! Előnévű az ELITE TVC. 64-ra ártírt változata. Ár: 350 Ft. Megrendelhet más saját fejlesztésű programot együtt! Horj István, Városlőd, Hyát u. 20. 6033

ZX-Spectrum + számítógép 200-300 ártírsolható programmal esetleg fénycsurával nyrúdron eladó. Kristóf Attila, Zalaegerszeg, Általégetű út 173. 11/26. 8900

Spectrum 48/128 programok olcsón, nagy választékban, garanciával kaphatók. Minden megrendelőnkhez szuper ajándékot kap. Választókérdést részletes katalógust és tájékoztatást küldök. Kiss Gábor, Győr, Vajkay u. 31. 9028

ZX-Spectrum (magról, Kompost, fénycsura) kb. 40 kezettől, irrodalmi. Irányár: 17 000 Ft. Seikosha GP305 printer is eladó. Gyulya Áron 1367-543

ZX-Spectrum (48 k), joystick + tartófény, fénycsura + tartófény + program Szabo magról eladó (12 000 Ft). Márkus Csaba, Zalaegerszeg, Klauza Gy. u. 6. 8900

ZX-Spectrum (48 k) alig használt, nagyon jó állapotban lévő számítógépet, félig működő magrólval, 3 db kezettől, 10 db szaknyelvi eladók. Ár: 11 000 Ft. Kiss Henrik, Budapest, Határ utca 103. 1213

ZX-Spectrum programok adok, veszek vagy cserélek. Klein Péter, Kompoli, Akácok út 18. 3356

ZX-Spectrum magról a legjobb játék programok eladók! Inyegyen katalógus! Pillay Gábor, Pécs Bajcsy-Zs. u. 4. 3/9. 7622

ET (8 Mhz-em) 640 kb RAM-al, 1.2 MB-os floppyval, négy 14 collos monitorral, MDG kártyával, 83 gombos billentyűzettel, valamint 2 db 12 MHz-es AI alapegység megvásárolásért Árcs eladó. Sárosi Sándor, Kőrcy, Zlatos út 1/A. Tel.: (34) 72-044

A BOSSZU és új **VALLEYGAT** 1-11. c. magról nyelvi, illusztrált káreljátékok megrendelhető: Réthly István, Esztergom, Bocskorosokút út 28. Árúk: lemezen 370, kezettől 340 Ft.

Commodore 64-es 880 mt Plus2-4/floppy szücs! Tausch u. Briefparter von Ingarn. Erdkudingen bei Rejtő 187-1280 von 179.

Eladom még nem használt 1551-es floppival 13 500 Ft-ért. Cserébe érdekel MPS 803 printer. Lukács Albert, Budapest, Klauza u. 6. IV/12. Tel.: 177-5240

Eladók Gmó kezékony és Amiga DDS kézikönyv magyar nyelvű fordításra. Vidra Nagy Leona, Bp. XII. György A. u. 30. A/2. 1125

Eladó: összes Mikromagazin, C-Ujság, csak együtt, féláron. Eladó még 64-es Sordern, magazin, Dwellt, Compute mt, 818 szaknyelvi Plus-4/16 és 1 db joystick. Rejtő/187-1280 17 óra után.

Eladó Angol-magyar, Magyar-angol szótár, lemez, fordítás C64-es és Plus-4-es gépre. Szilvási László, Budapest, Egresy út 113. 175. 1418

Ha a legfrissebb programokkal akarsz játszani, ha a pipód a teljeséggel igénylése, akéred kékben, akéredet Magyarországon a legolcsóbbban, az ma már nem vagy nélkülözök. Kérd részletes tájékoztatást! (Válaszboríték) Hyléi Zsolt, Sopron, Állomás út 5. 9400

Magyarországon már hetes Amiga és C64-es programját vagy Evi! Eddie-1! A legkedvezőbb árak Magyarországon! "THE BEST YOU CAN GET IN HUNGARY!" By ME. VAM. Levelezési Győr, Pf.: 40. 9007. 1058-os vázlat!

