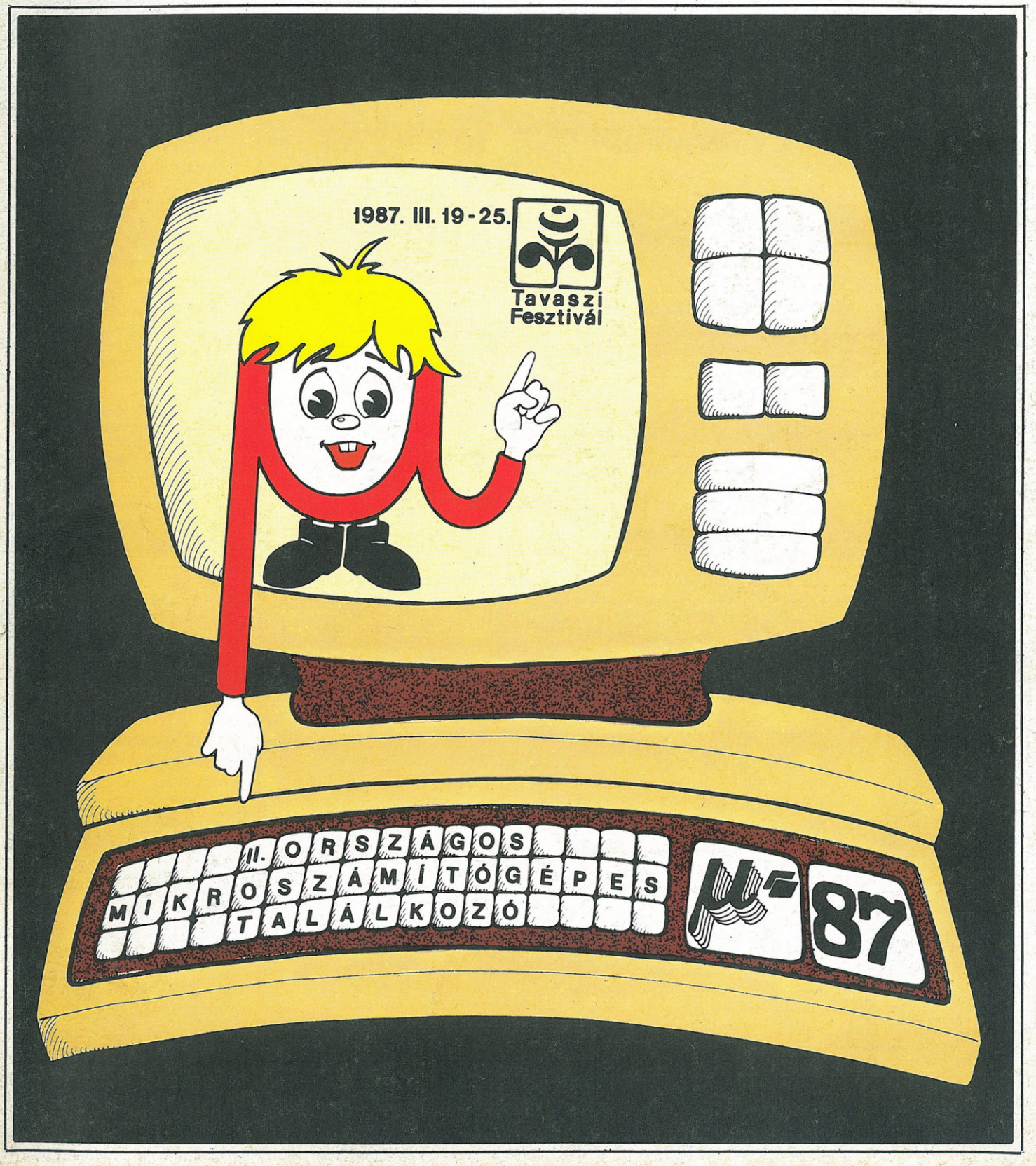
Ára: 30 Ft

mikro számítógép magazin



1987/3

5. ÉVFOLYAM
1987/3. SZÁM

μ mikro számítógép magazin

A NEUMANN JÁNOS SZÁMÍTÓGÉPTUDOMÁNYI TÁRSASÁG ÉS A KISZ KÖZPONTI BIZOTTSÁG LAPJA

A kiadvány a Tudományszervezési és Informatikai Intézettel együttműködve készül

A szerkesztőbizottság vezetője:
Kovács Győző

E számunkat szerkesztették:

Bakos Tamás
(programozástechnika)

Broczkó Péter
(hírek)

Kovács Győző
(levelezés)

Lindner László
(sakkprogramozás)

Petróczy Judit
(könyvek)

Simonyi Endre
(klub)

Varga András
(iskola-számítógép)

Felelős szerkesztő:
Könyves Tóth Pál

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

Levélcím
1371 Budapest
Pf. 433.

Kiadja az Ifjúsági Lap- és Könyvkiadó Vállalat

Felelős kiadó:
dr. Petrus György
igazgató

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

Terjeszti a Magyar Posta
Előfizethető a hírlapkézbesítő hivataloknál
és a Posta Hírlapelőfizetési és Lapellátási Irodáján
(1900 Budapest V., József nádor tér 1.)
vagy átutalással a 215—96 162 pénzforgalmi jelzőszámra.

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

Szikra Lapnyomda
Budapest (87-0010)
Felelős vezető:
Csöndes Zoltán vezérigazgató

INDEX: 25 629
ISSN 0236-6088

Címképünk:
Sárközi Antoinette munkája

Tartalom

μ'87	2
μINFORM	16
Vigyázat! Tolvaj!	28
Adok — veszek — cserélek	29
Olvastunk . . .	30
RAINBOW a μM-ban	34
Ki ad magyarázatot?	41
Egy számítástechnikai-gazdasági konszern	43
ISKOLA — SZÁMÍTÓGÉP	
„Folyékony logika"	3
Gépi kódban	5
Megszakításvezérelt digitális óra	6
Pontszerű töltések erővonalképe	9
DIÁKROVAT	
Hibaüzenetek olvasása	11
Térhatású ábrák	12
PROGRAMOZÁSTECHNIKA	
BASIC és gépi kód	13
OVERLAY—CHAIN—APPEND	13
Tömbök kezelése FORTH-ban	16
μPROGRAMOK	
Házi módosítások a GENS3, MONS3 programokon	19
Kalkulációs program adóelszámoláshoz	24
Hasznos rutinok C64-re	25
Lapdobás	25
Kétütemű motor működése	26
Programbetöltés soros illesztőn keresztül	26
Háromdimenziós ábrázoló geometria	27
μKLUB	
ZX Spectrum — helyi hálózatban	36
ZX Spectrum, mint digitális jeleket tároló oszcilloszkóp	37
Külső ROM C64-hez	38
A VC20 hardver adta lehetőségei	39
Menüprogram	40
SAKKPROGRAMOZÁS	
Bitek és figurák	42
FÓRUM	43
AZ OLVASÓ ÍRJA	44
JÁTÉKPROGRAMOK	45
KÖNYVEK	46
HÍREK — ÉRDEKESSÉGEK	48

„Az elektronikus digitális számológépek (sic!) kétségtelenül korunk legnagyobb jelentőségű műszaki alkotásai közé tartoznak. A feladatok, amelyek megoldására már ma is alkalmazzák őket, még inkább pedig azok, amelyek megoldására elvileg alkalmazhatók, meglepők, sokszor — mint például a fordítás és a sakkjáték esetében — egyenesen meghökkentők. (. . .) A határokat még korántsem látjuk világosan; annyi bizonyos, hogy az elektronikus számológépek új fejezetet jelentenek a technika történetében."

(Tarján Rezső: Gondolkodó gépek 1958.)

μ'87, a második Országos Mikroszámítógépes Találkozó

Tarján Rezsőtől, sokunk mentorának könyvéből — amelyik valószínűleg az első volt a magyar informatikai szakirodalomban — vettem a bevezető idézetet. Ő sajnos már nem érhette meg az informatikai gépeknek és módszereknek azt a társadalmi méretű elterjedését, amelyet az évenként rendezett Országos Mikroszámítógépes Találkozókon bemutatunk.

Amikor a szerkesztőségi cikkhez a rendelkezésemre álló Tarján-írások között a megfelelő idézetet kerestem, csodálkozva fedeztem fel, hogy írásaiban — noha műszaki „jóslásokra" szívesen vállalkozott — az informatikának mai mértékű terjedését nem említette. Írt a sakkozógépen és a fordítógépen kívül az automatizált gyárról, a számítógépes árurendelésről, általában a gazdasági életben való alkalmazásokról, de arról, hogy viszonylag nagy teljesítményű gépek jelennek meg majd az iskolákban, sőt a lakásokban is, ennek „jóslására" sem ő, sem nagyon sokan mások sem vállalkoztak.

Az idén másodízben rendezzük meg a népszerű nevén μ Találkozót, amelyen — részben a szakintézmények, részben pedig az amatőrök segítségével — azt mutatjuk be, hogyan hódítja meg az informatika mindazokat az alkalmazási területeket, amelyekre nemhogy 1958-ban, de még néhány évvel ezelőtt sem gondoltunk.

A μ'87-et éppen azzal szeretnénk megkülönböztetni például a BNV-től, hogy a látogatók az üzleti céllal rendezett kiállítás mellett az informatikának a kulturális életben, a családi körben, a tanításban és a tanulásban, és még ki tudja, milyen más területeken való alkalmazását is megláthatják.

Már a μ'86-on is meglepetést okozott, hogy milyen sokan érdeklődtek a számítógéppel komponált és a számítógép segítségével előadott zenei alkotások iránt. Egy egész ország csodálta meg az NJSzT számítógépes amatőreinek az alkotásait, amelyek még a professzionális számítógép-konstruktőrök elismerését is kivívták.

Az idén egy lépéssel ismét továbblépünk, remélve, hogy a közönség szívesen fogadja nem csak bevált programjainkat, de az újakat is.

A μ'87-en az idén harmadszor rendezzük meg az „A számítástechnika mindenkié, a számítástechnika mindenkiért" — népszerűen rövidítve SzMSzM — kiállítást. Az idén is arra kértük a kiállítókat, hogy azokat az olcsó eszközeiket, rendszereiket, programcsomagjaikat hozzák, amelyek széles körű érdeklődésre tarthatnak számot. Az idén először külföldi kiállítókat is meghívtunk, az ismert hazai gyártók és számítástechnikai intézmények mellett. Már a μ'86-on is igen nagy számban vettek részt kisvállalkozások, egy magánkiállítónk is volt! Idén a kisvállalkozásoknak és a magánvállalkozóknak különféle kedvezményeket adtunk, hogy a hazai „számítástechnikai paletta minden színe" megjelenjék a μ'87-en.

Az idén megrendezzük a megyei NJSzT szervezetek kiállítását is. A területi szervezetek maguk határozzák el, hogy mit fognak bemutatni. Az előzetes hírekből úgy tűnik, hogy itt is a megyék szakmai eredményein lesz a hangsúly, ami annál is inkább fontos, mert bizony ma még nincs meg az a szervezett lehetőség vagy piac, ahol egy-egy jó és vidéken készült program vagy műszaki alkotás megfelelő fórumot kaphatna.

A társadalom informatizálásának az egyik legfontosabb programja ma is az iskola-számítástechnika, de valószínűleg évekig az lesz még. Ismét bemutatjuk, hogyan alkalmazzák az iskolákban a számítógépeket, pályázatot írtunk ki diákoknak, tanároknak és iskoláknak is, hogy mindnyájan megláthassuk, hogyan lehet az iskolákban, tanórákon, szakkörökben, de foglalkozásokon kívül is a számítógépeket alkalmazni.

Ha már pályázatról van szó, az idén is díjazni fogjuk a számítógépes amatőrök legszellemesebb és műszakilag legjobb alkotásait (számítógépek, perifériák, áramköri megoldások stb.), hasonlóképpen az amatőr szoftverfejlesztők eredeti munkáit (operációsrendszer-elemek, segédprogramok, alkalmazói programok stb.) is. Az idén is lehet pályázni az ún. háztartási programokkal. Az elnevezés pontatlan, ti. a nem számítógépes szakemberek pályázhatnak olyan alkalmazói programokkal is, amelyeket például saját hivatali munkájuk megkönnyítésére találtak ki, de elfogadjuk számítástechnikai szakemberek pályázatát is, ha olyan programot hoznak, amelyet otthoni számítógépükön a ház körüli munkák megkönnyítésére használnak. Ugyanazzal a programmal akár két pályázaton is indulni lehet, például egy nyelvtanár pályázhat mondjuk egy angol segédigéket gyakorló programmal az iskolai, de a háztartási programok pályázatán is.

Hagyományosnak nevezhetjük a Szekszárdi Garay János Gimnázium, az NJSzT és a μM oktatási és játékprogram-pályázat győzteseinek a bemutatóját, ezeket a programokat a kiállítás látogatói természetesen ki is próbálhatják.

Nem hívhatjuk fel elégszer a figyelmet az egészségkárosultak helyzetére, hiszen az informatika az egyik nagyon fontos eszköze lehet az egészségkárosultaknak, amellyel magasan kvalifikált munkát végezhetnek. Szeretnénk, ha egyre több olyan eszköz születne, amellyel a társadalomnak ezen nagyon hátrányos helyzetben élő tagjai számára a munkába állást meg lehet könnyíteni.

A rendezőbizottság úgy véli, hogy nagyon sok látogatót fog vonzani az informatika humán és művészeti alkalmazásának a bemutatása. Az idén a párizsi IRCAM munkatársait hívtuk meg zenei bemutatóra, akik részben elmondják, de meg is mutatják azokat az eredményeket, amelyeket mind a zene komponálásában, mind pedig zenei művek számítógépen való előadásában elértek.

Az MTA SzTAKI és az Új Impulzus által rendezett számítógépes grafikai kiállítást, a Digitartot a μ'87-en folytatjuk. Reméljük, hogy itt olyan új alkotásokat is be tudunk mutatni, amelyeket már az első, őszi kiállítás, illetve az SzKI — SCI—L művészeti pályázata inspirált.

Kétségkívül érdekes színfoltja lesz a μ'87-nek a tv- és hangstúdió, amely a kiállítás tartama alatt folyamatos műsort „sugároz" mind a μ'87, mind pedig az Utazás '87 látogatói részére. A Magyar Televízió Képújság-stúdiója is részt vesz a μ'87-en, a nézők az aktuális eseményekről a Képújságban kaphatnak folyamatosan tájékoztatást.

Már a μ'86-on is nagy sikere volt a „boltoknak", ahol a látogatók nem csak nézhették a kiállított tárgyakat, de mindent meg is vásárolhattak. Az idén meghívást küldtünk mindazoknak a cégeknek, amelyek számítógépeket, alkatrészeket, perifériákat, de számítógépes médiumokat (szalag, floppy, leporelló, tisztítószerek stb.) árusítanak, valószínűleg az idén is jó üzleti forgalomra számíthatnak. A μ'86-on a vártnál nagyobb volt az érdeklődés a szakkönyvek iránt, ezért számítunk a könyvesboltok és a könyvkiadással is foglalkozó cégek részvételére.

Az idén a Műszaki Könyvkiadóval közösen a μ'87-re időzítettük az első informatikatörténeti mű

H. H. Goldstine: „A számítógép Pascaltól Neumann-ig"

kiadását, amelyet először a μ'87-en lehet megvásárolni. A könyv megjelenése alkalmából a μ'87 vendége lesz a szerző, H. H. Goldstine is, akivel találkozót szervezünk.

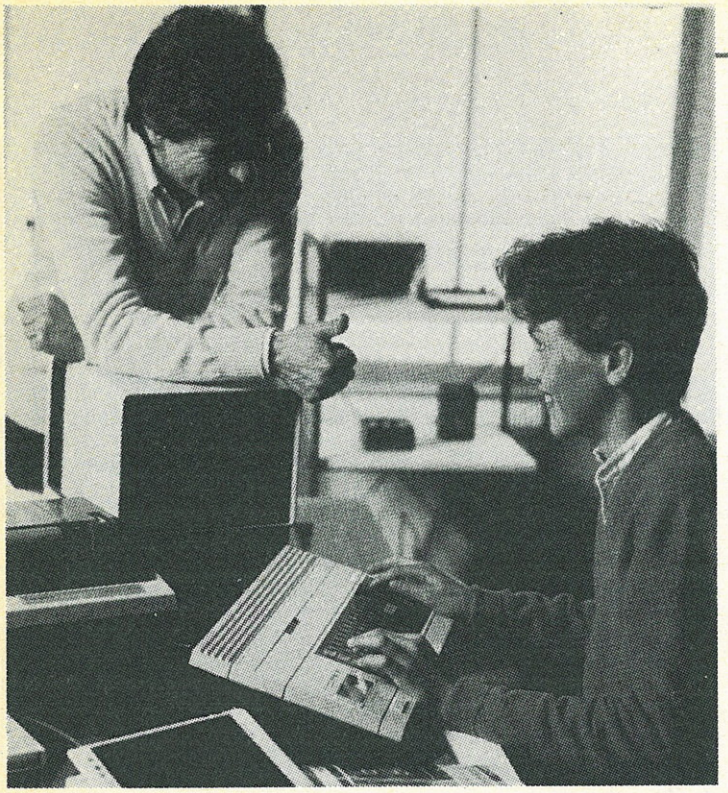
Az idén is lesz börze, ahol bárki eladhatja vagy elcserélheti számítástechnikai eszközeit, programjait. Azt reméljük, hogy meg tudjuk szervezni a számítástechnikai intézményeknél elfekvő inkurrens áruk folyamatos eladását is.

Az NJSzT Ifjúsági Bizottsága és Oktatási Szakosztálya a μ'87-tel egyidőben és a találkozó színhelyén rendezi meg az „Ifjú oktatók Kozma László konferenciáját", amely alkalomból nyílt ülésen tárgyalják meg a számítógépekkel való oktatásnak, illetve az informatika tanításának a még ma is sok vitát kiváltó kérdéseit.

A μ'87 valamennyi programját nem tudjuk felsorolni, a szerkesztőségi cikk írásakor még valamennyit nem is ismerjük. Az idén meglehetősen sok ad-hoc programot is szervezünk, amelyről a napilapokból, illetve a Tv Képújságból értesülhetnek az érdeklődők. Új programok megszervezésére így még az „utolsó pillanatban" is van lehetőség, ezért olvasóink javaslatait is várjuk.

A μ'87 rendezői azt szeretnék, ha mind a „szereplők", mind pedig a látogatók végül azzal a jó érzéssel összegeznék a látottakat, hogy a hazai számítástechnika ismét előbbre lépett a társadalom informatizálásában.

KOVÁCS GYŐZŐ



Fogékony korban:

„Folyékony logika"

A Minneapolisban működő JMH SOFTWARE MINNESOTA nevű vállalkozás Apple, Atari és Commodore típusú személyi számítógépekre oktatószoftver-csomagot készített. Bemutatunk néhányat termékeiből.

A legkisebbek számára C64-re készült az Analógiák című program. A logika kifejlesztését is segíti, de a nyelvtanítás egyik segédrutinjaként jól hasznosul. Indítás után a feladatok alternatívái közül választhat a gyakorló.

Az egyik feladatcsoport: a mondatokba a hiányzó végződéseket vagy teljes szavakat kell beírni.

Például:

A Hold olyan éjszaka, mint a Nap .....
(The Moon is tonight as sun is to .....)

A másik feladatcsoportban szintén hiányos a képernyőn megjelenő mondat, de itt a tanuló négy lehetséges megoldás közül az egyiknek a jelét (A, B, C, D) adja meg válaszképpen.

Például:

Az olló a vágáshoz olyan, mint a toll a .....-hoz.

A/ Írás
B/ Festés
C/ Törlés
D/ Papír

Rossz választásnál a számítógép letörli a képről az illető tippet, így a diáknak eggyel kevesebb lehetősége marad. Ha a harmadik próbálkozás sem sikerül, akkor a gép a lehetséges és jó megoldásokon kívül más tippeket is megad.

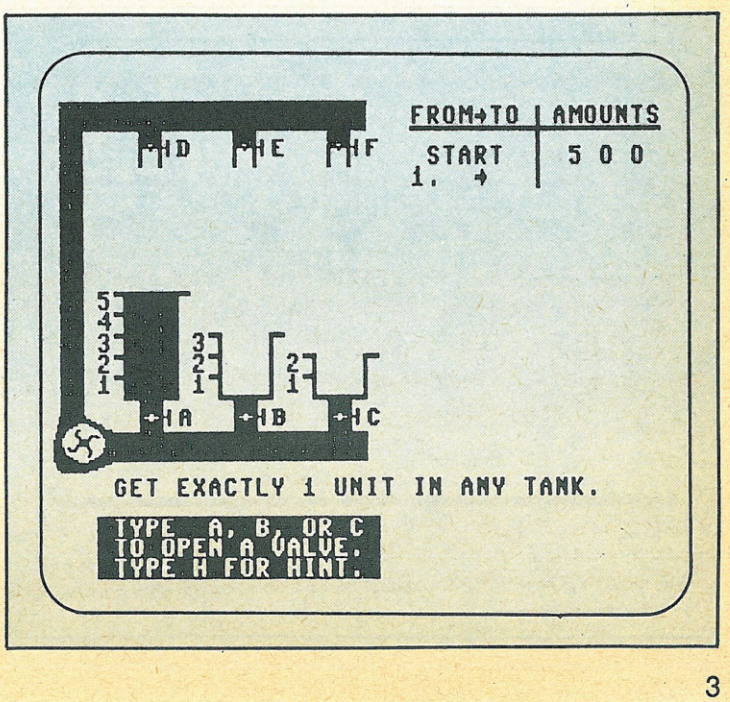
Kilenc-tíz évesek logikai és matematikai képességeit fejleszti a „Folyékony logika" című Atari és Commodore gépre tervezett játék. A gyermek rendelkezésére áll egy szivattyú, a csövek, 3 db tartály, 6 db szelep és a folyadék *(1. ábra)*. A program kijelöl egy vízmennyiséget, amit át kell tölteni a tartályok bármelyikébe. Az áttöltés a szelepek nyitásával irányítható. A képernyő szemlélteti az eddigi műveleteket, a mennyiségeket, és hogy melyik két szelep van nyitva *(2. ábra)*. További szabály, hogy mindig csak a legtöbb folyadékot tartalmazó tartályból lehet áttölteni. Amikor a tanuló elkészült a feladattal, a lépések számát tekintve összehasonlíthatja eredményét a gép minimál-megoldásával. Ha valaki nem tudná folytatni a műveleteket, akkor egy funkcióbillentyűt lenyomva „jó tanácsot" kaphat az éppen aktuális lehetőségek közül, mégpedig a legjobbat.

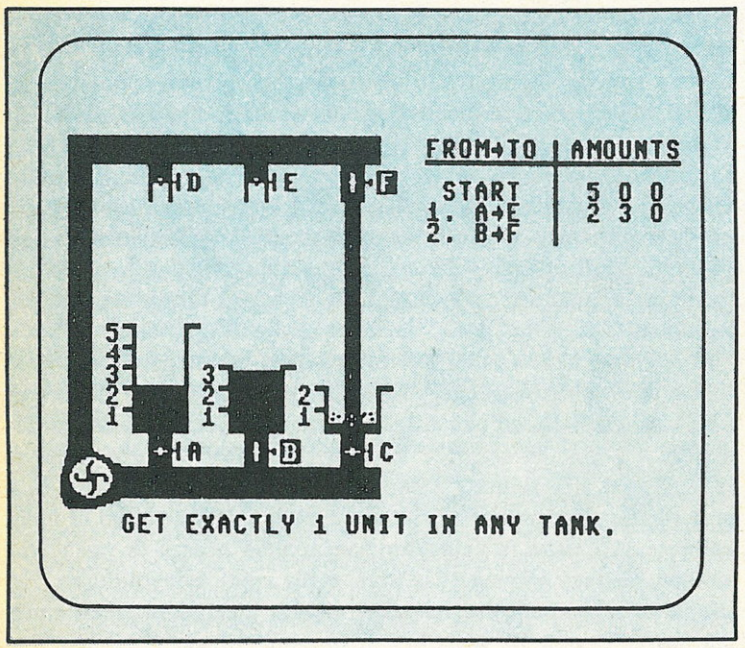
A feladat – akár a többi oktatóprogram – ismételhető, más adatokkal újraindítható, nehezebb fokozatokon folytatható, illetve egy másik program elérésének érdekében vissza lehet térni a lemezmenühöz.

A „Találd meg az utat!" című játék a megfigyelőképességet és a motorizálást fejleszti. A feladat egy kincsesládához vezető út megkeresése. A C64-re írt feladványban az utak hossza és variációja változó, egy-egy jó megoldás után pedig egyre bonyolultabb. Ha sikerül a botkormány segítségével az utat megtalálni, akkor még háromszor ugyancsak végig kell az úton menni a képen is kijelzett idő lejárta előtt.*(3. ábra)*

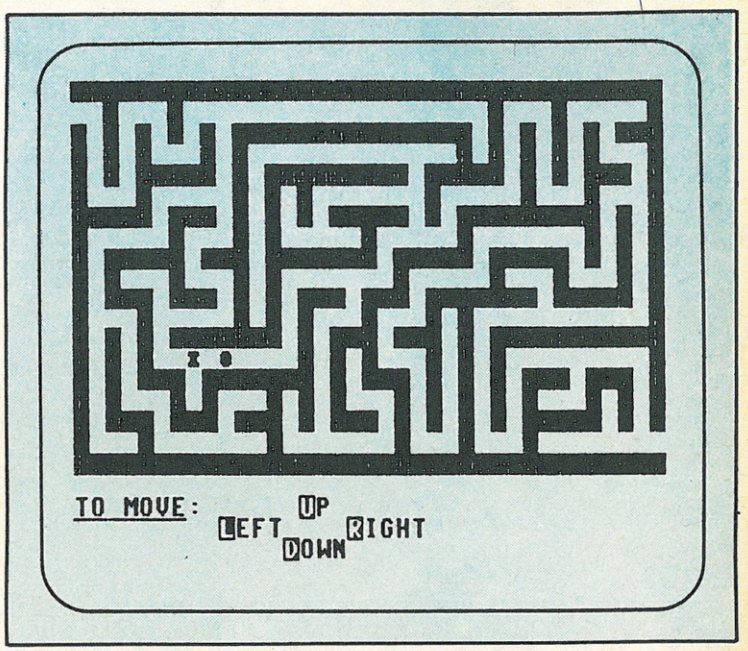
Szintén a logikai készséget méri fel a „Csodalabirintus" című C64 szoftver, de itt a jó reflexekre és az áttekintőképességre is szükség van. Egy labirintusból kell minél gyorsabban kijutni, mert a botkormánnyal irányítható kis figurát egy robot is veszélyezteti. A játékos veszít, ha a robot elfogja, vagy ha az hamarabb kijut a labirintusból. Kezdéskor van egy kis idő az áttekintésre. Az első három menetben a robot mindig a „jobbkézszabályt" alkalmazza, azaz minden keresztező folyosón – ha mód van rá – jobbra fordul. A negyedik szinttől már véletlenszerűen változhat a robot iránya. A játék képe a *4. ábrán* látható. Funkcióbillentyűk: S – újrakezdés vagy START; R – ismétlés; C – folytatás; E – END.

1. ábra





2. ábra



4. ábra

Az idegen nyelvek tanulását a szókincs felmérésével segítik. Összesen hét nyelv, közöttük a francia, a spanyol és a német közül lehet választani. A képernyőre kiírt angol szavak megfelelő idegen nyelvű párját kell megadni, és egyidejűleg két diák is dolgozhat egymással versenyezve. Egy-egy fordulóban 7 angol szót kell párosítani a választott nyelv megfelelő kifejezéseivel. Egy-egy ilyen csoportnak a szavait a számítógép egy nagyobb témakör kifejezéskészletéből emeli ki. A témakörök a következők: ételek; testrészek; napok, hónapok, dátumok; tantermi, szobai bútorok, tárgyak; melléknevek, jelzők.

Kisebbek számára az SI mértékegységrendszer metrikus átváltásait szemlélteti táblázatosan egy Apple gépre írt program, amely a nagyobbaknak már nehezebb feladatokat kínál. Az SI és más rendszerek közötti számításokat kell megoldaniuk *(5. ábra)*.

A földrajzi tájékozódást teszi próbára a mindhárom említett gépre elkészített program, amikor megjelöl egy területet (állam, város, hegység stb.) a vaktérképen, majd megad egy nyílirányt. Ekkor a tanuló az irány helyes vagy rossz (Y/N) voltát írja be. Ha körülbelül megfelelő a nyíl állása, akkor a botkormánnyal az adott területig kell elkormányozni, majd megadni a hosszúsági és a szélességi fokokat.

A halmazelmélettel kapcsolatban is készültek feladatok *(6. ábra)*.Itt Venn-diagrammal szemléltetett halmazokon gyakorolhatók a logikai műveletek (konjunkció, diszjunkció, komplemenshalmaz képzése). A program elindítása után az üres Venn-diagramok jelennek meg. Ezután a következő adatokat kéri a gép:

- Hány eleme legyen az „A” halmaznak?
- Add meg az „A” elemeit!
- Hány eleme legyen a „B” halmaznak?
- Add meg a „B” elemeit!

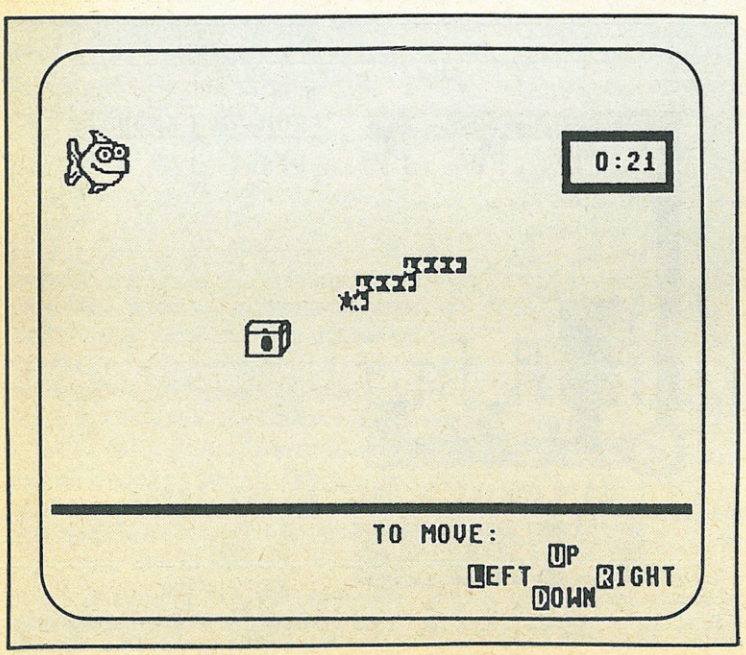
Ha a diák ezzel elkészült, akkor a logikai feladatok következnek. Helyes eredményhalmaz megadását a Venn-diagram látványos módosulása követi.

Az említetteken kívül szerepel a JMH szoftverlistáján összeadás, kivonás, szorzás, osztás (törtekkel is), a koordinátarendszerrel foglalkozó program, de könyvtárkezelő, helyesírási, a keringési szervrendszert bemutató, sőt még zenei anyag is.

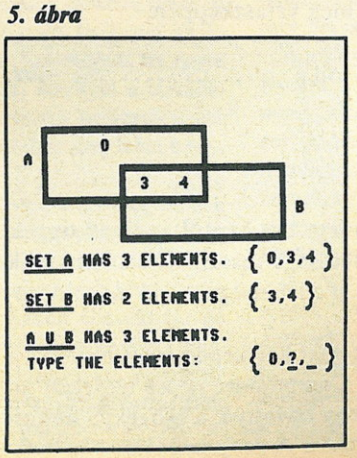
Az oktatás támogatásán kívül azért is fontos ez a kezdeményezés, hogy a számítástechnika az érdeklődők egyre fiatalabb nemzedékeinek gondolkodását csiszoló társává váljék.

HARGITAI PÁL

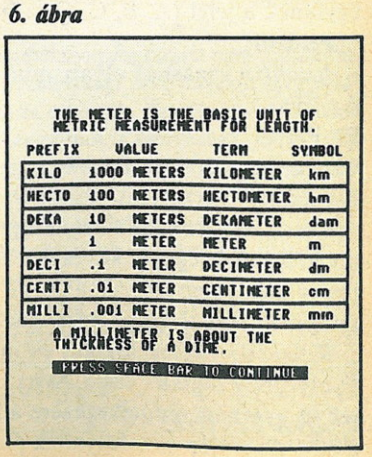
3. ábra



5. ábra



6. ábra



Gépi kódban

PRIMO

A Primo gépi kódú programozásának „felderítését" a G3 jelű programmal folytatva *(1. lista)*, egy pontot rajzolhatunk ki a képernyőre. E feladat elvégzéséhez a 00131 címen kezdődő szubrutint hívjuk meg a gépi kódú programunkból.

A gép szoftverkönyve szerint e szubrutin bemenő paramétereit az E és a D regiszterekben kell megadni; a korábban ismertetett módon az (5,5) koordinátájú pontot az LD D, 05 és az LD E, 05 utasításokkal.

A szubrutin meghívása a CALL utasítással és az azt követő címzéssel történik.

Címzés készítése táblázattal. A táblázat dőlt szedésű (kurzív) számai közül keresse meg azt a legnagyobbat, amelyik a címből levonva (—1)-nél nagyobb maradékot eredményez. (Esetünkben ez 0.) A maradékot kell a CALL utasítás után írni, majd azt a számot, amelyet a kiválasztott kurzív szám alatt látunk.

0	*256*	*512*	*768*	*1024*	*1280*	*1536*	*1792*	*2048*	*2304*	*2560*	*2816*	*3072*	*3328*	*3584*	*3840*
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
4096	*4352*	*4608*	*4864*	*5120*	*5376*	*5632*	*5888*	*6144*	*6400*	*6656*	*6912*	*7168*	*7424*	*7680*	*7936*
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
8192	*8448*	*8704*	*8960*	*9216*	*9472*	*9728*	*9984*	*10240*	*10496*	*10752*	*11008*	*11264*	*11520*	*11776*	*12032*
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
12288	*12544*	*12800*	*13056*	*13312*	*13568*	*13824*	*14080*	*14336*	*14592*	*14848*	*15104*	*15360*	*15616*	*15872*	*16128*
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
16384	*16640*	*16896*	*17152*	*17408*	*17664*	*17920*	*18176*	*18432*	*18688*	*18944*	*19200*	*19456*	*19712*	*19968*	*20224*
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
20480	*20736*	*20992*	*21248*	*21504*	*21760*	*22016*	*22272*	*22528*	*22784*	*23040*	*23296*	*23552*	*23808*	*24064*	*24320*
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
24576	*24832*	*25088*	*25344*	*25600*	*25856*	*26112*	*26368*	*26624*	*26880*	*27136*	*27392*	*27648*	*27904*	*28160*	*28416*
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
28672	*28928*	*29184*	*29440*	*29696*	*29952*	*30208*	*30464*	*30720*	*30976*	*31232*	*31488*	*31744*	*32000*	*32256*	*32512*
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
32768	*33024*	*33280*	*33536*	*33792*	*34048*	*34304*	*34560*	*34816*	*35072*	*35328*	*35584*	*35840*	*36096*	*36352*	*36608*
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
36864	*37120*	*37376*	*37632*	*37888*	*38144*	*38400*	*38656*	*38912*	*39168*	*39424*	*39680*	*39936*	*40192*	*40448*	*40704*
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
40960	*41216*	*41472*	*41728*	*41984*	*42240*	*42496*	*42752*	*43008*	*43264*	*43520*	*43776*	*44032*	*44288*	*44544*	*44800*
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
45056	*45312*	*45568*	*45824*	*46080*	*46336*	*46592*	*46848*	*47104*	*47360*	*47616*	*47872*	*48128*	*48384*	*48640*	*48896*
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
49152	*49408*	*49664*	*49920*	*50176*	*50432*	*50688*	*50944*	*51200*	*51456*	*51712*	*51968*	*52224*	*52480*	*52736*	*52992*
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
53248	*53504*	*53760*	*54016*	*54272*	*54528*	*54784*	*55040*	*55296*	*55552*	*55808*	*56064*	*56320*	*56576*	*56832*	*57088*
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
57344	*57600*	*57856*	*58112*	*58368*	*58624*	*58880*	*59136*	*59392*	*59648*	*59904*	*60160*	*60416*	*60672*	*60928*	*61184*
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
61440	*61696*	*61952*	*62208*	*62464*	*62720*	*62976*	*63232*	*63488*	*63744*	*64000*	*64256*	*64512*	*64768*	*65024*	*65280*
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

```
1 REM G3
100 C=0 : CC=0
105 CLS
110 DIM K%(20)
120 CC=VARPTR(K%(0))
128 REM : LDD,05 : LDE,05 : CALL,00131
129 REM : RET
130 POKECC,22,5,30,5,205,131,0,201
140 C=CALL(CC)
```

A PROGRAM	A KOD
LDD,05	22,5
LDE,05	30,5
CALL,00131	205,131,0
RET	201

1. lista

```
10 REM G/ATSZAMOLO

20 INPUT "MI A SZAM";S

30 PRINT"AZ ELSO SZAM  :";

      S-256*INT(S/256)

40 PRINT"A MASODIK SZAM:";

      INT(S/256)
```

2. lista

```
1 REM G4
100 C=0 : CC=0
105 CLS
110 DIM K%(20)
120 CC=VARPTR(K%(0))
127 REM : LDD,05 : LDB,55 : LD(HL),B
128 REM : LDB,(HL) : CALL,00131
129 REM : DJNZ,(-9): RET
130 POKECC,6,55,22,5,112,94,205,131,0,
      16,247,201
140 C=CALL(CC)
```

A PROGRAM	A KOD
LDB,55	6,55
LDD,05	22,5
LD(HL),B	112
LDE,(HL)	94
CALL,00131	205,131,0
DJNZ(-9)	16,247
RET	201

3. lista

Címzés készítése programmal. A *2. listán* látható program segítségével a címzés meghatározható.

Természetesen egy pont kirajzolásánál nem érződik a gépi kódú programrész sebessége. A G4 jelű program *(3. lista)* viszont már egy vonalat rajzol ki a képernyőre. A kirajzoló ciklus ismétlési számát (a vonal hosszát) a B jelű regiszterben adjuk meg: (LD B,55).

A DJNZ utasítás csökkenti a B regiszter értékét 1-gyel. Ha B regiszter tartalma 0, akkor a program tovább fut. Ha B regiszter tartalma >0, akkor a program futása a DJNZ utasítás utáni értéknek megfelelő ugrással folytatódik. Esetünkben ez (—9), ami azt jelenti, hogy az LD D,05 utasításra ugrunk vissza. A D és az E regiszterek ismételt feltöltése azért szükséges, mert a 00131-es címen kezdődő szubrutin e regiszterekben lévő értékeket megváltoztatja.

A negatív számokat a következő módon „kódolva" lehet megadni:
(KÓD) = 256 + (negatív szám)
Például esetünkben: 256 + (—9) = 247

SOMOGYI GYÖRGY

Megszakításvezé

Mikor és mire jó?

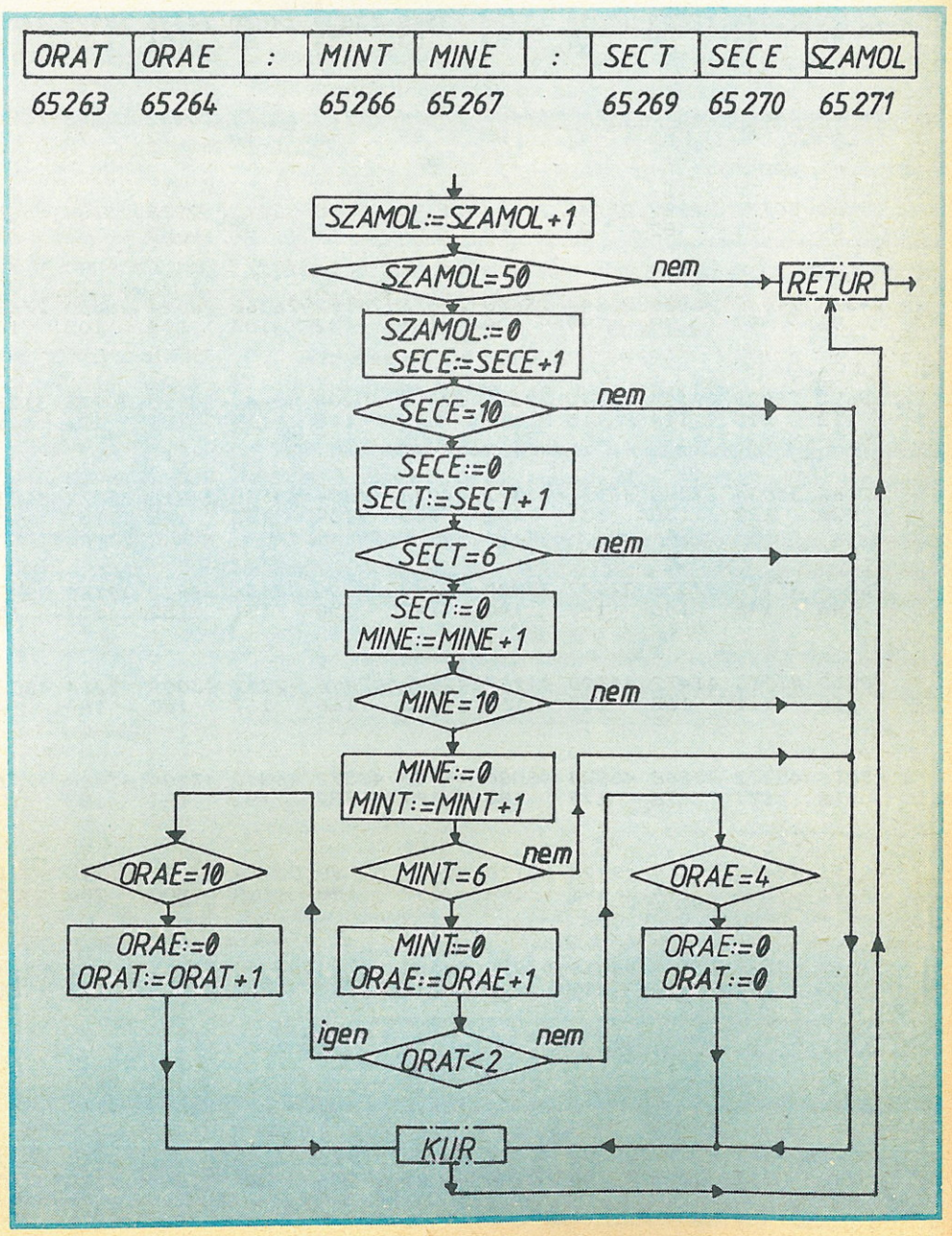
A BASIC program *(1. lista)* elindítása után a képernyő jobb felső sarkában megjelenik a 6 digites óra alaphelyzetből — 00:00:00 — indítva, és a menü. A számbillentyűk segítségével a következő lehetőségek közül választhatunk: az órát megállíthatjuk vagy újra elindíthatjuk; a kijelzést letilthatjuk vagy engedélyezhetjük, miközben az óra jár; beállíthatjuk (a kijelzés 24 órás); tárolhatunk egy adott időpontot, és ahhoz képest mérhetünk időtartamot. A program csak egy általam kigondolt demonstráció.

Az óra a NEW parancs hatására megáll, tehát ilyenkor újra kell indítani. LOAD, SAVE, BEEP esetén a parancs végrehajtása után magától megy tovább. Az óra egyébként független az éppen futó programtól, csak ez a négy parancs van rá hatással. Alkalmas egy programon belül az idő kijelzésére, időtartam mérésére stb. A későbbiekben leírom a változtatási, beállítási lehetőségeket is.

