MÁGORINÉ HUHN ÁGNES
PUSKÁS ALBERT

Számítógép az általános iskolában

Módszertani Közlemények Könyvtára 10.
Tanárok Könyvtára 1.
Szeged, 1986.

MÁGORINÉ HUHN ÁGNES
PUSKÁS ALBERT

Számítógép az általános iskolában

1986

TANÁROK KÖNYVTÁRA 1.
MÓDSZERTANI KÖZLEMÉNYEK KÖNYVTÁRA 10.
Szerkesztette: DOBCSÁNYI FERENC
Technikai szerkesztő: BÁLINT JÓZSEF

Lektorálta:
DR. MAKAY ÁRPÁD
kandidátus, egyetemi docens
DR. SZENDREI JÁNOS
kandidátus, főiskolai tanár

ISBN 963 01 7883 4
ISSN 0237—9651

TARTALOMJEGYZÉK

ELŐSZÓ

Örvendetes, hogy az általános iskoláinkban egyre több helyen van — s reméljük még több helyen lesz — személyi számítógép. Az utóbbi években ezek elterjedése miatt a könyvpiacon is sok, a BASIC nyelvet, illetve különböző személyi számítógéptípusokat ismertető könyv jelent meg. Úgy ítéljük azonban, hogy az általános iskolai tanárokat e témába bevezető és munkájukat segítő könyv eddig kevés jelent meg. A számítógépek elterjedésében nagy szerepe van a játékoknak, de csak ebben az irányban keresni a fejlődés, és a fejlesztés irányát, nem lenne szerencsés. A számítógép szerepe a napi munkában, a termelő és nem termelő szférában egyaránt növekszik, s lehetőségeinek kihasználásában csak a kezdet kezdetén vagyunk. Az általános iskolai anyag is számos alkalmazási lehetőséget kínál a számítástechnikai alapismeretek elsajátításához. A tantárgyak közül elsősorban a matematika és a természettudományi tárgyak (fizika, kémia), valamint a technika keretében van lehetőség számítástechnikai alkalmazásokra. Egyre növekszik azonban a számítógép szerepe az idegen nyelvek tanításában is.

A jelen kötet első részében a BASIC programozási nyelv bevezető, alapvető ismereteit tartalmazza. Ezt követően olyan témákat dolgoz fel, amelyek a matematikán belül a tanulói, illetve a tanári munkát segítik a számítógép felhasználása révén. Majd hasznos példákon keresztül mutatja be a számítógép felhasználásainak lehetőségeit a természettudományi és humán tárgyak területén, valamint az iskolai adminisztrációs munkában. Végül a felhasznált és ajánlott irodalom felsorolása segíti az olvasót ismereteinek további bővítésében.

A Módszertani Közlemények Könyvtára keretében — az egyes kötetek tartalmától és célkitűzésétől függően — elsősorban az általános iskolákban működő tanárok számára írt köteteket jelentetünk meg Tanárok Könyvtára sorozat formájában, amelynek ez az első kötete.

Reméljük, hogy ezen a módon is szolgálhatjuk a pedagógusok szakmai továbbképzését.

Szeged, 1986. augusztus

Dr. Szendrei János

1. SZÁMÍTÓGÉP AZ ISKOLÁBAN

1970-ben készült el a számítástechnikai oktatás programja. A program meghatározta a képzés célját és tartalmát, a képzésben résztvevő oktatók felkészítésének ütemét, a szükséges számítógépeket. Az általános és középiskolai tanulók részére a program az algoritmikus gondolkodásmódra nevelést, a számítástechnikai alapok megismerését irányozta elő.