NEC FD 1157 lemezmagjátok, 3M OSD2 lemezek Enterprise programokkal eladók. Kőrcy, Szolnok 5. Pf.: 49. 5005

ROCKMOTION, SCREEN-GRAPHIC, DOKU-64 és egyéb programok/lemez eladók. Medve Attila, Záhony, Kárpát u. 37. 4625

CSERÉLEK

Commodore PCI német nyelvű kézikönyvét, PC 10/20/40 MS-DOS 3.2 és OS-BASIC 3.2 referenciakönyvét angol nyelven, esetleg magyarra cserélnék. Szabó Zoltán, Budapest, Ády E. u. 129. B. 95. 1221

Commodore programokat cserélek vagy eladók kezettől (10 Ft/db). Cseré esetén listát kérek. Mészáros Zoltán, Baja, Klauza u. 11. 1/6. 6500

C64 játékok és felhasználói programokat cserélek lemezen. Pálvics Zsolt, Budapest, Hegedűs Gy. u. 45-47. 1136

C64 és Enterprise játékok és felhasználói programok cseréje kezettől és lemezen. Színfalas programok! Listát kérek és küldök. Bálint Zoltán, Dunavarsány, Iskolka u. 47. 2336

C64 és C Plus/A-es programokat cserélek lemezen. Pálvics Zsolt, Budapest, Hegedűs Gy. u. 45-47. 1136

C64 és Enterprise játékok és felhasználói programok cseréje kezettől és lemezen. Színfalas programok! Listát kérek és küldök. Borsos István, Mosonmagyaróvár, Hold út 8/B. 9200

Enterprise programok cseré-vétel-eladás. Listát kérek! Tóth Zoltán, Ogldás, Szék. u. 9. 2700

IBM PC-re felhasználói és játékok programot keresek. Listát kérek! Csak cserétre érdekel! Cím: Krauthorn Mihály, Székesfehérvár, Növök K. u. 22. 11375. 8000

Ez a rovatunk KODEX 2000 szövegszerkesztővel készült.

Ismét Garay-verseny

Pályázati felhívás

A Neumann János Számítógéptudományi Társaság, a székszárdi Garay János Gimnázium és a Mikroszámítógép Magazin szerkesztője 1990-re is programirásai pályázatot és programozási versenyt hirdet általános iskolás, középiskolák és elsőéves főiskolai, illetve egyetem hallgatók részére az alábbi kategóriákban:

1. Bármilyen iskolai tantárgyhoz kapcsolódó vagy bármilyen ismeret megtanulását segítő tanulóprogram (courseware), illetve
2. Új általános iskolai tanulóprogram fejlesztése.

A pályázatokat az alábbi gépekre lehet beküldeni: HT—1080Z, Sinclair ZX-Spectrum, Commodore 16, Commodore Plus/4, Commodore 64, Videoton TV-Computer, IBM PC és kompatibilis gépek, PS/2.

Az időszerű, kísérletekkel, a pályázati felhívást néhány külföldi diákszervezetnek is megküldjük.

A zsűri külön bírálja el a pályázathoz benyújtott programleírásokat (dokumentációkat), a legjobb(ak) különdíjat kap(nak). Ugyancsak különdíjat kapnak azok a legjobb tanulóprogramok, amelyeket a szerzők valamilyen magas szintű szerzői nyelv segítségével fejlesztettek.

A pályázat benyújtási határideje: 1990. február 15.

A pályázati programjaitak mágneszalag-kezelés (a kezettől többször felvéve) vagy mágneszalag (floppy), rövid kezelési leírását mellékelve (mit tud, hogyan működik a tanulóprogram vagy a

játék, hogyan kell kezelni, mi a tanulóprogram célja, milyen korszerű tanulóanyagok szánták, kapcsolódik-e valamilyen tantárgyhoz, vagy a célja például ismeretterjesztés, esetleg vizsgára való felkészítés vagy felkészülés stb.), jellegesen (a jelleg, valamint a pályázat neve és címe), iskolája nevét és címét egy külön lezart borítékban) küldjék el a Mikroszámítógép Magazin szerkesztőségének címére: 1371 Budapest, Pf. 433.

A szerkesztőségnek joga van a pályázaton részt vevő programok közlésére, amirt a szokásos honoráriumot fizeti.

A döntő 10—10 résztvevőjét a beérkezett pályázatok alapján az NJSZ Tagozati, a Garay Gimnázium tanáraiból és a Mikroszámítógép Magazin munkatársaiból álló előztéri választja ki. Az előztérinek joga van — ha nem ezerből belegendő pályázat, vagy a pályaművek színvonala nem éri el a kívánt szintet — a döntő résztvevőinek számát csökkenteni.

A döntő, amelyen az előztéri által kiválasztott programok versenyeznek, Székszárdon, a Garay János Gimnáziumban rendezendő Garay-napok alkalmából, 1990. március 13—14-én lesz.

A döntőbe jutott versenyzőket a Garay János Gimnázium vendégül látja.

Mindkét kategóriában díjat kap az első három helyezett. A zsűri ezenkívül különdíjakat is kiadhat. A zsűrinek joga van a díjak számát csökkenteni vagy a kategóriákat összevonni.