A gépi kódú rutin

A ZX—Spectrum alaphelyzetben, bekapcsolás után 1-es megszakítási módban (IM 1-ben) dolgozik. Ez azt jelenti, hogy 20 ms-onként megszakítja az éppen folyó feladatának végrehajtását és egy „restart" rutinra ugrik (rst 38H), ahol a FRAMES

Az óravezérlő rész működésének folyamatábrája



érelt digitális óra

ZX—SPECTRUM

1. lista

```
10 REM az ORA lehetosegeit bemutato prog
ram
15 BORDER 6: PAPER 6: BRIGHT 1: INK 0
20 CLS: PRINT AT 3,10; INVERSE 1;"LEHET
OSEGEK": INVERSE 0
30 PRINT AT 6,0;"1 az ORA (el)inditasa:";
40 PRINT AT 8,0;"2 az ORA megallitasa:";
50 PRINT AT 10,0;"3 a kiiras letiltasa:";
60 PRINT AT 12,0;"4 a kiiras engedelyeze
se:";
70 PRINT AT 14,0;"5 az ORA beallitasa--)";
80 PRINT AT 16,0;"6 idopont kiolvasasa-)";
90 PRINT AT 18,0;"7 idotartam merese---)";
100 LET K$=INKEY$: IF K$="" THEN GOTO 100
110 IF CODE K$<49 OR CODE K$>55 THEN GOTO
100
120 GO SUB ((CODE K$)-48)*100+8000
130 GOTO 100
140 STOP
8100 REM az ORA elinditasa
8110 PRINT AT 8,26;"   ";: PRINT AT 6,26;
"<--";
8120 RANDOMIZE USR 65122
8130 RETURN
8200 REM az ORA megallitasa
8210 PRINT AT 6,26;"   ";: PRINT AT 8,26;
"<--";
8220 RANDOMIZE USR 65115
8230 RETURN
8300 REM a kiiras letiltasa
8310 PRINT AT 12,26;"   ";: PRINT AT 10,
26;"<--";
8320 POKE 65206,24: POKE 65207,45
8330 RETURN
8400 REM a kiiras engedelyezese
8410 PRINT AT 10,26;"   ";: PRINT AT 12,
26;"<--";
8420 POKE 65206,0: POKE 65207,0
8430 RETURN
8500 REM az ORA beallitasa
8510 PRINT #0;: INPUT"Az idopont 06:15:05
 alakban=              "; LINE T$
8520 LET ORAT=65263: FOR I=0 TO 7: PRINT
 AT 14,24+I; CHR$(CODE T$(I+1)): POKE ORAT
+I,CODE (T$(I+1)): NEXT I
8530 RETURN
8600 REM az idopont kiolvasasa
8610 DEF FN P(T)=(PEEK (T)-48)*100+ PEEK
(T+1)-48
8620 DEF FN Q(U)=3600*FN P(U)+60*FN P(U+3)
+FN P(U+6)+0.02*PEEK 65271
8630 LET KEZDET=FN Q(65263)
8640 FOR I=0 TO 7: PRINT AT 16,24+I;CHR$(
PEEK(65263+I)): NEXT I
8650 PRINT AT 18,21;"          ";
8660 RETURN
8700 REM idotartam merese
8710 LET VEG=FN Q(65263): LET DELTA=VEG-
KEZDET
8720 LET H=INT (DELTA/3600): LET M=INT (
(DELTA-H*3600)/60): LET S=DELTA-H*3600-M*
60: LET S=(INT (S*100))/100
8730 PRINT AT 18,21;"          ";: PRINT
 AT 18,21;H;":";M;":";S;
8740 RETURN
8998 STOP
8999 CLEAR 65100
9000 LET A$="fff3f5c5d5e5dde5fde521f7fe3
47efe323869360021f7fe06022b343e3abe202c36
302b343e36be202336302b10eb"
9010 LET B$="2b342b3e32be23200c3e34be201

036302b363018093e3abe200436302b34"
9020 LET C$="0000060811184021efff6004fdd
090608dd7e001214dd2310f7d11323c110db"
9030 LET D$="fde1dde1e1d1c1f1fbc930303a3
0303a303000"
9033 REM ********************************
9035 LET KONTR=0: LET INV=65129: LET M$=
A$+B$+C$+D$+"*": LET A=INV
9040 LET P=16*(CODE M$-48-(39*(M$>"£")))
+CODE M$(2)-48-(39*(M$(2)>"£"))
9050 LET KONTR=KONTR+P: POKE A,P
9060 LET M$=M$(3 TO )
9070 IF M$="*" THEN GOTO 9100
9080 LET A=A+1
9090 GOTO 9040
9100 PRINT "Az ellenorzo szam= ";KONTR
9105 REM ++++++++++++++++++++++++++++++++
9110 DATA 62,63,237,86,237,71,201,62,9,
237,71,237,94,201
9115 RESTORE 9110
9120 FOR I=65115 TO 65128: READ A: POKE
I,A: NEXT I
9130 REM ++++++++++++++++++++++++++++++++
9150 REM a szamjegyek UDG kodjai
9160 REM ################################
9170 DATA 24,36,36,36,36,36,24,0
9171 DATA 4,4,4,4,4,4,4,0
9172 DATA 28,4,4,28,32,32,60,0
9173 DATA 28,4,4,28,4,4,28,0
9174 DATA 32,32,36,60,4,4,4,0
9175 DATA 60,32,32,56,4,4,56,0
9176 DATA 24,32,32,56,36,36,24,0
9177 DATA 24,4,4,4,4,4,4,0
9178 DATA 24,36,36,24,36,36,24,0
9179 DATA 24,36,36,28,4,4,24,0
9180 DATA 0,0,24,0,24,0,0,0
9185 RESTORE 9170
9190 FOR I=65272 TO 65359: READ A: POKE
I,A: NEXT I
9199 REM ********************************
9200 REM Koosz Tamas Kapuvar 1986.X.21.
9201 REM ********************************
9210 STOP
9999 SAVE "time" LINE 8999
```

```
 1                ;Az ORA elinditasat es
 2                ;megallitasat vegzo
 3                ;rutinok
 4                ;KOOSZ TAMAS Kapuvar 1986
 5                        ORG  65115
 6 FE5B 3E3F              LD   A, 63   ;IM 1 beallitasa
 7 FE5D ED56              IM   1       ; az ora kikapcsolasa
 8 FE5F ED47              LD   I, A
 9 FE61 C9                RET
10 FE62 3E09              LD   A, 9    ;IM 2 beallitasa
11 FE64 ED47              LD   I,A     ;az ora elinditasa
12 FE66 ED5E              IM   2
13 FE68 C9                RET
14               ;IM 2 alkalmaval a ZX Spectrum
15               ;a 9*256+255-os cimen levo
16               ;szamnak (ketbyte-os) megfelelo
17               ;cimre ugrik es vegrehajtja az
18               ;ott talalhato rutint. Ez most
19               ;konkretan a 65129.
20               ;A gepi kodu program elhelyezese
21               ;elott ki kell adni a CLEAR 65100
22               ;parancsot, mert ekkor a NEW utan
23               ;sem torlodik az ott elhelyezett
24               ;rutinunk. NEW utan IM 1-es modba
25               ;kerul a gep, ekkor az ora megall.
26               ;RANDOMIZE USR 65122 kiadasa utan
27               ;azonban ujra elindul.
28                       END
```

2. lista

rendszerváltozókat állítja (lásd az eredeti gépkönyv 18. fejezetét), majd a billentyűzet figyelése, kódolása következik. Mindezek után visszatér eredeti feladatához.

Mi áttérhetünk egy másik, maszkolható — vagyis szoftverrel beállítható — megszakítási (interrupt) módra, a 2-esre. Ez a programozó számára nagyon hatékony lehet, mivel lehetőséget nyújt saját rutinjainak „állandó", minden megszakításkor egyszeri futtatására.

Az IM 2-ben a megszakítójel után a PC tartalma elmentődik, majd a gép az (A)*256+255 címen kezdődő kétbájtos számon mint címen tárolt gépi kódú program végrehajtásába kezd. Ennek lefutása után visszatér félbeszakított, eredeti munkájához.

A *2. listán* két gépi kódú rutin található. Az első az IM 1, a második az IM 2 beállítását végzi. A ZX-Spectrumnál IM 1-ben az I regiszter tartalma 63. A jelen programban IM 2-ben az I regiszterbe 09H-t töltünk; hatására a gép megnézi, hogy a 9*256+255=09FFH címen milyen kétbájtos szám található. Ezt a PEEK függvénnyel magunk is megkereshetjük:

```
LET A=PEEK(2559)+256*PEEK(2560):
PRINT A
```

Az A értéke 65129 lesz; tehát megszakításkor erre a címre ugrik a gép, és az itt található rutint hajtja végre. A-nak ilyen választása azért jó, mert a 48 k-s Spectrum RAM-jának a végére esik. Ha a RAMTOP-ot 65100-ra állítjuk be, akkor kb. 400 bájt áll a gépi kódú programunk rendelkezésére.

A *3. lista* az óra vezérlését és a kiírást végző gépi kódú program. Mindjárt elsőnek az RST 38H utasítás látható; enélkül megszűnne kapcsolatunk a számítógéppel a billentyűzeten keresztül. A regiszterek tartalmának elmentése mindig fontos, mert a rutinból való visszatéréskor hiányuk zűrzavart eredményezhetne. A lista ezután két fő részre bontható: a számláló—vezérlő és a kiíró részre. A rutin végén lévő kilenc bájtból az utolsóban (SZÁMOL) tároljuk az eltelt 20 ms-ok számát; az első nyolcban a megjelenő számjegyek és ":" ASCII kódját. A bemutatott blokkdiagram és a megjegyzések segítik a vezérlő rész egyébként sem nehéz működésének megértését.

Kiírásra csak minden másodpercben kerül sor, de ez a 3. lista 19. sorában levő 69H szám 3AH-ra cserélésével megváltoztatható: ui. ilyenkor a KIIR részre ugrunk, azaz minden ötvened másodpercben a digitális óra valamennyi számjegye újra kiíródik, ami még „állandóbb" kijelzést eredményez.

Ha a kiírást le szeretnénk tiltani, akkor az 57—58-as sorok két NOP utasítását JR RETUR-ra cseréljük (18H és 2DH). A 60-as sorban a DE regiszterpárban az óratízesek karaktere első bájtjának képernyő RAM-beli címe található (4018H=16408). Ezt átírva, a kijelzés egy-egy képernyőharmadon belül tetszőlegesen áthelyezhető.

A kiírt számjegyek és a kettőspontkaraktereket leíró nyolc-nyolc bájt tárolóhelyének kezdőcímét a 66-os sorban az IX regiszterpárba töltjük. Itt saját tervezésű karaktereket használok; a tárolás kezdőcíme FEF8H=65272 dec. Ha megelégszünk a

3. lista

```
 1                ;A ZX SPECTRUM IM 2-es
 2                ;megszakitasmodjat
 3                ;felhasznalo digitalis
 4                ;kvarcpontossagu ORA
 5                ;KOOSZ TAMAS Kapuvar 1986
 6                        ORG  65129
 7 FE69 FF                RST  38H
 8 FE6A F3                DI
 9 FE6B F5                PUSH AF
10 FE6C C5                PUSH BC
11 FE6D D5                PUSH DE
12 FE6E E5                PUSH HL
13 FE6F DDE5              PUSH IX
14 FE71 FDE5              PUSH IY
15 FE73 21F7FE            LD   HL, SZAMOL ;.02 sec-kent
16 FE76 34                INC  (HL)
17 FE77 7E                LD   A, (HL)
18 FE78 FE32              CP   50
19 FE7A 3869              JR   C, RETUR ;ha (A)<50
20 FE7C 3600              LD   (HL),0 ;(SZAMOL):=0
21 FE7E 21F7FE            LD   HL, SECE+1
22 FE81 0602              LD   B, 2    ;ismetlesek szama
23 FE83 2B        CIKLUS: DEC  HL      ;HL a SECE-re mutat
24 FE84 34                INC  (HL)
25 FE85 3E3A              LD   A, 58   ;CHR$(58)="9"+1
26 FE87 BE                CP   (HL)
27 FE88 202C              JR   NZ, KIIR
28 FE8A 3630              LD   (HL), 48 ;SECE:=0 (vagy MINE)
29 FE8C 2B                DEC  HL      ;HL a SECT(vagy MINT)-re
30 FE8D 34                INC  (HL)
31 FE8E 3E36              LD   A, 54   ;CHR$(54)="6"
32 FE90 BE                CP   (HL)
33 FE91 2023              JR   NZ, KIIR
34 FE93 3630              LD   (HL), 48 ;SECT:=0 (vagy MINT)
35 FE95 2B                DEC  HL
36 FE96 10EB              DJNZ CIKLUS
37 FE98 2B                DEC  HL      ;HL az ORAE-re mutat
38 FE99 34                INC  (HL)
```

```
39 FE9A 2B                DEC  HL      ;HL az ORAT-re mutat
40 FE9B 3E32              LD   A, 50   ;CHR$(50)="2"
41 FE9D BE                CP   (HL)
42 FE9E 23                INC  HL      ;HL az ORAE-re mutat
43 FE9F 200C              JR   NZ, IDE ;ha ORAT<2
44 FEA1 3E34              LD   A, 52   ;CHR$(52)="4"
45 FEA3 BE                CP   (HL)
46 FEA4 2010              JR   NZ, KIIR
47 FEA6 3630              LD   (HL), 48 ;ORAE:=0
48 FEA8 2B                DEC  HL
49 FEA9 3630              LD   (HL), 48 ;ORAT:=0
50 FEAB 1809              JR   KIIR
51 FEAD 3E3A      IDE:    LD   A, 58   ;CHR$(58)="9"+1
52 FEAF BE                CP   (HL)
53 FEB0 2004              JR   NZ, KIIR
54 FEB2 3630              LD   (HL), 48
55 FEB4 2B                DEC  HL
56 FEB5 34                INC  (HL)
57 FEB6 00        KIIR:   NOP          ;van KIIRAS
58 FEB7 00                NOP          ;ha JR RETUR - nincs
59 FEB8 0608              LD   B, 8    ;ORAT-tol SECE-ig
60 FEBA 111840            LD   DE, 4018H ;a 0.sor 24.oslop
61 FEBD 21EFFE            LD   HL, ORAT
62 FEC0 C5        VISSZA: PUSH BC
63 FEC1 D5                PUSH DE
64 FEC2 7E                LD   A, (HL)
65 FEC3 E60F              AND  15      ;ASCII-bol szamma alakit
66 FEC5 DD21F8FE          LD   IX, MINTA
67 FEC9 CB27              SLA  A
68 FECB CB27              SLA  A
69 FECD CB27              SLA  A       ;szorzas 8-cal
70 FECF 0600              LD   B, 0
71 FED1 4F                LD   C, A

72 FED2 DD09              ADD  IX, BC ;a minta kezdete IX-ben
73 FED4 0608              LD   B, 8    ;egy karakter = 8 byte
74 FED6 DD7E00    CIKL:   LD   A, (IX+0)
75 FED9 12                LD   (DE), A
76 FEDA 14                INC  D       ;egy sorral lejjebb
77 FEDB DD23              INC  IX
78 FEDD 10F7              DJNZ CIKL
79 FEDF D1                POP  DE
80 FEE0 13                INC  DE
81 FEE1 23                INC  HL
82 FEE2 C1                POP  BC
83 FEE3 10DB              DJNZ VISSZA
84 FEE5 FDE1      RETUR:  POP  IY
85 FEE7 DDE1              POP  IX
86 FEE9 E1                POP  HL
87 FEEA D1                POP  DE
88 FEEB C1                POP  BC
89 FEEC F1                POP  AF
90 FEED FB                EI
91 FEEE C9                RET
92 FEEF 30303A30  ORAT:   DB   48,48,58,48,48,58,48
92 FEF3 303A30
93 FEF6 30        SECE:   DB   48
94 FEF7 00        SZAMOL: DB   0
95 FEF8 00        MINTA:  DB   0
96                ;FEF8-tol a 0,1,2,...,9,: karakterek
97                ;10*8 byte-janak tarolasa
98                        END
```

Spectrum saját számjegyeivel, akkor ide 80H, 3DH-t írjunk (3D80H = 15744 dec).

Ha mindent kipróbáltunk és tárolni akarjuk a gépi kódú rutinjainkat, akkor ezt a BASIC program begépelése és GO TO 8999 után — ui. ekkor a gépi kódokat elhelyeztük a RAM-ban — SAVE "ÓRA" CODE 65115, 245 paranccsal megtehetjük. A visszatöltés CLEAR 65100:LOAD "ÓRA" CODE 65115, 245-tel lehetséges.

Felhasználói programért cserébe mind a két programot szívesen másolom kazettára annak, aki címemre kazettát küld, megjelölve az üres helyeket.

KOÓSZ TAMÁS
tanár
Kapuvár, Fürdő u. 22.
9330

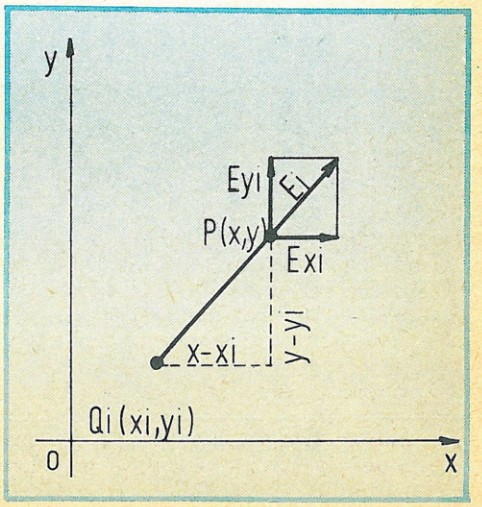
PTA—4000

Pontszerű töltések erővonalképe

A Coulomb-törvény értelmében a töltések a térben erőhatást fejtenek ki egymásra. A térnek az a része, amelyben ez az erő hat, az elektromos tér vagy mező.

Az elektromos térerősség a teret jellemző vektormennyiség. Iránya a tér valamely pontjában az oda elhelyezett pozitív pontszerű próbatöltésre ható erő irányával egyezik meg, nagysága a fellépő erő és a próbatöltés nagyságának hányadosa. Egyetlen nyugvó pontszerű töltés elektromos térerősségének nagysága tőle r távolságban: $E = kQ/r^2$. Ha a teret több töltés hozza létre, akkor az egyes pontokban a térerősség nagysága az egyes töltésektől származó térerősségek vektori összege.

Az elektromos erőteret Faraday nyomán erővonalakkal szemléltetjük. Ezek olyan görbék, amelyekhez húzott érintő a tér minden pontjában az ott uralkodó elektromos térerősség irányába esik. A térerősség nagyságát az erővonalakra merőleges egységnyi felületet metsző erővonalak száma jellemzi. Az elektrosztatikus erőtér erővonalai mindig töltésekből indulnak és töltéseken végződnek. Irányításuk a megállapodás szerint olyan, hogy az erővonalak pozitív töltésekből indulnak és negatív töltéseken végződnek.

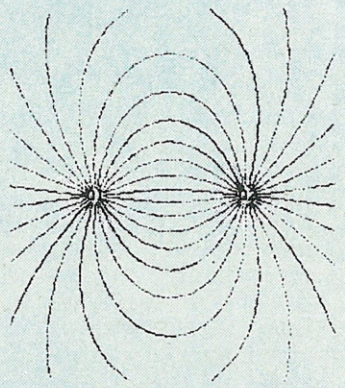


Ha a teret létrehozó töltések összege nem zérus, az erővonalakat úgy tekintjük, hogy egy megfelelő nagyságú és előjelű, végtelen távoli töltésből is indulnak, illetve ott is végződnek az erővonalak. Az erővonalak nem valóban létező fizikai objektumok, bevezetésükre csak a szemléletesség céljából volt szükség. Nem a töltések nagyságáról, hanem egymáshoz viszonyított arányukról és elhelyezkedésükről adnak képet.

A program az erővonalakat a következőképpen határozza meg. Adott pontban összegzi az egyes töltések által keltett térerősségek x és y összetevőit, majd az így meghatározott eredő térerősségvektor irányába

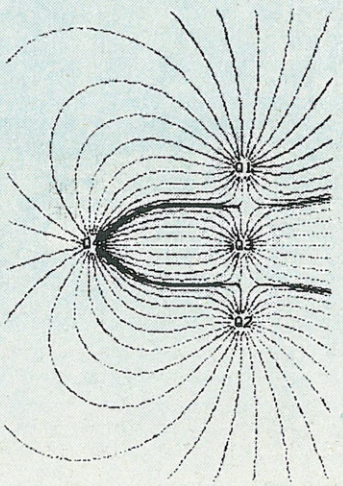
```
 5:CLEAR :ON ERROR GOTO 170
10:INPUT "TOLTESEK SZAMA=";N,"Y TAR
   TOMANY=";W:N=N-1:DIM X(N),Y(N),Q
   (N)
20:GRAPH :CSIZE 1:GLCURSOR (107,-W)
   :SORGN
30:FOR I=0TO N:WAIT 0
35:CLS :PRINT I+1;".X KOORDINATA=";
   :INPUT X(I)
40:CLS :PRINT I+1;".Y KOORDINATA=";
   :INPUT Y(I)
50:CLS :PRINT I+1;".Q TOLTES=";:
   INPUT Q(I):QQ=QQ+Q(I)
60:CLS :COLOR (1+2*(Q(I)>0)):
   GLCURSOR (X(I)-5,Y(I)-3):LPRINT
   "Q";STR$ (I+1):NEXT I
65:INPUT "1C-BOL INDULO EROVONALAK:
   ";M,"MOD (0,1) =";G
70:QQ=SGN QQ+(QQ=0):RADIAN :K=5000:
   COLOR 2:FOR S=0TO N:IF SGN (Q(S)
   *QQ)<0GOTO 180
80:FOR J=1TO M*ABS Q(S):Q=π/M*2*J/
   ABS Q(S)
90:X=6*SIN Q+X(S):Y=6*COS Q+Y(S):
   GLCURSOR (X,Y):H=0
100:O=0:P=0:I=0
105:R=(X(I)-X)*(X(I)-X)+(Y(I)-Y)*(Y(
   I)-Y):IF R<35.5GOTO 170
110:E=K*Q(I)/R:R=√R
120:O=O+E*(X-X(I))/R:P=P+E*(Y-Y(I))/
   R:I=I+1:IF I<=NGOTO 105
125:E=√(O*O+P*P)
130:DL=3:IF H=1LET DL=DL*((X*X+Y*Y)/
   14000)
140:X=X+QQ*O/E*DL:Y=Y+QQ*P/E*DL
150:REM
160:ON H+1GOTO 200,230
170:NEXT J
180:NEXT S:GLCURSOR (-107,-W)
190:TEXT :COLOR 0:LF 2:CSIZE 1:FOR I
   =0TO N
192:LPRINT "Q";STR$ (I+1);"=";Q(I);"
   C";:LCURSOR 11
194:LPRINT "X";STR$ (I+1);"=";X(I);:
   LCURSOR 23
196:LPRINT "Y";STR$ (I+1);"=";Y(I)
198:NEXT I:LF 4:END
200:LINE -(X,Y):IF ABS X<108AND ABS
   Y<WGOTO 100
210:IF G=0LET H=1:GOTO 100
220:GOTO 170
230:PRINT X,Y:IF ABS X<108AND ABS Y<
   WGLCURSOR (X,Y):H=0
240:GOTO 100
```

1. ÁBRA



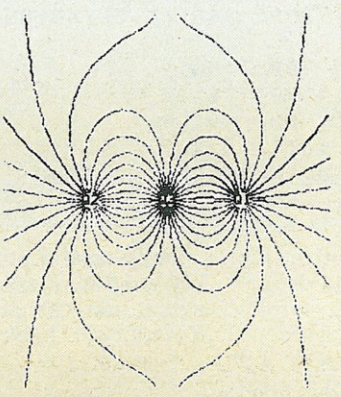
Q1=-1C X1=-50 Y1= 0
Q2= 1C X2= 50 Y2= 0

2. ÁBRA



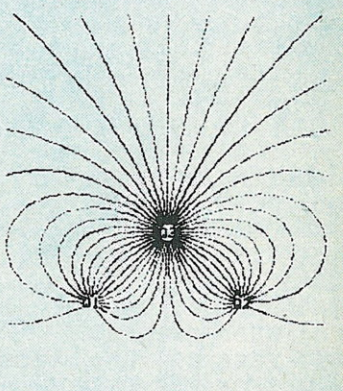
Q1= 1C X1= 50 Y1= 50
Q2= 1C X2= 50 Y2=-50
Q3= 1C X3= 50 Y3= 0
Q4=-1.5C X4=-50 Y4= 0

3. ÁBRA



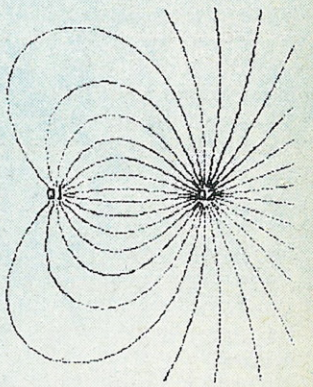
Q1= 2C X1= 50 Y1= 0
Q2= 2C X2=-50 Y2= 0
Q3=-3C X3= 0 Y3= 0

4. ÁBRA



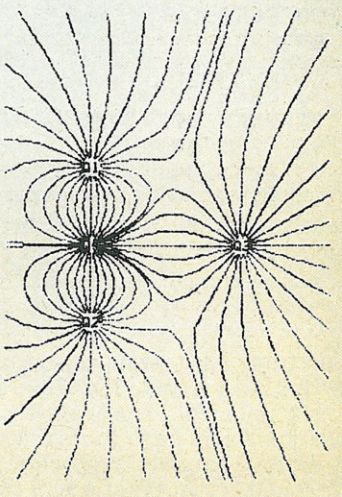
Q1= 2C X1=-50 Y1=-65
Q2= 2C X2= 50 Y2=-65
Q3=-5C X3= 0 Y3=-20

5. ÁBRA



Q1=-1C X1=-50 Y1= 0
Q2= 2C X2= 50 Y2= 0

6. ÁBRA



Q1= 1C X1=-50 Y1= 50
Q2= 1C X2=-50 Y2=-50
Q3= 1C X3= 50 Y3= 0
Q4=-1.5C X4=-50 Y4= 0

egy lehetőleg kis DL szakaszt vesz, a szakasz másik végpontjaként megkapva az erővonal következő pontját. Ez az iterációs eljárás természetesen csak közelítés, de DL kellően kicsire választásával tetszőlegesen pontossá tehető.

$$r_i^2=(X-X_i)^2+(Y-Y_i)^2$$

$$E_i=k\,\frac{Q_i}{r_i^2}$$

$$E_{xi}=\frac{E_i\,(X-X_i)}{r_i}$$

$$E_{yi}=\frac{E_i\,(Y-Y_i)}{r_i}$$

eredők:

$$E_x=\Sigma E_{xi}$$
$$E_y=\Sigma E_{yi}$$

$$E=\sqrt{E_x^2+E_y^2}$$

$$X'=X+\frac{E_x}{E}\ DL$$

$$Y'=Y+\frac{E_y}{E}\ DL$$

Az erővonalak a töltések körüli 6 egység sugarú kör pontjaiból indulnak. Egy erővonal ábrázolásának akkor van vége, ha valamely töltés 6 egység sugarú környezetén belül került, vagy az ábrázolt tartományon kívülre, illetve attól kellően távolra. Az erővonalak a töltések összegének megfelelően vagy csak a pozitív, vagy csak a negatív töltésekből indulnak. Ha a töltések összege zérus, akkor is a pozitív töltésekből indulnak.

A töltéseket egy olyan derékszögű koordináta-rendszerben helyezhetjük el, melyben az X tartomány rögzített (−108, 108), az Y tartományt pedig mi választhatjuk meg (a példáknál 120, 150 közötti érték). Mi határozhatjuk meg az egységnyi töltésből induló erővonalak számát is: értékük 26, 24, 13, 9, 15, 24 volt.

A programban választható egy MÓD érték; Ø választásra az *1., 2., 4., 6. ábránál* volt szükség; ezekhez tartozik olyan erővonal, melynek egy része a tartományon kívül esett. Ø értéknél a program nem hagyja abba az erővonal vizsgálatát, amikor az elhagyja a vizsgált tartományt, csak akkor, ha az erővonal „végtelenül messze" kerül a tartomány origójától, azaz amikor a távolság értéke túlcsordul. A sebesség növelése érdekében ilyen esetekben DL értéke a távolság négyzetével arányosan növekszik.

A program PTA—4000 (PC—1500) számítógépre készült, de kisebb módosításokkal más gépeken is futtatható.

MÉSZÁROS CSABA

HIBAÜZENETEK OLVASÁSA

Gyakran okoz bosszúságot — főleg fájlkezelés közben —, hogy a DOS hibaüzeneteit csak programból lehet olvasni. Ezen segít a rutinom. A gép bekapcsolása után töltsük be a BASIC-betöltőt, és indítsuk el. Ha olvasni akarjuk a DOS üzenetét, adjuk ki a SYS 49152 utasítást, és a szöveg a képernyőre íródik. Ha csak a gépi kódú részt vesszük fel, majd töltjük be, akkor új program írása előtt adjuk ki a NEW parancsot.

MARÓTI BENEDEK

```
DISK STATUS                                                    SEITE:1

C000                0          *=49152        ; DISK STATUS - $C000
FFBA                1 SETLFS   =65466 ; $FFBA
FFBD                2 SETNAM   =65469 ; $FFBD
FFC0                3 OPEN     =65472 ; $FFC0
FFC6                4 CHKIN    =65478 ; $FFC6
FFCF                5 CHRIN    =65487 ; $FFCF
FFD2                6 CHROUT   =65490 ; $FFD2
FFB7                7 READST   =65463 ; $FFB7
FFE7                8 CLALL    =65511 ; $FFE7
C000 A9 0F         11          LDA #15        ; LOGIKAI FILE SZAM
C002 A2 08         12          LDX #8         ; EGYSEGSZAM
C004 A0 0F         13          LDY #15        ; MEGNYITASI MOD
C006 20 BA FF      20          JSR SETLFS     ; PARAMETEREK BEIRASA
C009 A9 00         21          LDA #0         ; FILENEV HOSSZA
C00B 20 BD FF      30          JSR SETNAM     ; FILENEV BEIRSA
C00E 20 C0 FF      40          JSR OPEN       ; A FILE MEGNYITASA
C011 A2 0F         41          LDX #15        ; MASODLAGOS CIM
C013 20 C6 FF      50          JSR CHKIN      ; A 15-OS INPUTCSATORNA
C016 20 CF FF      60 W        JSR CHRIN ;    EGY KARAKTER OLVASASA
C019 20 D2 FF      70          JSR CHROUT     ; A KARAKTER KIIRASA
C01C 20 B7 FF      80          JSR READST     ; ST BEOLVASASA
C01F C9 00         81          CMP #0         ; HA ST = 0
C021 F0 F3         85          BEQ W          ; AKKOR UGRAS VISSZA
C023 20 E7 FF      90          JSR CLALL      ; A CSATORNA LEZARASA
C026 60            95          RTS            ; VISSZATERES BASIC-BE
C027               99          .END           ; VEGE

ZEILEN:26   SYMBOLE:9   FEHLER:0

CHKIN =FFC6 CHRIN =FFCF  CHROUT=FFD2  CLALL =FFE7 OPEN  =FFC0
READST=FFB7 SETLFS=FFBA  SETNAM=FFBD  W      =C016
```

```
1 FORI=49152TO49190:READA:POKEI,A:S=S+A
2 NEXT:IFS<>5547THENPRINT"ADATHIBA":END
3 SYS49152:CLR:NEW
4 DATA169,15,162,8,160,15,32,186
5 DATA255,169,0,32,189,255,32,192
6 DATA255,162,15,32,198,255,32,207
7 DATA255,32,210,255,32,183,255,201
8 DATA0,240,243,32,231,255,96

READY.
```

Térhatású ábrák

Programjaim írásakor látványos ábrák készítésére törekedtem; arra, hogy kétváltozós függvényeket három dimenzióban ábrázoltathassak.

Az *1. program* megértéséhez tudni kell, hogy szükség volt egy egyenletrendszerre (nevezzük ezt most koordinátatranszformációnak), amely tetszőleges térbeli ponthoz hozzárendeli képernyőbeli megfelelőjét. Ha x, y, z a térbeli és i, j a képernyőpont koordinátái, az

$i = x + y/2 \cdot \sqrt{2}$

$j = z + y/2 \cdot \sqrt{2}$ $\quad (2 \cdot \sqrt{2} = 2{,}8284)$

egyenletekkel axonometrikus képet állíthatunk elő, melyen az x tengely vízszintes, a z függőleges, az y 45 fokos, az y tengellyel párhuzamos szakaszok a képen félszeresükre rövidülnek.

A program annak a felületnek a képét rajzolja ki, melyet az egy sorban megadott kétváltozós függvény állít elő. A függvény értékét csak egy négyzetháló rácspontjaira határozza meg (tízesével halad végig az x és y értékeken), és az így kapott pontokat egyenes szakaszokkal összekötve alakítja ki a felület hálószerű képét. A 10—50 ciklus határozza meg a soron lévő szakasz végpontjainak x, y koordinátáit, a 100-as sor a z = f(x, y) koordinátákat, a 110—120-as sorok a megfelelő képpont-koordinátákat. 900—1070-ig történik a megjelenítés és a takart részek törlése.

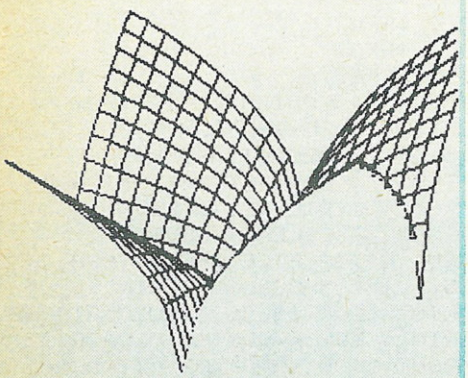
A koordinátatranszformációt megváltoztatva elérhetünk perspektív képet is. A szempontok, melyek alapján az új egyenleteket némi koordinátageometriával levezettem, a következők: adott a térben egy nézőpont és egy képsík; tetszőleges térbeli pont képe az őt és a nézőpontot összekötő egyenes metszéspontja a képsíkkal. Az egyszerűség kedvéért képsíknak az xz síkot választottam, a nézőpont koordinátáit DATA-ként adtam meg. Lehet kísérletezni a változtatásukkal *(2. program)*.

A takarás megoldása mindkét programban éppoly egyszerű, mint amilyen tökéletlen: hátulról haladunk előre, és az éppen felrajzolt vonal és az alsó képernyőszegély közötti területet „fehérítjük", így a később rajzolt részek eltakarják a régebbieket. (Az yz síkkal párhuzamos szakaszoknál nincs szükség törlésre, így a törlést a t változó kapcsolgatja ki-be. Magát a törlést az 1040-es sor végzi el.) Így a felületekre alulról nem láthatunk rá. Változtathatunk ezen az 1030-tól kezdődő sorok *3. program* szerinti módosításával. Akkor a program csak az éppen kirajzolt és az eggyel későbbi vonal közötti sávot tisztítja meg. Csak az axonometrikus változatot tökéletesíthetjük így, a perspektivikus program javításához valamivel több matematikai ügyeskedés kell. A lista könnyebb megértéséhez: $3{,}5355 = 10/2 \cdot \sqrt{2}$,10 a lépésköz.

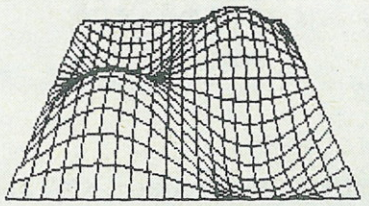
Nem titkolom el a program gyengéit sem: 1. lassú; 2. nem szereti a nem folytonos és az értelmezetlen függvényeket, 3. se a túl meredek vonalakat; 4. jól mutat, de mire jó?

SZABÓ PÉTER

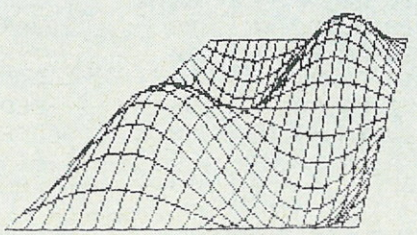
1. program



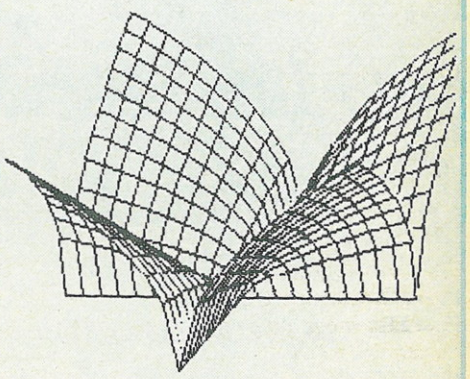
```
  1>DEF FN f(x,y)=SQR ABS ((x-1
00)*(y-100))
  10 FOR y=200 TO 0 STEP -10
  20 FOR x=0 TO 200 STEP 10
  30 LET x1=x+10: LET y1=y
  40 IF x<200 THEN LET t=1: GO S
UB 100
  50 LET x1=x: LET y1=y-10
  60 IF y>0 THEN LET t=0: GO SUB
 100
  70 NEXT x
  80 NEXT y
  90 STOP
 100 LET z=FN f(x,y): LET z1=FN
f(x1,y1)
 110 LET i=x+y/2.8284: LET j=z+y
/2.8284
 120 LET i1=x1+y1/2.8284: LET j1
=z1+y1/2.8284
 900 IF t=0 AND i>=0 AND i1>=0 A
ND i<=255 AND i1<=255 AND j>=0 A
ND j1>=0 AND j<=175 AND j1<=175
THEN PLOT i,j: DRAW i1-i,j1-j: R
ETURN
1000 LET di=ABS (i1-i): LET dj=A
BS (j1-j)
1010 LET k=di*(di>=dj)+dj*(di<dj
)
1020 LET ei=(i1-i)/k: LET ej=(j1
-j)/k
1030 FOR n=0 TO k
1040 IF i>=0 AND i<=255 AND j>=0
 AND j<=175 THEN PLOT i,j: IF t
THEN DRAW INVERSE 1;0,-j
1050 LET i=i+ei: LET j=j+ej
1060 NEXT n
1070 RETURN
```



```
  1 DEF FN f(x,y)=30*SIN (PI/10
0*x)*SIN (PI/100*y)
  2 READ x0,y0,z0
  10 FOR y=200 TO 0 STEP -10
  20 FOR x=0 TO 200 STEP 10
  30 LET x1=x+10: LET y1=y
  40 IF x<200 THEN LET t=1: GO S
UB 100
  50 LET x1=x: LET y1=y-10
  60 IF y>0 THEN LET t=0: GO SUB
 100
  70 NEXT x
  80 NEXT y
  90 STOP
 100 LET z=FN f(x,y): LET z1=FN
f(x1,y1)
 110 LET i=x-y*(x0-x)/(y0-y): LE
T i1=x1-y1*(x0-x1)/(y0-y1)
 120 LET j=z-y*(z0-z)/(y0-y): LE
T j1=z1-y1*(z0-z1)/(y0-y1)
 900 IF t=0 AND i>=0 AND i1>=0 A
ND i<=255 AND i1<=255 AND j>=0 A
ND j1>=0 AND j<=175 AND j1<=175
THEN PLOT i,j: DRAW i1-i,j1-j: R
ETURN
1000 LET di=ABS (i1-i): LET dj=A
BS (j1-j)
1010 LET k=di*(di>=dj)+dj*(di<dj
)
1020 LET ei=(i1-i)/k: LET ej=(j1
-j)/k
1030 FOR n=0 TO k
1040 IF i>=0 AND i<=255 AND j>=0
 AND j<=175 THEN PLOT i,j: IF t
THEN DRAW INVERSE 1;0,-j
1050 LET i=i+ei: LET j=j+ej
1060 NEXT n
1070 RETURN
2000 DATA 100,-250,200
```



2. program



```
1030>LET jn=FN f(x+3.5355,y-10)+
(y-10)/2.8284: LET jn1=FN f(x+k*
ei+3.5355,y-10)+(y-10)/2.8284
1040 LET ejn=(jn1-jn)/k
1050 FOR n=0 TO k
1060 IF i>255 OR j<0 OR j>175 TH
EN GO TO 1110
1070 PLOT i,j: IF y*t=0 THEN GO
TO 1110
1080 IF jn<0 THEN DRAW INVERSE 1
;0,-j: GO TO 1110
1090 IF jn>175 THEN DRAW INVERSE
 1;0,175-j: GO TO 1110
1100 DRAW INVERSE 1;0,jn-j
1110 LET i=i+ei: LET j=j+ej: LET
 jn=jn+ejn
1120 NEXT n: RETURN
```

3. program

BASIC és gépi kód

Legutóbb a feltételes ugróutasításokkal ismerkedtünk meg. Most a használatukra látunk egy egyszerű példát. Megismerünk néhány újabb címzési módot, melyek alkalmazására a következő részben találunk példákat.

A mintaprogram

Mintaprogramunk BASIC betöltőprogramja az 1986/5. számban jelent meg, és akkor az időmérés bemutatására szolgált. A rutin egyszerű feladatot old meg: letörli a képernyőt és 999 darab csillagkaraktert ír fel rá.

A lista disassemblerrel készült. Most ennek alapján lépésről lépésre megvizsgáljuk a rutin működését. A képernyőre a BSOUT KERNAL rutinnal írunk. A BSOUT sok részfeladatot hajt végre, ez okozza, hogy rutinunk viszonylag lassan működik. Előnye viszont e módszernek, hogy a mintaprogram mindhárom géptípuson változtatás nélkül futtatható — eltekintve a VC20 23 × 22 karakteres képernyőméretétől.

A $033C . . . $033E címen lévő utasítások a BASIC-beli PRINT CHR$(147); utasítás feladatát hajtják végre. A pontosvessző azt jelöli, hogy a BSOUT végrehajtása során a kurzor egy pozíciót lép jobbra, de nem kerül automatikusan a következő sorba, mint egy PRINT utasítás esetén.

A következő két utasítás „felhúzza a ciklust". A rutin az X és Y regisztereket ciklusszámlálóként használja. Az említett utasítások ezeknek a kezdőértékét állítják be. Hogy miért éppen az itt látható értékekre, arra még alább visszatérek. Megjegyzem, hogy a számlálásra az indexregiszterek helyett két memóriabájt is megfelelne, de ez semmilyen előnnyel nem járna, csak a rutin mérete lenne nagyobb.

A következő utasítás a csillagkarakter ASCII kódját tölti az A regiszterbe.

Két egymásba ágyazott ciklust láthatunk ezután. A belsőt a $0347 . . . $034B, a külsőt a $0347 . . . $034E címeken lévő utasítások alkotják. A közös ciklusmag a $0347 című utasítás: ez a BSOUT hívásával egy csillagot ír a következő képernyő-pozícióba.

```
033c  a993        lda #$93
033e  20d2ff      jsr $ffd2
0341  a0e7        ldy #$e7
0343  a204        ldx #$04
0345  a92a        lda #$2a
0347  20d2ff      jsr $ffd2
034a  88          dey
034b  d0fa        bne $0347
034d  ca          dex
034e  d0f7        bne $0347
0350  60          rts
```

Nézzük a belső ciklus végrehajtását. A DEY utasítás eggyel csökkenti az Y regiszter tartalmát, egyúttal beállítja az N és Z feltételbiteket. Minket most az utóbbi érdekel. A Z bit értéke addig Ø, míg Y tartalma nullától különböző érték. Ekkor a BNE utasítás a ciklusmagra adja a vezérlést. Ha Y tartalma nullára csökken, a Z bit 1-re állítódik, és a futás a BNE-t követő utasításon folytatódik.

Ugyanez játszódik le a külső ciklusban is, de az Y regiszter helyett az X szerepel benne. Ez összesen négyszer fut le, az X regiszter négy különböző értékénél. Számoljuk meg, hogy a belső ciklus — ezzel együtt a ciklusmag — hányszor hajtódik végre. Amikor X-ben 4 van, a belső ciklus 231-szer fut le. X többi előforduló értékénél a belső ciklus minden alkalommal 256-szor hajtódik végre. A ciklusmag végrehajtásainak száma: 231 + 3 * 256 = 999. Tehát a rutin ennyi csillagot fog a képernyőre írni.

Megfigyelhető, hogy a DEX végrehajtásakor az Y regiszterben $ØØ van. Ez a következő DEY végrehajtásakor $FF-re változik, aminek következményeképpen a DEY-t követő BNE a $0347 címre ugrat. Ez — mint néhány, különböző helyeken publikált programból kiderült — nem mindenki számára nyilvánvaló.

Azt is érdemes megfigyelni, hogy a disassembler által készített listán a gépi kódban a relatív ugrási távolság látható, míg a BNE emlékeztetőkód (idegen szóval mnemonik) után már a tényleges ugrási cím.

Az egyszerű indexelt címzési módok

Az egyszerű indexelt címzési módok az utasításban szereplő memóriabájt címét befolyásolják: az operandusban található értékhez hozzáadódik a művelethez tartozó indexregiszter tartalma úgy, hogy ezt a tartalmat előjel nélküli egész számnak tekinti. A módosítás értéke a $ØØ—SFF (Ø—255) tartományba eshet. Az összegzésből létrejövő cím adja a műveletben ténylegesen részt vevő memóriabájt címét.

Négy egyszerű indexelt címzési mód van, ezeket két szempont szerint csoportosíthatjuk: 1. attól függően, hogy a címzést módosító regiszter az X vagy az Y; 2. attól függően, hogy nullalapos vagy abszolút címet módosít az indexelés — más szóval: egy- vagy kétbájtos a művelet operandusa.

Az indexelt címzési mód alkalmazására a következő alkalommal láthatunk példákat. A további két indexelt címzési módot is a következő részben ismerhetjük meg.

(„Az ismeretlen C16" című sorozat első részében jeleztem, hogy a közölt információk helyenként kapcsolódnak ezekhez az írásokhoz. Jelen sorozatban idáig elmulasztottam ezt megemlíteni — most pótolom.)

BARNA LÁSZLÓ

COMMODORE 64

OVERLAY— CHAIN— APPEND

Az 1986. 10. szám 25. oldalán, amikor a program titkosításáról volt szó, találkoztunk azzal a feladattal is, hogy miként lehet a titkosítást végző programot a titkosítandó programmal összekapcsolni. A feladatot általánosítva így fogalmazhatjuk meg: hogyan lehet két önálló programot egy egységgé összekapcsolni? A titkosító program után begépelhető a másik, de ez hosszadalmas bíbelődéssel járhat. Egyszerűbb megoldásnak tűnt, hogy be kell tölteni a titkosítandó programot, majd a BASIC progamszerkesztő segítségével a viszonylag rövid titkosító programot eléje írni. Sajnos azonban ez az út nem járható, egyrészt mert a titkosító program egy utasítássora két képernyősornál hosszabb, másrészt mert nem szabványos BASIC kódokat tartalmaz.

Hasonló feladat előtt állunk akkor is, ha két hosszú program összeolvasztására készülünk úgy, hogy elkerüljük az ismételt begépelést. Ebbe a témakörbe tartozik az is, ha olyan hosszú programot akarunk futtatni, ami egyszerre nem fér el a tárban, ezért a feladatot részekre bontva vagyunk kénytelenek elvégezni.

Az alábbiakban megvizsgáljuk, hogy milyen kapcsolatba hozható a tárban elhelyezett program egy külső tárolón elhelyezett programmal. A programok ilyen jellegű kapcsolatának megteremtését OVERLAY vagy átfedéstechnikának nevezik.

A kapcsolatot minden esetben a külső tárolón elhelyezett program betöltésével hozzuk létre. Ennek kulcsszava a LOAD, ezért egyrészt a LOAD különböző működési lehetőségeit, másrészt a BASIC program térbeli elhelyezkedését és az elhelyezkedést jelző mutatókat elemezzük. (Az utóbbiról részletesebb ismertetés volt az októberi BASIC RAM felosztás, illetve a novemberi Programtitkosítás c. cikkekben.) Most csak a mutatók elnevezését és elhelyezkedését ismétlem meg.

43—44 SOB (Start Of Basic) BASIC-terület eleje alaph: 2049

45—46 SOV (Start Of Variables) Változók kezdete alaph: 2051
47—48 SOA (Start Of Arrays) Tömbök kezdőcíme alaph: 2051
49—50 EOA (End Of Arrays) Tömbök vége alaph: 2051
51—52 BOS (Bottom Of Strings) Sztringterület alja alaph: 40960
55—56 TOM (Top Of Memory) BASIC-terület vége alaph: 40960

Nézzük tehát a LOAD-ot!

BASIC-ben négyféle LOAD-funkciót valósíthatunk meg:

1. Parancsmódban másodlagos cím nélkül. Pl. :LOAD"TEST",8
2. Parancsmódban másodlagos címmel. Pl. :LOAD"TEST",8,1
3. Utasításmódban másodlagos cím nélkül.
4. Utasításmódban másodlagos címmel.

Ez a felosztás talán fölöslegesnek tűnik, pedig komoly gondokat okozhat, ha nem ismerjük az egyes funkciók közötti különbségeket.

CHAIN1—A

```
1000  REM  CHAIN1-A
1010  :
1100  A = 1:A$ = "CHAIN1"
1110  PRINT "[CD]";A,A$
1120  LOAD "CHAIN2-A",8
```

CHAIN2—A

```
2000  REM  CHAIN2-A
2010  :
2020  POKE 45, PEEK(174): POKE 46, PEEK
      (175): CLR
2100  B = 2:B$ = "CHAIN2"
2110  PRINT "[CD]";A,A$: PRINT B,B$
2120  END
```

CHAIN1—B

```
1000  REM  CHAIN1-B
1010  :
1100  A = 1:A$ = "CHAIN1" + ""
1110  PRINT "[CD]";A,A$
1120  END
```

CHAIN2—B

```
2000  REM  CHAIN2-B
2010  :
2100  B = 2:B$ = "CHAIN2"
2110  PRINT "[CD]";A,A$: PRINT B,B$
2120  END
```

CHAIN1—C

```
1000  REM  CHAIN1-C
1010  :
1020  POKE 45,0: POKE 46,64: CLR
1100  A = 1:A$ = "CHAIN1" + ""
1110  PRINT "[CD]";A,A$
1120  LOAD "CHAIN2-C",8
```

CHAIN2—C

```
2000  REM  CHAIN2-C
2010  :
2100  B = 2:B$ = "CHAIN2"
2110  PRINT "[CD]";A,A$: PRINT B,B$
2120  END
```

Parancsmód

Azt jelenti, hogy a LOAD funkciót közvetlenül a billentyűzetről a kulcsszó beírásával, majd a RETURN lenyomásával aktivizáljuk. Ekkor a következők történnek:

— betöltődik a program a tárba,

— a SOV mutató felveszi a tárban elhelyezett program utáni első szabad bájt címét,

— egy CLR utasítás hajtódik végre, ami a többi mutatót igazítja a SOV-hoz,

— megjelenik a READY felirat, és a vezérlés visszatér a beviteli várakozó ciklusba.