Az 1980-as évek elején a mikroszámítógépek, a személyi számítógépek tömeges megjelenésével kedvezően változott a helyzet: kidolgozást nyerhetett egy iskolaszámítógép-program. A program első lépésként az 1982/83. tanévben a középfokú oktatási intézményeket látta el számítógépekkel, és — elsősorban szakköri foglalkozások keretében — a számítástechnikai alapismeretek oktatását tűzte ki célul. A középiskolai program folytatása mellett az 1985/86. tanévben megindulhatott az általános iskolák kísérleti számítástechnikai programja is. 1985-ben — ha nem is tömegesen — de megkezdődött az általános iskolák számítógéppel való ellátása is. A közoktatásba kerülő számítógépek a HT—1080Z, PRIMO és C—16 gépek lettek.

Az iskolaszámítógép-program megvalósítása során elkezdődtek, majd folytatódtak a pedagógiai kísérletek egyrészt a számítógépek iskolai felhasználásáról a számítástechnikai oktatás megvalósítása érdekében, másrészt a számítógépnek a tanítás-tanulás folyamatában való alkalmazásáról. A hazai és nemzetközi tapasztalatok felhasználásával e kutató munka irányítói és végrehajtói elsősorban az Országos Pedagógiai Intézet, a Tudományszervezési és Informatikai Intézet és a pedagógusképző intézmények voltak.

Az OPI az általános iskolai számítástechnikai program egyik fő feladatául egy olyan módszertan kidolgozását tűzte ki, amely a számítástechnikai módszerek alkalmazását beépíti a különböző tantárgyakba,

megismertetve a számítógép alkalmazási lehetőségeit, hogy ezáltal elősegítse a tantárgyi ismeretanyag megértését a személyi számítógép oktatógépként vagy szimulálóeszközként történő alkalmazásával. A feladat megvalósítása érdekében a fő cél azoknak a lehetőségeknek, módszereknek a feltárása és kidolgozása volt, hogy hogyan lehet a személyi számítógép alkalmazási területeit a tantervi anyaggal és követelményekkel úgy összhangba hozni, hogy a tanítási-tanulási folyamatban javulást, időmegtakarítást érhessünk el, a tanulók problémamegoldó készségét, gondolkodás módját pedig kialakíthassuk, fejleszthessük.

A fenti feladatok és célok megvalósításához kívánunk e kiadvánnyal is hozzájárulni, melyet elsősorban az általános iskolai matematika tanárok számára állítottunk össze, de példákat mutattunk be más tárgyakat tanítók részére is.

A számítógép tanítási órán való alkalmazásának alapelveiből mindenek előtt kiemeljük, hogy a szaktanárnak két alapvető cél elérése érdekében kell döntésre jutnia. Az egyik:

a tanuláselméleti cél, vagyis mennyire alkalmas a számítógép

az új ismeretek feldolgozására,
az új ismeretek demonstrálására,
az új ismeretek szimulálására,
valamint az ismeretek gyakorlására, gyakoroltatására.

A másik: *a tananyagtaxonómiai cél,* vagyis milyen kritériumrendszer alapján választható ki a tantervi anyagból az az anyagrész, melynek számítógépes feldolgozása hatékonyabbá teheti a tanítási-tanulási folyamatot.

Kiadványunk további fejezeteit ezen általános pedagógiai feladatok és célok megvalósítása alapján állítottuk össze, természetesen nem mindenre kiterjedően, csupán csak példákkal illusztrálva gondolatainkat, ajánlásainkat.

Befejezésül óva intünk mindenkit a számítógép mindenáron való alkalmazásától, idézve — kissé átfogalmazott formában — Szűcs Pál (Az audiovizuális oktatás hatékonysága c. könyvéből vett) szavait:

„A tanítási-tanulási folyamatban a pedagógusé volt mindig a vezető szerep, és az is marad. ... ha a 'számítógépet' komplex módon használjuk fel a tanítási-tanulási folyamatban, akkor az oktatási folyamat hatékonysága jelentősen növelhető. A médiumok számának növelésével azonban nem növekszik automatikusan a hatékonyság, sőt az is lehetséges, hogy egyes témakörökben a hagyományos 'oktatási' módszer eredményesebb lehet, mint a 'számítógép' alkalmazása."