Salátné Bocsi Éva:

Adatállományok tervezése és kezelése
(Budapest, 1989. SZÁMALK, 153 oldal.
Ára: 230,— Ft.)



A könyv a számítógéppel támogatott adatfeldolgozó rendszerek külső tárolóeszközökön tárolandó adatállományainak tervezésével foglalkozik. A megalkotásról szól az első rész, a második rész pedig a logikai és a fizikai adatmodell meghatározásait tartalmazza, bemutatva és felhasználva az adatkezelés hagyományos útjának eszközeit.

Minden gazdasági szervezetnek információra van szüksége, az információkat pedig a szervezet adatfeldolgozási funkciója biztosítja. Az adatfeldolgozás-hoz szükséges erőforrások (eszközök, emberek, adatok), az e funkciókat megvalósító eljárások és az eredmény, az információ együttesen alkotják az információrendszert. Az adat- és eljárásrendszer csak úgy láthatja el jól az információrendszeren belüli szerepét, ha a valós világnak megfelelő és az adott szervezet szempontjából lényeges tényeket és összefüggéseket tartalmaz. Ezek sajátosságait modellezéssel lehet feltárni. A modellezés az információrendszer elemzésének és fejlesztésének eszköze.

A kötet alapozó jellegű, elsősorban kezdő szakemberek számára készült, de használnak fogorvosok, mérnökök, akik korábbi ismereteket kívánják feleleveníteni, mivel az adatkezelés hagyományos útjának, eljárásainak alapos ismeretét a fejlettebb módszerek alkalmazáshoz sem nélkülözhetik.

Bernhardt, Rolf:

A számítógéppel támogatott tervezés CAD/CAM alapfogalmai
(Budapest, 1989. Műszaki Könyvkiadó, 85 oldal.
Ára: 98,— Ft.)

A mérnöki tervezésben a képernyő szerepe nagymértékben megváltozott. A hagyományos rajzolás technikák egyre inkább háttérbe szorultak. Az új számítógépes eljárások egyre tökéletesebbek, és a gyakorlatban mind jobban használhatók. Ezért egyre több vállalat foglalkozik a CAD/CAM bevezetésével.

A könyv a konstrukciós tervezés területeinek bemutatásán és a szükséges fogalmak értelmezésén keresztül ismerteti a kiindulási helyzetet, majd a CAD/CAM bevezetésének alapkérdéseit és a grafikat középpontba állító tervezérendszerrel szemben támasztott követelményeket. Elemzi a tervezés mai és a jövőben kialakítandó munkamódszereit, a szükséges és a rendelkezésre álló információkat, a darabjegyzék készítését és a gyártástervezést is. A CAD/CAM technológia mai állását három részen elemzi a szerző: az első a rendszer felépítésével, a második a számítógéppel történő kommunikációval foglalkozik, a harmadik pedig a hardver, illetve a szoftver sajátosságait tárgyalja. Foglalkozik a könyv a számítógéppel támogatott rajzolással és a CAD segítségével végzett tervezéssel, mivel a CAD/CAM alkalmazá-

sát nagyon gondosan elő kell készíteni. Ismerteti a CAD/CAM bevezetésének legfontosabb szervezési kérdéseit és a tapasztalatokat.

Végül a CAD/CAM környezeti hatásainak áttekintése, az irodalomjegyzék, a szakszövegek és azok értelmezése teszi teljessé a könyvet.

dBase III PLUS
Novell NetWare kapcsolat
FoxBase+, CLIPPER
1—2. kötet

(Budapest, 1989. LSI ATSZ, 435 oldal.
Ára: 479,— Ft.)

A könyv célja az IBM PC-re készült adatbázis-kezelő programcsomagok: a dBase III Plus, a Clipper és a FoxBase+ ismertetése.

A kötet a számítógépes adatfeldolgozás fejlődésének rövid áttekintés után az itt vizsgált rendszerek alapjait, a relációs adatbázis-kezelés filozófiáját ismerteti. A dBase-2 leíró fejezeteket egy-egy, lehetőség szerint minden esetre kiterjedő példa követi. A dBase-hibabejelentő magyar megfelelőjét nem mindig szó szerint, hanem valódi jelentésük szerint adja meg, és magyarázatokkal is kiegészíti.

Részletesen ismerteti a dBase és a lokális hálózatok, például a Novell NetWare kapcsolatát, a hálózati programcsomagot, telepítését annak feltételeivel és használatát, a dBase-ből elérhető hálózat jellegű és adatvédelmi funkciókat, titkosítási lehetőségeket — felhasználások módjával együtt —, a hozzáférési jogok kezelését.