Utasításmód

Ugyanez akkor kerül utasításmódban végrehajtásra, ha a LOAD kulcsszó a programban van, és a program soron következő végrehajtandó kulcsszava éppen a LOAD lesz. Ilyenkor betöltődik a program a tárba, majd a vezérlést a SOB által jelzett program kezdőcímén lévő első BASIC utasítás kapja meg.

Fontos megfigyelés: ilyenkor a mutatók értéke nem változik, és a program nem a LOAD utasítást követő utasításnál, hanem a program első utasításánál folytatódik.

Másodlagos cím

A program betöltésének kezdőcímét fogja meghatározni az alábbiak szerint.

— A másodlagos cím hiánya (Ø) azt jelenti, hogy a program elhelyezését a SOB által jelzett címen kell elkezdeni. Ha a SOB-ot nem változtattuk meg, akkor az új program az előző program helyére kerül.

— Ha a másodlagos cím 1, akkor az elhelyezési címet nem a SOB-ból, hanem a háttértárolón a program előtt tárolt mutatóból vesszük. Ez a mutató a SAVE-kor íródott fel.

A fentiek ismerete magyarázatot adhat néhány rejtélyre. Ilyen szituációkra gondolunk: Parancsmódban betöltünk egy BASIC programot, és azt hibátlanul futtatjuk le. Mivel úgy gondoljuk, hogy szükségünk van egy gépi kódú segédprogramra is, azt szintén parancsmódban betöltjük például a szalagpufferbe (828—1019 tárcímek). Most újra elindítjuk a BASIC programot, de ez rövid működés után furcsa dolgokat produkál. Ha ekkor kétségbeesetten megpróbáljuk legalább SAVE-vel a zabolátlan programunk kimentését, ismét nem várt dolgot tapasztalunk: a néhány blokkból álló programunk kimentése valahogy túl sokáig tart.

Mi a magyarázata mindennek? Lássuk sorban:

— A parancsmódban betöltött gépi kodú program után a SOV átállítódott.

APPEND1—A

```
1000  REM  APPEND1-A
1010  :
1100  A = 1:A$ = "APPEND1"
1110  PRINT "[CD]";A,A$
1120  GOSUB 2100: END
```

APPEND2—A

```
2000  REM  APPEND2-A
2010  :
2100  B = 2:B$ = "APPEND2"
2110  PRINT "[CD]";B,B$
2120  RETURN
```

APPEND1—B

```
1000  REM  APPEND1-B
1010  :
1100  A = 1:A$ = "APPEND1"
1110  PRINT "[CD]";A,A$
1120  X = PEEK(45) + PEEK(46) * 256:X
      = X - 2
1130  POKE 44, INT(X / 256): POKE 43,X
      - PEEK(44) * 256
1140  LOAD "APPEND2-B",8
1150  GOSUB 2100
1160  :
1170  END
```

APPEND2—B

```
2000  REM  APPEND2-B
2010  :
2020  POKE 45, PEEK(174): POKE 46, PEEK
      (175): CLR
2030  POKE 43,1: POKE 44,8
2040  GOTO 1150
2050  :
2100  B = 2:B$ = "APPEND2"
2110  PRINT "[CD]";A,A$: PRINT B,B$
2120  RETURN
```

APPEND1—C

```
1000  REM  APPEND1-C
1010  :
1020  POKE 45,0: POKE 46,64: CLR
1050  :
1100  A = 1:A$ = "APPEND1" + ""
1110  PRINT "[CD]";A,A$
1112  :
1114  X = PEEK(174) + PEEK(175) * 256
      :X = X - 2
1116  POKE 44, INT(X / 256): POKE 43,X
      - PEEK(44) * 256
1130  :
1140  LOAD "APPEND2-C",8
1150  GOSUB 2100
1160  :
1170  END
```

APPEND2—C

```
2000  REM  APPEND2-C
2010  :
2030  POKE 43,1: POKE 44,8
2040  GOTO 1150
2050  :
2100  B = 2:B$ = "APPEND2"
2110  PRINT "[CD]";A,A$: PRINT B,B$
2120  RETURN
```

— A BASIC program a változóit nem a kívánt helyre kezdi elhelyezni, és így az nem kívánt tárterületeket ír át.

— Amikor a SAVE-vel próbálkozunk, ez a program kimentését a SOB-nál kezdve a SOV-ig hajtja végre, de a SOV értéke most kisebb a SOB-nál, így először a SOB fölötti teljes tárterületet menti ki, majd a számlálást nulláról indítva folytatja a mentést a SOV-ig.

A megoldás: ilyen esetben először a gépi kódú programokat és utána a BASIC programot töltjük be.

Hasonló buktatót rejt az is, ha egy BASIC programutasítás utasításmódban meghív (LOAD) egy másik programot, de a második program hosszabb, mint az első. Ekkor — mivel a SOV értéke most nem változott — a változók ugyanarra a tárterületre kerülnek, ahol a program vége is elhelyezkedik, így a változók elrontják a program végét.

A megoldás a C64 BASIC-ben nem ismert CHAIN utasítás. Ezt a következő módon szimulálhatjuk.

CHAIN

A CHAIN (láncolás) a programok közötti olyan kapcsolat, amikor a tárban lévő program helyére, annak folytatásaként egy külső tárolóról új programot töltünk be, majd azt elindítjuk.

— Direkt módban a LOAD és a CHAIN közötti különbség csak annyi, hogy LOAD után a programot külön paranccsal kell indítani.

— Utasításmód: CHAIN1—A CHAIN2—A programpár

Utasításmódban a betöltött program automatikusan indul. Az egyetlen tennivaló a betöltött program első sorában olyan utasítás elhelyzése, ami a SOV változó értékét beállítja. Erre példa a CHAIN1—A és a CHAIN2—A programpár. Betöltve és elindítva a CHAIN2—A-t, az behívja és elindítja a CHAIN1—A-t. Bár a CHAIN2—A hosszabb, mint az első program volt, a változók mégsem rontják el a program végét, mert a SOV a program vége utáni bájtra mutat. A futás eredményéből azt is látjuk, hogy az első program változóit a második program nem ismeri.

A CHAIN speciális esete, ha az első program helyére úgy akarjuk betölteni a második programot, hogy ez az első programmal előállított változókat is használni tudja. Ilyenkor külön területet jelölünk ki a programnak és külön a változóknak.

Direkt mód: CHAIN1—B CHAIN2—B programpár

— Betöltjük a CHAIN1—B programot.

— Kijelöljük a program és a változó terület határát. Például: POKE45,Ø:POKE46,64 (most 16384 a SOV értéke).

— Elindítjuk az első programot. (A változók a SOV fölött lesznek elhelyezve.)

— A program lefutása után a többi változó mutatóit elmentjük egy szabad tárterületre, például 49152-től kezdődő címekre.

— FORX=ØTO7:POKE49152+X,PEEK(45+X):NEXT

— Most betöltjük a CHAIN2—B-t.

— Ezután fordított sorrendben visszaállítjuk a mutatókat.

— FORX=7TOØSTEP—1:POKE45+X,PEEK(49152+X):NEXT

A második programot nem RUN, hanem GOTO paranccsal indítjuk el:

— GOTO 2ØØØ

A program ekkor kiírja az első és a második program változóit is. A programban definált sztringváltozókat csak úgy tudtuk átvenni az első programból, hogy azokat a programterületről áthelyeztük a sztringterületre egy +"" művelettel az 1ıØØ-as sorban.

Utasításmód: CHAIN1—C CHAIN2—C programpár

Utasításmódban ugyanezen feladat megoldása a CHAIN1—C és CHAIN2—C programpárral sokkal egyszerűbb. Itt nincs más teendő, mint a CHAIN1—C program betöltése és elindítása. A többi a gép dolga. Érdemes megfigyelni a parancs- és az utasításmód közötti különbségeket ezeken a programpárokon keresztül is.

APPEND

Az APPEND (hozzáfűzés) a programok közötti olyan kapcsolat, amikor a tárban lévő program végéhez egy külső tárolóról egy programrészt fűzünk hozzá. Az összefűzés után a két programrész a tárban egységes programot alkot. Az összefűzés feltétele, hogy a két programrészben azonos sorszámok ne forduljanak elő, és a sorszámok az összefűzés után növekvő sorrendben kövessék egymást.

Az összefűzés megvalósításához ismerni kell az első program helyfoglalását a tárban. Azt már láttuk, hogy minden BASIC sor első két bájtja azt a tárcímet adja meg, ahol a soron következő BASIC utasítássor megtalálható a tárban. Ha ez a két bájt nulla, akkor ez azt jelenti, hogy nincs több BASIC utasítás, tehát itt van a program vége. Mivel alapértelmezésben egy SOV a két nulla bájtot tartalmazó tárcímek utáni címre mutat, a második programot a SOV—2 címtől kezdve kell betölteni ahhoz, hogy a második program a tárban úgy helyezkedjen el, mintha az előző program folytatása lenne. Az összefűzéshez tehát a következő lépések szükségesek:

— A SOB SOV—2-re állítása.

— A második program betöltése.

— A SOV-ot az összefűzött program végére kell állítani.

— A SOB visszaállítása eredeti értékére.

A BASIC sorok elején álló mutatókat a LOAD a SOB értékének megfelelően automatikusan átírja, lehetővé téve, hogy egy adott helyről kimentett BASIC programot más tárcímre is be lehessen tölteni.

Nézzük tehát a direkt és az utasításmód közötti különbségeket a mintaprogramok segítségével.

Direkt módban: APPEND1—A APPEND2—A programpár

— Betöltjük az APPEND1—A programot:
LOAD"APPEND1—A",8

— Kiszámítjuk a második program betöltési címét:
X=PEEK(45)+PEEK(46)*256:X=X—2

— Átírjuk a SOB-ot
POKE44,INT(X/256):POKE43,X—PEEK(44)*256:NEW

— Betöltjük a második programot:
LOAD"APPEND2—A",8

— Visszaállítjuk a SOB-ot:
POKE43,1:POKE44,8

Utasításmódban: APPEND1—B APPEND2—B programpár

Az eljárás hasonló a direkt módhoz, de a szükséges lépéseket most a programba építettük be. Vegyük észre, hogy a második program betöltése után a program futása nem a LOAD-ot követő utasításon folytatódik, hanem a SOB utáni első utasításon, ami nem más, mint a második program első sora! A főprogramba csak úgy lehetséges a visszatérés, ha a második program visszaállítja a SOB-ot, és olyan ugróutasítást tartalmaz, ami az első programra mutat.

APPEND1—C APPEND2—C programpár

Itt a különbség csak annyi, hogy a hozzáfűzött második program használhatja az első program változóit is. Az eljárás már teljesen ismerős kell hogy legyen a CHAIN és az APPEND előző változata alapján.

ZSOM BÉLA

Tömbök kezelése FORTH-ban

A FORTH nyelv szabványosításával foglalkozó Forth Interest Group (FIG) elnevezésű szervezet 1979-ben fogadta el azt a szabványt, amely 243 FORTH szó definíciójából állt. Ez az alapszókészlet nem kínál tömb kezelésére szolgáló definíciót, de tartalmazza a <BUILDS ... DOES> szerkezetet, amelynek segítségével új adatszerkezetek és utasítások értelmezhetők.

Tömb típusú adatszerkezet létrehozásakor a <BUILDS és DOES> közötti utasításokkal foglaljuk le a tömb számára a szükséges területet a memóriában. A terület a tömb szótári elemének PFA mezője. Ezek az utasítások fordításkor hajtódnak végre.

Egy szótári elem PFA mezejében az alábbiak egyike található:

— végrehajtási cím(ek) a végrehajtás sorrendjében, ha a szótári elem egy másodlagos FORTH szó,
— gépi kódú program primitív szó esetén,
— konstans,
— változó aktuális értéke,
— tömbelemek.

Mivel a tömböt leíró szótári elemet újabbak is követhetik, ezért egy tömb méretét a definiálása után már nem lehet megváltoztatni. Célszerű az egyes tömbelemeknek Ø kezdőértéket adni. A DOES >utáni utasítások segítségével számíthatjuk ki egy tömbelem címét. Ezek az utasítások végrehajtáskor működnek.

Egy m*n méretű kétdimenziós, 2 bájtos szavakból álló tömb a(i,j) elemének távolsága a tömb kezdőcímétől, mint a bázistól számítva.(

((i−1) *n+j)*2−1 bájt.

Látható a képletből is, hogy a tömb elemein kívül az egyes dimenziók maximális méretét is tárolnunk kell. Ezeket célszerű a PFA mező elejére helyezni, a tömb elemeit megelőzően.

n	m	a(1,1)	...	a(m,n)

↑
PFA mező kezdőcíme

A definíció:

```
: 2TOMB
  <BUILDS PDUP , , * 0
                    DO , LOOP
  DOES> DUP @ 4 ROLL 1 - * ROT +
2 * 3 + ;
```

A PDUP definíciója:

```
:PDUP (n1 n2 — n1 n2 n1 n2)
  OVER OVER;
```

A ROLL a verem (stack) n-edik elemét a paraméterverem tetejére helyezi; nem szerepel a standard FORTH-ban, de több implementációban megtalálható. Definíciója:

```
:ROLL
  SP @ SWAP 1 — 2 * DUP > R 2 +
+ DUP @ R > SWAP >R >R DUP 1 +
SWAP
  1 — R> 0
        DO DUP I — C @ 3 PICK I
        — C!
        LOOP
  DROP DROP DROP R> ;
```

A PICK a verem n-edik elemét a paraméterverem tetejére másolja; akárcsak a ROLL, ez sem standard FORTH szó.

```
: PICK
  SP @ 2 + SWAP 1 — 2 * + @ ;
```

Hozzunk létre egy 3*4-es tömböt TERULET néven:

```
3 4 2TOMB TERULET
Legyen a TERULET(2,3)=10
10 2 3 TERULET !
```

A tömb egy elemének kiolvasása a beíráshoz hasonló:

```
2 3 TERULET @.
  10 ok
```

A fentiek mintájára bevezethetünk 3 dimenziós tömböket is. Egy m*n*o szó méretű tömb a(i,j,k) elemének távolsága a tömb kezdőcímétől

((i−1)*n*o)+((k−1)*n+j))*2−1 bájt.

A 3 dimenziós adatszerkezet definíciója:

```
: 3TOMB
  <BUILDS DUP , SWAP DUP , ROT
DUP,**
  0 DO
    0,
  LOOP
  DOES> DUP @ SWAP 2 + DUP
        >R @ *4 ROLL 1 — *
        SWAP 1 — R>
        DUP >R @ * ROT+ + 2 *
        2 + R>+ ;
```

A két- és háromdimenziós tömbök értelmezése és kezelése elvben teljesen azonos:

```
3 4 2 3TOMB KOCKA
Legyen a KOCKA (1,2,2)=14
14 1 2 2 KOCKA!
```

Egy elem kiolvasása és kiírása a képernyőre:

```
1 2 2 KOCKA @.
14 ok
```

Ez a tömbkezelési eljárás beépíthető bármely gépen futó FORTH rendszerbe, mivel nincs gépfüggő része.

BALOGH LÁSZLÓ

A tartalomleírások az alábbi folyóiratokban megjelent programlistákról készültek:

A folyóirat neve	Kódja
64'er Magazin	64er
Chip Magazin	chip
Commodore Horizons	coho
Commodore Microcomputers	comi
Compute!	cute
Computer Persönlich	pers
Happy Computer	happ
hc - Mein Home-Computer	hc
mc - Zeitschrift	mc
Run /USA/	run
Sinclair User	sinc
Your Sinclair	ysin

A tartalomleíró szóláncokat permutáltuk, a szóláncvariánsokat pedig alfabetikusan rendeztük.

A folyóiratok a SZÁMALK szakkönyvtárában is föllelhetők. A másolás feltételeiről bővebb felvilágosítást a 853-111/251 telefonszámon kaphatnak.

A tartalomleírás egy szóláncból áll, majd a listában ezt követi a forrás megjelölése a folyóirat azonosítójával, a megjelenés dátumával és a cikk előkeréséséhez a kezdő oldalszám és a terjedelem megadásával. A mellékelt lista értelmezéséhez még az alábbiakat kell tudni. A tartalomleírás szóláncában elsőként a téma átfogó megnevezése, utána a számítógéptípus(ok), ezt követően a szűkebben jelölt tartalom meghatározása szerepel, majd esetlegesen néhány, a közleményt minősítő adat (például : cikksorozat).

A forráshely karaktersorozatát nyíl vezeti be, melyet a / jelig a folyóiratok azonosítója, a két / jel között az évszám, folyóiratszám és kötőjellel a kezdő oldalszám követi, a végén pedig a közlemény teljes oldalterjedelme áll.

Európa legjobb Commodore újságjáról, a 64'er Magazinról másféle információszolgáltatás is kérhető. Szolnoki Béla vállalja, hogy havi 20 Ft-ért (+ a postaköltség) elküldi a folyóirat tartalomjegyzékeinek megjegyzésekkel is ellátott fordítását - a lap 1986. 07. havi számától kezdődően bármelyikről, megbízás esetén folyamatosan. Ennek alapján a megrendelő kiválaszthatja, hogy számára melyik cikkek fontosak, és ezek fordítását is megkaphatja oldalanként 5,-Ft-ért (+ a postaköltség), ha ír a következő címre:

"Számítástechnikai információszolgáltatás". Szolnoki Béla, Budapest, Pf.: 400. 1446

µINFORM

PROGRAMLISTA
64er beviteli utmutato sor-osszeg ellenorzes(mse) ->64er/86.02-57/2
PROGRAMLISTA
adatatvitel(telefonos) c nyelv cp/m-80 ->pers/86.03-57/4
PROGRAMLISTA
adatbazis leKerdezes atari 400/800;xl;xe muveletgyorsitas fuzerKezeles ->cute/86.02-58/4
PROGRAMLISTA
animacio commodore 64 sprite-mozgatas az <impossible mission>-nal is alKalmazott techniKa alapelve ->happ/86.02-69/3
PROGRAMLISTA
apple ii alKalmazasi utmutato <speed calc> ->cute/86.02-88/14
PROGRAMLISTA
apple ii atari 400/800;xl;xe commodore 64 jateKprogram <high rise> ->cute/86.02-49/8
PROGRAMLISTA
apple ii monitorKep tarolas ->cute/86.02-79/2
PROGRAMLISTA
apple ii pascal programozas betutipus valtas ->pers/86.03-56/2
PROGRAMLISTA
apple ii programozasi fogasoK ->cute/86.02-82/3
PROGRAMLISTA
apple iie grafiKus programozas <applesoft>modositas felbontas javitasa ->mc/86.02-42/2
PROGRAMLISTA
apple atari commodore compute! beviteli utmutato ->cute/86.02-114/10
PROGRAMLISTA
apple jateKprogram <brain trainer> ->hc/86.02-63/7
PROGRAMLISTA
atari 400/800;xl;xe basic programozas hibajavitas ->cute/86.02-74/3
PROGRAMLISTA
atari 520st logo programozas rajzKeszites egerrel ->cute/86.02-77/2
PROGRAMLISTA
atari monitorprogram lemezegyseg(atari) <duex st> ->happ/86.02-86/4
PROGRAMLISTA
atari beviteli-sor osszegezes ->hc/86.02-42/1
PROGRAMLISTA
atari ciKKsorozat grafiKus programozas jateKprogram Keszites ->happ/86.02-98/3
PROGRAMLISTA
atari jateKprogram <caveman> ->hc/86.02-51/3
PROGRAMLISTA
barKacsolas commodore 64 adatatvitel i lehetosegeK buszvezerles ->64er/86.02-93/5
PROGRAMLISTA
basic programozas commodore 64 muveletgyorsitas ->cute/86.02-72/2
PROGRAMLISTA
checKsum-Kiegeszitesek a <run/85.08) litaihoz ->run/86.02-118/3
PROGRAMLISTA
ciKKsorozat commodore 64 <ascompiler 64> ->64er/86.02-68/3
PROGRAMLISTA
ciKKsorozat commodore 64 eprom beegeto programozas(eprom) ->64er/86.02-64/2
PROGRAMLISTA
commodore 128 <ultra hi-res> ->run/86.02-34/14
PROGRAMLISTA
commodore 128 sajat Karakterek generalasa ->run/86.02-42/3
PROGRAMLISTA
commodore 64 <softscroll> szoveg-gorgetes ->mc/86.02-41/1
PROGRAMLISTA
commodore 64 fuggvenyabrazolas grafiKus programozas muveszi grafiKa ->happ/86.02-107/2
PROGRAMLISTA
commodore 64 grafiKon nyomtatas <poster-hardcopy> alKalmazasi utmutato ->happ/86.02-65/2
PROGRAMLISTA
commodore 64 grafiKus programozas muveszi grafiKa sinus-fuggvenyeK alKalmazasa ->hc/86.02-82/2
PROGRAMLISTA
commodore 64 jateKprogram <butterfly> ->coho/86.02-41/2
PROGRAMLISTA
commodore 64 jateKprogram <fast-food chef> ->run/86.02-80/3
PROGRAMLISTA
commodore 64 jateKprogram <space snaKe> alKalmazasi utmutato gepiKodu program;elso resz ->coho/86.02-38/2
PROGRAMLISTA
commodore 64 KarakTermeret valtoztat

PROGRAMLISTA
nyomtato(mps 801)▮KarakterKep finomitas ->64er/86.02-75/2

PROGRAMLISTA
nyomtato(mps 801;802;803)▮KarakterKep modositas ->64er/86.02-100/6

PROGRAMLISTA
programozasi fogasoK▮commodore 16;116▮rovidprogram-gyujtemeny ->hc/86.02-86/1

PROGRAMLISTA
programozasi fogasoK▮commodore 64▮rovidprogram-gyujtemeny ->hc/86.02-84/2

PROGRAMLISTA
programozasi fogasoK▮sinclair▮rovidprogram-gyujtemeny ->hc/86.02-88/1

PROGRAMLISTA
programozasi fogasoK▮uzemeltetesi tanacsoK▮atari▮rovidprogram-gyujtemeny ->hc/86.02-90/2

PROGRAMLISTA
run▮beviteli utmutato▮commodore 64;128 ->run/86.02-126/2

PROGRAMLISTA
sinclair spectrum▮jateKprogram▮(alien) ->ysin/86.02-48/4

PROGRAMLISTA
sinclair spectrum▮jateKprogram▮(hot shot) ->ysin/86.02-44/2

PROGRAMLISTA
sinclair spectrum▮muveletgyorsitas▮(quicKy)▮Kazetta ->happ/86.02-89/2

PROGRAMLISTA
sinclair zx-81▮gepiKodu programoK betoltese ->chip/86.02-130/3

PROGRAMLISTA
sinclair▮oKtatas(autovezetes)▮utKozes szimulacio(feKtavolsag/sebesseg) ->hc/86.02-72/3

PROGRAMLISTA
struKturalt programozas▮basic programozas▮ciKKsorozat▮begepelo/maszKolorutin ->hc/86.02-92/2

PROGRAMLISTA
tavadatatvitel(telefonon)▮terminal uzemmod▮commodore 64▮(transbit)▮rs 232 vezerles▮soros fileoK Kezelese ->happ/86.02-52/5

PROGRAMLISTA
turbo-pascal▮fileKonvertalas▮dbase/turbo-pascal atteres ->pers/86.03-54/2

PROGRAMLISTA
turbo-pascal▮menubeepites▮gepfuggetlen program ->pers/86.03-48/2

PROGRAMLISTA
zene▮commodore 64▮(easymusic)▮poKe-helyettesito utasitasKeszlet ->coho/86.02-41/1

PROGRAMLISTA
zene▮commodore 64▮zongorabillentyuzet szimulacioja▮hangKimenet bovites ->64er/86.02-80/1

PROGRAMLISTA
as ->cute/86.02-70/3

PROGRAMLISTA
commodore 64▮lemezegyseg(1541)▮muveletgyorsitas▮fuzerbetoltes ->cute/86.02-64/3

PROGRAMLISTA
commodore 64▮lemezegyseg(1541)▮programvisszaallitas ->64er/86.02-81/2

PROGRAMLISTA
commodore 64▮lemeztar Kezeles ->run/86.02-74/2

PROGRAMLISTA
commodore 64▮muveletgyorsitas▮fuzervaltozo tarolas▮(garbage 64) ->64er/86.02-53/4

PROGRAMLISTA
commodore 64▮nyomtato(mps 801)▮grafiKus output ->64er/86.02-59/5

PROGRAMLISTA
commodore 64▮programozasi fogas(re-clear) ->64er/86.02-77/2

PROGRAMLISTA
commodore 64▮szovegfile formatalas▮(vizawrite)▮alKalmazasi tanacsoK ->64er/86.02-49/3

PROGRAMLISTA
commodore 64▮utasitasKeszlet bovites▮(worK-tool 64) ->hc/86.02-46/3

PROGRAMLISTA
dbase ii▮fileKezeles ->pers/86.04-48/4

PROGRAMLISTA
dbase ii/iii▮Keresoprogram(flexibilis) ->pers/86.03-49/2

PROGRAMLISTA
grafiKa▮pascal programozas▮turbo-pascal▮perspeKtiviKus abrazolas ->mc/86.02-36/5

PROGRAMLISTA
grafiKa▮pascal programozas▮turbo-pascal ->mc/86.02-78/4

PROGRAMLISTA
helyesbites(64er/85.11-85) ->64er/86.02-160/1

PROGRAMLISTA
helyesbites(64er/85.12-156) ->64er/86.02-160/1

PROGRAMLISTA
helyesbites(64er/85.12-82) ->64er/86.02-160/1

PROGRAMLISTA
helyesbites(mc/85.12-10) ->mc/86.02-14/1

PROGRAMLISTA
helyesbites▮(cute/85.12-108;86.01-110) ->cute/86.02-112/1

PROGRAMLISTA
matematiKai programozas▮mbasic programozas▮fuggvenybeado alprogram ->pers/86.04-42/3

PROGRAMLISTA
mbasic preprocesszor▮cimKezett programugrasoK ->chip/86.02-123/2

Házi módosítások a Gens3, Mons3 programokon

1. Ékezetes üzemmód a Gensben

Itt és a továbbiakban feltételezzük, hogy a Gens a 30000, a Mons az 59000 címre van töltve, más töltési címnél minden értelemszerűen eltolódik; továbbá a Gensben a Buffer size-ra Enterrel válaszoltunk.

Az ékezetes Gens lényege, hogy a kommentekben ékezetes betűket használunk: ha L kurzorunk van, akkor a symbol shifttel együtt lenyomott asdfgqwe2346ui billentyűkre rendre áöőüűÁÉéúíóÓÚf kódokat vesz be a program. Hiányzik még az ÖŐÜŰ, de ezek nélkül is elég magyarosan írhatunk.

A Gens wait-key rutinját kell módosítanunk. Először nézzük meg nagy vonalakban az eredeti működését (a rutin címe # 7615 = 30229):

1. lista

```
 20 #7615  PUSH IY
 30        LD   IY,#5C3A
 40        PUSH BC
 50        PUSH DE
 60        PUSH HL
 70 L761E  EXX
 80        PUSH AF
 85
 90 ;< az üzemmódtól függően C vagy L kurzor kiírása
100 ;inverz, villogó nyomtatásban, majd a normál nyomtatás
110 ;visszaállítása >
120
130 L7648  RES  5,(IY+1)    ;még nem volt billentyű - jelzés
140        CALL #02BF       ;billentyűzet leolvasása
150        BIT  5,(IY+1)    ;volt billentyűnyomás?
160        JR   NZ,L765F    ;ha igen, ugrás
170        LD   DE,#0A00    ;egyébként kis szünet jön
180 L7658  DEC  DE
190        LD   A,D
200        OR   E
210        JR   NZ,L7658
220        JR   L7648       ;majd újra a leolvasás következik.
225
230 L765F  CALL #7755       ;a kurzor törlése
240        LD   A,(#5C08)   ;a lenyomott billentyű kódja
250        CP   #06         ;CAPS lock
260        JR   Z,L767C     ;esetén ugrás
265
270        LD   B,#0B       ;11 párból álló táblázat van
280        LD   HL,L7689    ;ezen a címen. Ha valamely pár 1.
290 L766E  CP   (HL)        ;tagjával egyezik a bill.kód
300        INC  HL          ;(HL a 2. tagra mutat)
310        JR   Z,L7686     ;akkor ugrás a cseréhez
320        INC  HL
330        DJNZ L766E       ;(újabb pár vizsgálata)
335
340 L7675  POP  HL          ;egyébként vége a rutinnak.
350        POP  HL
360        POP  DE
370        POP  BC
380        POP  IY
390        RET
395
400 L767C  LD   HL,#5C6A    ;a MODE
410        LD   A,#08       ;3-as bitjét    (# CAPS LOCK *)
420        XOR  (HL)        ;átváltja
430        LD   (HL),A      ;beírja az új értéket
440        POP  AF
450        JR   L761E       ;vissza a wait-key-be
455
460 L7686  LD   A,(HL)      ;a kód felcserélése
470        JR   L7675       ;után visszatérés
480 L7689  DEFB #08,#18,#0C ;a 11 párból álló táblázat
490        DEFB #08,#0B,#0C
500        DEFB #07,#03,#C6 ;  ,    ,AND
510        DEFB "[",#C5,"]" ;[, OR, ]
520        DEFB #E2,"~",#C3 ;STOP, ~,NOT
530        DEFB "|",#CD,"\" ;|, STEP, \
540        DEFB #CC,"{",#CB ;TO, {, THEN
550        DEFB "}"         ;}
```

Ennek alapján a beavatkozás egyszerű: a kódcseréhez egy nagyobb táblázatot adunk meg a 29944 címen, ebben az ékezetes betűk számára szükséges cseréket is elhelyezzük. Ékezetes betűkre azonban csak L kurzor esetén váltunk át (a kommentekben úgyis kisbetűvel érdemes írni), így C kurzorral elérhetjük a ~/{|@#$& karaktereket is.

A 29944, illetve 29990 címre az alábbi táblázatokat kell elhelyezni (legcélszerűbb ezt előre megcsinálni egy kis BASIC programmal, és kimenteni):

```
cím       értékek vagy karakterek
29944     8          12
29946     12         8
29948     11         12
29950     7          3

29952     ' AND '    '['
29954     '<='       'Á'
29956     '>='       'é'
29958     '<>'       'É'
29960     'AT '      'í'
29962     'z'        'y'
29964     'y'        'z'
29966     'Z'        'Y'
29968     'Y'        'Z'
29970     ' OR '     'ú'
29972     ' STOP '   'ś'
29974     'NOT '     'ö'
29976     ' STEP '   'ő'
29978     ' TO '     'ü'
29980     ' THEN '   'ű'
29982     '#'        'í'
29984     '$'        'ó'
29986     '&'        'Ö'
29988     '@'        'ú'

29990     ']', '~', '!', '\', '{', '}'
```

1. táblázat

Látható, hogy felcseréltük a mi írógépeinknek megfelelően az y és z billentyűket; akinek ez nem tetszik, könnyen kiiktathatja a cserét úgy, hogy például a táblázatban az yz,YZ,zy,ZY párok helyett yy,YY,zz,ZZ párokat ír.

Ha már a wait-key rutin módosításánál tartunk, akkor érdemes a billentyűzetkattanást is megcsinálni, ami a Gensből szerintem nagyon hiányzik. A kattanást a rutinról való visszatérés előtt adjuk ki. Az 580-as sorban szereplő HL = 160 az eredetinél 2 egésszel magasabb hangot eredményez. Bárki az ízlése szerint más értékkel is kísérletezhet, a frekvencia ezzel fordított arányban változik.

A kiegészítő programot a 29900 címre tettük — lehet máshova is, a címek értelemszerű megváltoztatásával. A lista a következő:

```
 30 ;ÉKEZETES GENS  (kiegészítés és módosítás)
 40
 50        ORG  29000        ;=#74CC, a kiegészítő rutin
                              ==========================
 60
 70 ;a Gensből csak akkor kerül ide a vezérlés, ha a kódcsere
 80 ;táblázatban megtalálta az InPut karaktert.
 90 ;a B regiszter mutatja, hogy hátulról számolva hányadik
100 ;Pár szerint kell cserélni.
110 ;A=régi kód, (HL)=új kód
120
130        LD   C,A          ;a régi kód tárolása
140        LD   A,B
150        SUB  11           ;az utolsó 10 Pár kivételével
160        LD   A,(HL)
170        JR   NC,VISSZA    ;feltétlen csere
180
190        LD   HL,23658     ;ha a kurzor L,
200        BIT  3,(HL)
210        JR   Z,VISSZA     ;az utolsó 10 Párnál is OK.
```

```
220
230         LD   A,10          ;egyébként ebből a 10 párból
240         SUB  B
250         CP   6             ;az utolsó 4 elemnél (A=6..9)
260         JR   NC,VISSZ1     ;visszacsinálja a cserét:  #$&@
270
280         LD   HL,29990      ;ebből a táblázatból: ]^!\()
290         ADD  A,L           ;tovább cserél A=0..5 esetén
300         LD   L,A
310         LD   C,(HL)
320
330 VISSZ1  LD   A,C           ;a kód az A regiszterbe kerül
340 VISSZA  JP   #7676         ;vissza a Genshez
350
360
370         ORG  #B667         ;=46695
380
390 ;a Gens wait-key rutin egy részét módosítjuk.
    ;=================================
400 ;fordítás után áthelyezendő a #7667=30311 címre
410
420 L761E   EQU  #B61E
430
440         JR   Z,L768E       ;CAPS LOCK
450         LD   B,#17         ;23 Párból álló táblázat van
460         LD   HL,#74F8      ;ezen a címen.
470
480 L766E   CP   (HL)          ;keresés...
490         INC  HL
500         JP   Z,29900       ;találatnál ugrás a kieg. rutinra
510         INC  HL
520         DJNZ L766E
530
540         PUSH IX            ;IX és AF elmentendő, mielőtt a
550         PUSH AF
560         LD   D,#00
570         LD   E,(IY-1)
580         LD   HL,#00A0
590         CALL #03B5         ;kattanást kiadjuk.
600         POP  AF
610         POP  IX
620
630         POP  HL            ;vége a wait-key rutinnak.
640         POP  HL
650         POP  DE
660         POP  BC
670         POP  IY
680         RET
690
700 L768E   LD   HL,#5C6A      ;CAPS lock rutin: a MODE
710         LD   A,#08         ;3. bit (kisbetű/nagybetű)
720         XOR  (HL)          ;átváltása
730         LD   (HL),A
740         POP  AF            ;vissza új billentyű leolvasására!
750         JR   L761E
```

2. lista

Végül beszéljünk magukról az ékezetes betűkről! Ezek nyomtatási képét bármilyen címre tehetjük, és ha a 29944-en levő táblázatot BASIC programmal hozzuk létre, akkor még az is lényegtelen, hogy milyen betűt milyen kódhoz rendelünk, hiszen a táblázatba majd a megfelelő kód kerül be. Mivel azonban a keskenykarakteres ékezetes készletre ez már nem igaz (lásd a 2. pontot), ezért jobb, ha itt is ahhoz igazodunk, és az ékezetes betűket a következő sorrendben kódoljuk (144-től 161-ig):

ÁÉÍÓÖŐÚÜŰáéíóöőúüű

Akinek pedig nincs ékezetes betűkészlete, az alábbi táblázat alapján pótolhatja:

betű	cím	értékek
Á	65368	16, 112, 168, 136, 248, 136, 136, 0,
É	65376	16, 252, 160, 248, 128, 252, 0,
Í	65384	16, 32, 112, 32, 32, 32, 112, 0,
Ó	65392	16, 120, 164, 132, 132, 132, 120, 0,
Ö	65400	72, 120, 132, 132, 132, 132, 120, 0,
Ő	65408	40, 120, 212, 132, 132, 132, 120, 0,
Ú	65416	16, 168, 168, 136, 136, 136, 112, 0,
Ü	65424	80, 132, 132, 132, 132, 132, 120, 0,
Ű	65432	20, 40, 132, 132, 132, 132, 120, 0,
á	65440	16, 32, 112, 8, 120, 136, 120, 0,
é	65448	16, 32, 112, 136, 240, 128, 120, 0,
í	65456	16, 32, 0, 96, 32, 32, 112, 0,
ó	65464	16, 32, 112, 136, 136, 136, 112, 0,
ö	65472	80, 0, 112, 136, 136, 136, 112, 0,
ő	65480	40, 80, 112, 136, 136, 136, 112, 0,
ú	65488	16, 32, 0, 136, 136, 136, 112, 0,
ü	65496	80, 0, 0, 136, 136, 136, 112, 0,
ű	65504	40, 80, 0, 136, 136, 136, 112, 0,
³	65512	192, 32, 64, 160, 0, 0, 0, 0,
³	65520	192, 32, 192, 32, 192, 0, 0, 0,

2. táblázat

2. 64 kar/sor listázási lehetőség a képernyőn

A gépi kódú programozásnál célszerű, ha a programlistát az utasítások mellé írt magyarázatokkal egészítjük ki. Enélkül néhány hét múlva már saját programunkat sem értjük meg. Sajnos azonban a Gens3 ezt a technikát szinte lehetetlenné teszi, mert a 32 karakter/soros képernyő miatt a jobb oldali kommenteket folytatja a következő sorban a címkék és utasítások között, és így teljesen zavaros, áttekinthetetlen lesz a lista. (Aki nem hiszi, próbálja ki!) A megoldás: írjunk 64 karaktert egy sorba! Ehhez célszerűen az L parancsot kell megfejtenünk. Ezért jó, ha tudjuk, hogy a #8ea3 címen van a fő parancsok címtáblázata, amely a következő felépítésű: 19 × 3 bájtból áll, ebből az első a parancs nagybetűjele, utána van a parancs végrehajtási címe (lásd a 3. táblázatot). Megjegyezzük, hogy ugyanilyen táblázat található a #8eeb címen, ami viszont az E (szerkesztő) parancs alparancsait tartalmazza (lásd a 4. táblázatot).

A	32727	= #7FD7
B	30057	= #7569
C	32541	= #7F1D
D	31146	= #79AA
E	31257	= #7A19
F	31779	= #7C23
G	32246	= #7DF6
I	31104	= #7980
L	31662	= #7BAE
K	31740	= #7BFC
M	32355	= #7E63
N	32305	= #7E31
P	32005	= #7D05
R	31182	= #79CE
S	31243	= #7A0B
T	31881	= #7C89
V	35839	= #8BFF
W	31752	= #7C08
X	32657	= #7F91

3. táblázat

F	31655	= #7BA7
I	31547	= #7B3B
K	31471	= #7AEF
L	32620	= #7F6C
O	32653	= #7F8D
S	31569	= #7B51
R	31253	= #7A15
X	31542	= #7B36
Z	31552	= #7B40

4. táblázat

Most már nem okoz nehézséget az L és K parancsok visszafejtése. Elöljáróban megjegyezzük, hogy a Gens parancs általános alakja:

parancsjel n1, n2, s1, s2

ahol n1,n2 kétbájtos szám, s1,s2 pedig max. 20 karakteres sztringek (amit nem adunk meg, azt 0-nak, illetve üres sztringnek veszi a program). Amikor az egyes parancsrutinokra ugrunk, akkor már n1, n2, s1, s2 rendre a #8f49, #8f4b, #8f1b, #8f2f címre van lerakva. A legutolsó beadott parancs (editline) mindig a #90ad címen található. Az L parancs általános jelentése: listázás n1-től n2-ig.

```
30 ;a Gens L,K és W parancsok rutinjai
40 ;          visszafejtés
50 ;ha a Gens a 30000= #7530 címen van
60 ;általános forma:   betű,n1,n2,strin91,strin92
70
80 ;a belépéskor n1,n2 már rendre a #8F49, #8F4B címen van
90
100 ;a listázás n1-től n2-ig: L Parancs
110
120 L7BAE   LD   BC,(#8F49)  ;n1 -> BC
130         LD   A,B
140         OR   C           ;meg van adva n1?
150         JR   NZ,L7BB7    ;akkor jó
160         INC  C           ;egyébként legyen BC=1
170 L7BB7   CALL #7F35       ;a BC-edik sor megkeresése
180         RET  C           ;vissza, ha nincs ilyen sor!
190         SET  0,(IX+0)
200 L7BBF   LD   A,(#8F5B)   ;az egy adagban listázandó sorok
210         LD   (IX+1),A    ;számát a számlálóba teszi
220
230 ;egy sor kiírását végző részprg.
240
250 L7BC5   LD   C,(HL)      ;a HL a címmutató, a sorszámot
260         INC  HL
270         LD   B,(HL)      ;a BC-be tesszük
280         EX   DE,HL       ;a címmutatót DE-ben tároljuk
290         LD   HL,(#8F4B)  ;n2 -> HL (utolsó listázandó sor)
300         LD   A,H
310         OR   L           ;ha nincs megadva,
320         JR   NZ,L7BD3
330         LD   HL,#7FFF    ;akkor a max sorszámot veszi
340 L7BD3   SBC  HL,BC       ;ha ez a sor már nem listázandó
350         RET  C           ;akkor visszatérés._____
360
370         LD   H,B
380         LD   L,C         ;sorszám a HL-be megy
390         PUSH DE          ;a folyó cím tárolása
400         CALL #7F85       ;sorszám kiírása
410
420         POP  HL          ;a folyó cím
430         PUSH HL          ;(sorszám 2. byte)
440         INC  HL          ;szöveg 1. byte
450         LD   D,#06       ;a szöveg a 6. oszloptól kezdődik
460         CALL #7806       ;a szöveg kiírása
470
480         POP  HL          ;címmutató
490         DEC  HL          ;sorszám 1. byte
500         CALL #7F4F       ;HL<-köv.sor.cím, Z=1,ha vége a
```

```
510         RET  Z              ;listának, ekkor visszatér._____
    ;itt helyesebb lenne RET NC alkalmazása, ez megakadályozná
    ;a Gens elszállását, akkor is, ha valami folytán a lista
    ;végét jelző mutató nem sorvége utáni rekeszre mutat.
520
530         DEC  (IX+1)         ;kiírt sorok számlálása
540         LD   A,(IX+1)
550         DEC  A              ;ha még nem írt ki elég sort,
560         JP   P,L7BC5        ;akkor folytatja a listázást
570
580         CALL #76AD          ;billentyűt vár, kódját A-ba teszi
590         CP   #03            ;EDIT-et nyomtunk még?
600         RET  Z              ;akkor visszatérés._____
610         JR   L7BBF          ;újabb adag listázása következik.
620
630
640 ;a K[n] Parancs működése:
650
660         LD   A,(#8F49)      ;n mod 256 (alsó byte)
670         OR   A              ;még van adva?
680         JR   NZ,L7C04       ;ha igen, ugrás
690         LD   A,#0F          ;ha nincs megadva akkor legyen 15!
700 L7C04   LD   (#8F5B),A      ;ennyi sor lesz 1 adag listában.
710         RET
720
730
740 ;a W Parancs működése:
750
760         LD   A,#10          ;Paraméter a következő rutinhoz
770         CALL #755D          ;megnyitja a Printer csatornát, és
780         JR   L7BAE          ;az L Parancsnak megfelelően oda
790                             ;fogg listázni.
```

3. lista

A List rutint úgy módosítjuk, hogy a sorszámkiíró rutin hívásánál az eredeti #7f85 címet saját rutinunk címével helyettesítjük, ami NUMBER= #el00. Ez a rutin először is megnézi, hogy az L után nem 0 áll-e. Ha igen, akkor áttér a normális 32 karakteres kiírásra, tehát például
L0100 hatására hagyományos,
L100 hatására keskenykarakteres lista jön a 100-as sortól.

Ha az L után nem 0 áll, akkor a sorszámot keskeny karakterekkel kiírja, majd visszatér a List rutinba. Ugyanígy járunk el a szöveget kiíratónál is, ahol az eredeti CALL #7806 helyett CALL #el96-ra módosítunk. Ez a LIST64 rutin címe, amely szintén kezeli az L0... esetet. A kiegészítő rutin listája:

```
 30 ;86.09.11
 40 ;64karakteres Gens listához rutinok
 50
 60         ORG  #E100
 70
 80 ;a sorszámkiíró rutin
 90
100 NUMBER LD   A,(#90AE)      ;a Parancssor 2. karaktere
110         CP   48             ;L0... alak volt megadva?
120         JP   Z,#7F85        ;akkor normál listázás
130
140         PUSH HL
150         LD   HL,20640       ;a sor kiírása képernyő 21. sorába
160         LD   (CIM),HL       ;történik és
170         LD   A,65           ;64 karakter sorhossz lehet
180         LD   (POS),A
190         POP  HL
200
210         LD   DE,-10000      ;a sorszám kiírása a tízezres
220         CALL DIGIT          ; helyiértéktől kezdve
230         LD   DE,-1000
240         CALL DIGIT
250         LD   DE,-100
260         CALL DIGIT
270         LD   DE,-10
280         CALL DIGIT
290         LD   DE,-1
300         CALL DIGIT
310
320         LD   A,32
330         CALL PR4B           ;1 helyköz a szám után
340         XOR  A
350         LD   (TAB),A        ;a tabulátorPozíció nullázása
360         RET
370
380 ;egy számjegy kiírása rutin
390 DIGIT   XOR  A              ;nullától indul a számláló
400         ADD  HL,DE          ;kivonás
410         INC  A              ;számlálás, hányszor vontuk ki
420         JR   C,DIGIT+1      ;ha még nem fogyott el,visszaugrik
430         SBC  HL,DE          ;egyébként utolsó kivonást törli
440         ADD  A,47           ;a keresett számjegy kódja
450
460
470 ;A 4bites Print rutin (ascii és felh.graf. karakterek)
480
490 PR4B    CP   32             ;vezérlő kódoknál
500         RET  C              ;visszatérés
510         CP   128            ;lehet nyomni az ascii karaktert
520         JR   C,ASCII
530         CP   144            ;standard graf.nál vissza
540         RET  C
550         CP   165            ;kulcsszavakat sem fogad el
560         RET  NC
570
580         PUSH DE             ;felh.graf.karekterek esetén
590         LD   DE,UDG4-1152   ;ez a báziscím,
600         JR   KOZOS
610
620 ASCII   PUSH DE             ;a normál keskenykarakterek
630         LD   DE,CHARS4-256  ;esetén pedig ez.
640 KOZOS   PUSH BC
650         PUSH HL
660         LD   H,0
670         LD   L,A
680         ADD  HL,HL
690         ADD  HL,HL
700         ADD  HL,HL
710         ADD  HL,DE          ;HL=kar. nyomt.kép címe
720
730         LD   A,(POS)
740         DEC  A              ;kiírási Pozíció lép, ha van hely
750         LD   (POS),A
760         JR   Z,VEGE
770
780         EX   DE,HL          ;DE=kar.kép címe
790         LD   HL,(CIM)       ;felső képelem címe
800         PUSH HL
810
820         LD   B,8            ;8 képelem lesz
830         BIT  0,A            ;ha A páratlan, akkor egy 8bites
840         JR   NZ,HATUL       ;kép hátsó felébe kell írni
850
860 ELOL    LD   (HL),0         ;eddigi kép törlése
870         LD   A,(DE)         ;beírás hátulra
880         RRD                 ;(HL) elejébe megy.
890         INC  DE             ;a kar.képelem és a
900         INC  H              ;képelem frissítése
910         DJNZ ELOL           ;mind a 8 elemet!
920         POP  HL             ;a felső képelem címét lehívjuk
930         JR   CIMLEP         ;és nem változtatjuk meg.
940
950 HATUL   LD   A,(DE)         ;kar.képelem ->A
960         OR   (HL)           ;a hátsó 4 bit beírása a képelembe
970         LD   (HL),A
980         INC  DE             ;kar.képelem és
990         INC  H              ;képelem frisítése
1000        DJNZ HATUL          ;mind a 8 elemet!
1010        POP  HL             ;a felső képelem címe
1020        INC  HL             ;most nő.
1030
1040 CIMLEP LD   (CIM),HL       ;a felső képelem cím beírása
1050 VEGE   POP  HL
1060        POP  BC
1070        POP  DE
1080        RET                 ;a karakter kinyomtatva.
1090
1100
1110
1120 ;a szöveg kiírása
1130
1140 LIST64 LD   A,(#90AE)      ;a Parancssor 2.karakter
1150        CP   48             ;'0' ?
1160        JP   Z,#7806        ;akkor normál kiírás
1170
1180 LIST   LD   A,(HL)         ;aktuális karakter
1190        CP   59             ;Pontosvessző esetén esetleg
1200        JR   Z,NTABU        ;tabulálás szükséges.
1210        CP   9              ;tabulátor-karakter?
1220        JR   Z,NTABU
1230        CP   13             ;Enter esetén visszatérés
1240        JP   Z,#DFE         ;a scroll rutinon keresztül.
1250 CHAR   CALL PR4B           ;egyébként a karakter kiírható
1260 STEP   INC  HL             ;listázási cím növelése
1270        JR   LIST           ;és a listázás folytatása
1280
1290 NTABU  LD   A,(TAB)
1300        INC  A              ;a tabulátor ugratása
1310        LD   (TAB),A
1320
1330        LD   D,#E6          ;DE megcímezi a tabulátor
1340        LD   E,A            ;PozíciójÁnak értékét.
1350
1360        ADD  A,(HL)         ;tabu.Poz.száma+aktuális karakter
1370        DEC  A
1380        CP   59             ;Pontosv. és 1. Pozíció
1390        JR   Z,CHAR         ;esetén visszatérés kiírásra
1400
1410 SPACES LD   A,(POS)        ;az aktuális Pozíció
1420        LD   B,A            ;(tárolás B-ben)
1430        LD   A,32           ;után szÉszeket ír
1440        CALL PR4B           ;(legalább egyet!)
1450        LD   A,(DE)         ;amíg a tabulátorPozíciót
1460        ADD  A,B
1470        CP   67
1480        JR   NC,SPACES      ;el nem értük.
1490
1500        LD   A,(HL)         ;aktuális kar.
1510        CP   59             ;Pontosvessző esetén
1520        JR   Z,CHAR         ;kinyomtatásra, egyébként a kar.
1530        JR   STEP           ;átugrására visszatérés
1540
1550
1560        ORG  #E200
1570 CHARS4 DEFS 768            ;a keskeny karakterkészlet
1580 UDG4   DEFS 168            ;keskeny felh.graf.karakterek
1590 POS    DEFS 1              ;kiírási Pozíció
1600 CIM    DEFS 2              ;és ennek címe a képernyőn
1610 TAB    DEFS 1              ;a tab. tábl. tételszáma
1700
1710        ORG  #E600          ;a tabulátor táblázat
1730        DEFB 6,13,18,30     ;a tabulátor állásai
```

4. lista

Amint a listából látható, a 64 karakteres listázásba még egy funkciót bevettünk: a kommentárok tabulálását. Ha a pontosvesszső nem a sor elején van, akkor a rutin tabulátornak tekinti. Tehát ahhoz, hogy a megjegyzések szép sorban legyenek egymás alatt, elegendő, ha az utasítás operandusa után egy pontosvesszőt teszünk, még helyköz sem kell. Ezzel időt és memóriát takarítunk meg.

A végrehajtás egy kis haditervet igényel:

a) A keskeny karakterek nyomtatási képeinek 768 bájtos blokkját a tasword2-ből vehetjük, a 61184 címtől. Néhány bájtot a save előtt érdemes megváltoztatni, az eredeti karaktereknél szerencsésebb alakokat kapunk így: például jól megkülönböztethető a B,8,Ø, illetve az m és w, a W és Y (lásd az 5. táblázatot).

b) A felhasznált grafikus karakterek keskeny változatát a 6. táblázat alapján elkészítjük és felvesszük.

c) Betöltjük a Genset 30000-re (ékezeteset 29900-ra), a Monsot 59000-re és a 4. listát Genssel beírjuk. Ezután 5-ös opcióval fordítunk, és okvetlenül ellenőrizzük legalább a NUMBER és a LIST64 címeket (#elØØ és #el96).

d) Kilépünk a Gensből, belépünk a Monsba és módosítjuk az alábbi két hívást:

#7BD9 CALL #7F58 ; helyette #ElØØ=NUMBER

#7BE1 CALL #78Ø6 ; helyette #El96=LIST64

e) Betöltjük a keskeny karaktereket 57856-ra, a grafikusakat 58624-re.

f) Ezzel készen vagyunk, most már listázhatunk a Gensben. A kész programot pedig a Monssal együtt felvesszük.

Végül a szükséges táblázatok következnek.

karakter	cím	értékek
Ø	57984	Ø,3,5,5,7,5,6,Ø,
8	58048	Ø,2,5,2,5,5,2,Ø,
N	58224	Ø,6,5,5,5,5,5,Ø,
W	58296	Ø,5,5,7,7,7,2,Ø,
m	58472	Ø,Ø,7,7,7,5,5,Ø,

5. táblázat

karakter	cím	értékek
á	58624	1,2,5,5,7,5,5,Ø,
É	58632	2,7,4,6,4,4,7,Ø,
í	58640	1,2,7,2,2,2,7,Ø,
Ó	58648	1,2,5,5,5,5,2,Ø,
Ö	58656	5,Ø,7,5,5,5,7,Ø,
ő	58664	7,Ø,7,5,5,5,7,Ø,
Ú	58672	2,7,7,5,5,5,7,Ø,
Ü	58680	5,Ø,5,5,5,5,7,Ø,
ű	58688	7,Ø,5,5,5,5,7,Ø,
á	58696	1,2,6,1,7,5,7,Ø,
é	58704	1,2,2,5,6,4,3,Ø,
í	58712	1,2,Ø,6,2,2,7,Ø,
ó	58720	Ø,1,2,5,5,5,2,Ø,
ö	58728	Ø,5,Ø,2,5,5,2,Ø,
ő	58736	5,5,Ø,2,5,5,2,Ø,
ú	58744	Ø,1,2,Ø,5,5,7,Ø,
ü	58752	Ø,Ø,5,Ø,5,5,7,Ø,
ű	58760	Ø,5,5,Ø,5,5,7,Ø,

6. táblázat

3. A Monsban disassemblált szöveg átalakítása Gens formátumra

A Mons a T paranccsal visszafordított listát állít elő valamely memóriablokkról. A hiba csak az, hogy míg a tabulátorugrásokat a Gens 9-es kódokkal jelzi, a Mons ezt szóközökkel teszi. Szerencsére máskor a szövegben nem is fordul elő szóköz, így nincs más dolgunk, mint a 32-es kódokat 9-esre átváltani. A sorszámokat természetesen közben át kell ugrani.

Az átalakító rutin használata a következő. Először a Gensben megtudjuk X paranccsal a végcímet. A Monsban a text kérdésre a végcímet (hex!) adjuk be. Kilépve a Monsból, PRINT AT 21,Ø; USR 5896Ø és kész!

```
 90 ;3$!!
 95
100 ;TEXT TO GENS
110 ;a visszafordított listában a szPészeket
120 ;chr$ 9 -re cseréli
130
140         ORG  58960
150
160 GENS    EQU  30000
170 MONS    EQU  59000
175
180 ;a MONS a GENS lista mö9é tette a szöve9et!
185
190         LD   HL,(GENS+70) ;Gens lista vé9e
200         LD   DE,(MONS+4895) ;Mons lista vé9e
205
210 NUMBER  INC  HL           ;az aktuális cím léPtetése
220 CHAR    INC  HL
230         LD   A,(HL)       ;az aktuális karakter
240         CP   32           ;szPész?
250         JR   Z,TABULA     ;akkor azt tabulátorként kezeli.
260         CP   13           ;ha mé9 nincs vé9e a sornak,
270         JR   NZ,CHAR      ;akkor folytatjuk.
275
280         INC  HL           ;Sor vé9e: átlépjük az Entert.
290         PUSH HL
300         SBC  HL,DE        ;vé9e a listának?
310         POP  HL
320         JR   C,NUMBER     ;ha nem, tovább dol9ozunk.
325
330         LD   (GENS+70),HL ;az e9yesített lista vé9címe
340         JP   GENS+7Z      ;mele9 starttal a Gensbe!
345
370 TABULA  LD   (HL),9       ;a szPészt tabulátorra cseréli
380         JR   CHAR         ;folytatja a sort.
```

5. lista

4. A nyomtatóra egy sort küldő rutin

Mivel az eredeti printerfunkciókkal ugyanazok a problémák, mint a képernyőre listázásnál (lásd a 2. pontot), inkább saját magunk fogunk nyomtatni. A most következő gépi kódú rutin egy sort ír ki, ezt majd a BASIC főprogramból fogjuk használni (5. pont). Az indítási cím 58885, és az ITT+1=58891 címen mindig az aktuális sor kezdőcíme van.

```
 20 ;GENS3 (L)LIST
 30
 40         ORG  58885
 50
 60
 70         LD   A,3          ;Az LPrint csatornát me9nyitjuk
 80         CALL #1601
 85
 90 ITT     LD   HL,38278     ;A sor va9y lista kezdete.
100         LD   BC,VISSZA    ;ez lesz a visszatérési cím
110         PUSH BC           ;a ROM rutinból
120
130         PUSH DE
140         LD   E,(HL)
150         INC  HL
160         LD   D,(HL)
170         INC  HL
180         PUSH HL           ;szöve9 kezdőcím raktározása
190         EX   DE,HL        ;HL=sorszám.
200
210         LD   E,32         ;bevezető szPészek me9en9edettek
220         LD   BC,-10000
230         CALL #192A        ;először a tízezres helyiértéket
240         JP   #1A30        ;azután a többit íria ROM rutinnal
250
260 VISSZA  XOR  A            ;a sorszám nyomtatás után nullázza
270         LD   C,A
280         JR   TABU         ;tabulátort.
290
300
310         ORG  58823        ;a sor karaktereit vizs9álja
320
330 LIST    LD   A,(HL)       ;aktuális karakter
340         INC  HL           ;címnövelés
350         LD   C,A          ;az aktuális kód raktározása
360         CP   59           ;commentet a jobboldalra!
370         JR   Z,NTABU
380         CP   9            ;tabulátor karakter?
390         JR   Z,NTABU
400         CP   13           ;vé9e a sornak?
410         JR   Z,VEGE
420 CHAR    RST  16           ;e9yébként nyomtatja a karaktert
430         JR   LIST         ;és folytatja a sort.
440
450 VEGE    LD   (ITT+1),HL   ;a következő sor elejét állítja be
460         RET               ;visszatérés  előtt_________
470
480 NTABU   LD   A,(TAB)      ;a tabulátor leolvasása
490         INC  A            ;léPtetése
500 TABU    LD   (TAB),A      ;beírása.
510         LD   D,#E6        ;(DE) fo9ja mutatni, hoy hányadik
520         LD   E,A          ;oszloPba kell nyomtatni.
530
540         ADD  A,C          ;a tabulátor állás + aktuális kód
550         DEC  A
560         CP   59           ;ha a Pontosv. 1.Pozícióban van
570         JR   Z,CHAR       ;akkor minden marad
580
590 SPACES  LD   A,32         ;le9alább 1 szPész kinyomtatása
600         RST  16
610         LD   A,(DE)       ;a tabulátor kívánta Pozíciót
620         LD   B,A
625         LD   A,(23680)    ;a nyomtatási Pozícióból
640         SUB  B            ;kivonjuk, ha mé9 a tabu.Poz.
```

```
650         JR   C,SPACES     ;nagyobb, akkor újabb szpészt!
660
670         LD   A,C          ;ha az aktuális kár. nem chr$ 9,
680         CP   59           ;hanem pontosvessző, akkor kiírás
690         JR   Z,CHAR
700         JR   LIST         ;különben átugrás következik.
710
720 ;ellenőrizendő, hogy JEL <= #e600
730 JEL
740
750         ORG  #E600
760         DEFB 6,13,18,30
770 TAB
```

6. lista

5. BASIC főprogram a módosított Gens-Mons használatára

Az eddigi módosítások alapján van egy ékezetes Gensünk, továbbá egy megnövelt Monsunk, amihez ui. hozzácsaptuk a 2—4. pontok rutinjait, és így az ékezetes betűkkel együtt egy CODE 57600,7936 felvételt készítettünk. Mindezek működését a 7. lista BASIC programja a következőképpen vezérli.

10-es sor. Start, kissé gyorsított billentyűzet beállítása. A w(c) függvény kétbájtos peeket valósít meg. Ezután a gépi kódú részek beolvasása következik.

100-as sor. A menü kiírása. A válasz egy betű és egy számjegy, ez utóbbi = 1, ha nincs megadva; a számjegy a használandó microdrive száma. A tabulátorpozíciókat mindig visszaállítjuk, mert nyomtatáskor a 934-es sor elállítja. Ezután a megfelelő funkcióra ugrunk.

700-as sor. Ez a részlet a Gens listáját a kijelölt microdrive-ra viszi, és ellenőrzi a felvételt. Először a lista start- és végcímét (ez a lista vége utáni első szabad rekeszre mutat) olvassa le, majd ellenőrzi, hogy a lista első sorában kijelöltük-e a hosszmutató bájtokat. Ennek szabálya: a lista első sorát ";"-vel kezdjük, ezt két tetszőleges karakter követi, majd egy "!"-nek kell állnia, azután akármi lehet. A lista hosszát 200-as számrendszerben, fordított sorrendben írjuk fel (a maximális hossz 39999 bájt), majd a két „számjegyet" 32-vel megnövelve elhelyezzük a ";" és a "!" közé. Innen tudjuk majd a LOAD-nál a lista hosszát. A 32-es eltolás pedig azért kell, hogy vezérlő karakter ne kerülhessen a listába (717-es sor). Ha a hosszmutató nincs előkészítve, hibajelzés jön (716-os sor). Ha rendben van, akkor felvétel a kijelölt drive-ra, verify, majd visszatérés a menühöz.

800-as sor. Beolvasás a kijelölt drive-ról. A program megkérdezi a felvétel nevét, a lista hosszát — így lehetőség van olyan program beolvasására is, amely nem tartalmaz hosszmutatót. Ha a hossz helyett enterrel válaszolunk, akkor feltételezi, hogy van hosszmutató a beolvasandó listában. Beolvasás után, ha nem adtuk meg a hosszt és nincs hosszmutató sem, akkor hibaüzenetet kapunk, egyébként kiszámítja az LV végcímet. Ha a tárban már van egy lista, akkor a beolvasott rész ennek folytatása lesz, az újraszámozást azonban nekünk kell elvégeznünk a Gensben. Előfordulhat, hogy a hossz hibásan van megadva. Ez ellen védekezik a 845-ös sor: addig csökkenti a végcímet, amíg fenn nem áll, hogy a lista vége egy sor végével egyezik meg. Ellenkező esetben a Gens ugyanis elszáll. A Gens rendszerváltozóba tölti a lista végcímét, majd visszatér a menühöz.

900-as sor. Listázás a nyomtatóra. Feltételezzük, hogy a nyomtató legalább 64 karakter széles. Igazán akkor jön ki jól az ékezetes nyomtatás, ha olyan a nyomtató és a csatlakoztató Epromja, hogy a felhasznált grafikus krakterek nyomtatási képének bal felső 6 × 7-es részét nyomtatja ki (például Seikosha GP 100 és Microteam interfész).

Először a lista kezdőcímét adjuk át a gépi kódú programnak (lásd a 4. részt), majd a végcímet olvassuk le. A 931—934-es sorban lehetőségünk van bal oldalon margót hagyni: annyi szóközt gépelünk be. Ekkor módosul a tabulátortáblázat, amit a gépi kódú rutinok használnak (934-es sor).

Ha a nyomtató nem emel automatikusan lapot, akkor nekünk kell ezt megoldanunk: erre szolgál a 935-ös sor. Ilyenkor a listázást a lap tetején kezdjük el! Ide fog kerülni a fejlécként megadott szöveg és a lapszám, egy sor kimarad, 68 sor lista következik, majd a lap alján két sor újra kimarad (72 sor/lap). Ha a lapemelés automatikus, akkor a fejlécre enterrel válaszoljunk!

Egy sor kinyomtatását végzi a 940-es sor. A nyomtatási sorokat számoljuk a 995-ös sorban, és ha már a 71. sorba írnánk, akkor új lap, fejléc következik (996-os sor). Ha nincs több listasor, akkor a menü jön.

1500-as sor. Egy memóriablokkot felveszünk a kijelölt microdrive-ra és ellenőrizzük. Utána visszatérés a menühöz.

5000-es sor. A BASIC főprogram, a módosított Gens és Mons felvétele "m";1;-re.

SLIZ MIKLÓS

7. lista

```
  9 REM ----- START -----
 10 POKE 23609,20: POKE 23561,15: POKE 23562,4
 20 DEF FN w(c)=PEEK c+256*PEEK (c+1)
 30 CLEAR 29899
 40 LET gens=30000:          LET mons=59000
 50 LOAD *"m";1;"964/299"CODE
 60 LOAD *"m";1;"m64/576"CODE
 90 CLS : PRINT AT 5,0;" ÉKEZETES GENSMONS3, 6.2 verzió"'"
64 karakter/sor"''" © PrigoSoft , 1986.10.09."'" Sliz Miklós"
 97
 98 REM ----- menü -------
 99
100 POKE 23658,8: INPUT #1;AT 0,0;"PRINTER  CODE-SAVE  GENS   M
ONS  SAVE*  LOAD*  Text TO GENS     ENTER = STOP ";a$
101 IF A$="" THEN  STOP
102 LET n=1: IF LEN a$=2 THEN  LET n=VAL a$(2)
103 RESTORE 933: FOR i=58880 TO 58883: READ a: POKE i,a: NEXT i
105 IF a$(1)="S" THEN  GO TO 700
106 IF A$(1)="L" THEN  GO TO 800
110 IF a$="G" THEN  PRINT AT 21,0;: RANDOMIZE USR gens
115 IF A$="P" THEN  GO TO 900
116 IF A$(1)="C" THEN  GO TO 1500
120 IF a$="M" THEN  RANDOMIZE USR mons
130 IF a$="T" THEN  PRINT AT 21,0;: RANDOMIZE USR 58960
230 GO TO 100
697
698 REM ----- save* -------
699
700 LET L0=FN W(GENS+6695)
710 LET LV=FN W(GENS+70)
715 IF PEEK (L0+2)=CODE ";" AND PEEK (L0+5)=CODE "!" THEN  GO T
O 717
716 PRINT "hosszmutató?": GO TO 100
717 LET hossz=lv-l0: POKE l0+4,INT (hossz/200)+32: POKE l0+3,ho
sz-200*INT (HOSSZ/200)+32
720 PRINT " SAVE *'M';";n;";";
725 INPUT A$: PRINT A$;"'CODE ";L0;",";LV-L0
730 SAVE *"M";n;A$CODE L0,LV-L0
735 VERIFY *"M";n;A$CODE L0,LV-L0
740 GO TO 100
797
798 REM ------ load* ------
799
800 INPUT " LOAD *'M';";STR$ n;";";A$
805 LET L0=FN W(GENS+70)
810 INPUT "CODE ";STR$ L0;",";H$
830 LOAD *"M";n;A$CODE L0
833 IF H$="" AND PEEK (L0+5)<>CODE "!" THEN  GO TO 716
835 IF h$="" THEN  LET lv=l0+PEEK (l0+3)-32+200*(PEEK (L0+4)-32
): GO TO 845
840 LET LV=L0+VAL H$
845 IF PEEK (lv-3)=13 OR PEEK (lv-2)=13 OR PEEK (lv-1)<>13 THEN
 LET lv=lv-1: GO TO 845
850 POKE GENS+71,INT (LV/256)
860 POKE GENS+70,LV-256*PEEK (GENS+71)
870 GO TO 100
900
901 REM ------ Printer ------
902
920 POKE 58891,PEEK (GENS+6695)
921 POKE 58892,PEEK (GENS+6696)
930 LET LV=FN W(GENS+70)
931 INPUT "balmar96";a$: LET x=LEN a$
932 RESTORE 933
933 DATA 6,13,18,30
934 FOR i=58880 TO 58883: READ a: POKE i,a+x: NEXT i
935 LET SOR=-1E8: INPUT "fejléc ? enter=semmi ";N$: LET LAP=1:
IF n$>"" THEN  LET SOR=3: GO TO 996
940 LPRINT a$;: RANDOMIZE USR 58885: LPRINT
995 LET SOR=SOR+1: IF SOR=71 THEN  LET SOR=3: LET LAP=LAP+1: LP
RINT : LPRINT
996 IF SOR=3 THEN  LPRINT A$;n$;" - ";LAP: LPRINT
997 IF FN W(58891)>=LV-1 THEN  LPRINT : GO TO 100
999 GO TO 940
1500
1501  REM ------ CODE  SAVE ------
1502
1510 POKE 23658,0
1520 INPUT " SAVE  CODE "/"from"," TO ",C1,C2
1525 LET l=c2-c1+1
1540 INPUT "name ";a$: PRINT " SAVE *'M';";N;";";a$;"'CODE ";c1
;",";l
1550 SAVE *"m";N;a$CODE c1,l
1555 VERIFY *"m";N;a$CODE c1,l
1590 GO TO 100
4990
4991 REM ------ szaporítás ------
4995
5000 CLEAR : SAVE *"m";1;"run" LINE 10
5010 VERIFY *"m";1;"run" LINE 10
5020 SAVE *"m";1;"m64/576"CODE 57600,7936
5025 VERIFY *"m";1;"m64/576"CODE 57600,7936
5030 SAVE *"m";1;"964/299"CODE 29900,8380
5035 VERIFY *"m";1;"964/299"CODE 29900,8380
```

Kalkulációs program adóelszámoláshoz

ZX-SPECTRUM

Az egész világon terjednek a típusprogramok: kalkulációs, szövegszerkesztő, adatkezelő stb. szoftverek. Ezekkel a gyakorlati életben felmerülő problémák jelentős részét meg lehet oldani. Ily módon egyszerűen és rövid idő alatt, programnyelvek ismerete nélkül is rendkívül hasznos programok állíthatók össze. Hogy ez mennyire igaz, azt egy példával szeretném bizonyítani.

A feladat egy gazdasági munkaközösségben a tagok személyi jövedelmének kiszámítása és nyilvántartása, beleértve a TBI-járulékot és a jövedelemadót. Ezt végzi el az a program, amely a ZX—Spectrum számítógépen futó VU—CALC kalkulációs segédprogrammal együtt működik, de az elve minden más gépen használatos kalkulációs programra adaptálható. A VU—CALC segédprogramot a Mikroszámítógép Magazin az 1984/2. számában részletesen ismertette. Jelen leírás épít a VU—CALC-ban szerzett gyakorlatra. Maga a programozási munka csak a celláknak (vagy boxoknak) az *1. lista* szerinti feltöltéséből áll. E listán a cellák azonosítóiban a numerikus karakter jelzi a VU—CALC megfelelő oszlopát, a betű pedig a sorát.

A program által készíthető nyilvántartás résztáblázatokból épül fel. Ezek méretét a VU—CALC-beli 18 × 4 cella (box) határozza meg, amit a számítógépes feldolgozás szempontjából ablaknak nevezünk, valamint a sor- és oszlopkoordináták. Így az egyes résztáblázatok tartalma éppen megfelel a képernyőn látható képek és a kinyomtatás méretének. A tag nevén kívül a résztáblázatok személyenként tartalmazzák a nyereségelőlegnek az aktuális (tárgyhavi) számítás/kifizetés időpontjáig járó részét, az ebből korábban felvett összeget, továbbá a tárgyhavi elszámolást, részletezve a levonásokkal:

— költséghányad,
— TBI-járulék,
— jövedelemadó.

A résztáblázatok felépítése

1. *ablak 1—4.oszlop = cím és megnevezés*
   Ez a résztáblázat tetszőlegesen kitölthető. Például: Mikrobi GM. Budapest. Adóelszámolás
2. *ablak 5—8. oszlop = számítások elvégzése*
   Minden egyes tag jövedelmének számítá-

Megjegyzések a programlistához

1. Az 5-ös oszlopban csak szöveges kifejezések szerepelnek.
2. A 6-os oszlopban csak matematikai kifejezések szerepelnek.
3. A 7-es oszlopban a B, F, O, R sorba Ø-t kell írni, a többi üresen marad.
4. A 8-as oszlopban is csak a B, F, O, R sorban szerepel kifejezés, a többi üresen marad.
5. A program 200 000 Ft adóalapig jó. E fölött további sorok beszúrása szükséges.
6. A B6 cellába tetszőleges érték beírható.

1. lista

```
                Jovedelem nyilvántartás
                   Programlista

A5:NEV      A6:          A7:          A8:
B5:keresetB6:10000   B7:0         B8:B6-B7
C5:-10%ktsC6:B6*.1   C7:          C8:
D5:--------D6:--------D7:--------D8:--------
E5:maradekE6:B6-C6   E7:          E8:
F5:-  TBI F6:E6*.1   F7:0         F8:F6-F7
G5:--------G6:--------G7:--------G8:--------
H5:AdoalapH6:E6-F6   H7:          H8:
I5:     2% I6:H6*.02 I7:          I8:
J5:     6% J6:(H6-20000)*(H6>20000)*.04
K5:    10% K6:(H6-40000)*(H6>40000)*.04
L5:    20% L6:(H6-60000)*(H6>60000)*.1
M5:    30% M6:(H6-80000)*(H6>80000)*.1
N5:    45% N6:(H6-100000)*(H6>100000)*.15
O5:-adoosz06:&I6:N6  07:0         08:O6-O7
P5:+10%ktsP6:C6      P7:          P8:
Q5:--------Q6:--------Q7:--------Q8:--------
R5:kifizetR6:H6-O6+C6 :0         R8:R6-R7
```

2. lista

	05	06	07	08
A	BARNA I			
B	kereset	129000	63000	66000
C	-10%kts	12900	6300	
D				
E	maradek	116100	56700	
F	- TBI	11610	5670	5940
G				
H	Adoalap	104490	51030	
I	2%	2089	1020	
J	6%	3379	1241	
K	10%	2579	441	
L	20%	4449	0	
M	30%	2449	0	
N	45%	673	0	
O	-adoosz	15618	2702	12916
P	+10%kts	12900	6300	
Q				
R	Kifizet	101772	54628	47144

3. lista

	13	14	15	16
A	SZALAI			
B	kereset	26900	5400	21500
C	-10%kts	2690	540	
D				
E	maradek	24210	4860	
F	- TBI	2421	486	1935
G				
H	Adoalap	21789	4374	
I	2%	435	87	
J	6%	71	0	
K	10%	0	0	
L	20%	0	0	
M	30%	0	0	
N	45%	0	0	
O	-adoosz	506	87	419
P	+10%kts	2690	540	
Q				
R	Kifizet	23973	4827	19146

sa itt zajlik, az 1. lista szerinti programmal. A képernyőn csak az aktuális számítás eredménye jelenik meg *(2. lista)*. A kapott eredményeket minden egyes futtatás után el kell helyezni a nyilvántartás megfelelő résztáblázatában, vagyis a tag sorszáma szerinti ablakba át kell vinni.

3. ablak 9—12. oszlop= 1. tag adatai

4. ablak 13—16. oszlop= 2. tag adatai (3. lista)

.

.

.

n+ 2. ablak a—(a+ 3). oszlop= n. tag adatai

Itt *a* értéke 4*n*+5.

Az első kivételével minden résztáblázat felépítése egyforma.

A számítás módja az első hónapban

Az A5 cellába a tag nevét, a B6 cellába a tag bruttó nyereségét írjuk be. Ezután CALC parancs kiadását követően a 6. és 8. oszlopban megjelennek a számított értékek. Végül az oszlopokat nyilvántartás céljából transzfer paranccsal átvisszük a megfelelő résztáblázatba:

t, 5, a Az *a* értéke: 4*n*+5

t, 6, a+1

t, 7, a+2 (*n* a tag sorszáma)

t, 8, a+3

Természetesen a kiadott parancsba az „a" kiszámított értékét — ez 9-től 57-ig terjedő természetes szám lehet — kell beírni.

A számítás módja a további hónapokban

Először visszahozzuk a számítandó tag adatait résztáblázatból:

t, a+1, 7

(Az „a" értéke ugyanaz, mint korábban.) Ez lesz az előző időszak oszlopa.

Innentől a számítás megegyezik az első hónapban követett módszerrel. Megfigyelhető, hogy a 6-os oszlop a teljes, a 7-es oszlop a megelőző, a 8-as oszlop a tárgyidőszak kifizetéseit tartalmazza.

E program tárolására az s (save) parancs, betöltésére az 1 (load) parancs szolgál VU—CALC üzemmódban. Kinyomtatása a p paranccsal érhető el. Vigyázni kell, mert csak a képernyő tartalmát nyomtatja ki. Ha bármilyen ok miatt a gép kiesik a kalkulációs üzemmódból, GOTO 3300 segítségével térhetünk vissza.

A VU—CALC betöltése a szokásos: LOAD, és a program automatikusan indul.

SZALAI GÉZA

Hasznos rutinok C64-re

A program három fő részből áll: a rutinszétválasztóból és a két rutinból.

A rutinszétválasztó a 828-839 (Ø33C — Ø347) címeken helyezkedik el. Először meghívja azt a KERNAL rutint, amely megvizsgálja, hogy vessző következik-e, és ezután betölt az X regiszterbe egy nyolcbites számot. Ezt a számot megvizsgálja: ha 1, akkor a 84Ø-es (Ø348) címre, a 1. rutinra ugrik, ha 2, akkor a 857-es (Ø359) címre, a 2. rutinra ugrik. Ha valami más szám van ott, a program megáll.

Az egyik rutin a kurzor pozicionálását teszi lehetővé. Ami egy kicsit furcsa benne, az az, hogy először az Y koordinátát kell megadni. Működése egyszerű. Bekér egy nyolcbites számot, az lesz az Y koordináta, majd bekér egy újabb nyolcbites számot, az lesz az X koordináta. Ezután meghívja a PLOT KERNAL rutint, amely az X,Y regiszterek értéke szerint a kurzort az adott helyre irányítja. A kurzorpozicionáló rutin a 84Ø—856 (Ø348—Ø358) címeken helyezkedik el. Hívása SYS 828,1,Y koord, X koord.

```
5 X=0
7 FORI=828TO875:READA:X=X+A:POKEI,A
8 NEXTI
9 IFX<>6362THENPRINT"HIBA AZ ADATOKBAN"
20 END
30 DATA 32,241,183,224,1,240
40 DATA 5,224,2,240,18,96
50 DATA 32,241,183,138,72,32
60 DATA 241,183,138,168,104,170
70 DATA 24,32,240,255,96,32
80 DATA 241,183,142,32,208,32
90 DATA 241,183,142,33,208,32
95 DATA 241,183,142,134,2,96
```

BASIC-betöltő

A másik rutin a keret, a papír és a tinta színét állítja be. A KERNAL rutin segítségével bekér egy nyolcbites számot, ez lesz a keretszín, majd egy újabb számot, ez lesz a papírszín, és végül egy harmadik számot, amely a tintaszín lesz.

A rutin a 857—875 (Ø359—Ø368) címeken helyezkedik el. Hívása: SYS 828,2, keretszín, papírszín, tintaszín.

ROSNER GÁBOR

A teljes program

```
b33C                    1          *=828
D020                    5 KERE     =$D020;KERETSZIN TARCIME
D021                   10 ALAP     =$D021;ALAPSZIN TARCIME
0286                   15 KARA     =$0286;KARAKTERSZIN TARCIME
B7F1                   20 SZAM     =$B7F1;EGYBYTE-OS SZAM BEVETELE
FFF0                   25 CURS     =$FFF0;KURZORPOZICIONALAS TARCIM
033C 20 F1 B7          30          JSR SZAM
033F E0 01             40          CPX #01
0341 F0 05             50          BEQ CU
0343 E0 02             60          CPX #02
0345 F0 12             70          BEQ PO
0347 60                80          RTS
0348 20 F1 B7          90 CU       JSR SZAM
034B 8A               100          TXA
034C 48               110          PHA
034D 20 F1 B7         120          JSR SZAM
0350 8A               130          TXA
0351 A8               140          TAY
0352 68               150          PLA
0353 AA               160          TAX
0354 18               170          CLC
0355 20 F0 FF         180          JSR CURS
0358 60               190          RTS
0359 20 F1 B7         200 PO       JSR SZAM
035C 8E 20 D0         210          STX KERE
035F 20 F1 B7         220          JSR SZAM
0362 8E 21 D0         230          STX ALAP
0365 20 F1 B7         240          JSR SZAM
0368 8E 86 02         250          STX KARA
036B 60               260          RTS
036C                  270          .END

ALAP  =D021  CU    =0348  CURS  =FFF0  KARA  =0286  KERE  =D020  PO    =0359
SZAM  =B7F1
```

Lapdobás

COMMODORE 64

Vállalati C64-es adatfeldolgozásnál gyakran előfordul összefüggő adatblokkok tablózása. A program ezek lapdobásos kirását oldja meg.

Első része a 251-es bájton számolja a kiadott PRINT#4 utasításokat. A második rész ellenőrzi, van-e elegendő hely a legnagyobb adatblokk számára. Ha nincs, lapot dob és nullázza a 251-es bájtot.

A tablózás menete a következő lehet:

1. Betöltjük az *1. lista* szerinti BASIC-töltő programot, és a 251-es bájtot nullázzuk (POKE 251,Ø).

2. A 252-es bájtra az előforduló legnagyobb adatblokk sorainak számánál 2-3-mal nagyobb számot írunk be (például: POKE 252,1 Ø).

```
10 REM    LAPDOBAS 'BASIC BETOLTOJE'
20 :
50 FOR I=828 TO 906
60 READ A:POKE I,A:X=X+A+I
70 NEXTI
80 IF X<>79385 THEN PRINT"HIBAS":END
90 SYS828:NEW
100 END
110 DATA 162,03,189,70,3,149,114,202,208
120 DATA 248,96,76,74,3,230,122,208,02
130 DATA 230,123,32,121,0,201,152,208,09
140 DATA 152,72,164,251,200,132,251,104
150 DATA 168,76,121,0,169,72,56,229,252
160 DATA 197,251,16,30,138,72,162,4,32
170 DATA 201,255,169,72,56,229,251,133
180 DATA 251,162,0,32,215,170,232,228
190 DATA 251,208,248,169,0,133,251
200 DATA 104,170,96
```

1. lista

```
4 ;  -------------------------
5 ;
6 ;   KESZITETTE : TOTH GYULA
7 ;
8 ;  -------------------------
10 *=$033C
20 LDX #$03
30 LP LDA CIM-1,X
40 STA $0072,X
50 DEX
60 BNE LP          ;20-60 A CHRGET RUTIN
70 RTS             ;ATIRASA
75 ; A PRINT#4 UTASITASOK SZAMOLASA
76 ; ES AZOK SZAMANAK TAROLASA A $FB-BEN
80 CIM JMP UJCIM
90 UJCIM INC $7A
100 BNE KI
110 INC $7B
120 KI JSR $0079
130 CMP #$98       ;A PRINT#4 TOKENJE
140 BNE VISSZA
150 TYA
160 PHA
190 LDY $00FB
210 INY
230 STY $00FB
240 PLA
250 TAY
280 VISSZA JMP $0079
305 ; A KIIRT SOROK SZAMANAK ELLENORZESE
330 LDA #$48       ;EGY LAP 72 SOR
340 SEC
350 SBC $00FC
380 CMP $00FB
390 BPL VISSZ1
391 ; LAPDOBAS ES A $FB NULLAZASA
392 TXA
394 PHA
400 LDX #$04
410 JSR $FFC9      ;SOREMELES ELOKESZITESE
420 LDA #$48
430 SEC
440 SBC $00FB
450 STA $00FB
460 LDX #$00
510 HR JSR $AAD7 ;SOREMELES
520 INX
530 CPX $00FB
540 BNE HR
550 LDA #$00
555 STA $00FB
560 PLA
570 TAX
600 VISSZ1 RTS
```

2. lista

```
10 REM PELDAPROGRAM
20 OPEN4,4:POKE251,0
30 INPUT"KEREM AZ ADATBLOKK HOSSZAT";X
40 INPUT"KEREM A DATUMOT ";D$
50 POKE 252,X
60 FOR I=1 TO 30
70 GOSUB 170
80 PRINT#4,"     SOR-1"
90 PRINT#4,"     SOR-2"
100 PRINT#4,"     SOR-3"
110 PRINT#4,"     SOR-4"
120 PRINT#4,"     SOR-5"
130 PRINT#4,"     =====":PRINT#4
140 NEXT I
150 PRINT#4:CLOSE4:END
160 :
170 SYS867
180 IF PEEK(251)<>0 THEN RETURN
190 Z=Z+1
200 PRINT#4,CHR$(16)"30"Z".          ";D$
210 PRINT#4:RETURN
```

3. lista

3. Behívjuk a *2. listájú* tablózó BASIC programot, melyben minden adatblokkot kiíró rész előtt szerepel a SYS 867, a lapot laptetőre állítjuk, és futtatjuk a programot.

A példaprogram *(3. lista)* minden lap tetejére felírja a lapszámot és a dátumot is.

TÓTH GYULA

Programbetöltés soros illesztőn keresztül

COMMODORE 64

A programrészlet RS232 interfészen keresztül sztring formájában érkező sorokat fűz be a programba. Az ötlet lényege: mivel a C64 billentyűpuffere 10 karaktert tud fogadni, és ha ezt 13-mal töltjük fel, akkor a későbbiekben paraméterezett nyomtatás során a programból kilépve automatikusan be lehet új sorokat vinni, illetve parancsokat lehet végrehajtani.

A 63000-es sorban a kurzor pozicionálása után megnyitjuk a soros illesztőt az adatvonal paramétereivel azonosan, és beolvassuk az első sztringet, melynek szerepe kettős. Egyrészt gyakorlatban tapasztalható, hogy a fájl megnyitása után soros illesztő esetén az első karakter gyakran téveszt, ezt szűrjük ki egy fel nem használt sztring átvételével, másrészt ez a sztring adja a szinkronizálást. Nem kezdődhet számmal, és nem lehet olyan, ami értelmezhető parancsként is. Amennyiben ez a sztring nem a helyén érkezik, kiíródik a képernyőre, és a gép szintaxis hiba jelzésével leáll.

9 programsor bevételezése és kiíratása után lezárjuk a fájlt, és a gép „RUN 63000" sztringet ír ki, mely a programrészlet újbóli indítását, és ezzel újabb 9 sor bevételezését kezdeményezi.

A 63006-os sorban a billentyűpuffert (631—640. memóriacím) CR kóddal (13) töltjük fel, és a pufferpointerbe (198. cím) betöltjük a pufferben lévő karakterek számát (10), majd a kurzort a bal felső sarokba vezéreljük, és átállunk parancs üzemmódba. Ezt követően a sorok bevételezése automatikus.

A betöltés végén egy „%" karaktert vár a

```
63000 PRINT"■00":OPEN1,2,2,CHR$(134):INP
UT#1,A$:FORI=1TO9:B$=""
63001 GET#1,A$:IFA$=""THEN63001
63002 IFA$=CHR$(13)THENPRINTB$:GOTO63005

63003 B$=B$+A$:IFB$=CHR$(10)+"%"THENPRIN
T"R,63007":I=10:GOTO63005
63004 GOTO63001
63005 NEXT:CLOSE1:PRINT"R,63000"
63006 FORI=0TO9:POKE631+I,13:NEXT:POKE19
8,10:PRINT"■":END
63007 PRINT"■00":FORI=0TO8:PRINT1+63000:
NEXT:PRINT"2C ■(147)":GOTO63006
```

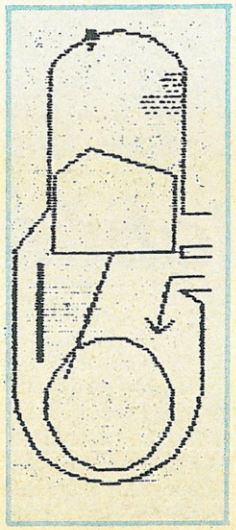
programrészlet. Ezt kiértékelve a gép kiírja a „RUN 63007" sztringet, mely parancs üzemmódban a teljes programrészlet kitörlését vonja maga után. Vigyázni kell, hogy a kinyomtatandó (megjelenítendő) sztringek hossza ne haladja meg a 80 karaktert, mert ez esetben a túlcsorduló részt a gép nem értelmezi, a sor bevétele után a kurzor a csonka sorra ugrik, és a következő bevételezésnél szintaxis hibával áll le, vagy ha tudja is értelmezni, kiesik a szinkronizmusból. A hibák leggyakoribb oka, hogy olyankor küldünk adatokat, amikor a vevő fogadásukra nincs még kész. A sztringekben rövidített kulcsszavak is lehetnek (PRINT helyett ? stb.), mert a gép ezeket is értelmezi.

A programok futás közbeni változtatására is alkalmas az alapötlet. Használatával a futó program a lemezről vagy kazettáról beolvasás közben módosítható, vagy sorokat törölhet önmagában a futás részeredményeitől függően. Ilyenkor a változók elvesznek, ezért mentésükről és újbóli betöltésükről gondoskodni kell.

KRAUSS OTTÓ

A kétütemű motor működése

ZX-SPECTRUM



A program metszetben megjeleníti a kétütemű motort, és bemutatja a működés különböző fázisait: szívás, sűrítés, robbanás (munkavégzés), kipufogás. Az s billentyűvel a program megállítható, a c billentyűvel pedig kimásolható a kívánt rajzállapot.

```
   1 REM © Nyitrai 86'
   5 REM 2 utemu motor mukodese
  10 GO SUB 290: BORDER 7: PAPER
1: INK 6: CLS
  20 CIRCLE 125,39,24
  30 PRINT AT 19,0;"Ket utemu";A
T 20,0; "motor";AT 21,0;"mukodes
e"
  40 PLOT 102,160: DRAW 0,-50: D
RAW -12,-15: DRAW 0,-50: DRAW 70
,0,PI: DRAW 0,30: DRAW -8,5: DRA
W 15,0: PLOT PEEK 23677,PEEK 236
78+12
  50 DRAW -15,0: DRAW 0,4: DRAW
15,0: PLOT PEEK 23677,PEEK 23678
+12: DRAW -15,0: DRAW 0,53: DRAW
 -50,0,2: PLOT 100,90: DRAW 0,-3
```

```
5: DRAW -1,0: DRAW 0,35
  60 PRINT AT 0,14; INK 7; BRIGH
T 1;"#"
  70 FOR a=(2*PI)-PI/6 TO 0 STEP
 -PI/5
  80 FOR z=0 TO 1
  90 LET b=126+20*COS a: LET c=3
9+20*SIN a
 100 PLOT OVER 1;b,c: DRAW OVER
1;126-b,45
 110 LET d=c+45
 120 OVER 1: PLOT 105,d: DRAW 44
,0: DRAW 0,28: DRAW -32,10: DRAW
 -13,-10: DRAW 0,-28: OVER 0
 130 IF INT c=59 THEN GO SUB 210
 140 IF INT c=56 AND b>126 THEN
PRINT AT 0,0;"Munkavegzes";AT 1,
14;"    ";AT 2,13;"      ";AT 3,1
3;"    "
 150 IF INT c=21 AND b>126 THEN
GO SUB 230
 160 IF INT c=38 AND b<126 THEN
PRINT AT 0,0;"Suritesi      "
 170 IF INT c=48 AND INT b<126 T
HEN GO SUB 280
 180 IF INKEY$="c" AND z=0 THEN
COPY
 190 IF INKEY$="s" AND z=0 THEN
GO TO 190
 200 NEXT z: NEXT a: GO TO 70
 210 PRINT AT 0,0;"robbanas    ";
AT 1,14; INK 2; BRIGHT 1;"      ";
AT 2,13;"        ";AT 3,13;"      
": IF z=0 THEN FOR x=0 TO 10: BE
EP .01,10: BEEP .01,20: NEXT x
 220 RETURN
 230 PRINT AT 0,0;"           ";
 OVER 1: PLOT 134,122: DRAW 20,-
20, PI/2: PLOT 155,102: DRAW 16,
0: PLOT 166,106: DRAW 5,-4: DRAW
 -5,-4: PRINT AT 9,22;"Kipuffoga
s"
 240 PRINT AT 9,0;"Uzemanyag";AT
 10,5;"fel": PLOT 96,40: DRAW 0,
55: DRAW 20,22: DRAW -5,0: POKE
23677,PEEK 23677+5: DRAW 0,-5: O
VER 0
 250 IF z=0 THEN FOR x=1 TO 100:
 IF INKEY$="c" THEN COPY
 260 IF z=0 THEN NEXT x
 270 RETURN
 280 OVER 1: PLOT 150,85: DRAW -
12,0: DRAW -5,-20: DRAW -4,5: PO
KE 23677,PEEK 23677+11: DRAW -6,
-5: PRINT AT 11,21;"Uzemanyag";A
T 12,21;"beszivas";AT 13,25;" ve
ge": OVER 0: GO TO 250
 290 RESTORE 300: FOR x=0 TO 14:
 READ y: POKE USR "a"+x,y: NEXT
x
 291 RETURN
 300 DATA BIN 00001110,BIN 00001
110,BIN 00011111,BIN 00011111,32
,BIN 11001110,BIN 00000100,BIN 0
0001100,BIN 01011010,0,BIN 10101
001,0,BIN 01001010,0,BIN 0010010
0,0
 310 SAVE "motor" LINE 1
 530 LET szamitas=szamitas+1: RE
AD d
 540 LET c(szamitas,1)=c+COS (PI
/6): LET c(szamitas,2)=d+(-c*SIN
 (PI/6))
 550 GO TO 525
 610 PLOT 150,0
 620 FOR n=1 TO szamitas: DRAW c
(n,1),c(n,2): NEXT n
 630 LET e=magassag*COS (PI/6):
LET f=magassag*SIN (PI/6)
 640 PLOT 150,0: DRAW e,f
 650 FOR n=1 TO szamitas: DRAW c
(n,1),c(n,2): DRAW -e,-f: DRAW e
,f: NEXT n
 660 INPUT "Kimasoljam ? i/n ";
LINE k$: IF k$="i" THEN COPY
 999 STOP
1000 FOR n=0 TO 7: READ a: POKE
USR "a"+n,a: NEXT n: DATA 48,72,
72,48,0,0,0,0
1005 PRINT TAB 10; INVERSE 1;"3-
D RAJZPROGRAM"; INVERSE 0
1050 PRINT AT 20,0;"Nyomj egy go
mbot a                rajzkeszites
hez": PAUSE 0: GO TO 40
9000 DATA -50,0,-10,20,40,20,20,
-40,1000
9999 SAVE "3D rajz"
```

Háromdimenziós ábrázoló geometria

A program megkérdezi a kért idom magasságát, nézetei számát, és három nézetben lerajzolja, végül megjelenteti három dimenzióban is. Bármelyik nézet kinyomtatható. A 9000-es DATA sort bármilyen megrajzolt egysíkú ábrára ki lehet cserélni, amelyet a program háromdimenziósról átrajzol.