A Clipper ismertetése az 1985-ös téri verzióból indul ki, de kiegészül az 1986-os és 1987-es verziók tapasztalataival is. Ezt a programrendszer-leírást is példa egészíti ki.

A FoxBase+ felülről teljesen kompatibilis a dBase III Pluszal és önmagában is használható. Bár a nyugati piacon már megjelennek a 2-es verziók, itt még a Foxbase 1.21 verziója az elterjedt, így a szerző is ezt ismerteti, kitérve a 2-es verzió újodásaira is.

Új képkorszak határán
A számítógépes grafika és animáció
kezdete Magyarországon
Összeállította: Paternák Miklós
(Budapest, 1989. SZÁMALK, 188 oldal.
Ára: 380,— Ft.)

A nagyon szép kiállítású kötettel az volt a kiadó fő célja, hogy megmutassa és egyúttal dokumentálja a számítógépes művészet (grafika és animáció) első éveinek, évtizedének magyarországi fejleményeit. Mindez a nagyközönség számára is érdekes és élvezhető formában, színes képanyaggal és informatív frások összegyűjtésével tette.

A könyv a különböző pályákon, szakmákban dolgozó emberek munkáit gyűjti egybe. A sokféleségben, amely a bemutatott anyagból is észrevehető, néhány közös szempont mégis iránymutató volt: elsősorban a kép, annak számítógépes segítségével megmutatható formája, módosulása, az eltérések, a különböző irányok feltárása. Bár a szerkesztő törekedett a teljességre, nem kívánta az összes magyar alkotót bemutatni; célja volt viszont, hogy lehetőleg minden létező képtípus és alkotói attitűd képviselve legyen, beleértve a közvetlen előzményeket is.

Jelzőként, a hobbi és az alkalmazott-alkalmi használattal szintén tartalmaz a kötet karikatúrát, emblémát és digitális portréit is.

A záró rész bőséges, színes képanyaga bemutatja a technika lehetőségeit, illetve korlátait, valamint azt is, hogyan lehet egy eszköz korlátain szellemi koncentrációval túljutni.

A mágnesszalag reneszánsza

A szakemberek már-már úgy gondolták, hogy a mágnesszalagot előbb-utóbb mindenhonnan kiszorítja az adatok jóval tömörebben tároló optikai lemez. Most azonban egy japán cég olyan mágnesszalagot fejlesztett ki, amely ismét versenyképes, mert rajta kétszer sűrűbben helyezhetők el az adatok, mint az optikai lemezen. Ezen az új szalagon négyzetmilliméterenként kétféle bit fér el. A digitális magnetofonhoz, képmagnetofonhoz és számítógéphez egyaránt használható új mágnesszalag az a titka, hogy megvalósították rajta az eddig csak mágneslemezen alkalmazott merőleges mágnesezést.

A hagyományos mágnesszalagon vagy mágneslemezen a mágnesréteg elemi szemcséi vízszintesen helyezkednek el, s ezáltal hosszuk korlátozza a rajtuk rögzíthető adatok mennyiségét. A függőleges szemcsék jóval sűrűbbek lehetnek, éppúgy, ahogy állva több ember fér el egy szobában, mint fekvve. A szemcséket azonban nagyon nehéz függőlegesen elhelyezni. A japán cég e feladattal úgy birkózott meg, hogy a 0,2 mikrométer vastagságú mágnesréteget légrákkolt térben, elektronsugárral felgőzöltetett kobalt-krom ötvözetből alakította ki. Ennek hosszúságú kristályait a rájuk irányuló, jól fókuszált mágneses mező függőlegesen irányban rendezi el. Ezután az egyébként gyorsan bomló kristályokat oxidréteggel és az arra juttatott szerves anyaggal tartosítja.

Az új mágnesszalagok mintapéldányát már bemutatták.

Szeszkártya

Franciaországban egy olyan zsebszámológépet szabadalmaztattak, amelynek segítségével a gépkocsivezető könnyen kiszámíthatja, mikor és mennyit hit biztonsággal, mielőtt kocsiba ül. A franciául allocarte-nek nevezett kis szerkezet akkora, mint a legkisebb, legvékonyabb számológép, így a mellényzsebben vagy a börtárcában könnyen elfér. Mivel a legtöbb nyugati országban hasonlóan a francia törvények is megengednek csekély mennyiségű szeszfogyasztást vezetést előtt, ami elérheti a 0,08 véralkohol szintet, nem mindegy, hogy pontosan mekkora is az a véralkohol-mennyiség, amivel az autós még a törvény szabta határon belül marad. A számítógéppel a vezető kiszámíthatja magának, hogy testülányának függvényében a főbb szeszitalokból, a vezetés megkezdése előtt hány órával, mennyit fogyaszthat.