NYITRAI LÁSZLÓ

```
   5 REM © Nyitrai 86
   6 REM abrazolo geometria
  10 BORDER 1: PAPER 2: INK 6: C
LS : RESTORE : GO TO 1000
  40 DIM a(100,2): DIM b(100): L
ET szamitas=0: LET a1=0: LET a2=
0: LET a3=0
  50 READ c: IF c=1000 THEN GO T
O 61
  51 LET a1=a1+c: IF a1<a2 THEN
LET a2=a1
  52 IF a1>a3 THEN LET a3=a1
  60 LET szamitas=szamitas+1: RE
AD d: LET a(szamitas,1)=c: LET a
(szamitas,2)=d: GO TO 50
  61 LET szelesseg=a3-a2
  69 REM * FORGASSIKOK *
  70 INPUT "Az idom magassaga,(m
in 2,max 50) ? ";magassag: IF ma
gassag<2 OR magassag>50 THEN GO
TO 70
  72 CLS
  75 PRINT AT 0,0;"Az idom forga
tasa: ": INPUT "Nezetek szama ?
(min )2"'nezet: LET nezet=INT ne
zet: IF nezet<2 THEN GO TO 75
  76 LET nezet=nezet-1
  77 LET pp=(127+szelesseg/2)-a3
  80 FOR n=0 TO PI/2+.001 STEP P
I/(2*nezet)
  90 FOR m=1 TO szamitas: LET b(
m)=a(m,2)*COS n: NEXT m
  95 LET a=(INT ((1(PI/2-n)*(90/P
I*2))*100+.5))/100
 100 CLS : PRINT "Szog: ";a;"°";
: PLOT pp,70: FOR m=1 TO szamito
s: DRAW a(m,1),b(m): NEXT m: IF
n=0 THEN GO TO 200
 150 LET x=magassag*SIN n: PLOT
pp,70: DRAW 0,-x: FOR m=1 TO sza
mitas: DRAW a(m,1),b(m): DRAW 0,
x: DRAW 0,-x: NEXT m
 200 INPUT "Kimasoljam ? i/n ";
LINE k$: IF k$="i" THEN COPY
 220 NEXT n
 225 REM *RAJZKESZITES*
 230 LET pp=ABS a2
 250 CLS : PRINT AT 0,0;"Az idom
 harom nezetben :"
 260 PLOT pp,70: FOR n=1 TO szam
itas: DRAW a(n,1),a(n,2): NEXT n
 270 PLOT 170,70: FOR n=1 TO sza
mitas: DRAW 0,a(n,2): NEXT n: PL
OT 170,70: DRAW magassag,0: FOR
n=1 TO szamitas: DRAW 0,a(n,2):
DRAW -magassag,0: DRAW magassag,
0: NEXT n
 280 PLOT pp,10: FOR n=1 TO szam
itas: DRAW a(n,1),0: NEXT n: PLO
T pp,10: DRAW 0,magassag: FOR n=
1 TO szamitas: DRAW a(n,1),0: DR
AW 0,-magassag: DRAW -0,magassag:
 NEXT n
 290 INPUT "Kimasoljam ? i/n ";
LINE k$: IF k$="i" THEN COPY
 300 REM * FERDE VONAL *
 310 LET x=COS (PI/4)*(magassag/
2): LET pp=120+szelesseg/2
 315 CLS : PRINT AT 5,0;"Azidom
harant nezetben"
 320 PLOT pp,10: FOR n=1 TO szam
itas: DRAW a(n,1),a(n,2): NEXT n
: PLOT pp,10: DRAW x,x: FOR n=1
TO szamitas: DRAW a(n,1),a(n,2):
 DRAW -x,-x: DRAW x,x: NEXT n
 330 INPUT "Kimasoljam ? i/n ";
LINE k$: IF k$="i" THEN COPY
 499 REM *3-D RAJZ *
 500 CLS : RESTORE 9000: PRINT "
AZ idom nezete 3 dimenzioban"
 520 DIM c(50,2): LET szamitas=0
 525 READ c: IF c=1000 THEN GO T
O 600
```

Vigyázat! Tolvaj!

Az előző alkalommal már foglalkoztunk a mágneslemez sávos szerkezetével és a sávok szektorokra osztásával. Most nézzük, hogyan épül fel egy adatblokk! Az *1. ábra* a mágneslemez egy sávjának részét ábrázolja. Láthatjuk, hogy az egyes blokkokat és a blokk fejrészét az adatblokktól szinkronjel választja el. Ez olyan speciális jelsorozat, ami a lemez formattálásakor hardver úton kerül a lemezre. Nem más, mint öt $FF bájt, azaz 40 1-es értékű bit. Mint látni fogjuk, a GCR kódolás miatt ugyanilyen jelsorozat a szinkronjelen kívül nincs a lemezen. A DOS a szinkronjel segítségével találja meg a blokkok elejét. Használata nélkül az adatblokkok egybefolynának, így nem lehetne azokat különválasztani.

A blokk fejrésze

A rendszer a blokk fejrészének adatai szerint azonosítja az író-olvasó fej pozícióját, vagyis, hogy a lemez melyik részén dolgozik. Itt hívom fel az olvasó figyelmét, hogy a „1541 User's Manual" című könyv 54. oldalán és a dr. Úry László: Commodore 64 című könyv 180. oldalán található ábra hibás, téves információt tartalmaz. A helyes elrendezés a *2. ábrán* látható.

Szinkronjel. Erről ismeri fel a DOS, hogy adatblokk vagy blokkfej következik. Negyven 1-es értékű bit.

A blokkfej azonosítója *(header block ID).* Minden esetben $Ø8. Ez tudatja a rendszerrel, hogy a szinkronjel után a blokkfej következik. Az adatrész előtt ez $Ø7.

Blokkfejellenőrző összeg *(header block checksum).* A blokkfej adatainak ellenőrzésére szolgál. Magát az értéket a sávszám, szektorszám és a két ID karakter közötti EOR (kizáró vagy) művelet elvégzése után kapjuk. Ha ez nem egyezik meg az adatok olvasása után kiszámított értékkel, akkor hibajelzést kapunk.

Szektorszám *(sector number).* Megadja, hogy az illető blokk melyik szektorban van.

Sávszám *(track number).* Megadja, hogy az illető blokk melyik sávban van. A DOS erről állapítja meg, illetőleg ellenőrzi, hogy az író-olvasó fejet helyesen pozicionálta-e.

ID karakterek. Az ID (lemezazonosító) második és első karaktere; erről ismeri fel a DOS a lemezcserét. Azért veszélyes több lemeznek ugyanazt az ID-t adni, mert akkor nem tudja megkülönböztetni az egyes lemezeket, és így felülíródnak rajta az adatok.

$ØF bájtok. A blokkfej végét jelölik.

Blokkfejrés *(header gap)* Ez nyolc $55 bájt. A rendszer sosem olvassa, ez alatt az idő alatt készül fel az adatok fogadására. A 1541-es és a 4040-es meghajtó esetén eltérnek egymástól, ezért ezek csak olvasáskompatíbilisek.

A blokk adatrésze *(3. ábra)*

Szinkronjel. Nyolc $FF bájt, azaz negyven 1-es értékű bit.

A blokk adatrészének azonosítója *(data block ID).* Minden esetben $Ø7 bájt. Ezzel különbözteti meg a DOS a szinkronjel után a blokkfejet az adatrésztől.

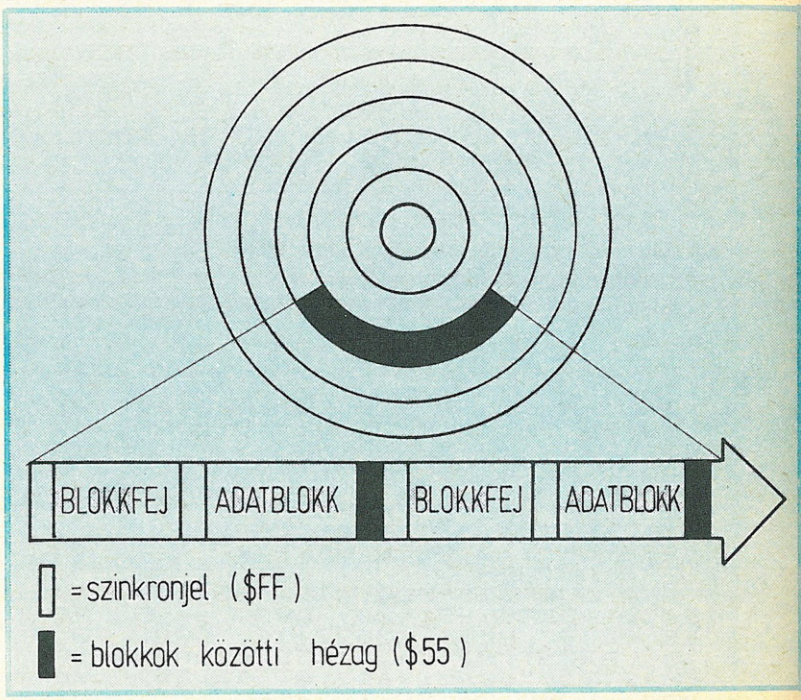
Az adatblokk ellenőrző összege *(data block checksum).* A 256 adatbájt közötti EOR (kizáró vagy) művelet elvégzéséből adódik.

$ØØ bájt. Az adatblokkot záró két bájt. A DOS formattáláskor írja fel őket azért, hogy az adathossz esetleges elcsúszása esetén kiegyenlítse a blokkot, és ne sérüljön meg a szinkronjel.

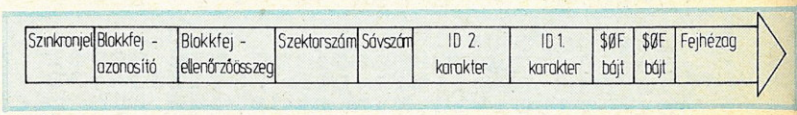
Szektorok közötti rés *(inter-sector gap).* $55 bájt, amelynek hosszúsága 4 és 12 bájt között változik.

Hibafelismerő (GCR) kód

A 1541-es meghajtónak két írásmódja van: a szinkronjel mód és a normál adatbájt mód. A DOS a szinkronjel ($FF), a szektorok közötti rés, valamint a blokkfej és az adatblokk közötti rés ($55) kivételével az adatokat GCR (Group Code Recording) kódban tárolja.



1. ábra



2. ábra

3. ábra

GCR konverzióval a rendszer minden bájt alsó és felső négy bitjéből ötöt állít elő, így egy GCR bájt tíz bitből áll. Ezt a konverziós rutint a meghajtó memóriájában a $F78F címen találhatjuk (lásd: Das grosse Floppy-Buch, 251. oldal). A GCR konverzióhoz szükséges táblázat a következő:

Hexadecimális	Bináris	GCR
$ØØ (Ø)	ØØØØ	Ø1Ø1Ø
$Ø1 (1)	ØØØ1	Ø1Ø11
$Ø2 (2)	ØØ1Ø	1ØØ1Ø
$Ø3 (3)	ØØ11	1ØØ11
$Ø4 (4)	Ø1ØØ	Ø1110
$Ø5 (5)	Ø1Ø1	Ø1111
$Ø6 (6)	Ø11Ø	1Ø110
$Ø7 (7)	Ø111	1Ø111
$Ø8 (8)	1ØØØ	Ø1ØØ1
$Ø9 (9)	1ØØ1	11ØØ1

$ØA (1Ø)	1Ø1Ø	11Ø1Ø
$ØB (11)	1Ø11	11Ø11
$ØC (12)	11ØØ	Ø11Ø1
$ØD (13)	11Ø1	11Ø1
$ØE (14)	1111Ø	11110
$ØF (15)	1111	1Ø1Ø1

Nézzük meg például, hogy a $12 átváltásához milyen lépésekre van szükség:

1. lépés: hexadecimális-bináris átváltás.
   $12 (18) =ØØØ1ØØ1Ø
2. lépés: a bájt felső négy bitjének GCR konverziója.
   ØØØ1xxxx=$1(1)=Ø1Ø11
3. lépés: a bájt alsó négy bitjének GCR konverziója.
   xxxxØØ1Ø=$2(2)=1ØØ1Ø
4. lépés: a GCR kód egyesítése (konkatenációja).
   Ø1Ø11+1ØØ1Ø=Ø1Ø111ØØ1Ø

Most pedig lépésről lépésre bemutatjuk, milyen bitműveleteket végez a rendszer egy adatblokk felírása előtt. Tekintsük az adatblokk egymást követő négy bájtját. Legyenek ezek a $Ø8, $1Ø, $ØØ, $12 számok.

1. lépés: hexadecimális-bináris átváltás

$Ø8	$1Ø	$ØØ	$12
ØØØØ1ØØØ	ØØØ1ØØØØ	ØØØØØØØØ	ØØØ1ØØ1Ø

2. lépés: bináris-GCR átváltás.
   1. A 8 bites adatbájtok alsó és felső négy bitjének elkülönítése:
   ØØØØ1ØØØ ØØØ1ØØØØ ØØØØØØØØ ØØØ1ØØ1Ø
   Ø Ø Ø Ø 1 Ø Ø Ø Ø Ø Ø Ø 1 Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø 1 Ø Ø 1 Ø
   2. A bájtok alsó és felső négy bitjének GCR megfelelői:
   Ø1Ø1Ø Ø1ØØ1 Ø1Ø11 Ø1Ø1Ø Ø1Ø1Ø Ø1Ø1Ø Ø1Ø11 1ØØ1Ø
   3. A számok 10 bit GCR-bájt megfelelői:
   Ø1Ø1ØØ1ØØ1 Ø1Ø11Ø1Ø1Ø Ø1Ø1ØØ1Ø1Ø Ø1Ø111ØØ1Ø
3. lépés: 10 bit GCR-bájtról 8 bit GCR-bájtra történő konverzió.
   1. A 10 bit GCR-bájt konkatenációja:
   Ø1Ø1ØØ1ØØ1Ø1Ø11Ø1Ø1ØØ1Ø1ØØ1Ø1ØØ1Ø111ØØ1Ø
   2. A bitsorozat nyolcbitenkénti szétválasztása:
   Ø1Ø1ØØ1Ø Ø1Ø1Ø11Ø 1Ø1ØØ1Ø1 ØØ1Ø1ØØ1 Ø1111ØØ1Ø
4. lépés: bináris-hexadecimális átváltás

Ø1Ø1ØØ1Ø	Ø1Ø1Ø11Ø	1Ø1ØØ1Ø1	ØØ1Ø1ØØ1	Ø1111ØØ1Ø
$52	$56	$A5	$29	$72

Tehát a $Ø8, $1Ø, $ØØ, $12 számsornak a GCR megfelelője: $52, $56, $A5, $29, $72.

Láthatjuk, hogy a $FF GCR megfelelője 1Ø1Ø11Ø1Ø1, tehát a DOS egyértelműen megkülönbözteti a $FF adatbájtot a szinkronjeltől.

Ezek után nézzük, hány bájtból áll valójában egy adatblokk.

	Adatbájt	GCR-bájt
Szinkronjel	5*	5
Blokkfej	8	1Ø
Fejrés	8*	8
Szinkronjel	5*	5
Adatok	26Ø	325

A *-gal jelzett bájtoknál nincs GCR konverzió.

A Floppy Disc Controller (FDC) vagyis a lemezmeghajtó vezérlője a GCR kódot hiba- és szinkronjel-felismerésre is használja. Mivel a Hamming-távolság kettő, ezért hibajavításra nem alkalmas, csak az adatrögzítési hibák felismerésére. Hiba esetén az FDC a hibás adatot $FF bájttal cseréli fel.

KOVÁCS P. ATTILA

ADOK-VESZEK-CSERÉLEK

Ebben a rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás: közületeknek gépelt soronként (6o karakter) 1oo,-Ft, magánszemélyeknek az első sor 5o,-Ft, minden további sor 2o,-Ft. Az NJSZT tagjainak az első három sor ingyenes. Hirdetéseiket a szerkesztőség címére várjuk.

48 K-s ZX-Spectrumhoz felhasználói és játékprogramokat cserélek. Válaszokat listával kérek: Kernbaum Antal, Bicske, Kisfaludy út 25. I. 29. 2060.

ZX-Spectrum 48 K, 14 000 forintért és GP-5OS nyomtató 15 000 forintért eladó. Varga Zoltán, Székesfehérvár, Hosszúsétatér u. 17. III/2. 8000.

ZX-Spectrumra 1000 darabszámú játékprogram -készletemet csere folyamán szeretném bővíteni. Válaszokat a programok listájával kérek: Cseh Attila, Debrecen, Kapitány út 16. 4033.

ZX-Spectrum 48 K számítógépre játékprogramot és segédprogramokat cserélek. Somlói Ferenc, Tatabánya, Mártírok u. 68. III/1 2800.

ZX-Spectrumhoz eladó 16 K-s ROM. Új BASIC parancsok (renum, hexaszám, hexadump, onerror, delline, variables, meleg reset). EPROM-okat beégetem. Puzsár István , Budapest , Zrínyi 125/A 1196.

48 K-s ZX-SPECTRUM-ra játékprogramokat cserélek. Farkas Tibor, Sarkad, Táncsics u. 1. 5720.

C-64-es játékprogramokat cserélek, magnót használókkal Varga Károly, Kapuvár, Lenin u. 11. 9330.

Commodore 64-es számítógép, adatrögzítő magnóval, a teljes Data Becker könyvsorozattal, programokkal együtt eladó 25 000 Ft-ért. Sári Gábor, Szolnok, Rózsa Ferenc út 6. 5000.

Commodore 64-es számítógépre játékprogramokat cserélnék. Válaszokat a programok listájával kérek: Turóczy Péter, Nyiregyháza, Korányi Frigyes u. 74. fsz. 1. 4400.

C-64-re vagy Spectrumra írt horoszkópprogramot keresek. Cserébe programokat tudok küldeni, esetleg megvenném. Nagyon sürgős ! Tel.: 84/II-494

C-16-os játékprogramokat cserélnék. Tóth Mihály, E-mőd, Akác u. 9. 3432.

Commodore 16-os, 20-as, 64-es, 128-as géppel rendelkezőkkel kapcsolatot keresek kb. 4ezer programom van. Gyermán Sándor, Zrenjanin Rade Koncara 23/V. 23000. Jugoszlávia

16 K-s ZX81 számítógép olcsón eladó. Megvételre keresem a Mikroszámítógép Magazin 1986/6. számát. 48 K-s ZX-Spectrumhoz játék-és egyéb programokat cserélek, assembly fordítóprogramot keresek. Hidvégi András, Komló, Ifjúság útja 34. 7300.

16 K-s ZX81 + botkormány +Joystick Interface(mini) + gépkönyvek eladók.Ára: 6 000 Ft. Taucz Ernő, Leninváros, Munkácsy u. 44. 3580.

16 K RAM ZX81-hez bővítő olcsón eladó. Alig használt játékprogramot is tudok adni. Sörös Róbert, Budapest, XI. Bartók Béla u.17. 1114.

VC 20-hoz 16 K RAM cartridg eladó. C-64-es játékprogramokat cserélek kazettán. Kiss János, Hódmezővásárhely, Árpád u. 2. IV/2. 6800.

Casio PB-200 -as géphez keresek FA-3 típusú interfészt. Beregszászi Attila, Mátészalka, Alkotmány u 13. 4700.

A 64'er újság olvasóival szeretném a kapcsolatot felvenni, programcsere céljából.Horváth-Papp Imre, Debrecen, Görgey u. 4. II/19. 4032.

PRIMO tulajdonosokat keresek programcsere céljából, számítógépekkel kapcsolatos könyveket eladok. Kívánságra részletes listát küldök, válaszboríték ellenében. Varsányi Gábor, Kiskőrös, Árpád u. 12. 6200.

EPROM-programozó készülékhez 28 lábas IC-foglalatok eladók (390 Ft/db). Csányi Gábor, Mezőtúr, Ifjusági lkt. 13. III/16. 5400.

TV-COMPUTER és ZX81 programokat cserélek. Menyhárt Tibor Debrecen, Gyepü sor,12. 4033.

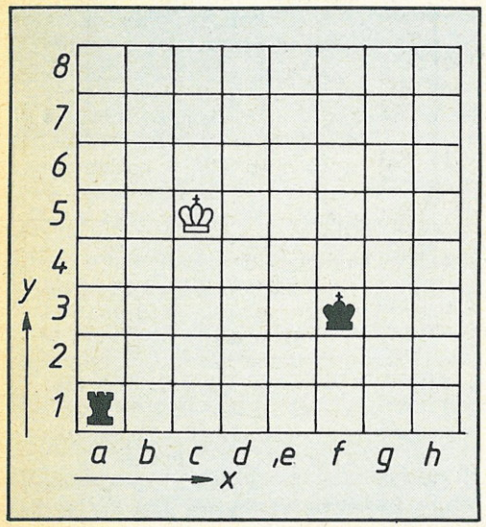
LASER 2001-es géppel, valamint német nyelvű leírással rendelkezem. Keresek hasonló géptipussal rendelkezőket tapasztalatcsere céljából. Címem: Kerekes Mihály, Keszthely, Zalka Máté út 19/A/2/3, 8360, tel.: 12-463

pontosabban reméljük, önök olvasták a „Végjáték" nevű program fejlesztésének az előzőekben megkezdett bemutatását (lásd az 1986/11-12. számban). Most ezt folytatjuk. Röviden emlékezetbe idézzük: a gép királlyal és bástyával játszik, a gép kezelője — az ellenfél — ellen. A kiinduló állás az *1. ábrán* látható.

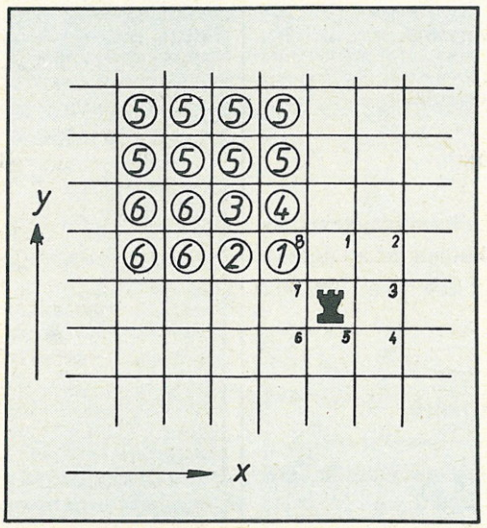
A program megírásához a Pascal nyelvet választottuk. Az előző cikkben röviden bemutattuk a főprogram vázát (vezérlési struktúráját), az adatábrázolást és az ellenfél lépését ellenőrző-elfogadó procedúrát.

bástya: al, el

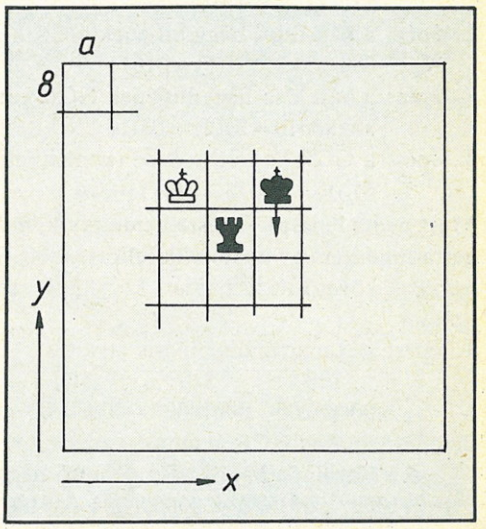
A második — a parti 4. — lépésében a gép mindig a bástyával lép előre úgy, hogy
— királya védelmébe kerüljön,
— az ellenfél királya a bástya és a tábla 8. sora között maradjon és
— a bástya a fenti két követelmény betartása mellett annyit haladjon előre y irányban, amennyit csak lehet.



1. ábra. A kiinduló állás



2. ábra. A két király relatív helyzete a bástyához képest, ahogy a program látja
A bekarikázott számok az ellenfél királyának, a be nem karikázottak a gép királyának helyzeteit jelölik a bástyához képest



*3. ábra. Az (*12*) relatív helyzet, melyben a gép a királyával „y" irányban hátra lép*

Fontolgattuk, milyen „stratégiát" válasszunk a gép soron következő lépésének meghatározásához (ez a program leglényegesebb része), és három lehetőség közül — tudva, hogy még sok más, esetleg sokkal jobb megközelítés is létezhet — kikötöttünk amellett, hogy az állásokat a gép szempontjából a két király bástyához viszonyított helyzete alapján különböztetjük meg (lásd a *2. ábrát*).

Utaltunk arra, hogy így majd 6*8=48-féle megkülönböztetett álláshoz be kell programozni a *megfelelő lépést*. (Mivel a szerző nem sakkjátékos, ezért az olvasóra bízza, hogy a bemutatott *programozási* megközelítés alapján a saját maga által *„legmegfelelőbbnek"* tartott lépéssel helyettesítse az itt közölteket.)

Most tehát a *lépésgen* nevű procedúra (szubrutin) bemutatása következik, melyben a gép lépéseit úgy határozzuk meg, hogy a gép a bástyával és a királlyal fokozatosan az a8 sarokba szorítja az ellenfél királyát, és ott mattot ad neki. (Ez jogosan vitatható egyszerűsítés: egy nagyobb, bonyolultabb programmal az ellenfél lépéseitől függően ki lehetne értékelni, hogy melyik lehet a *legmegfelelőbb* sarok.)

A gép lépéseinek megválasztása szempontjából a *„parti"*-t három részre lehet osztani.

Az első részben a gép a bástyával meglehetősen mechanikusan a saját királya mellé (védelmébe) lép. A gép első lépése — valójában a „parti" 2. lépése, mert először az ellenfél léphet — mindig ez:

A második részben — mint már szó volt róla — a gép a két király bástyához viszonyított relatív helyzete alapján 6*8=48 állást különböztet meg, és az ezekhez az állásokhoz „beprogramozott" lépést teszi meg, fokozatosan az a8 sarok felé szorítva az ellenfelet. A lépésgen procedúra hívja az ezt megvalósító lépgen2 procedúrát.

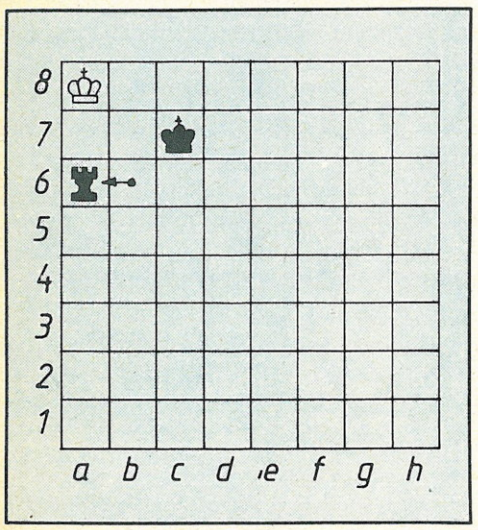
A harmadik részben az ellenfél mozgástere már annyira lecsökkent, hogy nem használható tovább a lépgen2, mert egyrészt ügyelni kell, nehogy patthelyzet alakuljon ki, másrészt, hogy esetleg már mattot lehet adni.

A parti harmadik részében a lépésgen procedúra hívja a lépgen3 procedúrát.

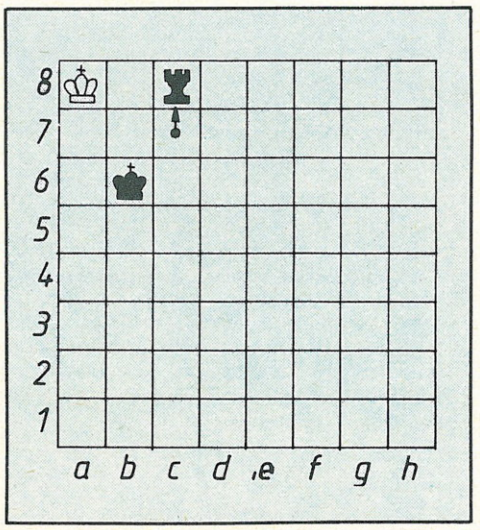
Most nézzük meg, hogy alakul ez Pascalban (vázlatosan):

```
PROCEDURE lépésgen;
  IF l<5 THEN BEGIN (* Ha az első részben vagyunk *)
    IF l=2 THEN (* Ha a gép első lépése következik *)
    állás [saját bástya, x]:=PRED(állás[saját király, x]);
    IF l=4 THEN (* Ha a gép második lépése következik *)
    állás [saját bástya, y]:=PRED(állás[ellenkirály, y]);
      IF állás [saját bástya, y]>'4' THEN
      (* Ha kilépett volna a király védelméből, akkor: *)
      állás [saját bástya, y]:='4';
      (* Akkor visszahúztuk a védelmébe *)
  (* Ide jön a lépés kijelzése a képernyőre *)
```

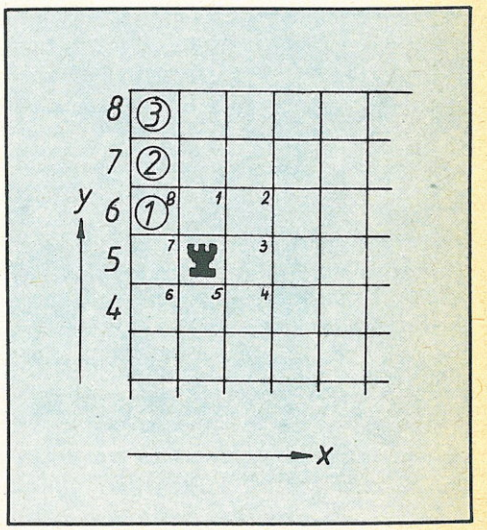
```
    END (* Első rész *)
    ELSE (* Akkor a második vagy a harmadik részben
    vagyunk *)
    IF második rész THEN [lépgen 2] (* Külön procedúra *)
    (* Ha nem a második, akkor a harmadik részben vagyunk *)
                    ELSE [lépgen 3] (* Külön procedúra *)

    l:=l+1; (* Lépésszámláló *)
END; (* Lépésgen procedúra)
```



4. ábra. Az egyik matthelyzet



5. ábra. A másik matthelyzet



6. ábra. Az állások egyik javasolt megkülönböztetése a „parti" harmadik részében
A bekarikázott számok az ellenfél királyának, a be nem karikázottak a gép királyának helyzeteit jelölik a bástyához képest

```
(* Megjegyzés:
   VAR második rész: BOOLEAN;
        .
        .
        második rész:=((xb>'b') AND (yb<'7')) *)
```

Mint említettük, kritizálható a gép első két lépése: lehetne a bástya veszélyeztetése nélkül olyan lépéseket is generálni, melyek az ellenfél mozgásterét már az első lépésekben jobban szűkíthetnék — így a mattig vezető lépések száma csökkenthető lenne. A program felépítése szinte kínálja az utólagos finomítást. Ezt az olvasóra bízzuk.

Most foglalkozzunk a lépéseket a parti második részében (a gép harmadik lépésétől) a parti harmadik részének eléréséig generáló lépgen2 procedúrával.

```
PROCEDURE lépgen2;
VAR     xe,xs,xb:CHAR;
        ye,ys,yb:CHAR;
BEGIN xe:=állás[ellenkirály, x];ye:=állás[ellenkirály, y];
      xs:=állás[saját király, x];ys:=állás[saját király, y];
      xb:=állás[saját bástya, x];yb:=állás[saját bástya, y];
(* Az xe,ye,xs,ys,xb és yb segédváltozók feltöltése megtörtént *)
(* Ø a "K<>"eset kezelése. Lásd később! *)
(*1) (* Az ellenkirály 1-es relatív pozíciója esetén *)
IF (xe=PRED(xb) AND (ye=SUCC(yb)) THEN BEGIN
(*11*) (* A saját király nem lehet az 1-es relatív helyzetben,
       mert az ellenkirály ütné: kizárt eset *)
(*12*) IF((xs=SUCC(xb)) AND (ys=SUCC(yb))) THEN
       ys:=PRED (ys) ELSE BEGIN
(*13*) IF((xs=SUCC(xb)) AND (ys=yb)) THEN
       ys:=SUCC(ys) ELSE BEGIN
          .
          .
          .
       END END END END END
END (*1*) ELSE BEGIN
(*2*)
(* És így tovább *)
```

```
          .
          .
          .
(*Ø A „K<>Ø" eset kezelése. Lásd később!)
          .
          .
          .
(*1*)
  IF<az ellenkirály az 1-es pozícióban van a bástyához képest>
THEN
(*11*) (* Kizárt eset, mert ha a saját király is az első pozícióban
       lenne, akkor a két király ütné egymást *)
(*12*) IF<a saját király a 2-es pozícióban van>THEN

  [a saját királlyal lépjünk egyet „y" irányban vissza]

                                        ELSE BEGIN
```

Kicsit másképpen fogalmazva tehát: hogy a lépgen2 az általa elvileg megkülönböztetett 6*8=48 fajta állásban hogyan lépjen, így oldja meg:

```
0.1000
0.2000 (************************************)
0.3000
0.4000     PROGRAM VEGJATEK1(INPUT,OUTPUT);
0.5000
0.6000 (************************************)
0.7000
0.8000 TYPE
0.9000       KOORDT=(X,Y);
1.0000       (*A SAKK-TABLA KET KOORDINATAJA*)
1.1000       XSETTIP=SET OF 'A'..'H';
1.2000       (*X TARTOMANYA *)
1.3000       YSETTIP=SET OF '1'..'8',
1.4000       (*Y TATOMANYA *)
1.5000       NBETUT=SET OF 'A'..'H';
1.6000       (*A KEZELES "BARATSAGOSABBA"
1.7000       TETELEHEZ*)
1.8000       LEPEST=0..30,
1.9000       (*A LEPES-SZAM-VALTOZO TIPUSA *)
2.0000       POZT=ARRAY[KOORDT] OF CHAR;
2.1000          (* EGY-EGY FIGURA HELYET
2.2000       TARTALMAZO KOORDINATA-PAR
2.3000       TIPUSA *)
2.4000       FIGURAT=(SK,SB,EK);
2.5000       (*A HAROM FIGURA:SAJAT KIRALY ES
2.6000         BASTYA,MEG
2.7000         AZ ELLENFEL KIRALYA*)
2.8000       ALLAST=ARRAY[FIGURAT] OF POZT;
2.9000       (*A HAROM FIGURA ALLASA*)
3.0000       SMPNT=(SAKK,MATT,PATT,NORMAL);
3.1000
3.2000 (************************************)
3.3000
3.4000 VAR   L:LEPEST;(*A LEPES-SZAMLALO*)
3.5000       XSET:XSETTIP;  YSET:YSETTIP;
3.6000       NSET: NBETUT;
3.7000       KS,BS,KE:POZT;
3.8000       (*A HAROM FIGURA POZICIOJA*)
3.9000       SMPNV:SMPNT,
4.0000       (*EZ A VALTOZO TARTALMAZZA,
4.1000         HOGY SAKK,...VAN-E*)
4.2000       ALLAS:ALLAST;
4.3000       (*AZ ALLAS-T TARTALMAZO VALTOZO*)
4.4000       JATEK:ARRAY[LEPEST] OF ALLAST;
4.5000       (* A JATEK EGESZ MENETE,
4.6000       AZAZ AZ EGYMAST KOVETO ALLASOK*)
4.7000       VEGF:BOOLEAN,
4.8000       (* VEGE A "PARTI"-NAK:
4.9000 FELADTA,MATT *)
5.0000       FELAD:BOOLEAN;
5.1000      (*A JATEKOS FELADJA-E A JATEKOT*)
5.2000       UJBOL:BOOLEAN,
5.3000      (*A JATEKOS KIVAN-E UJABB "PARTIT"
5.4000      JATSZANI*)
5.5000       VALASZ:CHAR;
5.6000      (*SEGED-VALTOZO A DIALOGUSHOZ*)
5.7000       K:0..3;
5.8000  (*SEGED-VALTOZO A "LEPGEN2"-PROC-HOZ*)
5.9000       (* A "K" NEVU SEGED-VALTOZO-BAN
6.0000         AZT AZ INFORMACIOT TAROLJUK,
6.1000          HOGY AZ ELLENFEL KIRALYANAK
6.2000        "ULDOZESE" KOZBEN A BASTYA
6.3000         LESZAKADT-E KIRALYTOL,
6.4000         ES HA IGEN, HOGYAN *)
6.5000
6.6000 (************************************)
6.7000
6.8000
6.9000
7.0000
7.1000 PROCEDURE LEPGEN2;
7.2000 (*EZT A SUHRUTIN-T A "LEPESGEN"
7.3000 PROCEDURA HIVJA
7.4000   A GEP MASODIK LEPES UTAN A PARTI
7.5000 MASODIK
7.6000   RESZEBEN A GEP E SZERINT LEP,
7.7000 FOKOZATOSAN AZ "A8" SAROKBA
7.8000   SZORITVA AZ ELLENFEL KIRALYAT *)
7.9000 VAR
8.0000     XE,XS,XB:'A'..'H',
8.1000     YE,YS,YB:'1'..'8',
8.2000     TAKARX,TAKARY:BOOLEAN;
8.3000
8.4000     (*FIGYELEM: A PROCEDURAK LOKALIS
8.5000  VALTOZOI AZ EGYES HIVASOK
8.6000       KOZOTT "ELTUNNEK", EZERT KELL
8.7000  PELDAUL A "K" VALTOZOT
8.8000       UN. GLOBALIS VALTOZOKENT,
8.9000  A PROGRAM ELEJEN DEKLARALNI *)
9.0000
9.1000 BEGIN
9.2000   XE:=ALLAS[EK,X];  YE:=ALLAS[EK,Y];
9.3000   XS:=ALLAS[SK,X];  YS:=ALLAS[SK,Y];
9.4000   XB:=ALLAS[SB,X];  YB:=ALLAS[SB,Y];
9.5000
9.6000    (*C*)
9.7000    (*HA A BASTYA AZ ELOZO LEPESBEN
9.8000  LESZAKADT VOLNA A KIRALY-TOL: *)
9.9000    IF K<>0   THEN BEGIN
10.0000      IF K=1 THEN BEGIN XS:=PRED(XS);
10.1000 YS:=SUCC(YS) END;
10.2000      IF K=2 THEN YS:=SUCC(YS);
10.3000      IF K=3 THEN XS:=PRED(XS);
10.4000      K:=0 END (*0*)
10.5000   ELSE BEGIN
10.6000    (* FIGYELJUK MEG, HOGY "K" ERTEKET
10.7000  AZ ALABBIAKBAN AZOKNAL A
10.8000       LEPESEKNLL ALLAITJUK BE "0"-TOL
10.9000 KULONBOZO ERTEKRE, AHOL
11.0000       A BASTYA "ELSZAKADT" A SAJAT
11.1000 KIRALYATOL. EZ OSSZESEN
11.2000       7 ESETBEN TORTENHET MEG. 3-FELE
11.3000 KORREKCIOT ALKALMAZUNK
```

```
11.4000       A KOVETKEZO LEPES-BEN *)
11.5000
11.6000    (*1*)
11.7000    IF ((XE=PRED(XB)) AND
11.8000       (YE=SUCC(YB)))
11.9000       THEN BEGIN
12.0000    (*11*) (*KIZART ESET*)
12.1000
12.2000    (*12*) IF ((XS=SUCC(XB)) AND
12.3000             (YS=SUCC(YB))) THEN
12.4000             YS:=PRED(YS) ELSE BEGIN
12.5000    (*13*) IF ((XS=SUCC(XB)) AND (YS=YB))
12.6000             THEN
12.7000             YS:=SUCC(YS) ELSE BEGIN
12.8000    (*14*) IF ((XS=SUCC(XB)) AND
12.9000             (YS=PRED(YB))) THEN
13.0000             YS:=SUCC(YS) (*PONTOSITANDO*)
13.1000             ELSE BEGIN
13.2000    (*15*) IF ((XS=XB) AND (YS=PRED(YB)))
13.3000             THEN
13.4000             XS:=PRED(XS) ELSE BEGIN
13.5000    (*16*) IF ((XS=PRED(XB)) AND
13.6000 (YS=PRED(YB))) THEN
13.7000             XS:=SUCC(XS) ELSE BEGIN
13.8000    (*17*) (*KIZART ESET*)
13.9000
14.0000    (*18*) (*KIZART ESET*)
14.1000
14.2000           END END END END END
14.3000 (* 5-SZOR "END" *)
14.4000    END (*1*) ELSE BEGIN
14.5000
14.6000    (*2*)
14.7000    IF ((XE=PRED(PRED(XB))) AND
14.8000 (YE=SUCC(YB))) THEN BEGIN
14.9000    (*21*) IF ((XS=XB) AND
15.0000             (YS=SUCC(YB))) THEN
15.1000             XB:=PRED(XB) ELSE BEGIN
15.2000    (*22*) IF ((XS=SUCC(XB)) AND
15.3000             (YS=SUCC(YB))) THEN
15.4000             XS:=PRED(XS) ELSE BEGIN
15.5000    (*23*) IF ((XS=SUCC(XB)) AND
15.6000             (YS=YB)) THEN
15.7000             BEGIN XS:=PRED(XS);
15.8000             YS:=SUCC(YS) END ELSE BEGIN
15.9000    (*24*) IF ((XS=SUCC(XB)) AND
16.0000             (YS=PRED(YB))) THEN
16.1000             XS:=PRED(XS)
16.2000             (*PONTOSITANDO*)
16.3000             ELSE BEGIN
16.4000    (*25*) IF ((XS=XB) AND (YS=PRED(YB)))
16.5000             THEN
16.6000             XB:=PRED(XB)
16.7000            ELSE BEGIN
16.8000    (*26*) IF ((XS=PRED(XB)) AND
16.9000              (YS=PRED(YB))) THEN
17.0000             XB:=PRED(XB);
17.1000    (*27*) (*KIZART*)
17.2000
17.3000    (*28*) (*KIZART*)
17.4000
17.5000  END END END END END (*5-SZOR "END" *)
17.6000  END (*2*) ELSE BEGIN
17.7000
17.8000    (*3*)
17.9000    IF ((XE=PRED(PRED(XB))) AND
18.0000    (YE=SUCC(SUCC(YB))))  THEN BEGIN
18.1000    (*31*) IF ((XS=XB) AND
18.2000             (YS=SUCC(YB))) THEN
18.3000             XB:=PRED(XB) ELSE BEGIN
18.4000    (*32*) IF ((XS=SUCC(XB)) AND
18.5000             (YS=SUCC(YB))) THEN
18.6000             YB:=SUCC(YB) ELSE BEGIN
18.7000    (*33*) IF ((XS=SUCC(XB)) AND
18.8000             (YS=YB)) THEN
18.9000             YB:=SUCC(YB) ELSE BEGIN
19.0000    (*34*) IF ((XS=SUCC(XB)) AND
19.1000             (YS=PRED(YB))) THEN
19.2000             BEGIN XB:=PRED(XB); K:=1 END
19.3000             (*VIGYAZAT*)
19.4000             ELSE BEGIN
19.5000    (*35*) IF ((XS=XB) AND
19.6000             (YS=PRED(YB))) THEN
19.7000             XB:=PRED(XB) ELSE BEGIN
19.8000    (*36*) IF ((XS=PRED(XB)) AND
19.9000             (YS=PRED(YB))) THEN
20.0000             XB:=PRED(XB) ELSE BEGIN
20.1000    (*37*) IF ((XS=PRED(XB)) AND
20.2000             (YS=YB)) THEN
20.3000             YB:=SUCC(YB) ELSE BEGIN
20.4000    (*38*) (*KIZART*)
20.5000
20.6000           END END END END END END END
20.7000 (*7-SZER "END" *)
20.8000    END (*3*) ELSE BEGIN
20.9000
21.0000    (*4*)
21.1000    IF ((XE=PRED(XB)) AND
21.2000       (YE=SUCC(SUCC(YB)))) THEN BEGIN
21.3000    (*41*) (*KIZART ESET*)
21.4000
21.5000    (*42*) IF ((XS=SUCC(XB)) AND
21.6000             (YS=SUCC(YB))) THEN
21.7000            YB:=SUCC(YB) ELSE BEGIN
21.8000    (*43*) IF ((XS=SUCC(XB)) AND
21.9000               (YS=YB)) THEN
22.0000            YB:=SUCC(YB) ELSE BEGIN
22.1000    (*44*) IF ((XS=SUCC(XB)) AND
22.2000               (YS=PRED(YB))) THEN
22.3000            YS:=SUCC(YS) ELSE BEGIN
22.4000    (*45*) IF ((XS=XB) AND
22.5000               (YS=PRED(YB))) THEN
22.6000            BEGIN XS:=PRED(XS);
```

```
22.7000            YS:=SUCC(YS) END ELSE BEGIN
22.8000    (*46*) IF ((XS=PRED(XB)) AND
22.9000               (YS=PRED(YB))) THEN
23.0000            YS:=SUCC(YS) ELSE BEGIN
23.1000    (*47*) IF ((XS=PRED(XB)) AND
23.2000               (YS=YB)) THEN
23.3000            YB:=SUCC(YB) ELSE BEGIN
23.4000    (*48*) (*KIZART ESET*)
23.5000
23.6000    END END END END END END
23.7000           (*6-SZOR "END" *)
23.8000    END (*4*)  ELSE BEGIN
23.9000
24.0000    (*5*)
24.1000    IF ((XE<XB) AND
24.2000           (YE>SUCC(SUCC(YB)))) THEN BEGIN
24.3000    (*51*) IF ((XE=PRED(PRED(XB))) AND
24.4000           (YE=SUCC(YB))) THEN
24.5000           XS:=SUCC(XS) ELSE BEGIN
24.6000    (*52*) IF ((XS=SUCC(XB)) AND
24.7000           (YS=SUCC(YB))) THEN
24.8000           YB:=SUCC(SUCC(YB)) ELSE BEGIN
24.9000    (*53*) IF ((XS=SUCC(XB)) AND
25.0000           (YS=YB)) THEN
25.1000           YB:=SUCC(YB) ELSE BEGIN
25.2000    (*54*) IF ((XS=SUCC(XB)) AND
25.3000           (YS=PRED(YB))) THEN
25.4000           BEGIN YB:=SUCC(YB); K:=1 END
25.5000           (*VIGYAZAT*)
25.6000           ELSE BEGIN
25.7000    (*55*) IF ((XS=XB) AND
25.8000           (YS=PRED(YB))) THEN
25.9000           BEGIN YB:=SUCC(YB); K:=1 END
26.0000           (*VIGYAZAT*)
26.1000           ELSE BEGIN
26.2000    (*56*) IF ((XS=PRED(XB)) AND
26.3000           (YS=PRED(YB))) THEN
26.4000           BEGIN YB:=SUCC(YB); K:=2 END
26.5000           (*VIGYAZAT*)
26.6000           ELSE BEGIN
26.7000    (*57*) IF ((XS=PRED(XB)) AND
26.8000           (YS=YB)) THEN
26.9000           YB:=SUCC(YB) ELSE BEGIN
27.0000    (*58*) IF ((XE=PRED(XB)) AND
27.1000           (YE=SUCC(YB))) THEN
27.2000           YB:=SUCC(SUCC(YB));
27.3000
27.4000           END END END END END END END
27.5000           (*7-SZER "END" *)
27.6000    END (*5*) ELSE BEGIN
27.7000
27.8000    (*6*)
27.9000    IF ((XE<PRED(PRED(XB))) AND
28.0000  (YE<=SUCC(SUCC(YB))))   THEN BEGIN
28.1000    (*61*) IF ((XS=XB) AND (YS=SUCC(YB)))
28.2000           THEN
28.3000           XB:=PRED(XB) ELSE BEGIN
28.4000    (*62*) IF ((XS=SUCC(XB)) AND
28.5000           (YS=SUCC(YB))) THEN
28.6000           BEGIN XB:=PRED(XB);K:=3 END
28.7000           (*VIGYAZAT*)
28.8000           ELSE BEGIN
28.9000    (*63*) IF ((XS=SUCC(XB)) AND (YS=YB))
29.0000           THEN
29.1000           BEGIN XB:=PRED(XB);K:=1 END
29.2000          (*VIGYAZAT*)
29.3000           ELSE BEGIN
29.4000    (*64*) IF ((XS=SUCC(XB)) AND
29.5000           (YS=PRED(YB))) THEN
29.6000           BEGIN XB:=PRED(XB);
29.7000           K:=1 END (*VIGYAZAT*)
29.8000           ELSE BEGIN
29.9000    (*65*) IF ((XS=XB) AND (YS=PRED(YB)))
30.0000           THEN
30.1000           XB:=PRED(XB) ELSE BEGIN
30.2000    (*66*) IF ((XS=PRED(XB)) AND
30.3000           (YS=PRED(YB))) THEN
30.4000           XB:=PRED(PRED(XB)) ELSE
30.5000           BEGIN
30.6000    (*67*) IF ((XS=PRED(XB)) AND
30.7000           (YS=YB)) THEN
30.8000           BEGIN XS:=SUCC(XS);
30.9000           YS:=PRED(YS) END ELSE BEGIN
31.0000    (*68*) IF ((XS=PRED(XB)) AND
31.1000           (YS=SUCC(YB))) THEN
31.2000 (*$J+++) (*UTASITAS A COMPILERNEK
31.3000            AZ "UGRAS-MELYSEG"
31.4000           ELFOGADASARA*)
31.5000           XB:=PRED(PRED(XB))
31.6000
31.7000         END END END END END END END
31.8000          (*7-SZER "END" *)
31.9000    END (*6*)
32.0000
32.1000
32.2000    END END END END END END;
32.3000    (* 6-SZOR "END" *)
32.4000
32.5000
32.6000 WRITELN('A GEP HUZASA AZ ',L:3,'-IK
32.7000                          LEPESBEN:');
32.8000 IF ((ALLAS[SK,X] =XS) AND
32.9000 (ALLAS[SK,Y]=YS)) THEN
33.0000    WRITELN(ALLAS[SB,X],
33.1000            ALLAS[SB,Y],'---->',XB,YB)
33.2000 ELSE WRITELN(ALLAS[SK,X],
33.3000            ALLAS[SK,Y],'---->',XS,YS);
33.4000 ALLAS[SK,X]:=XS; ALLAS[SK,Y]:=YS;
33.5000 ALLAS[SB,X]:=XB; ALLAS[SB,Y]:=YB;
33.6000
33.7000 END; (*LEPGEN2*)
33.8000
33.9000
34.0000
34.1000
34.2000
```

(*13*) IF < a *saját király* a 3-as pozícióban van > THEN

a saját királlyal lépjünk egyet „y" irányban előre

ELSE BEGIN (Lásd a *3. ábrát.*)

(*14*) IF és így tovább.

.

.

-

A 6*8=48 eset közül összesen 7 esetben nem tudott a szerző ellenállni a kísértésnek, és a bástyával úgy lépett a gép nevében, hogy az „leszakadt" a királyától. (Lásd a később részletesen bemutatott lépgen2-ben a 34., 54., 55., 56., 62., 63., 64. állást.) A részletesen kidolgozott lépgen2 ezt a következő lépésnél figyelembe veszi (Ø-ás eset!), és a királlyal a bástya után lép.

(K=1, K=2 és K=3 szerint háromfajta korrekciós lépést tehet ilyenkor.)

Az „IF THEN ... ELSE" konstrukció egyszerűvé teszi, hogy ha a gép egy állásra ráismert, akkor a többit már ne vizsgálja. Ahogy a gép a keresésben előre halad, a program ugrásmélysége nő. Egyes compilerek a hatékonyság érdekében az ugrásmélységet beállítják valamilyen előzetesen maximált értékre. Ezt túllépni csak akkor tudjuk, ha a megfelelő helyen *direktívát* adunk a compilernek, hogy az adott környezetben engedje az ugrásmélység túllépését. A szerzőnek — az általa használt compilernél — ezt meg kellett tennie a „(* $J ++ *)" furcsa kommentár formájában. Az adott compiler ebből megérti a kérést, hogy engedje az előzetesen beállított (default) ugrásmélység megnövelését további két fokozattal.

Most pihenésképpen hagyjuk el egy rövid időre a lépgen2-t, és vizsgáljunk meg egy olyan triviális részletet, hogy miként „veszi be" a gép az ellenféltől a következő lépést jelentő koordinátapárt, azaz a két karaktert:

```
PROCEDURE kérdez; (* Az ellenfelet kérdezi a gép a következő
                     lépésről vagy arról, hogy feladja-e a „par-
                     tit" *)
VAR xbe, ybe: CHAR;
    megfelel: BOOLEAN;
    BEGIN
    REPEAT (* Amíg a beadott lépés meg nem felel *)
    megfelel:=FALSE;
    WRITELN          ('Kérem, lépjen a királlyal, vagy küldjön
                      egy „v" betűt, ha fel kívánja adni.')
    WHILE NOT EOLN DO GET (INPUT);
    (* Az előző sorra „technikai okokból" van szükség *)
    GET (INPUT); xbe:= INPUT↑;
    GET (INPUT); ybe:= INPUT↑;
    IF ((xbe='v') OR (xbe='V'))
    THEN BEGIN felad:=TRUE;
                megfelel:=TRUE END
         ELSE felad:=FALSE;
    IF NOT felad THEN BEGIN
    megfelel:=FALSE;

    ellenőriz (xbe, ybe, megfelel);

    (* Ez az előző cikkben bemutatott procedúra *)
    IF NOT megfelel THEN<kijelzi ezt az ellenfélnek>
    END (* nem adta fel *)
    UNTIL megfelel; (* Azaz, amíg az ellenfél lépése meg nem
                       felel *)
    IF NOT felad THEN BEGIN
    <kijelzi az ellenfélnek>;
    l:=l+1; (* lépésszámláló *)
    END ELSE
  WRITELN ('Tehát ön feladta a partit. Köszönöm a játékot.')
END; (* A „Kérdez" procedúra vége *)
```

Az ellenfél így a lépését az alábbi módon adhatja be:

(xbe) (ybe) (adatátvitel)

vagy — ha feladja a partit:

(v) (adatátvitel)

Az ellenfél a kis „v" helyett megnyomhatja a nagy „V" billentyűt is. Az „adatátviteli" billentyű gépenként, terminálonként más és más lehet (lehet például egy „RETURN" feliratú billentyű is).

Általános tapasztalat, hogy a Pascal nyelv READLN (xbe) beépített procedúrájával *„óvatosan"* kell bánni. A bemutatott megoldásra szeretnénk felhívni a kezdő Pascalozók figyelmét, sok felesleges próbálgatást lehet vele megtakarítani.

A „Kérdez" procedúrában xbe és ybe típusát célszerűtlen lett volna így deklarálni:

```
VAR xbe:'a'..'h';
    ybe:'1'..'8';
```

Ilyen megoldás mellett ugyanis, ha az ellenfél melléütne, tehát nem a megfelelő billentyűt nyomná meg a klaviatúrán, akkor a program futását biztosító *run-time-system* megszakítaná a futást és hibát jelezne. Ebből a helyzetből csak a program újbóli indításával lenne kiút.

A „VAR xbe,ybe:CHAR;" deklaráció mellett viszont az ellenfél melléütéseit az előző cikkben bemutatott *„ellenőriz"* procedúra anélkül lekezelné, hogy a program futását meg kellene szakítani. (Megjegyezzük, hogy hasonló problémáink lettek volna „VAR xbe:'a'..'h'; ybe: INTEGER;" deklaráció esetén is.)

Most pedig lássuk részletesen a lépgen2 procedúrát (lásd a listát).

A *harmadik* rész akkor kezdődik, ha az
(xb='b') OR (yb='7')
feltétel teljesül.

A matthelyzet kétféle lehet (*4.* és *5. ábra*).

Visszafelé indulva, a matthelyzetből könnyen rájöhetünk arra, hogy a matt eléréséhez elsősorban arra kell vigyáznunk, hogy királyunk (a gép királya) *„jó ütemben"* kísérje „kívülről" a bástyát. Ha nem kísérné jó ütemben, akkor a gép a király egy oldallépésével korrigálja az ütemet, hogy azután biztos lépésekben haladjon az elkerülhetetlen matt felé, melyet majd a felhasználónak is jelezni kell.

A harmadik részben működésbe lépő *lépgen3* procedúra felépítése, szerkezete a szerző elképzelése szerint lényegében ugyanolyan lehet, mint a lépgen2-é, csak a figyelt állásokat célszerű egy kicsit módosítva megválasztani — például a *6. ábra* szerinti módon.

A szerző bevallja, hogy nincs maradéktalanul megelégedve a bemutatott, illetve eltervezett megoldással. Úgy értékeli, hogy az általa „beprogramozott" stratégia *nem elég intelligens,* túlságosan direkt: ugyanis állásonként megmondja a gépnek, hogy pontosan mit lépjen. Valószínűleg azonban egy ilyen egyszerű végjátéknál módszere nem marasztalható el teljesen, bár csak egy kicsit is bonyolultabb helyzetben ez az út már nem lenne járható: túl sok lenne a közvetlenül megvizsgálandó eset. Az állások finomabb megkülönböztetésével, a bástya merészebb mozgatásával még ez a stratégia is javítható. Nagyon örülnék, ha az olvasók törnék a fejüket, hogyan lehetne az itt közöltnél egyszerűbb, *intelligensebb* és jobb megoldásokat találni.

-KE-

RAINBOW a µM-ban

Amint korábban már jeleztük, módunkban áll cikkeket, programleírásokat átvenni az amerikai RAINBOW számítógép magazintól. A magazin teljes egészében a TANDY RADIO SHACK Color Computer (az USA-ban népszerűvé vált nevén: CoCo) számára írt havilap.

A mi mércénkkel mérve szokatlan, hogy egyetlen géptípusnak egy egész havilapot szenteljenek. Pedig a RAINBOW a maga kb. 300 oldalával nem is az egyetlen, kizárólag a CoCo-val foglalkozó újság. Legalább további öt önálló CoCo-lap létezik még, mi legalábbis ennyit ismerünk: Color Computer Magazine, Color Computer News, Micro Color Journal, CoCo Ads, Spectrogram. Ismerünk továbbá olyan folyóiratokat, amelyek nagyobb terjedelmet biztosítanak a CoCo-nak a lapon belül, mint a mi magazinunk a C64-nek vagy a Sinclair gépeknek. Közéjük tartozik a 80 Micro; CoCo-val foglalkozó szekciója a „HOT CoCo", amelyet 1983-tól 1986 februárjáig azonos címmel önálló kiadványként jelentetett meg a kiadó.

Az elmondottak alapján a gépnek rendkívül népszerűnek kell lennie az USA-ban. Erre utal az is, hogy több mint 1 millió darabot adtak el belőle. Nyilvánvaló ezek után, hogy megszámlálhatatlan hardver- és szoftverház lát üzletet a fejlesztésben a CoCo számára. Példaként említjük a kaliforniai Microworks által kifejlesztett képdigitalizálót, amellyel másodpercenként nyolc kép állítható elő 256×256 bontásban. A szoftverfejlesztések közül kiemelkedő az OS—9 operációs rendszer (Microware), amely a mini- és nagygépeken futó UNIX rendszer adaptációja 8 bites rendszerre. (A CoCo processzora valójában nem igazán 8 bites: számos 16 bites utasítást tartalmaz.)

A CoCo-t 1980 óta forgalmazzák. Hazánkban nem ismerik, noha a nálunk népszerű, azonos árkategóriájú típusok egyike sem tud összességében ennél a gépnél többet.

A CoCo-tól kapható többlet csak részben köszönhető a processzornak (6809E). A felhasználó igen könnyen kezelhető, nagy hatásfokú és sebességű BASIC-et kap kézhez; a BASIC interpretert a Microsoft cég készítette. Bár az 1980-ban készült változat egy-két módosításon átesett azóta, a változtatásokat a kompatibilitási problémákat kizáróan oldották meg. Az interpreter egyébként sok hasonlóságot mutat az IBM PC BASIC-jéhez.

A RAINBOW cikkeiből válogatva természetesen olyan cikkek megjelentetésére törekszünk, amelyek közérdeklődésre tarthatnak számot. Az elméleti cikkek feltétlenül ilyenek, szándékunkban áll azonban programokat is közzétenni. A programok valamilyen mértékben nyilván gépfüggőek. Igyekszünk azonban majd a programokat oly mértékben leegyszerűsíteni, továbbá olyan magyarázatot adni, hogy — kivált a MICROSOFT BASIC ismerőinek — ne jelentsen gondot a programok adaptálása.

„STYLIST" — Epson-nyomtatók kezelője

Elsőként az Epson- vagy Epson-kompatíbilis nyomtatók birtokosainak adunk egy igen jól hasznosítható programot, amellyel az Epson valamennyi betűképe könnyen előállítható, szükségtelenné téve a nyomtatói formákhoz tartozó kódok örökös keresgélését. A programot a nem Epson- vagy Epson-kompatíbilis nyomtatóval dolgozó olvasóink is kipróbálhatják a kódok átírásával a saját nyomtatójukhoz specifikált kódokra.

A program menüre szervezett. A betűkép a megfelelő billentyű lenyomásával egyszerűen változtatható. A meghívott kód mindaddig érvényben marad, amíg a nyomtatót ki nem kapcsoljuk, vagy a program azt meg nem változtatja. A program szövegszerkesztővel együtt is, de írógépként is működhet.

Ahhoz, hogy a kódokat bármikor megváltoztathassuk, a nyomtatónak állandóan fogadóképes állapotban kell lennie. Ezért a program is állandóan figyeli a nyomtató állapotát, és jelzi, ha a printer esetleg nem fogadóképes. A CoCo esetében a $FF20 (65314)-en található adat jelzi a nyomtató fogadóképességét (a 200—210-es és a 710—720-as sorokban).

A program kezelési útmutatót nem igényel, a menü mindenről tájékoztat. Rendkívül egyszerű működését az alábbiakban ismertetjük.

A program két képernyőt igényel. Az elsőn a „Print Stylist" menü jelenik meg; itt lehet a megfelelő betűváltozatot kijelölni. A kínálat a következő: pica és elit mint betűtípus, továbbá kurzív, kiemelt, vékonyított, széthúzott, nyomott, kétszeresen kicsinyített és aláhúzott forma. Ha a kívánt betűképhez tartozó billentyűt lenyomjuk, a megfelelő menüsorban megjelenik egy "*" — visszajelezvén a választást. Ugyanannak a billentyűnek az ismételt lenyomására a "*" jel eltűnik. Az ENTER (más géptípusokon RETURN, END LINE) gomb érintése szükségtelen.

A nyomtatókódok táblázata

Az Epson nyomtató kódja	Magyarázat
CHR$(27)"2"	Sortávolság: 6LPI (sor/inch)
CHR$(12)	Lapozás
CHR$(27)CHR$(64)	Normál üzem
CHR$(27)"A"CHR$(0)	Sortávolság: 0
CHR$(27)"P"	Betűtípus: pica (elite helyett)
CHR$(27)"M"	Betűtípus: elite (pica helyett)
CHR$(27)"4"	Betűkép: italic (kurzív)
CHR$(27)"5"	Italic ki
CHR$(27)"W1"	Betűkép: expanded (széthúzott)
CHR$(27)"WO"	Expanded ki
CHR$(15)	Betűkép: compressed (nyomott)
CHR$(18)	Compressed ki
CHR$(27)"S1"	Betűkép: subscript (vékonyított)
CHR$(27)"T"	Subscript ki
CHR$(27)"E"	Betűkép: emphasized (kiemelt)
CHR$(27)"F"	Emphasized ki
CHR$(27)"G"	Betűkép: double strike (kétszeresen fedett)
CHR$(27)"H"	Double strike ki
CHR$(27)"–1"	Betűkép: underlined (aláhúzott)
CHR$(27)"–0"	Underlined ki

```
100 REM ************************
110 REM *                      *
120 REM *    S T Y L I S T     *
130 REM *  FOR EPSON PRINTERS   *
140 REM *                      *
150 REM * THE RAINBOW MAY 1986 *
160 REM *                      *
170 REM ************************
180 A¤=' PRINT STYLIST SELECTION MENU':DV=-2
185 TR¤='':FOR I=1 TO 80:TR¤=TR¤+CHR¤(172):NEXT I:TR¤=TR¤+'**** END OF
 80'
190 CLS
200 PE=PEEK(65314) AND 1
210 IF PE=0 THEN 230
220 PRINTé194,'** PLEASE TURN PRINTER ON **':GOTO 200
230 X=1:POKE 150,1:CLS
240 PX¤='PRINTER IS NOW SET FOR THIS PRINT STYLE'
250 A1¤='*':B¤='9600':BCT=5
260 GOTO 1110
270 REM - LINE INPUT ROUTINE
280 CLS
290 PRINT'       LINE INPUT SCREEN'
300 PRINTé32,'1...5....10...15...20...25...30.'
310 PRINTé96,'..35...40...45...50...55...60...'
320 PRINTé160,'65...70...75...80'
330 PRINTé192,'----------'
340 PRINTé320,'----------CHARACTERS--'
350 PRINTé448,'ENTER '!' FOR PRINT STYLIST MENU';
390 PRINTé224,TR¤
430 PRINTé224,;:LINE INPUT X¤
440 IF X¤='!' THEN 490
450 PRINT#DV,CHR¤(27);'2';X¤
480 GOTO 340
490 CLS
500 REM - PRINTER DRIVER ROUTINE
510 IF SW=1 THEN 1070
520 IF X=32 THEN PRINTé0,A¤
530 PRINTé32,A1¤;'A = PICA          WRITTEN BY'
540 PRINT A2¤;'L = ELITE          GENE SHORT'
550 PRINT A3¤;'I = ITALIC'
560 PRINT A4¤;'X = EXPANDED'
```

```
570 PRINT A5¤;'C = COMPRESSED'
580 PRINT A6¤;'S = SUBSCRIPT'
590 PRINT A7¤;'E = EMPHASIZED'
600 PRINT A8¤;'D = DOUBLE-STRIKE'
610 PRINT A9¤;'U = UNDERLINE'
620 PRINT' R = RESET ALL PRINTER CODES'
630 PRINT' B = BAUD RATE - ';B¤
640 PRINT' P = PRINT STYLE'
645 PRINT' ? = DEMO PROGRAM'
650 PRINT' ! = SINGLE-LINE 'TYPEWRITER''
660 PRINT' Q = QUIT';
670 O¤=INKEY¤
680 IF X=32 THEN 710
690 PRINT@31-X,LEFT¤(A¤,X)
700 X=X+1
710 PE=PEEK(65314) AND 1
720 IF PE=0 THEN 780
730 CLS
740 PRINT@195,'** PRINTER IS OFF LINE **'
750 'PE=PEEK(65314) AND 1
760 IF PE=0 THEN 490
770 GOTO 750
780 IF O¤='Q' THEN 1500
790 IF O¤='!' THEN 280
800 IF O¤='P' THEN 940
810 IF O¤='R' THEN 1110
820 IF O¤='A' THEN 1140
830 IF O¤='L' THEN 1160
840 IF O¤='I' THEN 1180
850 IF O¤='X' THEN 1210
860 IF O¤='C' THEN 1240
870 IF O¤='S' THEN 1270
880 IF O¤='E' THEN 1300
890 IF O¤='D' THEN 1330
900 IF O¤='U' THEN 1360
910 IF O¤='B' THEN 1390
920 IF O¤='?' THEN 960
930 GOTO 670
940 PRINT@DV,CHR¤(27);'2';PX¤
950 GOTO 500
960 SW=1:PX¤='PRINT STYLIST DEMONSTRATION'
970 N=4
980 FOR XY=1 TO 60
990 RN=RND(N)
1000 IF XY=1 THEN RN=3
1010 IF XY=2 THEN RN=4
1020 IF XY=19 AND A5¤='*' THEN RN=4
1030 IF XY=20 THEN N=3:RN=9
1040 IF XY=35 THEN N=4:RN=8
1050 IF XY=45 THEN N=7:RN=5
1060 ON RN GOTO 1180,1210,1270,1240,1360,1300,1330,1140,1160
1070 PRINT#DV,CHR¤(27);'2';PX¤
1080 NEXT XY
1090 PRINT#DV,CHR¤(12)
1100 SW=0:PX¤='PRINTER IS NOW SET FOR THIS PRINT STYLE'
1110 GOSUB 1510:A1¤='*':A2¤=' ':A3¤=' ':A4¤=' ':A5¤=' ':A6¤=' ':A7
¤=' ':A8¤=' ':A9¤=' '
1120 PRINT#DV,CHR¤(27);CHR¤(64);CHR¤(27);'A';CHR¤(0)
1130 GOTO 500
1140 GOSUB 1510:IF A1¤='*' THEN 500
1150 A1¤='*':PRINT#DV,CHR¤(27);'P':A2¤=' ':GOTO 500
1160 GOSUB 1510:IF A2¤='*' THEN 500
1170 A1¤=' ':PRINT#DV,CHR¤(27);'M':A2¤='*':GOTO 500
1180 GOSUB 1510:IF A3¤='*' THEN 1200
1190 A3¤='*':PRINT#DV,CHR¤(27);'4':GOTO 500
1200 A3¤=' ':PRINT#DV,CHR¤(27);'5':GOTO 500
1210 GOSUB 1510:IF A4¤='*' THEN 1230
1220 A4¤='*':PRINT#DV,CHR¤(27);'W1':GOTO 500
1230 A4¤=' ':PRINT#DV,CHR¤(27);'W0':GOTO 500
1240 GOSUB 1510:IF A5¤='*' THEN 1260
1250 A5¤='*':PRINT#DV,CHR¤(15):GOTO 500
1260 A5¤=' ':PRINT#DV,CHR¤(18):GOTO 500
1270 GOSUB 1510:IF A6¤='*' THEN 1290
1280 A6¤='*':PRINT#DV,CHR¤(27);'S1':GOTO 500
1290 A6¤=' ':PRINT#DV,CHR¤(27);'T':GOTO 500
1300 GOSUB 1510:IF A7¤='*' THEN 1320
1310 A7¤='*':PRINT#DV,CHR¤(27);'E':GOTO 500
1320 A7¤=' ':PRINT#DV,CHR¤(27);'F':GOTO 500
1330 GOSUB 1510:IF A8¤='*' THEN 1350
1340 A8¤='*':PRINT#DV,CHR¤(27);'G':GOTO 500
1350 A8¤=' ':PRINT#DV,CHR¤(27);'H':GOTO 500
1360 GOSUB 1510:IF A9¤='*' THEN 1380
1370 A9¤='*':PRINT#DV,CHR¤(27);'-1':GOTO 500
1380 A9¤=' ':PRINT#DV,CHR¤(27);'-0':GOTO 500
1390 GOSUB 1510:BCT=BCT+1:IF BCT>5 THEN BCT=1
1400 ON BCT GOTO 1450,1460,1470,1480,1440
1440 B¤='9600':POKE 150,1:GOTO 500
1450 B¤=' 600':POKE 150,87:GOTO 500
1460 B¤='1200':POKE 150,41:GOTO 500
1470 B¤='2400':POKE 150,18:GOTO 500
1480 B¤='4800':POKE 150,7:GOTO 500
1500 PRINT#DV,CHR¤(27);'2':CLS:END
1510 PRINT#DV,CHR¤(27);'A';CHR¤(0)
1520 RETURN
```

Különböző kombinációk alakulnak több lehetőség egyidejű érvényesítésével. Bármikor ellenőrizhető a gép által, hogy milyen írásképet kértünk valójában: ehhez nyomjuk le a "P" billentyűt ("Print Option" rövidítéseként van a "P" billentyű kijelölve); hatására egysornyi szöveg (PRINTER IS NOW SET FOR THIS PRINT STYLE) jelenik meg. A baud értékének beállítását a nyomtató által megkövetelt értékre a "B" billentyű lenyomásával érhetjük el. Az "R" billentyűvel a nyomtató alapállapotba kerül (RESET). Ezt a két billentyűt is, a "P" billentyűhöz hasonlóan, a hozzájuk rendelt funkció angol neve kezdőbetűinek megfelelően jelölték ki, és természetesen e funkciók esetében is kijelölhető bármely más billentyű. Gépfüggő viszont maga az átviteli sebességet beállító rutin (1390—1480-as sor). Az 1440—1480-as sorokban a B$ sztringből kiolvasható a beállítható baudsebesség; ténylegesen a gép felőli oldalon a 150. (decimális) memóriarekesz átírásával lehet beállítani. A rutin más géptípusokhoz könnyen átírható, de párhuzamos kimenet esetén az egész rutin felesleges.

A menü megjelenésekor azonnal szabad választani, noha a képernyő még nem teljes: a menünek még nincs címe. A cím ui. ("PRINT STYLIST SELECTION MENU"; lásd 180-as sor) csak most kezd jobbról balra „begyalogolni" az ernyőre (lásd 690—700-as sorok!). Ennek a megoldásnak természetesen nincs a program futása szempontjából jelentősége; ki is hagyható — pusztán látványosság. A "Q" billentyű lenyomásával a programból kiszállunk, visszatérünk a BASIC-hez. A programban kiválasztott kód a nyomtató számára továbbra is érvényes. Ha most betöltünk egy szövegszerkesztőt — vagy más, a nyomtatót kezelő programot —, printerünk a kiválasztott variációval fog dolgozni.

A másik képernyő egysoros írógépként működik. A program soronként 80 karakter nyomtatását tételezi fel, a CoCo alfanumerikus képernyője (ahol jelen esetben a program kezelhető) viszont 16 sor, 32 oszlop formátumú. A 80 karakter hosszúságú sor tehát a képernyőn három sort igényel. A problémát áthidalandó a program 300—320-as sorai az oszlopokat beszámozzák. A gépelt szöveg az ernyő közepére kerül egy grafikus karakterekkel kijelölt területre, a számozásnak megfelelően pozicionálva. A szöveg LINE INPUT utasítással a billentyűkről a memóriába jut. Ez az utasítás abban különbözik az INPUT-tól, hogy aktivizálásakor nem jelenik meg kérdőjel az ernyőn, továbbá csak sztringet kezel, viszont bármilyen karaktert elfogad, még idézőjelet is. Ha nem gépelünk be semmit, csak megnyomjuk az ENTER gombot, soremelés és kocsi vissza a következmény.

Ez az egysoros írógép alkalmas nyomtatványok kitöltésére is. Ha például egy nyomtatványon egy bekeretezett részt kell kitölteni, amely az 50. oszlopban kezdődik, 50 szünet után gépeljük be a kívánt szöveget.

A "!" billentyű lenyomásával visszatérhetünk a menühöz, ahogy erről a képernyő állandó tájékoztatást ad. A "?" gombbal a programnak a demonstrációs szakaszát aktivizálhatjuk. Egy teljes oldal jelenik meg különböző betűvariánsokkal. A szöveg minden sorban "PRINT STYLIST DEMONSTRATION" lesz, a betűformák viszont találomra és soronként váltakoznak. A készletből való kiválasztás a véletlenszám-generátorra van bízva, így lehetnek olyan változatok, amelyek egy gépelt oldalon többször is megjelennek, mások viszont egyáltalán nem. Ha több oldalt gépeltetünk, az eredmény mindig más és más lesz.

A CoCo véletlenszám-generátora a következőképpen működik. Ha A = Ø, a generált szám: Ø<n<1; ha A>Ø, a generált szám 1≦n≦A; ha A<Ø, a véletlenszám-generátor új alapról indul. A syntax: RND(A).

A program a felesleges részek, például a demonstrációs rész elhagyásával szubrutinként is csatolható más programokhoz, ahol ilyen jellegű printerkezelésre van szükség. Ha a szubrutinkénti alkalmazás a cél, és az adott géptípus BASIC-jében nincs újraszámozó utasítás, célszerű az egyes sorokat ennek figyelembevételével számozni.

Néhány további útmutató a CoCo BASIC-jének megértéséhez. A PRINT@n, : a szöveget pozicionálja. A 32 × 16 karakteres alfanumerikus képernyő karakterhelyeit Ø—511-ig sorban megszámozták: az n helyére Ø—511 közötti számot írhatunk. Az egyes sorok első karaktere: sorszám (Ø—15) * 32. Az "@" jel rövidítés; jelentése: at (-nál, -nél).

A PRINT#n utasítással a kimenetet választhatjuk meg a kiíratáshoz. n = Ø: képernyőre; n = −1: magnóra; n = 1: lemezre; n = −2: nyomtatóra.

A programban az n helyén a DV változó (device number) áll. Értéke itt konstans: −2. (Lásd a 180-as sorban!)

A nyomtató a svéd ábécét használta, ezért a listában $ helyett Ü és @ helyett É jelent meg.

VÁGH ISTVÁN

Nos, igen! ZX-SPECTRUM – helyi hálózatban

Ha Spectrummal kapcsolatban helyi hálózatot emlegetünk, nem szabad a nagy gépek 10 Mbit/s vagy még gyorsabb, nagy hatótávolságú, mindenféle modemmel támogatott rendszerére gondolnunk. Az Interface 1 (a későbbiekben If—1) lehetőségei ennek csupán egyfajta szimulációját adják, de sokszor ez is nagyon hasznos. A mindössze 32 kbit/s sebességgel (mikrogépeknél azért ez is elég tekintélyes!) és a legfőljebb 5-6 méteres távolsággal is ügyes dolgokat lehet készíteni. Az If—1-be kétféle, egymástól eléggé különböző soros interfészt építettek be. Az egyik a tényleges helyi hálózathoz szükséges LAN (Local Area Network), a másik pedig egy félig-meddig szabályos RS232C.

A Spectrum LAN rendszerbe egyszerre maximum 64 gép léphet. Az adatforgalom általában két, konkrétan kiválasztott gép között megy végbe. Ilyenkor az utasítás kiadása nem jelenti feltétlenül annak végrehajtását; a magnót nem érdekli, hogy a gép vételkész-e. Ha elküldtünk egy írással kapcsolatos parancsot, de a célállomás még nem készült fel az adatok fogadására, mindössze azt látjuk, hogy a gép elveszi a vezérlést: a border besötétül. Alapállapotban feketére vált, de ez igény szerint megváltoztatható. Amikor a másik gép kiad egy ránk vonatkozó olvasóutasítást, a border villogni kezd, és az információ átkerül a másik gépbe.

Az információcsere másik formájában az egyik gép tölti be a főállomás szerepét, a többi pedig mellékállomás. Ilyenkor a főállomás íróutasítást küld, de az információ célját nem jelöli meg konkrétan, hanem állomásszámként Ø-t ad, és a mellákállomások is a Ø-ás géptől várják az adatokat. A főállomás nem várja ki, amíg a többi jelzi vételkészségét: az utasítás a kiadás után rögtön végre is hajtódik. Az információban érdekelt gépeknek tehát már az adás elkezdése előtt késznek kell lenniük az adatok fogadására.

Egy komolyabb kiépítésű hálózatban gyakran előfordul, hogy egyszerre két vagy több gép is bejelenti igényét ugyanarra az állomásra. Ekkor a kisebb állomásszámú gépnek van prioritása. Ebből is látszik, hogy nem érdemes és nem is szabad azonos állomásszámokat használni, mert ebből bonyodalom származhat: általában egyik gép sem kapja meg a vezérlést.

A LAN tehát általános célú, aránylag nagy sebességű kapcsolatot létesít a gépek között. Hibája, hogy az egyik géphez kötött meghajtó vagy más periféria a vezérlő gépe kihagyása nélkül, vagyis közvetlenül nem kezelhető. Bár ez nem is általános LAN funkció, de mivel majdnem annyit érne, mint az egész együtt, és mert egyhamar nem várható hardverfejlesztés Amstardéknál, „boldogult" Sinclairék — még annak idején — tehettek volna az If—1-be egy ilyen célú vezérlőáramkört . . .

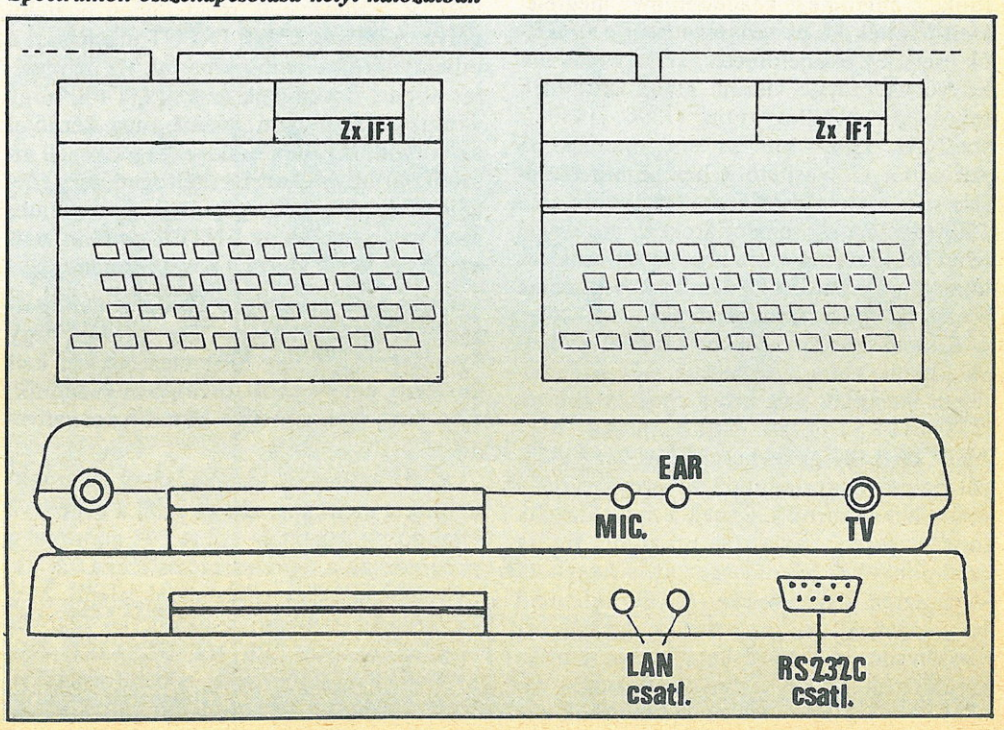
A másik interfész, az RS232C használhatóságában elmarad a LAN lehetőségeitől. Az, hogy nem szabványos, már kizárja az ilyen interfészt használó perifériák többségét: esetleg néhány rendszer és a kevés szabályellenes nyomtató jöhet szóba. Két gép között megvalósulhat információs vonal is, de ez sokkal lassabb és kényelmetlenebb, mint a LAN.

Ezek után nézzük meg utasításaikat! Mivel a LAN rendszerben 64 gép szerepelhet, a kavarodások elkerülésére mindegyikhez hozzá kell rendelni egy állomásszámot: ez 1—64-ig terjedhet. A későbbiekben tárgyalt adatcsere-forgalomban majd mindig ezekre fogunk hivatkozni. Bekapcsoláskor a rendszerben egyetlen gépnek sincs állomásszáma (tulajdonképpen még If—1-e sincs), de ha az If—1-et valamilyen tekintetben érintő utasítást adunk ki, tehát a külső ROM-ot felélesztjük, a gépünk száma 1. lesz. Ha eredetileg nem az 1-es állomásszám volt betervezve, akkor ezt az előzőekben leírtak miatt érdemes megváltoztatni. Az azonosító szám beállítását a FORMAT utasítással hajtjuk végre. Ennek szintaktikája: FORMAT "n"; a. Itt az "n" jelenti a LAN (Network) csatornakódját és az állomásszámot. A pontosvessző helyett If—1 utasításoknál megfelel a vessző is. Sajnos a saját gépünk számát kulcsszóval nem tudjuk lekérdezni, így ha a számot elfelejtettük, akkor a 23749-es címen lévő NET STATION rendszerváltozót kell kiolvasnunk.

A LAN funkciók közül először nézzük a legáltalánosabbat: a programok átvételét. A funkció nagyon hasonlít a kazettás egységgel kapcsolatos utasításokhoz. Amit megtehettünk a magnónál, itt is szabad. A LOAD, SAVE, MERGE, VERIFY kulcsszavak után csillagot kell tennünk, majd "n", vessző vagy pontosvessző, valamint állomásszám következik. Ezek után minden magnóutasítást használhatunk: CODE, SCREEN$, LINE, továbbá az adathalmazra vonatkozó utasításokat mint DATA a() vagy DATA a$(). A vonalon az adatok csak sorosan továbbíthatók, mivel a vezeték sodrott érpárú. Ez hasonlít a kazettás tároláshoz — azzal a különbséggel, hogy míg a magnókezelő rutin csak 8 bites puffert használ (L regiszter), a LAN a továbbítás előtt egy, a memóriában létrehozott 2 kbites puffert tölt fel.

A LAN másik funkciójában adatokat továbbít a célállomás(ok) felé. Ez kicsit bonyolultabb a programátvitelnél. Itt már nem elég kijelölni a NETWORK üzemmódot, ehhez még egy streamet is hozzá kell rendelni (egyszerre többet is lehet). Miután kijelöltük az „n" csatornát és az állomásszámot, megnyitunk egy streamet a LAN számára: OPEN#X; "n"; a., ahol X a stream (0—15), az *a* pedig az adatátvitelben a

Spectrumok összekapcsolása helyi hálózatban



cél- vagy az adóállomás száma, — attól függően, hogy mi vesszük vagy adjuk az adatokat. Ha ez rendben, akkor az X streambe írhatunk: PRINT#X, . . ., illetve onnan olvashatunk: INPUT#X, . . . Az adatok először egy 2 kbites pufferben gyűlnek, és az átvitel csak akkor kezdődik, ha ez megtelt.

Egy stream — hacsak nem zárjuk le és nyitjuk újra — csak egyféle funkcióra alkalmas. Ha először írtunk bele, akkor később nem olvashatunk, és fordítva. Tehát egy stream funkcióját mindig az első utasítás határozza meg. Ha mégis kiadunk funkcióval ellentétes utasítást, akkor »READING TO A „WRITE FILE"«, vagy »WRITING TO A „READ FILE"« hibaüzenetet kapunk. Ha a streambe írtunk, akkor dolgunk végeztével feltétlenül zárjuk le a csatornát: CLOSE#X. Ha ezt nem teszszük meg, akkor a vevő továbbra is adatokra fog várni.

A másik soros átviteli lehetőség az RS232C. Ennek a csatlakozója az If—1 hátoldalán a LAN aljzatoktól balra található. Az If—1-féle RS232C-nek két nagy hibája van. Az egyiken kis barkácsolással lehet segíteni: a szabványos RS232C csatlakozó egy 25 pólusú D, az If—1-beli viszont a botkormánynál megismert 9 pólusú. A másik probléma már kicsit bonyolultabb: a szabvány a logikai 0 szintnek +15 V-ot, a logikai 1 szintnek −15 V-ot ír elő, míg az If—1 a TTL szinteket használja. Ezen csak körülményesen, szintváltóval lehet segíteni. Az adatátvitel itt is soros formátumú, de még maximális sebességnél is sokkal lassabb, mint a LAN, mert a nyolc adatbitet egy startbit előzi meg, és 2 stopbit zárja.

Az RS232C átvitelnél a LAN-hoz hasonlóan itt is először egy FORMAT utasítást kell kiadni: FORMAT "", v. Az idézőjelbe itt is a csatornakód kerül; v a sebesség baudban. Az If—1 csak kilencféle sebességet ismer, így az általunk adott adatot mindig kerekíti. Csatornakódként a *t* és a *b* betűt választhatjuk a következők szerint.

A *t* (TEXT) egy szövegcsatornát jelöl ki. Ez az üzem nyomtató használatakor előnyös lehet, de egyébként bonyodalmas, mivel a nem nyomtatható karaktereket és a vezérlőkódokat el sem küldi.

A *b* bináris csatorna nem vizsgálja az információt, így minden adat eljut a vevő géphez. Ezt a csatornát általános célú adatforgalomra jelöljük ki. Ha programot szeretnénk egyik gépből a másikba továbbítani, akkor gépünket a *b* csatornára állítsuk, majd az utasításokat a LAN-hoz hasonlóan adjuk ki. Például:

LOAD*"n" CODE helyett LOAD*"b" CODE.

Az RS232C-n is mehet tisztán adatforgalom, de ennek formátuma szintén megegyezik a LAN-éval.

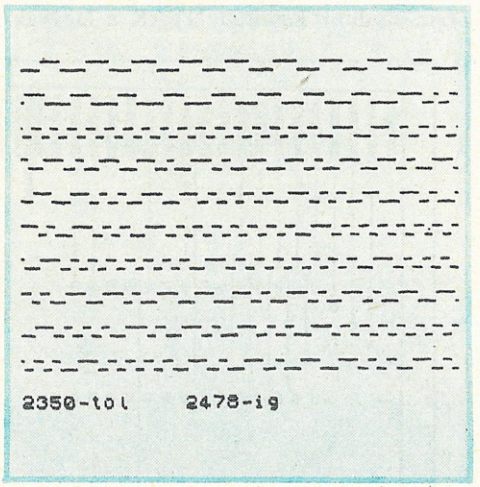
VARGA ISTVÁN

ZX-SPECTRUM

mint digitális jeleket tároló oszcilloszkóp

Előfordulhat, hogy ismeretlen digitális jelsorozatot oszcilloszkópon kell elemeznünk. Ha a vizsgálandó jel nem periodikus, tároló oszcilloszkópra van szükség. A magnetofonbemenet segítségével egy csatorna a hardver *átalakítása nélkül* vizsgálható ZX-Spectrumon. Az *1. programmal* 8 másodpercnyi jelet tárolhatunk a memóriában 50 µs felbontással. Ez azt jelenti, hogy a vizsgálandó jel néhány kHz frekvenciájú lehet.

A program a memóriában bárhol működik, de célszerű annak felső részében elhelyezni. A vizsgált jel a memóriában bitfolytonosan helyezkedik el a 40000-től 59999-ig terjedő tartományban.



A ZX-Spectrum magnójele

A kezdő cím a HL regiszterpárba kerül, a DE-be pedig a feltöltendő bájtok számánál 255-tel nagyobb szám. Ha ezeket az adatokat meg akarjuk változtatni, vegyük figyelembe, hogy a program csak a 32768 cím fölött működik helyes időzítéssel, alatta az ULA prioritása miatt értékelhetetlen eredményt kapunk. A programban levő két NOP utasítás biztosítja, hogy minden bit feltöltése ugyanannyi időközönként történjék, függetlenül attól, hogy a bájt első vagy közbenső bitje. A program egy körülfordulása 173 órajelnyi. Ez 3,5 MHz órajel esetén 49,43 µs; 1,2 százalékkal kevesebb, mint 50. Ez a hiba általában elhanyagolható.

A megjelenítésre a *2. program* szolgál. Egy ernyőre 10 sorban 128 ms-nyi jelet rajzol fel, amelyet nyomtatóra is kivihetünk. A 8 másodpercnyi felvételnek általában csak egy részletére vagyunk kíváncsiak. Ezt az ernyőn előrelapozással megkereshetjük, és az érdekes időtartományokat feljegyezzük. A programot leállítjuk és GO TO 10-zel újraindítjuk. Így tetszőleges időpillanattól kezdhetünk. Új adatokat GO TO 2-vel olvastathatunk be.

```
FF78            0010        ORG  65400
FF78 F3         0020        DI
FF79 21409C     0030        LD   HL,9C40H
FF7C 111F4F     0040        LD   DE,4F1FH
FF7F 0E06       0050        LD   C,6
FF81 0608       0060 NAGY   LD   B,8
FF83 C5         0070 KIS    PUSH BC
FF84 41         0080        LD   B,C
FF85 10FE       0090        DJNZ -2
FF87 DBFE       0100        IN   A,(254)
FF89 CB07       0110        RLC  A
FF8B CB07       0120        RLC  A
FF8D CB16       0130        RL   (HL)
FF8F C1         0140        POP  BC
FF90 0E06       0150        LD   C,6
FF92 10EF       0160        DJNZ KIS
FF94 23         0170        INC  HL
FF95 1B         0180        DEC  DE
FF96 0E02       0190        LD   C,2
FF98 3E00       0200        LD   A,0
FF9A B2         0210        OR   D
FF9B 00         0220        NOP
FF9C 00         0230        NOP
FF9D 20E2       0240        JR   NZ,NAGY
FF9F FB         0250        EI
FFA0 C9         0260        RET
                0270        END

KIS     FF83
NAGY    FF81
#       5DCE
```

1. program

```
   1 CLEAR 39999: LOAD "SZKOP"CO
DE 65400
   2 CLS : PRINT "Barmely gomb i
ndituja a felvetelt": PAUSE .5
   3 IF INKEY$="" THEN GO TO 3
   5 RANDOMIZE USR 65400: LET e=
0: GO TO 15
  10 INPUT "Kezdet, ms ";e
  15 LET i=INT (40000+e*2.5)
  20 CLS : PRINT AT 20,0;(i-4000
0)*.4;"-tol"
  30 FOR k=16384 TO 16576 STEP 6
4: GO SUB 100: NEXT k
  40 FOR k=18432 TO 18624 STEP 6
4: GO SUB 100: NEXT k
  50 FOR k=20480 TO 20576 STEP 6
4: GO SUB 100: NEXT k
  55 PRINT AT 20,12;(i-40000)*.4
;"-ig"
  60 LET a$=INKEY$: IF a$="" THE
N GO TO 60
  62 IF a$="z" THEN COPY
  70 IF i<60000 THEN GO TO 20
  75 STOP
 100 FOR j=0 TO 31: LET c=PEEK i
: POKE k+j,c: POKE k+j+1024,255-
c: LET i=i+1: NEXT j: RETURN
```

2. program

Az *ábrán* a ZX-Spectrum magnójele látható. Az első és a második sor a bevezető jel, a második sor végén fejléc kezdődik.