A kártya gyorsan népszerűvé vált Franciaországban, ahol néhány hónap alatt már 50 ezer darabot adtak el belőle. Egy olyan országban, ahol egyetlen főútkészlet sem kiegészítő el borfogasztás nélkül, az ilyen alkalmazás nyilván hiányt pótol. Ma már ott tartunk, hogy sok gépkocsi-biztosító társaság reklámként adja ügyfeleinek a kis számológépet. Most készült el a szeszkártya angol nyelvű változata, amellyel a hatalmas amerikai és angol piacot célozzák meg.

A szabadalmaztatott vállalat elnöke arra számít, hogy a nyugat-európai egységes piac létrehozásáig, 1992-ig négymillió példányt is eladhatnak belőle.

Papirusztekercsek

Egy Nápolyban dolgozó norvég kutatócsoport új eljárással bont ki és olvas el régi papirusztekercseket. A kutatók annak a 1200 tekercsnek a megmentésére vállalkoztak, amelyek az i. e. 79. évi Vezúv-katóréskor pörkölődtek meg, és temetődtek el. Ezeket annak a házának a maradványai között találták, amely valószínűleg Julius Caesar apósáé volt. A tekercsek előbb zselatin és ecet keverékébe mártották, hogy a vékony papirusz megpuhuljon. Minden egyes tekercshez — az elszenesedett fokától függően — más-más összetételű keveréket alkalmaztak. A szöveget számítógépes alakfelismerő eljárással betűzték ki. A legtöbb irományt a görög Philodémeus írta, de találtak néhány régibb, latin nyelvű szöveget is.

Petőfi-maradványok

Buris László, a Debreceni Orvostudományi Egyetem Igazságügyi Intézetének vezetője nyilatkozott arról, hogyan azonosítaná Petőfi állírólagos földi maradványait.

Tavaly dolgozták ki Pittsburgban azt a számítógépes eljárást, amely a koponyacsontok alakjából, a lágyrészek vastagságának figyelembevételével számítógépes úton rajzolja meg a lehetséges arcmegjelenítési formákat. Ez jobb, mint a Geraszimov-módszer, amely a kutató antropológus művészi elképzeléseit is figyelembe veszi. A számítógépes megjelenítésben nincs ilyen szubjektív elem. Az elkészült rajzokat össze lehet hasonlítani a dagerrotípával és a festményekkel, s meg lehet állapítani, hasonlóak-e ezekhez, vagy éppen kizárható a hasonlóság.

Vírusterjesztő a bíróság előtt

Először állítottak bíróság elé Ausztráliában számítógépvírus terjesztése miatt egy főiskolai hallgatót. A 32 éves Deon Barylak ellen az a vád, hogy 1989 májusában szándékosan olyan programot vitt a Melbourne-i Swinburne főiskola számítógépébe, amely a tárolt adatok megsemmisítését okozhatta volna. Barylak visszautasította a vádat.

Katonai számítógépek az USA-ba

Az utóbbi időben a figyelem középpontjába került a Magyar Néphadsereg, amely számos változáson megy keresztül. Ezek közül a legnagyobb vitát a katonai kiadások csökkentése váltotta ki. Dr. Jazma Károly mérnök-ezredes, a vezérkar anyagtervezési és közgazdasági csoportfőnökének a helyettese elmondta, hogy a világ számos országával — s nemcsak a Varsói Szerződés országával — vannak kereskedelmi kapcsolataink. Legutóbb például a mérnökeink által kifejlesztett katonai számítógépeket exportáltunk az Egyesült Államokba, mintegy 12 millió dollárért. A gépeket a Műszertechnika Kiszövetkezett állította elő.

Jobb oldal
Bal oldal

Jobb felső 8 : Fém Korona

Jobb felső 5 : Műanyag borítás Hid

Jobb felső 3 : Hiányzó fog

Jobb felső 1 : Distalis Caries

Bal felső 5 : Occlusodistalis Amalgam tömés

Bal felső 6 : Mesiodistalis Caries

A fogsornak a Dental által kirajzolt képe

DENTAL

A Compogran Rendszerház olyan fogászati nyilvántartó és kezelőrendszer fejlesztett ki, mely segítséget nyújt a fogorvosoknak és a fogászatban dolgozóknak a gyógyászati munka jobb megszervezésében, az adminisztrációt gyorsabbá, pontosabbá tételében.