A gépi kódú program decimális listáját DATA sorokban elhelyezve, a programot bevihetjük a memória megfelelő helyére:

243,33,64,156,17,31,79,14,6,6,8,197,65,16,
254,219,254,
203,7,203,7,203,22,193,14,6,16,239,35,27,
14,2,62,0,
178,0,0,32,226,251,201

DR. MAROSSY KÁLMÁN

Külső ROM C64-hez

Ezzel a cikkel azoknak szeretnénk segíteni, akik programjaikat a gyári kártyákhoz hasonló külső ROM-ban akarják tárolni.

A külső ROM-kártya a gép amúgy is szűkös RAM-jából 8 kbájtot leválaszt. Két lehetőségünk van: vagy a teljes 8 kbájt RAM-ot helyettesítjük ROM-mal, vagy csak egy kis részét, ami a tulajdonképpen felhasználandó program szabad területre másolásához szükséges. Ez akár a kártya által kiválasztott RAM is lehet. Az átmásolás után a kártya lekapcsolható.

Az első esetben a program a leválasztott RAM helyén a ROM-ban futhat. A most szóban forgó kártya a BASIC interpreter alatt levő 8 kbájtot használja. Ez megegyezik a szabad BASIC munkaterület tetejével. A maximális programhossz tehát szintén ennyi.

A kártya megépítésénél szinte bármilyen EPROM, PROM szóba jöhet. A gép hátulján a „cartridge port" vezetékei között találhatjuk meg azokat a vonalakat, amelyek az EPROM megcímzését, engedélyezését és olvasását teszik lehetővé *(1. ábra)*. A címzésre és az adatok fogadására az ábrán látható vonalakra van szükség.

Az EPROM szelektálására a $\overline{\text{ROML}}$ vonalat használjuk, mely $8000-től $BFFF-ig aktív Ø, ha közben az EXROM vonalat földre kötöttük. A lekötést kapcsolóval valósítjuk meg.

Ha megcímeztük a kártyát, és a $\overline{\text{CS}}$ jelet is kiküldtük, akkor időzítési problémák nem léphetnek fel, ezért az $\overline{\text{OE}}$-t földre kötjük.

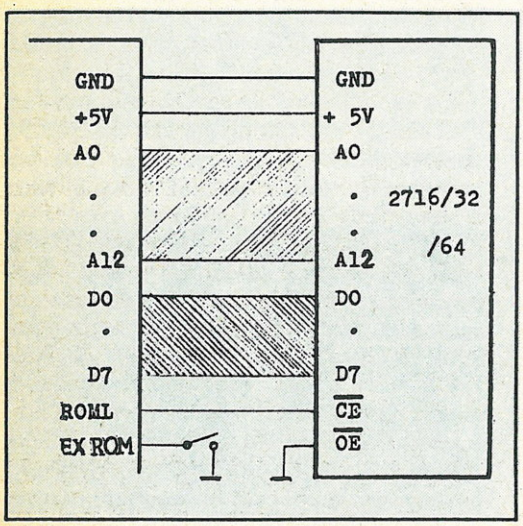
A kártyacsatlakozó sora a szabványos 0,1 inch osztású kell hogy legyen. (A nyomdai rajz itt ettől méretben eltérhet!)

Az általunk készített NYÁK a leggyakrabban használt EPROM-ok (2716, 2716 A, 2732 A és ezek CMOS változatai: 2516, 2732, 2764) befogadására képes *(2. a, b, c ábrák)*. A NYÁK-ot a *3. ábrán* mutatjuk be.

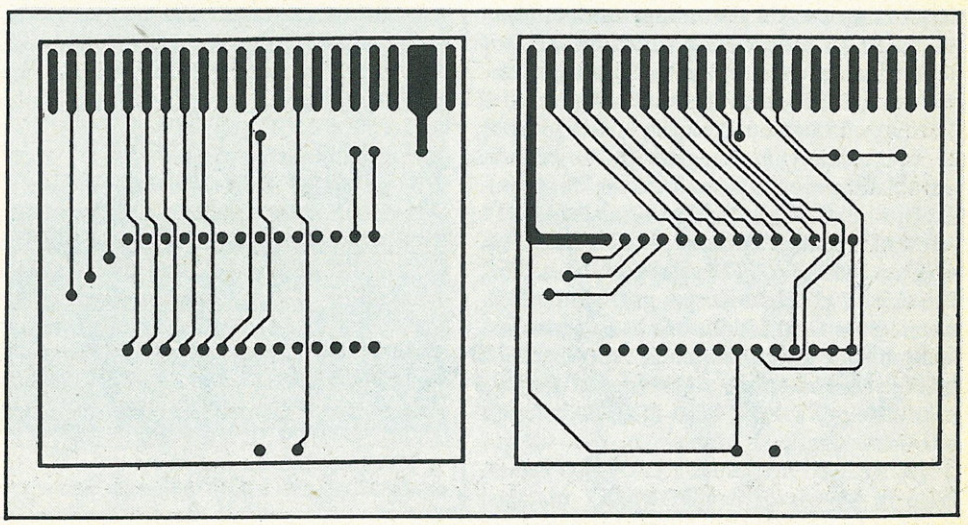
A kártyát működtető szoftverrel kapcsolatban néhány megjegyzés. Ha olyan kártya készítése a cél, amely a gép bekapcsolása után azonnal, automatikusan indul, akkor vegyük figyelembe, hogy a gép reset rutinja vizsgálja $8000-től a tárat. Ha itt a CBM 80 emelt karakterek ASCII kódjai találhatók ($ C3, $ C2, $ CD, $ 38, $ 30), akkor a programfutás a ($8000—$8001) címre ugrik. A másik két bájt az NMI vektora (RUN STOP/RESTORE).

A gép tulajdonképpen még inicializálandó; ez legegyszerűbben a KERNAL reset rutinjának némi módosításával érhető el.

MARTOS
Számítástechnika Kör Zalaegerszeg

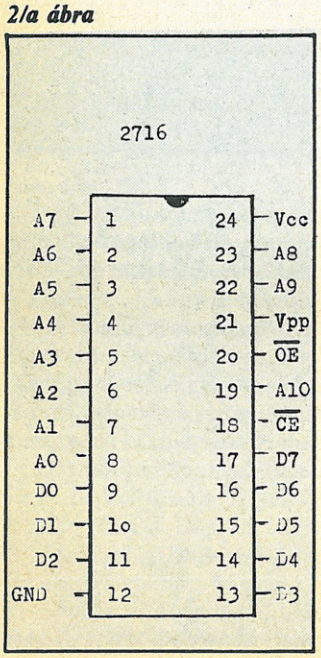


1. ábra

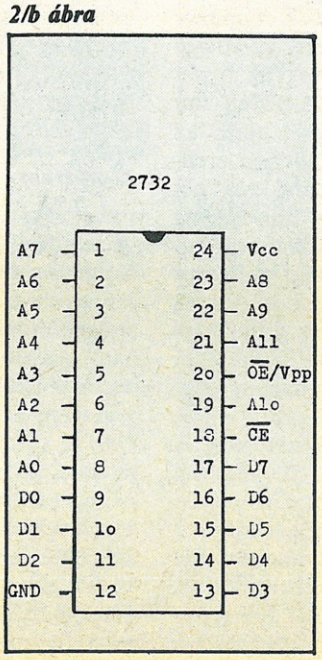


3. ábra

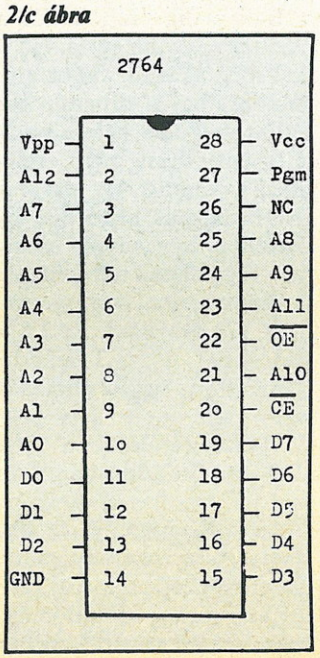
2/a ábra



2/b ábra



2/c ábra



A VC20 hardver adta lehetőségei

Hanggenerálás

A 6561-es chip öt regiszterének használatával lehet hangadásra kényszeríteni a VC20-as gépet. Ezek a regiszterek a következők.

36874-es tárcím: audiooszcillátor 1. (mély): a 0—6. bitek a frekvenciát, a 7. bit pedig a be- (=1), illetve a kikapcsolt állapotot jelzi.

36875-ös tárcím: audiooszcillátor 2. (közepes): bitfelosztás a 36874-es tárcímnél.

36876-os tárcím: audiooszcillátor 3. (magas): bitfelosztás a 36874-es tárcímnél.

36877-es tárcím: zajgenerátor (fehérzaj): a zaj milyenségét határozza meg, a 7. bit mint az előzőknél.

36878-as tárcím: az előző négy regiszter hangerejét határozza meg. Értéke 0—15 lehet. Többszínű üzemmódban (lásd itt később) vigyázni kell arra, hogy csak és csakis az alsó 4 bit tartalmát változtassuk meg.

A kiválasztott hang megszólaltatásához az alábbi értékek valamelyikét kell az első három regiszter egyikébe beírni, természetesen a hangerő és a zaj értékének meghatározása után.

OKTÁVOK:	1.	2.	3.	4.
HANG:				
C	135	195	225	240
C#	143	199	227	241
D	147	201	228	
D#	151	203	229	
E	159	207	231	
F	163	209	232	
F#	167	212	233	
G	175	215	235	
G#	179	217	236	
A	183	219	237	
A#	187	221	238	
H	191	223	239	

Többszínű üzemmód

A képernyő egy 8 × 8-as méretű karakterhelyén megjelenő pontok kétszínű üzemmódban vagy háttérszínűek (nem látszanak), vagy a színmemóriában levő kódnak megfelelő színűek. Többszínű üzemmódban összesen négy szín lehet egy karakterhelyen belül, a háttérszínnel együtt.

A többszínű üzemmód bekapcsolása: a színmemóriában a karakter helyének megfelelő bájt 3. bitjét magasra állítjuk. A színmemória alapkiépítésben a 38400—38884 címeken található, sorfolytonosan. Ha bővitve van a gép, akkor a 37888—38399 címeken lesz a színmemória. Többszínű üzemmód esetén a karakter alakját definiáló biteket párosával kell tekintenünk, és a két bit helyén a bitpár értékének megfelelően két azonos színű pont fog megjelenni a képernyőn. A bitpárok a következő színeket tartalmazzák:

00 = hátérszín: a 36879-es cím felső 4 bitje
01 = keretszín: a 36879-es cím alsó 3 bitje
11 = segédszín: a 36878-as cím felső 4 bitje
10 = a színmemóriában levő szín (alsó 3 bit)

Így például az „A" betű alakjai így néznek ki:

két színnel:

```
  XX
 XXXX
XX  XX
XXXXXX
XX  XX
XX  XX
XX  XX
```

több színnel:

```
DDAABBDD
DDCCCCDD
AABBAABB
AACCCCBB
AABBAABB
AABBAABB
AABBAABB
DDDDDDDD
```

bittérkép:

```
00011000
00111100
01100110
01111110
01100110
01100110
01100110
00000000
```

A többszínű módnál az azonos színű pontokat egyféle betűk jelölik.

Grafikus lehetőségek

Mivel a grafika a képernyő és a karaktergenerátor áthelyezésén alapszik, először ezek áthelyezésének problémáit kell tisztázni.

A képernyő áthelyezésére a következő utasítássort kell használni:

```
POKE 36869, (PEEK(36869)AND15) OR X:
POKE 36866, (PEEK(36866)AND127) OR 128*Y
```

ahol X és Y értéke a következőképpen alakulhat:

X	Y	A képernyőmemória kezdőcíme
128	0	0
128	1	512
144	0	1024
144	1	1536
160	0	2048
160	1	2560
176	0	3072
176	1	3584
192	0	4096
192	1	4608
208	0	5120
208	1	5632
224	0	6144
224	1	6656
240	0	7168
240	1	7690

A karaktermemória áthelyezésére a következő utasítást kell használni:

```
POKE 36869, (PEEK(36869)AND240) OR X
```

ahol X értéke a következő lehet:

X	A karaktermemória helye
4	4096
5	5120
6	6144
7	7168
8	8192
9	9216
10	10240
11	11264
12	12288
13	13312
14	14336
15	15360

Ha X értéke 0, 1, 2 vagy 3, akkor a gép a ROM-ból veszi a karaktereket. A nagyfelbontású grafikánál ugyanazokat az elveket használjuk, mint a felhasználói karakterek definiálásánál. A kapcsolat a VIDEO-RAM és a karaktermemória között a következő. Ha a karaktermemóriában minden bit nullára van állítva, akkor a képernyőn sincs egy pont sem kivilágítva. A nagyfelbontású grafika eléréséhez a VIDEO-RAM-ba betöltjük mind a 256 karakter kódját, majd a karaktergenerátor kezdőcímét egyenlővé tesszük a VIDEO-RAM kezdőcímével. Mivel 1-1 bit helyének meghatározása eléggé bonyolult, ezért minden további magyarázat nélkül leírok egy rövid grafikai programot.

```
 1 PRINT "SHIFT+CLR/HOME"
 2 POKE 36867,128
 3 POKE 36865,60
 4 F(8)=0:F(0)=128:F(1)=
   64:F(2)=32:F(3)=16
 5 F(4)=8:F(5)=4:F(6)=2:F(7)=1
 6 FORQ=0 TO 255
 7 POKE 7680+0,Q
 8 POKE 38400+Q,2
 9 NEXT Q
10 FOR Q=5120 TO 7160
11 POKE Q,0
12 NEXT Q
13 POKE 36869,253
14 POKE 36866, PEEK(36866) OR 128
15 POKE 36867,150
16 FOR C=0 TO 175
17 L=45+40*SIN (C/10)
18 A=5120
19 LR=L/8
20 LA=INT (LR)
21 A=A+(LA*176)
22 LR=(LR-LA)*8
```

THE USERS PORT

San Fernando Valley Commodore Users Group

Menüprogram

Amerikai utamon ígéretet kaptam a San Fernando Valley Commodore Users Group (Los Angeles) nevű klub vezetőjétől, David L. Oseas úrtól, hogy The Users Port című lapjukat küldeni fogják, és abból cikkeket vehetünk át. Ez meg is történt, így ezután rendszeresen közlünk ebből a lapból átvett cikkeket, rövid programokat. (SIMONYI ENDRE)

Sokakat érdekelhet, hogyan lehet rövid menüprogramból hosszabb programokat behívni. Ezt nemcsak BASIC, hanem ZOOM PASCAL programokból is megtehetjük.

Az *1.lista* az újság eredeti programja, magyar nyelvű üzenetekkel. A továbbiak ennek az elgondolásnak alapján készült saját programok.

A *2. lista* egy BASIC menüprogram, az előzőnek alkalmazására. Azt mutatja be, hogyan lehet tetszőleges hosszúságú BASIC programokat változó felhasználásával behívni. BASIC programok esetén törli a változó listát.

A *3. listán* az első program ZOOM PASCAL változata látható, azzal a módosítással, hogy megkérdezi a behívandó program nevét.

PÁL MAGDA

1. lista

```
10 PRINT"■■■■■"
15 REM KEPERNYO TORLES 5-SZOR CURSOR LE
20 PRINT"LOAD"CHR$(34)"PEN."CHR$(34)",8"
25 REM LOAD PARANCS KEPERNYORE IRASA
30 PRINT"TTT"
35 REM 4-SZER CURSOR FEL
40 POKE 631,13
45 REM A BEIRT <RETURN> BETOLTI A
46 REM PROGRAMOT
50 POKE 632,ASC("R")
60 POKE 633,ASC("U")
70 POKE 634,ASC("N")
75 REM AZ 50-70-IG TERJEDO SOROK BEIRJAK
76 REM A 'RUN' PARANCSOT A BILLENTYUZET
77 REM BUFFERBE
80 POKE 635,13
85 REM <RETURN>
90 POKE 198,5
95 REM KOZLI A GEPPEL,HOGY 5 KARAKTER
96 REM VAN A BILLENTYUZET BUFFERBEN
```

2. lista

```
5 CLR
10 PRINT"■"
15 PRINT"■"
20 PRINT"AZ ALABBI VALOSZINUSEG"
21 PRINT" SZAMITASI                "
30 PRINT"FOGALMAKRA VONATKOZO FELADATOK KOZUL
40 PRINT"LEHET VALASZTANI
50 PRINT"KLASSZIKUS FELADATOK:"
60 PRINT"1. 2. 3. 4. 5."
65 PRINT
70 PRINT"FUGGETLENSEG ES FELTETEL "
80 PRINT"6. 7. 8. 9."
85 PRINT
90 PRINT"VALOSZINUSEGI VALTOZO "
95 PRINT"VARHATOERTEK ES"
100 PRINT"SZORAS:"
110 PRINT"10. 11. 12. 13. 14. 15."
115 PRINT
120 PRINT"NEVEZETES ELOSZLASUK:"
130 PRINT"16. 17. 18. 19."
135 PRINT
140 PRINT"EGYUTTES ELOSZLASOK:"
150 PRINT"20. 21. 22. 23."
155 PRINT
160 PRINT"HATARELOSZLAS TETEL: 24."
165 PRINT
170 PRINT"REGRESSZIO: 25."
175 PRINT
180 PRINT"■";"A KERT FELADAT SORSZAMAT";
181 PRINT"■";" KEREM";
190 INPUT V$
200 IF VAL(V$)<1ORVAL(V$)>25 THEN GOTO10
210 N$="PELDA"+V$
211 PRINT"■■■■■"
220 PRINT"LOAD"CHR$(34)N$CHR$(34)",8"
221 PRINT"TTT"
222 POKE 631,13
223 POKE 632,ASC("R")
224 POKE 633,ASC("U")
225 POKE 634,ASC("N")
226 POKE 635,13
227 POKE 198,5
230 END
```

3. lista

```
PROGRAM HIVO;
 VAR NEV:STRING;
     CH:CHAR;
     L,J:INTEGER;
  BEGIN
  REPEAT
   WRITE('A BETOLTENDO PROGRAM NEVE:');
   READLN(NEV);
   L:=LENGTH(NEV);
  UNTIL L>2;
  IF NEV[2]<>':' THEN BEGIN
   WRITE('EGYSEG SZAM?:');
   READLN(CH);
   NEV:=CONCAT(CH,':',NEV);END;
   WRITELN(CHR(147),CHR(17),CHR(17),
   CHR(17),CHR(17),CHR(17),CHR(17));
   WRITELN('LOAD',CHR(34),NEV,CHR(34),
   ',8');
   WRITELN(CHR(145),CHR(145),CHR(145),
   CHR(145));
   MEM[$0277]:=CHR(13);
   MEM[$0278]:='R';
   MEM[$0279]:='U';
   MEM[$027A]:='N';
   MEM[$027B]:=CHR(13);
   MEM[$00C6]:=CHR(5);
  END.
```

23 CR=C/8
24 CA=INT(CR)
25 A=A+(CA*8)
26 A=A+LR
27 CR=INT((CR—CA)*8)
28 POKEA, PEEK(A) OR F(CR)
29 NEXT C
30 GET A$:IF A$=" " THEN 30
31 POKE 36869,240
32 POKE 36866,150
33 POKE 36867,174
34 POKE 36865,38

Amint látható, a program két részből áll: egy inicializáló és egy rajzoló részből. Az inicializáló rész átállitja a VIDEO-RAM, a karaktergenerátor kezdőcímét, valamint a képernyő méreteit és a szín RAM értékeit. Ez a rész az 1—15. sorokban van. A program lelkét a 18—28. sorok képezik, ugyanis itt van az a programrészlet, amely a C-edik oszlop L-edik sor koordinátájú pontot kivilágítja. A program utolsó sorai (30—34. sor) a képernyő- és a színmemória normális értékeit állítják vissza egy tetszőleges billentyű lenyomása után.

A rajzolás tehát a VC20-ason elég bonyolult és lassú, de gyakorlással, valamint gépi kódú programrészletek alkalmazásával ezen lehet segíteni.

A VIC chip 16 regiszterének lehetőségei

0. regiszter 36864-es tárcím: A bal felső karakter első pixeloszlopának távolságát állítja be a képernyő bal szélétől.
1. regiszter 36865-ös tárcím: A bal felső karakter felső pixelsorának távolságát állítja a képernyő tetejétől.
2. regiszter 36866-os tárcím: A 0—6. bitek az egy sorban levő karakterek számát mutatják, a 7. bitnek a képernyőmemória helyének meghatározásában van szerepe.
3. regiszter 36867-es tárcím: Ha a 0. bit=1, akkor 8×16-os karakterrel dolgozik a gép, egyébként normál, 8×8-as karakterméretet használ. A 7. bit a raszterszámláló legkisebb helyiértékű bitje. Az 1—6. bitek a képernyő oszlopszámának kétszeresét adják.
4. regiszter 36868-as tárcím: Kiegészítve a 3. regiszter 7. bitjével, az utoljára frissített rasztersor számát adja.
5. regiszter 36869-es tárcím: A VIDEO-RAM és a karaktergenerátor kezdőcíme.
6. regiszter 36870-es tárcím: Fényceruza vízszintes koordinátája.
7. regiszter 36871-es tárcím: Fényceruza függőleges koordinátája.
8. regiszter 36872-es tárcím: Játék I/0 X potméter érték.
9. regiszter 36873-as tárcím: Játék I/0 Y potméter érték.
10—14. regiszterek 36874—36878-as tárcímek: Lásd a hanggenerálásnál, a cikk elején.
15. regiszter 36879-es tárcím: A felső 4 bit a háttérszínkódot, az alsó 3 bit a keretszínkódot tartalmazza. Ha a 3. bit értéke 1, akkor a kiíratás inverz, ellenkező esetben normális. A háttér, a kertszínek, valamint az invertálás megválasztása:

POKE 36869,16*háttérszínérték+8*invertálási mód+keretszínérték.

Az ismertetés Nick Hampshire: VIC Revealed c. könyve alapján készült.

(td)

KI AD MAGYARÁZATOT?

Feladatom volt, hogy rajzoljak a DRAGON számítógép képernyőjére egy bogarat és nyomtassam ki, a nyomtatónak azt a lehetőségét kihasználva, hogy egy sorba 1920 pontot képes rajzolni.

A DRAGON képernyőjén 256 × 192 pont jeleníthető meg. A képernyőn elő is állítottam, azonban a papíron más jelent meg. Mivel a nyomtatás nagyon lassan haladt, módosítottam a programot.

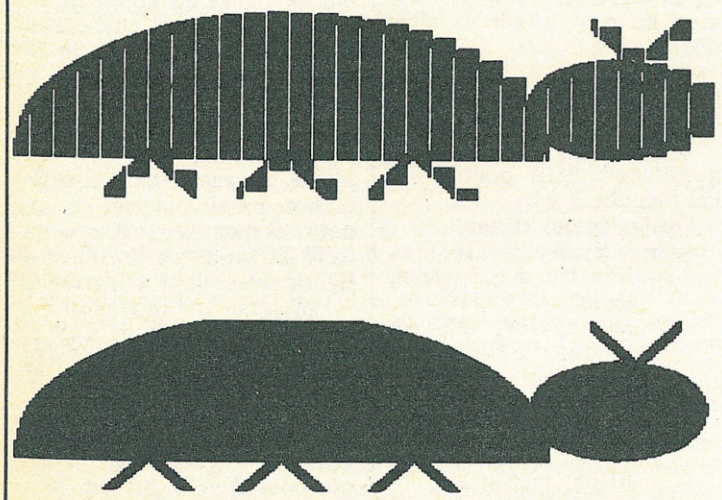
Mit változtattam? Kihagytam az X változót; így elmaradt több mint 5000 értékadás. Kiemeltem a második ciklusból a szorzást a ciklus elé: így elmaradt több mint 5000 szorzás. A hatványozás helyett osztottam: így elmaradt több mint 80 000 hatványozás, helyette lett több mint 40 000 osztás. Végül a rajz feletti és alatti üres területet elhagytam: így a rajzterület kb. 40 százalékkal csökkent. Meglepetésemre nemcsak a sebessége nőtt meg, hanem a nyomtatott rajz is megváltozott. Miért?

Az 1. lista és ábra az első, a 2. a második változat. A PRINT utasítások mind a nyomtatóra küldtek adatokat, az ESC Q az 1920 pont/sor beállító, ESC T Ø1 a sorközt minimumra állítja, az ESC W ØØØ8 az utána következő 16 bit által megadott 16 függőleges pontot nyolcszor megismétli. Végül a 13 és 10 a kocsivissza/soremelés. Ezek sem változtak. A DRAGON Microsoft—Basic változatot használ.

A kinyomtatás képernyőpontonként, azaz bitenként történt, ehhez bájtonként kiolvastattam a képernyőtartalmat, és a bájtot bitekre bontottam — először a hatványozással, majd az osztással. Ezután minden bitértéket nyolcszorozva nyomtattam ki, így az ábrát nyolcszorosára felnagyítottam.

Ellenőrizve mind a számítógép, mind a nyomtató hibátlanul működött.

SIMONYI ZSUZSA



```
10 PMODE4,1:PCLS:SCREEN1,1
20 CIRCLE(100,126),90,,1,.5,1:CIRCLE(214
,96),30:CIRCLE(220,84),6,,1,.5,1
30 LINE(10,126)-(190,126),PSET:LINE(214,
84)-(226,84),PSET:LINE(40,152)-(55,126),
PSET:LINE-(70,152),PSET:LINE(85,152)-(10
0,126),PSET:LINE-(115,152),PSET:LINE(130
,152)-(145,126),PSET:LINE-(160,152),PSET
:LINE(205,40)-(220,66),PSET:LINE-(235,40
),PSET
40 PAINT(214,96):PAINT(100,100)
50 PRINT#-2,CHR¤(24);CHR¤(27);"Q";CHR¤(2
7);"T01";:A=1536:FOR I=0TO 191:FOR J=1 T
O 30:X=PEEK(32*I+J+A):Y(0)=X:FOR K=1 TO
8:B=(2Ü(8-K))*INT(Y(K-1)/(2Ü(8-K))):Y(K)
=Y(K-1)-B:PRINT#-2,CHR¤(27);"W0008";
60 IF B<>0 THEN PRINT#-2,CHR¤(255);CHR¤(
255);:GOTO70 ELSE PRINT#-2,CHR¤(0);CHR¤(
0);:GOTO70
70 NEXT K:NEXT J:PRINT#-2,CHR¤(13);CHR¤(
10);:NEXT I:STOP
```

```
10 PMODE4,1:PCLS:SCREEN1,1
20 CIRCLE(100,126),90,,1,.5,1:CIRCLE(214
,96),30:CIRCLE(220,84),6,,1,.5,1
30 LINE(10,126)-(190,126),PSET:LINE(214,
84)-(226,84),PSET:LINE(40,152)-(55,126),
PSET:LINE-(70,152),PSET:LINE(85,152)-(10
0,126),PSET:LINE-(115,152),PSET:LINE(130
,152)-(145,126),PSET:LINE-(160,152),PSET
:LINE(205,40)-(220,66),PSET:LINE-(235,40
),PSET
40 PAINT(214,96):PAINT(100,100)
50 PRINT#-2,CHR¤(24);CHR¤(27);"Q";CHR¤(2
7);"T01";:A=1536:FOR I=36TO 152:D=32*I+A
:FOR J=1 TO 30:Y(0)=PEEK(J+D):C=256:FOR
K=1 TO 8:C=C/2:B=C*INT(Y(K-1)/C):Y(K)=Y(
K-1)-B:PRINT#-2,CHR¤(27);"W0008";
60 IF B<>0 THEN PRINT#-2,CHR¤(255);CHR¤(
255);:GOTO70 ELSE PRINT#-2,CHR¤(0);CHR¤(
0);:GOTO70
70 NEXT K:NEXT J:PRINT#-2,CHR¤(13);CHR¤(
10);:NEXT I:STOP
```

BITEK ÉS FIGURÁK

Dinamikus értékelőfüggvény

A dinamikus értékelés jelentősége

Mint tudjuk, amíg a sakkozó megtesz egy lépést, a változatok sokaságát gondolja át, értékeli, és ezek alapján dönt, hogy melyik lépést válassza az illető állásban. A számítógép az állás értékelésénél tulajdonképpen ugyanezt a műveletet végzi gépi eszközökkel. Ezt a minimax algoritmus ismertetésénél majd részletesen tárgyaljuk. Egyelőre csak azt kell tudnunk, hogy a gép is változatokat számít — hasonlóan az emberhez —, de nagy különbség, hogy a gépnek nincsen tapasztaláson alapuló intuíciós képessége, szemben a sakkozókkal. Mivel az eddig tárgyalt statikus értékelőfüggvény nem lehet tökéletes, ezért az állásértékelés finomítása miatt alkalmazzuk az úgynevezett dinamikus értékelőfüggvényt.

A dinamikus értékelőfüggvény a statikus értékeléssel szemben nem az előre számolt változat végpontjában kialakuló állást értékeli, hanem azokat a lépéseket, amelyekkel az az állás létrejött. Vagyis a végállás elemzésétől teljesen függetlenül, csak a kezdőállás függvényében a végállásig megtett lépéseket veszi egyenként figyelembe.

Egyes hadállásokban többféle nem egyenértékű lépéssorozattal is eljuthatunk egy adott végálláshoz. Nézzük például az *1. ábrát*. Világos a bemutatott állásból látszólag kétféle úton érheti el a mattot.
Az első — kényszerítő erejű — változat:

1. Vd2×g5+ Kg7–h7
2. Bf3–h3 (Hd7–f6)
3. Vg5×h6 matt.

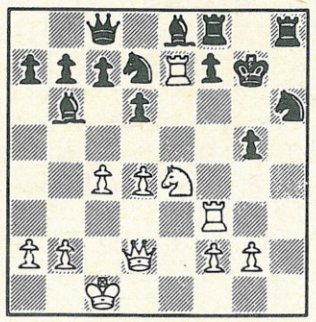
A második — nem kényszerítő erejű — lehetséges út:

1. Bf3–h3 (Hd7–f6)
2. Vd2×g5+ Kg7–h7
3. Vg5×h6 matt.

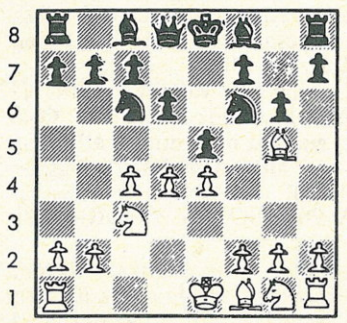
Láthatjuk, hogy kétféle úton is eljuthatunk a mattig, de valóságos partiban csak az első esetben tudja világos bemattolni sötétet, ugyanis 1.Bd1–h1 lépésre sötét növeli királya szabadsági fokát, azaz mozgékonyságát a Bf8–g8 lépéssel. Így királya az f8 mezőn el tud bújni a matt elől, és anyagi előnyének érvényesítésével megnyeri a játszmát:

1. Bf3–h3? Bf8–g8
2. Vd2×g5+ K7–f8!

Tehát a dinamikus értékelőfüggvénynek külön értékelnie kell a forszírt lépéseket az aktív és a passzív fél szemszögéből. Ilyen esetben megtörténhet, hogy hiába azonos a kezdő- és a végállás, az eltérő lépéssorozat miatt a dinamikus értékelőfüggvény más és más értékeket mutat.



1. ábra



2. ábra

A dinamikus értékelés hátránya, hogy dinamikus értékelőfüggvényünk közel egyenértékű lépéssorozatok esetében is nagyon különböző értékeket mutathat.

Felvetődhet a kérdés, hogy ezek szerint nem a végeredmény az érdekes? De igen. A dinamikus értékelőfüggvény használata ennek ellenére szükséges, mert statisztikusan sok hibás lépés kiszűrésére képes.

Milyen legyen a dinamikus értékelőfüggvény?

E függvényben a lépésszámtól, a táblán lévő figurák számától és anyagi értéküktől összegétől függően több feltétel teljesülését vizsgálhatjuk meg. Ilyen feltételek közé sorolhatjuk a sakkjáték több alapelvét és szabályát, amelyek közül a legfontosabbak:

1. A játszma elején ne lépjünk többet egy figurával, ameddig a többi tiszttel ki nem fejlődtünk.
2. Figyelembe vehetjük a sáncolást. Amelyik fél a lépéssorozat közben elsáncolt, pluszpontot kap.
3. Számon tarthatjuk az 50 lépéses szabályt, vagyis azt, hogy az utolsó 50 lépésben szerepelt-e gyaloglépés vagy ütés, mert ha nem, akkor az állás döntetlen.
4. Figyeljük a háromszori lépésismétlést. Ugyanis ilyenkor szintén döntetlen.

Egy lehetséges példa a sok közül (csak fiktív pontszámokat tüntettünk fel):

Gyaloglépések esetén:

1. ha a lépésszám < 10:
   - ha a, b, c, f, g, h gyalog lépett: –4,
   - ha d2–d4, e2–e4, d7–d5, e7–e5 történt: +4;
2. 10 < a lépésszám < 24:
   - ha a1(a8), b1(b8), c1(c8), f1(f8), g1(g8), h1(h8) mezőn álló király előtti gyalog lépett: –4;
3. 24 < lépésszám esetén nincs külön értékelés.

Huszár és futó lépése esetén:

- ha mozdulatlan figura lépett: +10;
- ha már mozdult a figura és a lépésszám < 9: –4 (ha a lépésszám > 9 akkor: 0);
- ha a lépésszám < 15 és a tiszt: d3-ra, d6-ra, e3-ra vagy e6-ra lépett úgy, hogy a megfelelő d2, d7, e2 vagy e7 mezőn gyalog áll: –15.

Bástya vagy vezér lépése esetén:

- ha a lépésszám < 7: –10;
- ha a lépésszám > = 7: 0.

Királylépés esetén:

- ha a királlyal lépett, de még nem sáncolt és a figura értékösszeg > 9 vezérnél: –8;
- ha rövidre sáncolt: 12;
- ha hosszúra sáncolt: 6.

Az ilyen értékelésnek az a hibája, hogy megbontja a szimmetriát. Ugyanis a statikus értékelésnél csak olyan szempontokkal számoltunk, amelyek mind világos, mind sötét figurákra egyszerre kiszámíthatóak voltak, és így könnyen képezhettük a kettő arányát. Jelen esetben ez képtelenség, mert egyszerre csak egy bábuval léphetünk. A függvényt kiegyenlíthetjük, ha csak páros féllépés után vesszük figyelembe a dinamikus értékelés eredményét. A tapasztalat viszont azt igazolta, hogy ennek az aszimmetriának nincs olyan nagy jelentősége.

Az előbb felsorolt lépésértékelést kiegészíthetjük más komponensekkel is, csak arra vigyázzunk, hogy a sok feltételvizsgálat ne legyen ellentmondó és ne lassítsa le programunkat. Így például figyelheti, melyik figura áll kötésben, illetve melyik figura ellépésével hozunk létre kötést. A *2. ábrán* látható állásban sötétnek a játszma folyamán az e7–e5 gyaloglépés megtételére kell törekednie, ugyanis csak így juthat a centrumban ellenjátékhoz. Ez viszont most azonnal nem megy, mert a lépéssel az f6 huszárját lekötné, majd a figuracserék után elveszítené. A program a figuravesztést csak a 2. d4×e5, d6×e5 3. Vd1×d8, Hc6×d8 4. Fg5×f6 lépések megvizsgálása után venné észre, ami nagyon sok időbe telne. Ezért a dinamikus értékelés használatával 7 féllépés kiértékelése helyett már hamarább abbahagyja az elemzést, mert az f6 huszár lekötése miatt — és mert egyedül a vezér védi — a létrejövő állást már korábban lepontozza.

A dinamikus értékelés feladata az is, hogy ütéskor csökkentse a figurák számának értékét ugyanúgy, mint az anyagi érték összegét, gyalogátváltozás esetén viszont ez utóbbit növelje. Gyorsabbá tehetjük így programunkat, mivel nem kell minden állás értékelésénél mind a figuraszámot, mind a figuraösszeget újra kiszámítani, csak az ütéseknél. Ezek száma viszont a legtöbb esetben elenyésző az egyéb lépések számához képest.

KOVÁCS P. ATTILA

Sejtetős propaganda

Egy számítástechnikai — gazdasági konszern: a COMPUTERMALL

„Több haszonra vágyik, versenyképesebb, tehetősebb kívánna lenni? Forduljon a Computermall-hez! A philadelphiai Computermall egy 50 000 négyzetlábnyi (kb. 5000 m²) „lehetőség", ahol 100 szinten szolgáltatások özöne, árubőség és számtalan kiállítás várja a kis- és középvállalkozókat."

Ezzel a reklámmal találkozhatnak az USA számítógép-gazdasági köreiben tevékenykedők, ha a Computermall címet viselő vállalkozás prospektusait forgatják. S amit az ilyen jó propaganda sejtet, érdemes alaposabban megvizsgálni: mit is nyújt pontosan a Computermall az érdeklődőknek?

A vállalkozás óriási mintatermében, remek installációkban tájékozódhat az érdeklődő az Amerikában forgalmazott számítógépekről.

Kaphatók itt teszt- és más jellegű programok, általában azonnal, esetleg minimális megrendelési, szállítási időközzel, de mindkét esetben igen alacsony áron. Ha még így is nehézségbe ütközik a vásárlás, a hitelosztályon elintézhető fogyasztói kölcsön bármely cikkhez fedezetet nyújt.

Működik itt szakirodalmi csoport, szerveznek előadásokat, szakmai oktatásokat. Ez utóbbiakat a tárgynak megfelelő és más számítógép-vezérelt eszközökön „élesben" illusztrálják, és gyakorlattal is kiegészítik. A tapasztalatlanabbakat vezetői, üzleti tanácsokkal segítik, megtoldva az általános útmutatást konkrét számítógépes szervezési ismeretekkel, például a szállítás vagy a megrendelések optimalizálásának elméletével, reklámpszichológiai alapismeretekkel, reklámfogásokkal stb.

A Computermall könyvtárában az aktuális referenciakönyvek, magazinok és más friss irodalmak között válogathat az ügyfél szabadidejében. Elfoglaltságait tetszőlegesen sorolhatja ugyanis, illetve megváltoztathatja, mert maga az üzletház és a szolgáltatásait összehangoló néhány decentralizált központi gép is 0—24 óráig, folyamatosan működik.

Csábító továbbá, hogy ha egy vállalkozó betársul, akkor el is adhat ezen a mamut-menedzser cégen keresztül akár hardver-, akár szoftvertermékeket, és ráadásul „minden eddiginél nagyobb profithoz" juthat. Ugyanis — ha nincs apparátusa, vagy nincs ideje a vállalkozónak az előbb említett reklámtanfolyamon részt venni — a Computermall biztosítja a professzionális szintű propagandát. Ezt szolgálja a televíziós és egyéb csatornák mellett a cég saját újságja, illetve annak hirdetései, melléklete. A telefonkapcsolat tökéletes: az USA bármely pontjáról elérhető (800 belső melléke van!) a kizárólag reklámigényekkel foglalkozó és információk adására létesítettt részleg. Az alkatrészek tekintetében is könnyítenek a vásárló gondjain: helyben, illetve megrendelés útján is beszerezhető a társult vállalatok tetszőleges terméke. A Computermall olyan eladókat foglalkoztat, akik "tökéletesen értik a kliensek problémáit, amellett szinte lehetetlen olyan kéréssel fordulni hozzájuk, amit rövid időn belül ne teljesítenének" — nyilatkozott hasonló hangon több gyártó is.

Mit kínálnak az első katalógusok? Az üzletház hardver-, szoftveranyagokat, -termékeket, nyomtatókat, távközlési, adatátviteli eszközöket, memóriát, interfészeket, robotrendszereket, másológépeket, processzorokat, biztonsági és riasztóberendezéseket és még sok mást állít ki és ad el több mint 5000 négyzetméternyi területén. A legfontosabb forgalmazók és egyben társcégek névsora: DEC, Texas Instruments, ADDS, Televideo, IBM, Centronics, Diablo, Teletype, MDB Systems, Anadex, Okidata, Epson, Qwint, Avanti Communications, Hazeltine, Visual Technology, Anderson Jacobs, General Data Communications, Racal Vadic, Lear Siegler.

A vállalkozás kulcsa, amely a cég koncepciójának alapjául szolgál, a következő: „Az üzletember nem egyszerűen számítógépet vásárol — neki egy probléma (problémák) megoldására van szüksége. A vásárló menekül a dzsungelből, hiszen nem tudja, hogy az esetleg tökéletes hardverfeltételek milyen szoftvertámogatást élveznek. Ha pedig a vásárláskor így nincs szerencséje, az katasztrofális lehet." Vagyis: a Computermall, megőrizve a számítógépgyártók versenyre ösztönzését, mégis komplex szolgáltatást nyújt.

A sok vállalat együtt, egy helyen áll rendelkezésre; az üzleti körök széles skálájú megrendeléseit szem előtt tartva, feltérképezve könnyebb a teljesítés és a tervezés is. A vásárlás, a választás leegyszerűsödik, az érdeklődő gyorsan rátalál a már említett lehetőségek bármelyikére, nem kell gyártótól gyártóig mennie. Ráadásul megtanítják a számítástechnikai üzleti élet legfinomabb fogásaira is, így idővel valóban az ígért haszonhoz jutnak az ügyfelek.

Hogy valóban sikeres lesz-e a sokszínű elképzelés, még nem tudjuk. Ez az eddigi legnagyobb vásárlócentrikus vállalkozás — mindenképpen követendő példaként szolgálhat ésszerűsége, gazdasági értéke. Létrehozásához, zökkenőmentes működéséhez azonban ott és máshol is nagy rugalmasság, technikai érettség szükséges.

(A cikk a philadelphiai Computermall Inc. kiadványa alapján készült.)

HARGITAI PÁL

FÓRUM

Kedves Halassy Béla!

Olvasom „Barátunk, a számítógép" című cikkét, és nem értem, hogy mit kíván a cikk bizonyítani, nem értem — vagy nem „fülik a fogam" megérteni — az intelmet.

Valamikor — nem is olyan régen — a számítástechnika a kevesek kiváltsága volt, egy kiválasztott klikké, akik féltékenyen őrizték gépkönyveiket és programozási leírásaikat, mint ahogy az egyiptomi papok őrizték valamikor titkaikat.

A laikus közvélemény ámulattal fogadta ezt a csodálatos technikai előrelépést, a vállalati felső vezetés pedig szemrebbenés nélkül kifizette a „számítógépes szervezés" címen kiszámlázott milliókat.

Aztán megérkezett a mikroszámítógép, és Pistike meg Lacika felfedezte, hogy nem olyan ördöngös dolog az a programozás, és a számítógépes szervezés legfontosabb alkotóeleme nem más, mint a „józan paraszti ész".

Ezek után Halassy arra inti a szeleburdi ifjakat, hogy ez a személyi számítógép még nem az igazi, mert ő már tudja, hogy az „igazi" számítógépek kezelése nem mindennapos feladat. Hát mi?

13 éve kenyérkereső szakmám a számítógép; elmondhatom, hogy legtöbbször mindennapos feladatokkal foglalkoztam, hiszen az élet mindennapos feladatokat produkál. Én nem hiszem, hogy bárkinek problémát okoz, hogy Magyarországon ma tízezrek BASIC-ben programoznak.

Ha a vezető beosztású apucinak vagy a háztartásbeli anyucinak van ideje, pénze és türelme, csak tanuljon matematikát, BASIC programozást. Kinek árt ez?? Nos, ki tudja, lehet, hogy holnap az a bizonyos vezető beosztás megszűnik, talán még a vállalat is megszűnik, és a papának vagy a háziasszonynak valamiiyen más munkahely/beosztás után kell néznie. Lehet, hogy az a kis BASIC programozás még nagyon jól jön egyszer!!!

Én azt hiszem, hogy azok a tízezrek, akik esténként személyi számítógépeikkel bütykölnek, sokkal jobban megértik ezt, mit Halassy. Nem hiszem, hogy bárki el tudná dönteni egy „tendenciáról", hogy az „vélt" vagy „valós". Mi a különbség? Az igazán eredeti ötletek nem „valós" tendenciák követéséből alakultak (ha egyáltalán létezik ilyesmi . . .). Tíz évvel ezelőtt Jobs és Wozniak egy kis garázsban bütykölte össze az első Apple mikrogépet annak ellenére, hogy mindenki meg akarta őket győzni, hogy erre nincs „valós" igény; Mitch Kapor meg félhivatásos zsonglőr volt, amikor a LOTUS 1-2-3 programcsomag ötletét felvetette néhány éve, és „igazi" programozók megpróbálták meggyőzni őt az ötlet alkalmazhatatlanságáról. Szerencsére ők nem hallgattak a „jó tanácsokra", és erényt csináltak a kényszerből.

Szívből remélem, nem sokan fogják megfogadni Halassy jó szándékú, de nagyon is vitatható álláspontját. Pistikék, apukák és mamák csak programozzatok, bütyköljetek azokon a mikrogépeken, és akkor nem lesz időtök tűnődni „vélt" vagy „valós" tendenciákon.

GEORGE LAZAR
478. Merritt
Oakland, CA.
USA — 94610

Kedves George Lazar!

Hozzászólásod valóban azt jelzi, hogy nem érted — vagy nem akarod érteni — az „intelmet". Idézőjelbe teszem ezt — nemcsak azért, mert Téged idézlek, hanem főképpen az értelmezése miatt: hiszen a szóban forgó írásom címe is utal rá, hogy nem valami ellen, hanem valamiért ragadtam tollat. Éppen Neked írtam a „Barátunk, a számítógép" című cikket.

Általában azoknak, akik értenek ugyan a számítógéphez, de kevesebbet látnak a problémákból. A számítógép új papjainak — a Te kifejezésed —, akik kicsit elzárkóznak attól, amit életnek nevezünk. Mert mit is mondasz?

Ténylegesen azt hiszed, hogy egy állását vesztett főnök BASIC programozásra fogja adni a fejét, és abból él majd — x-évesen? Idézem a szomszéd háziasszonyt, aki mikrogépes tanfolyamra jár: „Minek etették velünk ezt a BASIC-et? Az egy lehetetlen nyelv. Bezzeg a dBASE III-ra, amit könnyen és jól lehet használni, alig maradt időnk!".

Remélem, nem gondolod komolyan, hogy egy nem személyi számítógép kezelésének feladata azonos a PC kezelésével, amint ezt egyik kitételed sejtetni engedi. Mint ahogyan azt is csak kétkedve hiheted, hogy józan paraszti ésszel lehet jó számítógépes információrendszereket készíteni. Vajon akkor miért tartanak több éves szervezői tanfolyamokat?