Új beteg felvételek a személyi adatok, majd orvosi vizsgálatot is igénylő adatok felvétele történik. Ezek egy része általánosan a szájra, a fogak együttes állapotára, a másik része pedig az egyes fogakra vonatkozik. Az általános adatok felvétele után a felvételi állapot rögzítéséhez a rendszer a képernyőre rajzol egy ép fogsort. Az aktuális fogra mindig nyílt mutató, amelynek segítségével mindig a jelzett fogra vehetők fel a fogorvosok megállapításai. A felvételi állapot egyrészt a gépben tárolható, másrészt papírra is kirajzolható.

A régi beteg kezelésénél megtekinthető a beteg személyi adatai, a fogak kezelti állapota, valamint az utolsó kezelés utáni állapota is. Ekkor vezethető át az időközben bekövetkezett változások is.

A fogászati labor nevű alrendszer lehetővé teszi a labormunka felvételét, az elkészülés dátumának rögzítését, az adott napra elkészülő munkák listájának összeállítását. A betegrendelési naptár alrendszer az előjegyzést, a visszarendelést, az adott napra berendelt betegek listájának összeállítását és az előjegyzés módosítását teszi lehetővé.

A rendszer a magyaron kívül a német és az angol közeli nyelvet is használja.

topoLogic

A Geometria Kiszövetkezett a térinformatikai rendszerek alkalmazói számára topoLogic néven egy integrált szoftvert fejlesztett ki. Száztíz ezer C nyelvű forrással biztosítja, hogy a legkényesebb térinformatikai funkciók is egyetlen menüpont aktiválásával elérhetőek legyenek. Az alkalmazásfejlesztők számára elkészült a G-Lisp programnyelv, amely a szabályos Lisp nyelv térinformatikai funkciókkal bővített implementációja. A GQL lekérdező nyelv pedig a szabványos nyelv térinformatikai funkciókkal bővített változata. Az adatbázis-szerkezetek a folytonos térképezést a négyzetes technológia segítségével valósítják meg. A rendszer futtatható mikrogépen MS-DOS, OS/2, VAX/VMS operációs rendszer alatt. A négy fő funkciócsoport a Vektor, a Grid, a DTM és Geod önállóan és egy rendszerrel integrálva is üzemeltethető.

A topoLogic rendszert a Compair '89-en a Központi Statisztikai Hivatal különdíjával tüntették ki.

Vonalkódos áruház

Hároméves előkészítés után a teljes információrendszert számítógépre vitték a tatabányai Centrum Áruházban. A kidolgozó Kereskedelmi Szervező Intézet a legkorszerűbb kereskedelmi rendszert alakította ki. Jellemzője, hogy vonalkódos. Elég ha a pénztáros a leolvásó ceruzának „megmutatja” az árut; a vásárlást számítógép rögzíti, az eddiginél részletesebb információkat tartalmazó blokkot ad, s azonnal módosítja a készletet, sőt, ha az egy bizonyos szint alá csökken, az utánpótlásról is gondoskodik. Az idei év elején már várhatóan az áruház valamennyi pénztáránál lesz leolvásó (összel még csak az alsó és a felső szinten voltak ilyen pénztárgépek). A mintegy húszmillió forintba kerülő rendszer megtérülését öt évre becsülik, úgy, hogy a költségeket nem terheli a bevételre.

Szívórzó

Több millió devizaforientért vásárolt hatágyas szívórzó monitorrendszer Dániából a zalaegerszegi megyei kórház. A rendszer nem csupán mutatja a kivizsgált beteg EKG-ját, hanem elemzi is azt: nyomon követi az állapotromlást, és bizonyos értékek elérésekor riaszt. Ugyanakkor az állapotjavulást is regisztrálja, ezzel igazolva a terápia helyességét. Az ágy melletti monitorok képe az intenzív osztály és az orvosi szobák bármely részébe tv-láncon keresztül juttatható el. Így egyszerre több helyen is látható, ha például szívritmus-zavar jelentkezik. Az intelligens programrendszer a szívinfarktus okozta legveszélyesebb szövődményeket nyomon követi, és minden időben információt ad a gyógyszerelés változtatásának szükségességéről. Összefoglalva: az orvos a legkedvezőbb pillanatra a leghatásosabb terápiát avathozhat be, s az eredmény is azonnal mérhetővé válik.



Baracs Tibor, Budapest

Remélem, hogy észrevételeimet nem veszik rossz néven, hiszen az vezérelt az írásra, hogy kedvelem a lapot, és még sokáig szeretném a magaménak tudni. Azt is hiszem, hogy sokak nevében írok.

Attól tartok ugyanis, hogy a házi számítógépek tábora most is és mindinkább háttérbe szorul. Minket már cserben hagyott az Ötlet BIT-LET, kimaradtunk a Számítástechnika c. lapból, szinte tv-újság lett a Mikrovilág és így tovább. Maradt néhány gépfüggő kiadvány, de ezek nem adnak általános áttekintést. Mi tehát nem mondhatunk le a μ M-ról is! Érhető, hogy egy tudományos társaság lapja figyelni akar a hivatásos területre is, de hogy a házi számítógépek széles körét is számításba kell vennie, azt az alábbiakkal indokolom:

1. A sportból vett hasonlatnál élve: jó hivatásos és élsportolókak nevelni csakis tömegsportra építve lehet. Így nem képzelhető el nagyszámú számítástechnikus kinevelése sem a műkedvelők megfelelő irányítása, segítése nélkül.

2. Gyönyörű lenne, ha igen sok magyar család PC-t vásárolhatna, hiszen ez a lap szerint már a kispénzűek számára is elérhető. Az igazság azonban az, hogy a tízezres nagyságrendű befektetés is egyre nagyobb gondot okoz, nemhogy a százezres. A javasolt (1989/7. szám, 30. old.) tömeges bevásárló turizmus pedig (az elkerülhetetlen valuta- és vámgondokkal) nem lehet általánosan és nem is helyénvaló megoldás. Műkedvelő szinten most csak azok pécézhettek, akik a munkahelyükön (és általában munkaidőben) hozzáférhettek a géphez.

3. Bár pontosan nem lehet tudni, hogy milyen gépekre gondoltak, amikor azt írták, hogy eljön az az idő, amikor a „korszerűtlen” gépeket már nem foglalkoznak (1989/4. szám, 48. old.), nagyon remélem, hogy ez még sokáig nem vonatkozik a hazánkban tömegesen elterjedt kisgépekre (Commodore, Enterprise, Sinclair stb.), mert a műkedvelők csakis ezekkel rendelkezők.

4. Új programozási fogással, eljárással lassan már csak a Magazinban találkozhatunk. A szakkönyvek ritkulnak, és a színvonaluk is igen vegyes. A lap újtan azonban a géptársainkkal is tudást cserélhetünk. Magamról tudom, hogy a kezdőknek milyen nagy segítséget jelent a lap. Nem elég egy-néhány évjáratot átsegíteni a kezdés gondjain, az újak is kisgépeken kezdenek.

Ebből a szempontból szerintem sajnálatos lenne a hosszabb programlisták kilitásba helyezett elhagyása (1989/7. szám, 48. old.). Persze csak a jól megválasztott — felépítés, technika és tartalom szempontjából is mintaszerű — és újat is felmutató listákra gondolok. Főként pedig az ezekben tartozó, kellő részletességű leírásokat hiányolnám, amelyeknek

a lista a szükségszerű melléklete. Bár nagyon hasznos a rövid ötletek, fogások ismertetése is, csak ezekből senki sem fejlesztheti a programfelépítésre vonatkozó ismereteit, ehhez igenis hosszú programlista kell.

Különbös is téves az a felfogás, hogy a lista csak arra jó, hogy bepótyolgújk. Pótyógöttek nélkül is olvasható a lista, és hasznosítható, ami számunkra új. A kezdőknek gyakorolniuk is kell a listák olvasását. A lap nem hallgathat azokra, akik csak térképeket és örökélet-megoldásokat kérnek, a programról pedig úgy gondolják, hogy csak betöltésre és játékra jó. Az ő igényük is megváltozik, amint megpróbálkoznak a programzással.

Kérem tehát, hogy továbbra is tartsák céljuknak a kis házi számítógéppel rendelkezők ismereteinek fejlesztését, kedvtelésük segítését; esetenként pedig közöljenek hosszabb programokat is, főként olyanokat, amelyeknek gyakorlati haszna is lehet.

Egy más jellegű észrevétel. A lap olvasásánál, de főként a régi számokban való gyakori keresgélésnél volna jó, ha az egyes cikkeknel kis, egységes rendszerű jelzés tájékoztatná az olvasót arról, hogy a cikk általános jellegű, vagy valamilyen gép(ek)re vonatkozik. Sokszor az egész cikket át kell böngészni, vagy tanulmányozni kell a lista kifejezéseit, és csak akkor derül ki, hogy milyen gépről vagy milyen nyelvváltozatról van szó (például: 1989/7. szám: „Nem ördögösség!”; 1989/8. szám: „Ki ad magyarázatot?”; 1989/9. szám: „Nemes Tihamér verseny megoldása”). Szerintem nem elég rajzos szalaggal jelezni a közös jellegre (lásd Pécézzünk), hanem szöveg is kellene.