Ami a vélt és valós tendenciákat illeti: nem én bocsátkozom valamiféle bizonytalan jóslatokba. Az IBM és a többi számítógépes nagyvállalat alapos előrejelzési elemzéseket készít. Ezekről időnként konferenciákon, munkabizottságokban is beszámolnak. Ők — és nyomukban én is ezt teszem — úgy vélik, hogy a hagyományos programozói szakmának már nincs nagy jövője. Hamarosan élesen szétválik majd a gépet kezelni tudó (nem BASIC-ben!) „laikus" felhasználók tábora az őket kiszolgáló bonyolult rendszereket előállító szuperprogramozóktól. Ebben szabad kételkedni, de nem érdemes.

A programozás nem ördöngös dolog. A számítógépnek nem játékként, nem hobbiként, nem önmagunk gyönyörködtetésére való használatához viszont nem elegendő némi tehetség és vonzalom. Igen sokat kell tanulni, amit talán mi nem tettünk meg.

Remélem, nem haragszol, amiért e sorokban tegezlek. Elvégre egyszer majdnem szervező lettél nálunk. Valójában egyet akarunk, csak másként. Ennek reményében üdvözöllek, és örülök, hogy ismét hallottam Rólad.

DR. HALASSY BÉLA

„Barátunk a számítógép" címmel 1985-ben könyvet adott ki a Móra kiadó (Hobbi sorozat). Szerzője, Szikszai Csaba kérésének eleget téve, szívesen megemlítjük ezt a tényt. (Szerk.)

Karácsonyra gyűjtöttem össze a márciusi szám leveleit; ilyenkor több idő jut a tallózásra. Állandó lelkiismeret-furdalásom van, mert le kell rövidítenem az írásokat, és még így is a rovatnak jutó hely a levelek számához viszonyítva olyan kevés, hogy sok levélből még egy rövid részletet se tudok közölni.

Sok kérésnek nem tudunk eleget tenni, de az is előfordul, hogy a feltett kérdésekre sem tudunk felelni; ilyenkor levélíróinknak levélben válaszolunk, kérve megértésüket.

Duska Ákos, Budapest,

Fő u. 68. 1027

Egy nagy kéréssel fordulnék Önhöz. Az 1986. szeptemberi számukban megjelent egy cikk (program), a 21. oldaltól kezdődően a három dimenziós ábrázolásról. Sajnos, terjedelmi okok miatt csak az alaprutinok szerepelnek. Nekem C—64-es gépem van, és ezt a programot nagyon jól lehetne hasznosítani. Én sajnos nem értek még a bonyolultabb BASIC és Simon's BASIC programozáshoz, így nem tudnám a programot kiegészíteni.

Nagyon megkérném, ha egy mód lenne rá, hogy küldje el nekem a teljes programlistát.

Örültem a levélnek, mert így válaszul néhány dolgot el tudok mondani a programok cseréjével, az adásvétellel és a másolásokkal kapcsolatban. Előrebocsátom, hogy nem vagyok jogász, ennek ellenére megpróbálok pontosan fogalmazni. A μ M elég sok programot kap közlésre, a programot beküldő szerzők a honorárium fejében a közlés jogát átengedik a μ M-nak, a másolás jogát pedig az olvasóknak. Ez nem jelenti pl. azt, hogy az olvasók közül bárki ebből üzletet csinálhat és pl. egy kis módosítással vagy anélkül a programokat kazettán árulhatja, ti. az már a szerző jogait sérti. A μ M a beküldött programokat nem sokszorosítja le se programlista, se pedig kazetta formában, és azokat sem pénzért, sem ingyen nem terjeszti. Arra van lehetőségünk, hogy megkérjük a szerzőt, hogy küldje el a teljes programlistát, aki azután vagy megteszi, vagy nem. El kell még mondanom — szoftver amatőröktől kaptam az információt —, hogy úgy látszik, mintha az illegális másolások terén némi javulás mutatkozna. Azt hallottam, hogy pl. a HCC-ben tömörült amatőrök nagyon ügyelnek arra, hogy kalózmásolatok terjesztésével ne rontsák a magyar amatőrmozgalom jó hírét, védik mások szellemi termékeit és így a magukét is.

Levelét továbbítottuk a cikk szerzőjének.

Imhof Iván, Kecskemét,

GAMF Ságvári E. Kollégium, Izsáki u. 10. 6000

Megjelenése óta rendszeres vásárlója vagyok lapjuknak, eddig sok hasznos információhoz jutottam általa. Most egy kéréssel fordulok Önökhöz.

Elsőéves főiskolai hallgató vagyok. Eddig BASIC és Assembly nyelvű programokat írtam, most pedig szeretném megismerni a FORTH nyelvet. Sikerült hozzájutnom a C—64-re készült Performance FORTH programhoz, de a birtokomban levő FORTH könyvek nem tartalmazzák e program leírását. Lapjuk márciusi számában olvastam egy cikket, melyben Garay László több FORTH rendszert mutatott be. Ezek közt szerepelt a Performance FORTH is.

Szeretném a segítségüket kérni abban, hogy az említett program részletes leírását megszerezhessem, akár a cikk írójától, akár mástól. A felmerülő költségeket természetesen megtéríteném.

Levelét továbbítottam a szerzőnek.

Kerekes Mihály, Keszthely,

Zalka Máté út 19/A 8360

LASER 2001-es gépem van, néhány gyári játékkazettával, 2dS botkormányos és német nyelvű leírással. Társakat keresek, az Önök segítségével. A gépről minden érdekel.

Biztosan lesz társa, én azt hiszem.

Árus Zsolt, Gheorgheni,

Cart. Bucin, BL:6, Sc. A. Ap:18

Jud. Harghita, România 4200

Mindenekelőtt szeretném elismerésemet kifejezni azokért az erőfeszítésekért, amelyeket a magazin szerkesztősége tesz azért, hogy hónapról hónapra tartalmas, érdekes lapot vehessünk kézbe. Ez annál is értékesebb helybeli kollégáimnak és jómagamnak, lévén hogy a Számítástechnika mellett a Mikroszámítógép Magazin az egyetlen élő kapocs (még ha egyoldalú is) a szakma állandóan változó, terebélyesedő világával. És habár a cikkek jó részét csak mint esetleg átültethető, alkalmazható érdekességet böngészhetjük (lévén hogy csak egy Spectrumon dolgozhatunk egy iskolai szakkör jóvoltából, HT-t, Primót, Commodore-t csak képen láttunk), mégis mindig érdeklődéssel várjuk a lapot, megérkezése pedig esemény.

Ez azonban még nem elég ok arra, hogy a szerkesztőség postáját duzzasszam (legalábbis szerintem); ami mégis erre késztet, az egy rövid cikk, amely Ki ad magyarázatot címmel jelent meg a szeptemberi számban. S (mint lehet, mások is megtették) ezt már nem dicsérni akarom!

Hogy valaki, akinek Spectrumja van, az illető jelenséget rejtélyesnek találja, az könnyen megtörténhet, lévén hogy (bizonyosan) nem egy Spectrum-tulajdonos él (Magyarországon), aki nem szakember. (Ámbár, hogy valaki „szakkönyveket" olvasson, ahhoz szükséges, hogy konyítson a szakmához.) A szerkesztőségnek ellenben utána kellett volna járnia a „titoknak", mielőtt ország-világ elé tárja. Ugyanis: a cikket elolvasva egyik tanítványommal (szakkönyvek hiányában) kísérletezni kezdtünk, s még egy óra sem telt el, amikor már lényegében tudtuk, hogy mi a magyarázat. Ezt már csak megerősítette a szakkönyv, az Ipari Informatikai Központ kiadásában megjelent A ZX-Spectrum ROM programja című.

Tehát a magyarázat az, hogy a DRAW-rutin nem ellenőrzi a megadott szögértéket, mindössze azt vizsgálja meg, hogy 2 π-nek többszöröse-e, s ebben az esetben egyenes szakaszt rajzol. Ellenkező esetben jóhiszeműen rajzol akár 99 vagy 999 π-s körívet is, azaz egy, a kezdő- és végponton átmenő fura spirálist, amit talán görbe oldalú csillagsokszögnek lehetne nevezni. Hibajelzést tehát olyankor kapunk, ha az illető kör kilógna a képernyőből. A „forgó" három- meg ötszögek magyarázata pedig (ezek tulajdonképpen nagy sebességgel felrajzolt és nagyon sok ágú csillagsokszögek), hogy a rutin 252-re korlátozza az ívet alkotó egyenes szakaszok (elemi körívek) számát.

Hát ez az a titokzatos jelenség, amely valószínűleg onnan származik, hogy a rutin írója elmulasztotta (elfelejtette?) a szögértéket ellenőrizni és 2 π feletti érték esetén hibajelzést adni, esetleg a 2 π-vel való osztási maradékkal számolni tovább.

S ha már írásra szántam magam, hadd vessek fel egy problémát — a szoftverét. Nálunk sajnos mikrogépekre írt szoftvert hivatalosan venni nem lehet — marad a másolás lehetősége. Ilyen módon alapszoftverrel úgyahogy elláttuk magunkat (Beta Basic, Forth értelmező, Pascal fordító), de még ezek használatát megtanulni is komoly fejtörést okoz dokumentáció hiányában; ellenben az alkalmazói szoftverek területén (a játékprogramokat kivéve) teljes az űr. Márpedig mi a gépet minél hamarabb használni is szeretnénk (féléves sincs, gyakorlatilag csak ismerkedünk még vele, a saját programok száma minimális) nemcsak a számítástechnika, hanem más tantárgyak oktatásában is. Gondolom, ebben a szerkesztőség segítségünkre lehetne, ha mással nem, azzal, hogy közzéteszi problémánkat. Ezt előre is köszönöm!

Kérem nézzék el a rovatvezető gyengeségét, hogy a talán meg nem érdemelt dicsérő sorokat is közli. Ami a Spectrum oktatási alkalmazását illeti, sajnos sokat segíteni nem tudunk, miután Spectrum nem szerepel a magyar oktatásügy által az iskoláknak ajánlott gépek listáján, így ezekre a gépekre az oktatóprogramok kidolgozását hivatalosan nem támogatják. Én azt hiszem, hogy olvasóink közül lesznek néhányan, akik — levelét olvasva — küldeni fognak programokat és leírásokat. Én pedig sok sikert kívánok, a magyarázatot pedig köszönjük.

Marincsák János, Kisvárda,

Lenin út 20. 4600

Régi olvasója vagyok lapjuknak és különösen szívesen olvasom az Az olvasó írja című rovatot. De úgy érzem, hogy a „régi jó" gépekről (VC 2Ø, ZX 81, Texas 99—4/a) igen kevés szó esik mostanában, és inkább a „menő" (C—64, Spectrum, C—16) géptípusok kerültek előre és nem háttérbe, hanem teljesen kiszorítva a régebbi típusokat. Talán a VC 2Ø (hál'istennek) az egyetlen kivétel, mert az utóbbi időben a magazin a 64 mellett néhány szót „vesztegetett" erre a géptípusra is. Nekem is ilyen gépem van, és úgy érzem, méltánytalanul száműzték!

Ha egy módot találnak rá, közöljék levelem következő részét. Ezt előre is köszönöm. Kapcsolatot keresek olyan VC 2Ø-as tulajdonosokkal, akik szívesen cserélnének programokat, programozási ötleteket, programismertetőket stb . . .

Kérem, akit érdekel, írjon a címemre. Szeretném megtudni a HCC Commodore szekciójának a levélcímét is (a telefonszámmal sajnos nem sokat érek).

Érdekesen alakult a lap rovatainak a mérete és szerkezete az első szám óta. Talán az iskola-számítógép és sakkrovat valamelyest megrövidülhetne, és ezen a helyen szó eshetne a lassan feledésbe merülő, de közforgalomban nagy számban levő számítógépekről. Sajnos, nem olyanok a gépek árai, hogy a néhány éves fejlesztésű gépeket csak úgy hagyjuk porosodni és újat vegyünk mellé (nagy pénzen).

Az olvasó írja rovatban megjelent, a VIC 2Ø-hoz kapcsolódó hozzászólások íróinak is írtam, de sajnos, nagyon sokan még nem is válaszoltak. Igen sokat köszönhetek Auer Gábornak, akit az Önök lapján keresztül ismertem meg, és tőle származnak az első programjaim is.

Ismét csak a gyengeségemet bizonyítja, hogy közöltem a levél bevezető részét is, amelyben levélírónk az Olvasó írja rovatot dicséri. Ritka alkalom. A HCC Commodore szekciónak az NJSzT Titkárságára lehet levelet küldeni. Budapest 5. Pf. 240. 1368

Kovács Zsolt, Miskolc,

Egyetemváros E/2 210.

Érdeklődve olvastam lapuk utóbbi számában

a Commodore Plus 4-es gép ismertetőjét. A cikkel kapcsolatban néhány kérdésem lenne, mégpedig:

a) hozzájuthat-e magánszemély a BÁV-on kívül valamilyen formában? (ha lehet, akkor budapesti címen kívül szolnoki, ill. miskolci forgalmazói címeket is kérnék),

b) konkrétan vagy legalább nagyságrendileg milyen áron kerül forgalomba az alapgép és a hozzá illeszthető magnetofon?

c) ha megrendelés útján megvásárolható, hova forduljak igényeimmel?

Üzletkötéssel nem foglalkozunk, ezért levelét a Novotrade-hez továbbítottam. (Címük: Budapest XIII., Fürst S. u. 24/26. 1136.) Úgy tudom, hogy a SKÁLA Computer-S üzletei árulják időnként a Commodore+4 gépet 8000 forint körüli áron.

Pallagi László, Százhalombatta,

Rózsa Ferenc u. 3. 2440

Örömmel olvastam a Mikroszámítógép Magazinban, hogy ismét megrendezésre kerül a Garay pályázat. Különösen az a rész volt számomra kedvező, amely lehetővé tette, mint előfelvételis egyetemista, részt vegyek a pályázaton. Mivel a kiírt gépekhez katonai szolgálatom ideje alatt nem jutok hozzá, egy régebbi — 86 májusi — logikai játékprogramomat adom elbírálásra.

Örülök, hogy részt vesz a Garay pályázaton (nem is rossz ötlet így rövidíteni kedves gimnáziumom pályázatának az elnevezését). Én azt hiszem, hogy katonai szolgálata alatt is tud számítástechnikával foglalkozni, hiszen a laktanyában működnek a μ klubok. Azt javasolom, hogy kérdezze meg a parancsnokait, hogyan tud — ha ilyen éppen nincs — klubfoglalkozást szervezni az alakulat érdeklődő katonái részére. Sok sikert!

Gulyás András, Pomáz,

Eötvös u. 6. 2013

Az alkalmat felhasználva szeretném kifejezni elismerésemet a lap változatlanul magas színvonala és érdeklődésre számot tartó cikkei iránt. A számítástechnikával már közelebbi kapcsolatba került, de azt nem hivatásszerűen művelő olvasó számára kiválóan alkalmas az ismeretek bővítésére és a kíváncsiság kielégítésére.

Talán az egész oldalas, csak gépi kódban írt listákhoz (játékok) lehetne némi eligazító magyarázatot fűzni. Érdekes lehetne a hazai számítástechnikai folyóiratok, időszakos kiadványok áttekintő bemutatása.

Ez volt a célunk, a szerkesztőség nevében köszönöm, hogy törekvésünket nyugtázta.

Horváth Sándor, Kistelek,

Zrínyi u. 6. 6760

Ne haragudjanak, hogy levelemmel zavarom Önöket. Segítséget szeretnék kérni a lapjuktól.

A lényegre térek, vettem a kereskedelemben kapható SELTROW COLOR COMPUTER 200 és a hozzá való 2 db JOYSTICK-et. De nem lehet hozzá kapni játékprogram-kazettát sehol. Még hozzá való magyar nyelvű szakkönyvet sem. Tehát most itt állok tehetetlenül, mert nem értek hozzá. A magyar prospektus, amit kaptam, nem éppen kielégítő.

Mindenekelőtt játékkazettát szeretnék venni, vagy egyáltalán megtudni, hogy hol lehet hozzájutni. Ebben kérem az Önök segítségét. Nagyon kérem, válaszoljanak levelemre (válaszborítékot mellékelek).

Nagyon kérem, legyenek segítségemre, mert már sehova sem tudok fordulni.

Mindenütt azzal fogadnak, sajnos nincs ez a készülék elterjedve. Pedig a kereskedelemben kapható, a SKÁLA-COOP forgalmazza.

Erről az akcióról nem hallottunk, levelét elküldtük a Skála-Coop illetékes vezetőjének.

Bálint Attila, Nyíregyháza,

Gádor Béla köz 2. IV/19. 4400

7. osztályos általános iskolás tanuló vagyok, és a számítástechnika BASIC nyelvét már nagyjából elsajátítottam. Számítógépem nincs, de ha a 7. osztályos bizonyítványom is jó lesz, akkor kapok egy COMMODORE—64-est és szeretnék minél többet tudni róla, persze nemcsak erről az egy számítógépről, hanem általánosan mindegyikről, és továbbtanulni is ilyen téren szeretnék. Ezért szeretnék segítséget kérni leendő számítógépem megismeréséhez, hogy irodalmat ajánljanak főleg a gépi kódú programozás, de a BASIC nyelven kívül még más programozási nyelv elsajátításához is.

Mindenekelőtt C—64-et termő bizonyítványt kívánok. A géphez nagyon sok könyvet árusít pl. a Novotrade 2C Számítástechnikai Áruház (Bp. XIII., Balzac u. 35. Tel.: 402-954) és a Műszaki Könyváruház (Bp. VI., Liszt F. tér 9. Tel.: 420-353). A C—64-hez kapható szakirodalmat felsorolni túlságosan hosszú lenne, az egyik legutóbb megjelent könyv a Data-Becker—Novotrade kiadásában, Hornig—Trapp—Weltner: Tippek és trükkök Commodore 64-eshez. 239 Ft. (A μ M 86/11—12. számában közöltünk a könyvről rövid ismertetést.)

KOVÁCS GYŐZŐ

μ'87
ORSZÁGOS
MIKROSZÁMÍTÓGÉPES TALÁLKOZÓ
programja

Márc. 19.
Megnyitó és fogadás
— csak a meghívott
vendégek részére.

Márc. 20.
Amatőrök napja
— bemutatjuk a magyar
informatikai
amatőrök eredményeit.

Márc. 21.
Diák nap
— tiétek a μ'87!

Márc. 22.
Családi nap
— Mire használható
a számítógép otthon?

Márc. 23.
Kulturális és humán nap
— Zene, képzőművészet,
irodalom;
Vakok és gyengénlátók,
mozgássérültek napja.

Márc. 24.
Társasági nap
— a MTESZ-egyesületek
képviselőinek
és az NJSZT tagjainak
találkozója.

Márc. 25.
Felsőoktatási nap
— a találkozó bezárása.

COMMODORE—64

Játék POKE-ok

Nagy sikerük volt a Mikromagazin 1985/6. és 1986/1. számában megjelent Commodore „játék POKE"-oknak, ezért valószínűleg a C16-os játékok módosítására is kíváncsi a lap közönsége. Az itt közöltek szerint számos „életet" hosszabíthatnak meg olvasóink.

Az alábbi két játék tulajdonképpen BASIC program gépi kódú részekkel, ezért bennük a szokásos módon BASIC editorral javíthatunk, mint valami hibát.

RIG ATTACK	A 3ØØ-as sorban HL=HL−1 helyett HL=HL−Ø szerepeljen.
SPACE MISSION	A 75Ø-es sorban a ME=ME−1 helyett ME=ME−Ø a jó.

A táblázatban felsorolt programok teljesen gépi kódúak, ezért az átállításuk előtt MONITOR paranccsal gépi kódú monitorba kell lépnünk, majd a táblázat szerinti sorok beírása után X paranccsal ki kell lépni a monitorból. Ugyanezt egyszerűbb lenne POKE utasítással elintézni, de a játékok egy része nem szereti ezt a módszert, és OUT OF MEMORY ERROR hibajelzéssel jutalmaz érte. Természetesen a POKE utasítást nem is hajtja végre.

Fontos, hogy a táblázat sorait mindig külön sorba kell írni, és a sorok között — minden sor után — le kell nyomni a RETURN billentyűt. A >jel a billentyűzet jobb alsó sarkában található, a . felett.

A táblázatban egyes játékoknál megjegyzéssel ellátott sorok is vannak. Ezek nem tartoznak az „örökéletre" állításhoz. Hatásuk a következő:

négyzetek (MAJOR BLINK)	Eltünteti a szaladgáló négyzeteket, melyek zavarják a kezdő játékosokat.
pályaszám (ROBIN HOOD)	Ez annak jön jól, aki már minden reményt feladott. Ezzel végignézheti az összes pályát.
energy végtelenítése (BABY BERKS)	Az energiát állítja

	„örökenergiára".
idő (MAJOR BLINK, TUTTI FRUTTI, ROCKMAN)	Hatására az idő nem fogy a játék közben.
energy (BABY BERKS 2)	„Örökenergiára" állítás.
fuel (BAN)	Az üzemanyag fogyását akadályozza meg.
pajzs (DEFENCE)	Soha nem fogy el a pajzs.

Egyes játékok különböző néven ismertek. Ezeknél zárójelben a másik név is szerepel.

Más a helyzet a BMX RACER-nél, ugyanis előfordulhat, hogy az olvasó nem kalózmásolat útján jutott hozzá, és BASIC betöltőrésszel használja. Ebben az esetben a következő a megoldás:

MONITOR
L "£" Ø1 (lemeznél Ø8)
>236Ø EA EA EA
>FF16 CE A1
>2 Ø1 (botkormány használata esetén Ø1 helyett ØØ-t kell írni)
>21E1 EA (örökenergia)
X
SYS833Ø

A " "-ben a játék nevének kell állnia, a gépi kódú rész nevének. Nem biztos, hogy ez £ jel.

1. táblázat

NÉV	TARTALOM "játék POKE"
DARK TOWER	>1FCA EA EA EA
TIMESLIP	>14Ø4 EA EA
SPECTIPEDE	>1A87 EA EA
BABY BERKS	1A2Ø ØØ
	>2494 ØØ
	>1A5E EA (energy végtelenítése)
JETPAC	>1CFØ EA EA 4C F8 1C
(CUTHBERT IN SPC)	>119E életek száma
XZAP	>3665 EA EA 4C 7E 36
	>2CFF életek száma
SKRAMBLE	>39ØB EA EA EA
	3A8Ø életek száma
MAJOR BLINK	>298C ØØ
	>12B2 ØØ
	>131D EA EA (idő)
	>1323 EA EA (idő)
négyzetek:	>1BD1 FF
EVIL	>326A 4C EØ 3F EA
	>3FEØ A6 92 A9 Ø5 95
	B8 4C 6E>3FE8 32
TYCOON TEX	>249D EA EA EA EA EA
	4C C6 24
GALAXIANS (INVASION)	>25A3 EA EA
	>11ØØ életek száma
MAYHEM	>19EØ EA EA EA
STELLAR WARS	>2669 EA EA EA
	>2676 EA EA EA
SQUIRM	>2FAE EA EA EA
ROBIN HOOD	>2268 EA EA
	>21Ø5 ØØ EA
pályaszám	>1CB1 nehézségi szint (legfeljebb Ø4)
ZODIAC	>1FD2 ØØ
	>2Ø61 ØØ
MENEKÜLÉS	>121E EA EA EA
PUNCHY	>1Ø8A EA EA EA
	>1Ø2F életek száma
XARGON	>38E6 EA EA EA EA EA
	4C ØA 39
SPACE FREEKS	>1117 EA EA EA
SPACE SWEEP	>2375 EA EA EA
BABY BERKS 2	>261F ØØ
	>12CE ØØ
	>134F EA EA (energy)
TUTTI FRUTTI	>1F5Ø EA EA EA
	>1A27 A9 (idő)
BLAGGER	>354A EA EA EA
WILD WEST	>1EØØ EA EA
	>17D6 EA EA
CUTHBERT IN LAB	>1964 EA EA EA
	>19F8 EA EA EA
ATLANTIS	>2D3C EA EA
	>3AC8 EA EA
MR PUNIVERSE	>1FFC EA EA
	>3367 EA EA EA
OUT ON A LIMB	>13CB EA EA
ZAPPEM	>3Ø5Ø EA EA EA
BIG MAC	>1F2C EA EA
	>32BA EA EA EA
BAN	>2B5C EA EA
(HALL OF FAME)	>3C55 EA EA (fuel)
DEFENCE	>3861 EA EA
(DEFENDER)	>11AØ EA EA (pajzs)
FIRE ANX	>1C1F A9 Ø4
DORKS DILEMMA	>32FE A9 Ø5
ROCKMAN	>2194 EA
	211Ø EA (idő)

TUBA IMRE

Az ismeretlen C16

Amikor a címbeli állítást megváltoztatandó, e cikkek írásába belefogtam, a C16 belső világáról csak elvétve volt fellelhető magyar nyelvű információ. Azóta sokat változott a helyzet. Az előző számban ismertetett könyv megjelenése után alig egy hónappal, előttem van egy második ugyanerről a témáról, de még ez sem az, amelyikre számítottam.

Tóth Viktor:
A Commodore 16-os belső felépítése
(Budapest, 1986. NOVOTRADE, 429 oldal. Ára 99,— Ft.)

E könyvben minden benne van, amit én a jelen sorozatban — eredeti terveim szerint — közlésre szántam. És még sok más. Olyasmi is, amit én nem mertem határozottan leírni.

A BASIC és gépi kód című írások első részében utaltam rá, hogy az általam ismert Commodore gépeknél négy különböző processzortípussal találkoztam. Személyes találkozásról szó sincs, az egyes típusmeghatározásokat az irodalomból vettem. Dr. Úry László Commodore C16 könyvében határozottan 7501-es típusszám szerepel, ezzel szemben Székely Béla kollégám, aki velem egy időben és azonos helyen vette C16-osát, mesélte, hogy ő a gép belsejébe pillantva egyebek között egy 8501 feliratú áramköri tokot látott. Ugyanerről a témáról egy zárójeles mondat Tóth Viktor könyvéből: „(Egyes dokumentációk ettől eltérő — pl. 7501 — típusszámot közölnek; én eddig a nálunk forgalomba került gépeknél csak a 8501-essel találkoztam.)"

A kötet nagy részét itt is a ROM-lista teszi ki. A magyarázatok a nagyobb karaktermeret miatt még rövidebbek, mint az előző számban ismertetett könyvben, néhol akadnak a tapasztalatlan olvasót félrevezetőek is, de általában elég jónak mondhatók. Ebben a könyvben nincs keresztreferencia-lista, de nem is hiányzik. Van viszont a rendszer RAM-területéről leírás, amelyben azok az információk is megtalálhatók, melyek a Rendszerváltozók és I/O címek című füzetből hiányoznak (lásd az 1986/10. számban).

Az egész könyv — a borítótól eltekintve — a DATA BECKER cég INTERN köteteire emlékeztet, tartalmi felépítését és kivitelezését tekintve egyaránt. A szövegszerkesztővel írt, mátrixnyomtatón előállított anyag jól olvasható lenne, de sajnos a nyomdai munka színvonala egyenetlen, jó néhány hibás példánnyal találkoztam. A bevezető részben a szerző eláraszt bennünket a hardver- és szoftverinformációk tömegével, hibát csak elvétve találni, és általában nem túl jelentőseket. (Néhány mutatóba: a BASIC utasítássorszámok felső

korlátját 65 000-nek írja 64 000 helyett, a fogalommagyarázatoknál az interface szót „arc-közi"-nek fordítja.) Részletesen ismerteti a gépi kódú programozás elemeit, a KERNAL rutinokat, a monitor utasításait, a TED regisztertérképét, a B/K különleges lehetőségeit és még számos dolgot, amit a cikk leadásáig rendelkezésemre álló egy hét alatt nem volt időm áttanulmányozni.

Itt találtam meg a SETTMO első tisztességes leírását. A CLALL-ról szólva figyelmezteti az olvasót arra, hogy a rutin csak logikailag zárja le a fájlokat, a háttértárolón a fájl lezárása nem történik meg. Azért emelem éppen ezt ki, mert annak idején én elmulasztottam felhívni a figyelmet erre: egyik olvasóm kifogásolta is.

Összegezve: mindenkinek ajánlom a könyvet, aki a C16-ot nemcsak BASIC-ben akarja programozni. Ez az a kötet, amelyet kezemben tartva azt mondtam magamnak, hogy most már szép lassan befejezhetem a sorozatot, mert a C16 egyáltalán nem ismeretlen többé. Mindez 99 forintért!

A sorozat azonban mégsem fejeződik be. A könyv 66. oldalán röviden olvasható a BASIC-bővítéseknek egy új lehetősége, de a szerző a megoldás felderítését az olvasóra bízza. Erről a témáról a következő számban ír Gnädig Péter, aki megoldotta ezt a problémát.

BARNA LÁSZLÓ

EASY file-tól a MASTER 64-ig
Adatfeldolgozó programcsomagok Commodore 64-re
(Budapest, 1986.
LSI ATSZ,
254 oldal. Ára: 185,— Ft.)

A kötet öt adatfeldolgozó programcsomagot közöl módszertani útmutatóval. A programcsomagok általános ismertetését példákon mutatja be, és segítséget nyújt a felhasználók saját feladatainak megoldásához is. Külön fejezet foglalkozik az adatfeldolgozás néhány főbb szempontjával, a lemez- és az adatvédelemmel.

Radnai Rudolf:
Mikroprocesszoros berendezések vizsgálata
(Budapest, 1986.
Műszaki Könyvkiadó,
259 oldal. Ára: 70,— Ft.)

1971-ben az Intel félvezetőgyár előállította az első mikroprocesszort. A cég 4004 típusú áramkörének megjelenését ma már egy új ipari forradalom kezdetének tartjuk.

A mikroprocesszorok megjelenésével a számítástechnika végképp „emberközelbe" került. Felhasználásukkal gyártják az olcsó, nagy teljesítményű személyi számítógépeket és a programozható asztali kalkulátorokat, mikroprocesszorok vezérlik az intelligens terminálokat stb. Mivel alkalmazásuk által szoftver szinten alakíthatók ki egyedi tulajdonságok, ezért az igen nagy szériában gyártható hardverelemekből — programozással — egymástól eltérő tulajdonságú rendszerek is előállíthatók.

Nyilvánvaló előnyei mellett a mikroprocesszorokra épülő technika együtt jár azzal, hogy a felhasználóknak fel kell készülniük ezeknek a bonyolult digitális eszközöknek a vizsgálatára. A könyv ehhez nyújt segítséget, összefoglalva a mérési alapelveket, bemutatva a különböző mérőműszereket, ismertetve működésüket és kezelésüket.

Pataki Attila—Tallér Ferenc:
Fűtési rendszerek számítása személyi számítógéppel
(Budapest, 1986.
Műszaki Könyvkiadó,
278 oldal. Ára: 158,— Ft.)

A szerzők összeállításukkal segítséget kívánnak nyújtani a családi házak, társasházak fűtési rendszerének számítógépes méretezéséhez és ellenőrzéséhez — a hőszükséglet meghatározásától egészen a rendszer vezérléséig. A könyv az ipari felhasználóknak is hasznos ismereteket ad, de mindazoknak is, akik nem rendelkeznek számítástechnikai és hőtechnikai ismeretekkel. A közölt programok a Spectrum, a ZX 81 és a Commodore 64-en is futtathatók.

Főbb fejezetek: Hőtechnikai alapfogalmak. Családi házak hőtechnikai méretezése számítógéppel. Kazánok és kazánházak kialakítása személyi számítógépek segítségével. Kazánok, tüzelőberendezések méretezése, ellenőrzése személyi számítógéppel. A fűtési rendszer vezérlésének lehetőségei személyi számítógéppel.

Ada-Winter Péter—
Ada-Winter Dávid:
A ZX—Spectrum hardverleírás, gépi kódú programozás
(Budapest, 1986.
Műszaki Könyvkiadó,
358 oldal. Ára: 57,— Ft.)

A könyv a Magyarországon egyik legelterjedtebb személyi számítógéptípus, a Sinclair ZX—Spectrum azon felhasználóihoz szól, akik megismerkedtek gépük „könnyen hozzáférhető" szolgáltatásaival, BASIC nyelvű programozásával, és mélyebben is bele akarnak nyúlni a rendszerbe. A szerzők ismertetik a Spectrum, mint elektronikai rendszer tulajdonságait, a gép hardverfelépítését és szoftverrendszerét. A könyv alapján az olvasó maga is kísérletezhet saját illesztésekkel.

A főbb fejezetek: A ZX—Spectrum rendszerelemei. A rendszerkivezetések. A ZX-printer. A Spectrum ROM.

A függelék a Z80 mikroprocesszor utasításkészletét tartalmazza.

Rucz Lajos:
Rutinról rutinra.
Bepillantás a Sinclair Spectrum gépi kódú világába
(Budapest, 1986.
LSI ATSZ,
129 oldal. Ára: 149,— Ft.)

A könyvet a szerző elsősorban azoknak szánja, akik a ZX—Spectrum tulajdonosaként már megismerkedtek a gép kezelésével, a BASIC nyelvvel, és néhány játékprogram lejátszása után szeretnének a számítógéppel kapcsolatban mélyebb ismeretekre szert tenni.

A logikusan felépített könyv segít eloszlatni mindazoknak az idegenkedését, akik nem tudták eddig rászánni magukat, hogy megtanulják a gépi kódú programozást. Az ismerkedés megkönnyítésére az első fejezet részletes áttekintést ad az alapismeretekről és a Z80 processzor működéséről. A második fejezetben az olvasó választ kap arra a kérdésre, hogy mikor előnyösebb a BASIC helyett a gépi kódot használni. A harmadik rész a felhasználói programokban gyakran előforduló rutinokat ismerteti, így a Beta Basic és a Master Toolkit rutinjait is.

Tokodi Jenő:
A Laser mikroszámítógép-család
(Budapest, 1986.
LSI ATSZ,
144 oldal. Ára: 140,— Ft.)

A kötet a Laser mikroszámítógépek bemutatásán túl általános mikroszámítógépes ismereteket is nyújt. Az egyes témakörök feldolgozásánál mindig az általános elvekből indul ki, és az általánosan alkalmazható módszereket tárgyalja az utasítások és eljárások kapcsán. A Laser mikroszámítógép adta lehetőséget kihasználva a könyv anyaga alkalmas arra, hogy az olvasók áttekinthessék a „home computer" kategóriájú gépek jellemzőit, továbbá a beépített BASIC microsoft sajátosságai miatt megismerkedjenek az IBM PC/XT és az Apple II gépek BASIC programozásával.

A függelékben az olvasó kapcsolási rajzokat talál, melyek alapján házilag, itthon kapható alkatrészekből memóriabővítőt, botkormányt, kazettás egységet építhet.

Szuper metró

Automatikus vonatvezetési rendszert állítanak munkába a budapesti metró észak—déli, vagyis a 3-as, kék vonalán. Az ún. Interelektr vonatszabályozási rendszert a BKV és a francia Matra konszern szakemberei közösen dolgozták ki. Lényege: a metrószerelvények haladása állomástól állomásig, az indulástól a következő megállásig automatikus, nincs szükség a motorkocsi vezetőjének beavatkozására. Bármilyen veszélyhelyzetben — amiről a biztosító berendezésen keresztül kapott információból értesül az automatika — megállítja a vonatot. A mikroprocesszor által vezérelt fedélzeti rendszert különféle érzékelők tájékoztatják az éppen aktuális helyzetről, s ez kiváltja a beavatkozást.

Az új megoldás előnyei közé sorolják, hogy emelkedik a szerelvények szállítókapacitása, nő az utazási kényelem. Növekszik a forgalom biztonsága, és a becslések szerint 10-20 százalék energiamegtakarítás érhető el. Ezenkívül a metrókocsik személyzetét egy főre lehet csökkenteni.

Diétázóknak, fogyózóknak

Akiknek szigorúan be kell tartaniuk diétájukat, jól tudják, milyen nehéz úgy összeállítani a napi étrendet, hogy az változatos is legyen, és megfelelő mennyiségben tartalmazzon szénhidrátot, kalóriát, különféle vitaminokat, ásványi sókat. A Diabéteszes Társaság által Commodore 64-re kidolgozott program ezen a gondon próbál segíteni. Továbbra is az orvos határozza meg, hogy a páciens mennyi szénhidrátot, kalóriát fogyaszthat egy nap. A diabéteszeseknek például hatszor kell enniük naponta, és az sem mindegy, hogy egy étkezésre mennyi kalória, szénhidrát jut. Ezen adatok birtokában lehet összeállítani a napi étrendet. A számítógép sorra veszi az étkezéseket, s közben menüből lehet választani, mit is akarunk fogyasztani.

A program jó szolgálatot tehet a kórházaknak is: mentesítheti az orvosokat, diétásnővéreket a betegek étrendje összeállításának rutinmunkájától. Már készítik a programnak azt a változatát is, mely egészséges emberek, sportolók és a fogyni kívánók számára ad étrendvariánsokat.

Itthon: új C16 – IBM PC illesztő

Már sokan vállalkoznak a C64-nek az IBM PC-vel való összekapcsolására, de a C16 illesztése újdonságszámba megy. Elsősorban az IBM PC-vel kompatíbilis nagyobb számítógépeknek az egyszerűbb adatrögzítési és szövegszerkesztési munkák alól történő tehermentesítése céljából fejlesztette ki ezt a kapcsolatot a Data Manager Kisszövetkezet. Ez a termék három programból és egy illesztőből áll.

A Text nevű szövegszerkesztő program segítségével a C16-on lehet szövegeket bevinni vagy módosítani. Az így előállított szöveg kazettára vagy hajlékonylemezes tárolóra írható ki. A C16-on működő adatátviteli programmal lehetőség van az állományok illesztőre küldésére, illetve az illesztőn érkező állományok fogadására. Az IBM PC kompatíbilis gépen működő másik program kezeli az illesztőn érkező vagy átmenő jeleket.

A mintegy 30 ezer forintos illesztőnek már elkészült az idén egyre jobban terjedő Commodore Plus/4 és IBM PC közötti kapcsolatot biztosító változata is.

Átszakította a vonalat: első a magyar vonalkódolvasó!

A korszerű áruházi rendszerekben ma már világszerte elterjedt a vonalkódos árujelölés. A számítógépekhez csatlakoztatott vonalkódolvasó ceruzával dolgozó gyors pénztárosok látványa már hétköznapi dolognak számít a Skála-Metró áruházban is. Ezek az olvasók azonban még tőkés importból származnak.

A szocialista országok első vonalkódolvasó ceruzáját a hazai Szint Kisszövetkezet fejlesztette ki. Az ősszel nagy sikerrel bemutatott termék alkalmas mind a hagyományos, nyomdai úton előállított, mind pedig a hőnyomtatós papírra készített vonalkódok olvasására. Szabványos RS232 illesztőjén keresztül 300—9600 baud sebességgel tudja továbbítani a jeleket, s az illesztőn lévő tetszőleges feszültségnél alkalmazható. Elkészült a vonalkódolvasónak egy IBM PC billentyűzetfelülete is, amely minden szoftver nélkül bármely olyan meglévő rendszerhez alkalmazható, amely eddig billentyűzetről fogadta az adatokat.

Az olvasó intelligenciáját egy Z80 típusú mikroprocesszor biztosítja, s egy 64 kbájt méretű szoftver szolgál a vonalkódok felismerésére. A készülék több vonalkódtípus olvasására alkalmas, így például a Bar-kód, az EAN (európai) és az UPC (USA-beli) szabvány szerintiekre is.

A vonalkódolvasó kifejlesztésénél fontos szempont volt a csaknem százszázalékos olvasási biztonság mellett a ceruza ütésálló kialakítása. A tapasztalatok szerint ugyanis az áruházakban alkalmazott vonalkódolvasók legtöbbször a leesésük következtében romlanak el. A hazai változatnál magában a ceruzában csak a legszükségesebb és ütésre nem érzékeny elemeket helyezték el, a kényesebb alkatrészek az illesztőben kaptak helyet.

Már folyik a vonalkódolvasó továbbfejlesztése. Készül a pásztázó olvasósugarú változat. Ez esetben a pénztáros csak ráhelyezi az olvasófejet a vonalkódra, és az egy mozgó sugárral „letapogatja" a vonalkódot. A másik fejlesztési irány az igen vékony, 7 mm átmérőjű ceruza kifejlesztése.

A mindössze 20 márka tőkés anyaghányadot tartalmazó vonalkódolvasó hazai ára ötvenezer forint alatt van. Összehasonlításul meg kell jegyezni, hogy az NSZK-ban jelenleg forgalomban lévő vonalkódolvasó ceruzák ára 1700 márkánál kezdődik.

Osztrák Szilícium-völgy?

Salzburg körzetében a Siemens cég leányvállalata szoftverfejlesztő központot létesít. A telephely megválasztását az indokolta, hogy a salzburgi egyetemen új számítástechnikai intézetet alapítottak. A központ munkájának megindulását a salzburgi városi tanács 400 ezer schillinggel támogatja.

Az első szakaszban a szoftvertermelés 10-20 munkahellyel kezdődik. A továbbiakban a kutatási és oktatási tevékenység felfutásától függően fogják a központ létszámát növelni. Maga az építkezés fél évvel ezelőtt kezdődött meg.

Hazai videoprocesszorok

A Datacoop Kisszövetkezet gyors képfeldolgozó rendszert fejlesztett ki, mely alkalmas tetszőleges forrásból érkező videojel digitalizálására és a digitalizált képek tárolására memóriában. A bemenő, folyamatosan digitalizált kép és a memóriában tárolt kép is megjeleníthető ellenőrző képernyőn.

A beépített mikroszámítógép lehetővé teszi a tárolt kép (vagy képek) feldolgozását. Ez legegyszerűbb esetben a tárolt kép mátrixnyomtatón való kinyomtatását jelenti. A csatlakoztatott megjelenítő és billentyűzet segítségével a mikrogép számára vezérlőutasítások adhatók, így a kinyomtatott kép alá feliratok írhatók, sőt a képtartalom is módosítható.

Számtalan — egészségügyi, műszaki, építőipari — alkalmazási területe közül a textil- és a ruhaipari felhasználási lehetőségeket vázoljuk:

— Rajzok, textilminták, szabásminták tervezése, módosítása, optimalizálása;

— A megmunkáló gépen lévő álló vagy mozgó anyagoknak, illetve munkadaraboknak a gyártás folyamatában, annak megszakítása nélkül történő minőség-ellenőrzése;

— Automatizálási, szabályozástechnikai alkalmazás (optikai vezérlés, szabályozás);

— Gyors folyamatok „sorozatfényképezése", a képek kiértékelése;

— Káros rezgések, a rendellenes működés érzékelése;

— Infrakamerával a káros helyi melegedés kimutatása.

A különböző alkalmazásokhoz természetesen más-más berendezéskiépítés és szoftverellátás tartozik; a fejlesztés folyamatos.

A rovatvezető — a képfeldolgozó által elemi pontjaira bontva

VÁRJUK ÖNÖKET A μ '87

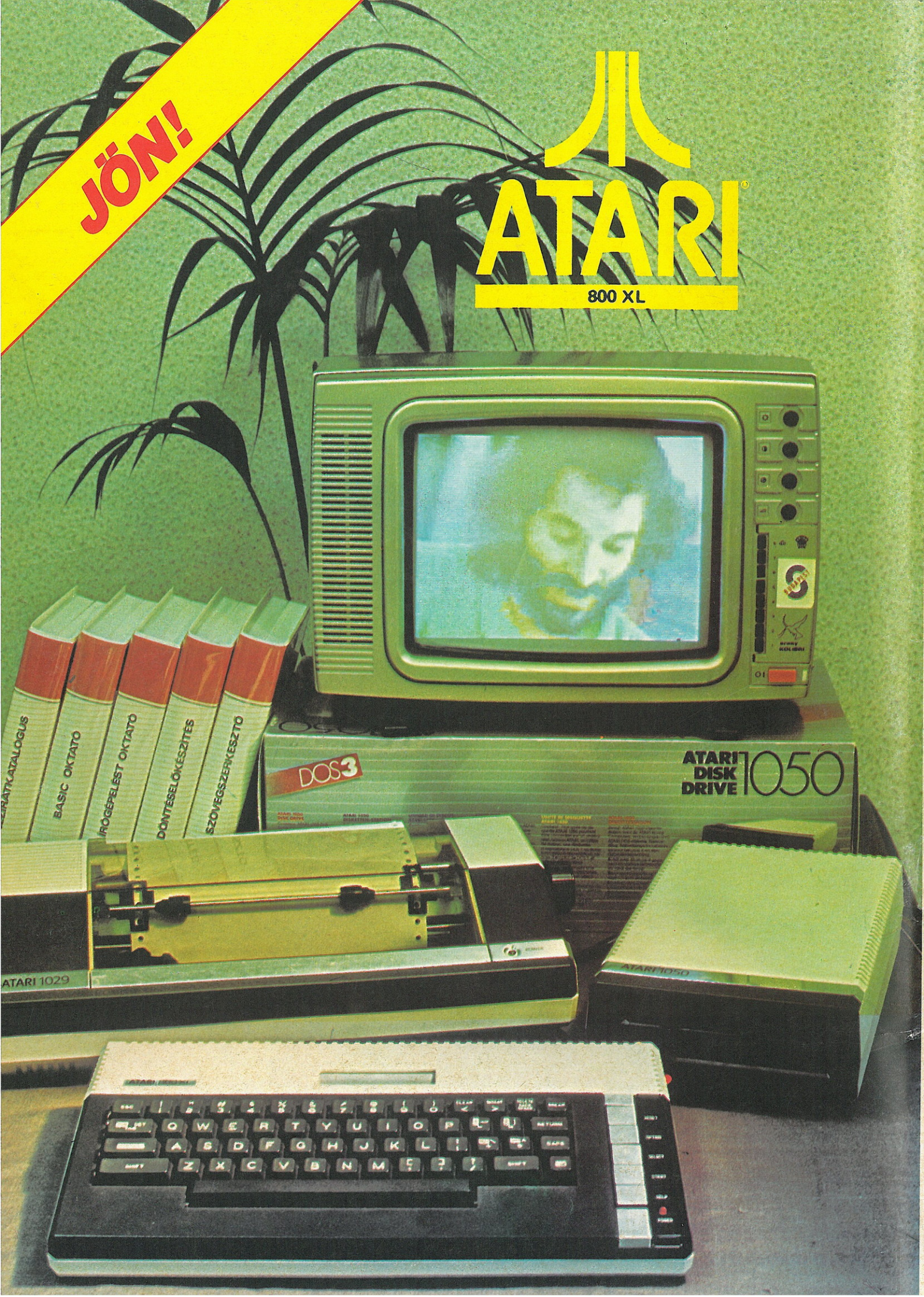
KIÁLLÍTÁS

25. PAVILONJÁBAN!

KERESSÉK A

FOTO-ELEKTRONIK

EMBLÉMÁT!


JÖN!
ATARI
800 XL
BASIC OKTATÓ
DÖNTÉSELŐKÉSZÍTÉS
SZÖVEGSZERKESZTŐ
DOS3
ATARI DISK DRIVE 1050
ATARI 1029
ATARI 1050