A μ M-t számonként veszem meg, mert az előfizetéssel már megjártam. Miután ezt a levelet megírtam, akkor jöttem rá, hogy miért kereselem hiába a 10. számot a hírlapúrsoknál. Azért, mert az a szokásom, hogy az új címlap megjelenését lesem. Most már tíz napja megütöközzel tapasztaltam, hogy nem fogott el még a 9. szám sem. Csak nagy nehezen jöttem rá, hogy a 10. számot is a 9. szám címlapjával álcázták. Nem értem, hogy miért gondolják azt (a SZÜV-el együtt), hogy az olvasó bosszantása jó reklám?!

Megállapításaival zömében egyetértünk. Sajnos — vagy hála istennek — a Magazin szerkesztésénél számos szempontot kell figyelembe vennünk, így előfordulhat, hogy ez vagy az a géptípus, illetve téma háttérbe szorul valami más érdekében. Szándékunk szerint a Magazinunk továbbra is vállalnia kell azt a szerepet, amelyet Ön a tömegsport támogatásához hasonlított.

Sokat nyoma a latba, amikor egy-egy géptípus mellé leesszük a voksunkat, hogy az adott gép milyen mértékben van jelen a hazai piacon, illetve mekkora az a mennyiség, amelyet effektíve használnak.

Ami a hosszú listák tervezett mellőzését illeti, sok múlik azon, vajon megéri-e közölni az adott listát, valójában hány felhasználó érdeklődésére tarthat számot.

A géptípusok jelzésének egységesítése — hogy ne kelljen végigbogarászni a cikket vagy a listát — helyenként valóban megoldhatatlan, a lap tervezett formai megújításánál természetesen gondolni fogunk rá.

A SZÜV-trükk az ismételt borítóval az Ön leveleinek tanúsága szerint elérte célját, még ha egy kis bosszantás révén is. A figyelemfelkeltés mint a reklám egyik fő eszköze ezúttal a kis-sé negatív érzelmi hatással érvényesült. Kérem, azért ne haragudjék...

Bobok Eleméne, Rakamaz

Mély megdöbbenéssel láttam az 1989/11. szám címlapján az el egyáltalán nem ide illő kompozíciót. Kisegítő könyvtárosként dolgozom, és örömmel figyelem a Magazin keressétjét, azt, hogy növekvő Magazinok milyen érdeklődéssel csapnak le a számítógépekhez segítségét nyújtó lapra.

Tisztelt Szerkesztő Úr! Szükség van ezekre a pornográfiai szaporító képekre? Hiszen gyermekeink mindenütt csak ezt a testrészt láthatják!

Gy — névtelen — Voit Olvasónk

Felháborítónak tartom, hogy már a Mikro-magazin címlapján is meztelen női fénykép „pompázik”? Új látszik, a férfiak csak nem tudják meghazudtolni alaptermészetüket, hogy — ahol csak lehet — megalázzák a női nemet!

Vagy talán másképpen nem tudják eladni a lapot? Eddig érdekelt, hogy mit írnak a gépekről, programokról, de ezután nem fog érdekelni. Nem vagyok maradi, de már idegesít ez a nagy „hüsvásár”. Önöket azért igényesebbnek gondoltam!

Hát ezt megkaptuk?! Bár névtelen levélírókat nem illetné válasz, most a névtelenség tényétől mégis eltekintünk. Valószínűnek tartjuk, hogy Tisztelt Olvasóinknak nincs betekintésük egy-egy nyomdászámiba. Ahhoz, hogy lapunk egyáltalán létezni tudjon, időnként — természetesen belül maradva a jó ízlés szabta határokon — alá kell vetnünk magunkat egyes megrendelőink kívánságának. Újy látszik, a nyomda „ördöge” is osztja két olvasónk véleményét (több levelet ebben a témában nem kaptunk), hiszen a címlapból végül is nem derült ki, hogy melyik cég hirdetéséről van szó.

Higgyék el, nem áll szándékunkban a női nem „megalázása”, sem pedig a pornográfia támogatása. Ez a címlap ugyanis megrendelésre készült, s ahogy mondják: nem jött össze.

AZ ÉV SZÁMÍTÓGÉPE '89



IBM Personal System/2, Modell 30



Toshiba 5200



Zenith Turbosport 386



Apple Macintosh II



Dell 325



Commodore Amiga 500

Ismertetésünket februári számunkban olvashatják.