2. ÖSSZEFOGLALÓ

a BASIC programozási nyelvről

A magyar közoktatásban a BASIC nyelvű személyi számítógépek terjedtek el, ezért e fejezetben szükségesnek tartjuk ennek a nyelvnek összefoglalását.

A BASIC nyelv megalkotója J. Kemeny magyar származású, amerikai matematikus (Dortmouth College, USA, 1963.). A BASIC angol szó, magyar jelentése alap, így a BASIC nyelv alapnyelvet jelent. Másrészt mint a „Beginner's All-purpose Symbolic Istruction Code" mozaik szava, a „kezdők általános célú szimbolikus nyelv"-ét jelenti. E nyelv *magas szintű* (a változók azonosítóival dolgozik), *interaktív* (párbeszédes üzemmódú), *univerzális* (bármely algoritmus megfogalmazható a nyelven) és *széleskörűen hozzáférhető* (a programok lefuttathatók bármely — BASIC-interpreterrel rendelkező — számítógépen).

A BASIC nyelvnek nincs általánosan elfogadott szabványa, így az egyes gépi reprezentációk igen széles skálájú specialitásokat alkalmaznak (pl. grafikus-, szín-, hang-kijelzések; perifériák kezelése stb.). Mi most csak a nyelv alapértelmezését foglaljuk össze, és ezeket használjuk a további fejezetekben bemutató programjainkhoz is, megjegyezve azonban, hogy milyen típusú gépre készültek, s az egyes gépi specialitások milyen jobb lehetőséget szolgáltatnak.

A nyelv szabályaiból a következőket foglaljuk össze:

1. Adatok, változók, műveletek
2. Parancsok
3. Utasítások
4. Függvények

2.1. Adatok, változók, műveletek

Alapvetően két adattípust különböztetünk meg: a szám típusú (numerikus) és a szöveg típusú (alfanumerikus vagy string (= füzér)) adatokat.

A számokat két írásmódban írjuk:

a) előjel, egészrész, tizedesjel, törtrész

pl. −13.25

b) előjel, egészrész, tizedesjel, törtrész, kitevőrész, ahol a kitevőrész felbomlik még a kitevőjelre, kitevő előjelre, kitevő alakra,

pl. −13.25 E−06

Megjegyzés: a számokban előfordulható jelek hosszáról, az írás engedményeiről és korlátjairól az egyes gépi változatok eltérően gondoskodnak (pl. a számjegyek száma 6 és 15 közötti, a pozitív előjel elhagyható stb.).

Szöveg típusú adatoknak az idézőjelek közé zárt tetszőleges karakterek sorozatát nevezzük,

pl. „KIS IMRE"

A karaktersorozat megengedett hossza gépenként változó (általában 16-tól 4096-ig).

Az adattípusoknak megfelelően kétféle változó típusról beszélünk: numerikus és string változókról. Megkülönböztetésükre a $ jelet használjuk, így

A egy numerikus változó,

A$ egy string változó

azonosítója. (Azonosítóul betűvel kezdődő és betűvel vagy számjeggyel folytatódó karaktersorozat használható, pl. AA, A1 stb. Általában a gépi reprezentációk csak az első két karakter szerint különböztetik

meg a változók neveit). Mindkét változó típusban megengedett az egy és két dimenziós tömbök használata pl.

V(10), M(5,3)

illetve

K$(5), L$(35,5)

A numerikus változókra értelmezve vannak:

az aritmetikai műveletek

összeadás	jele	+	
kivonás	jele	−	
szorzás	jele	*	
osztás	jele	/	
hatványozás	jele	↑	(vagy **),

a relációk

egyenlő	jele	=	
kisebb, mint	jele	<	
nagyobb, mint	jele	>	
kisebb, mint vagy egyenlő	jele	<=	
nagyobb, mint vagy egyenlő	jele	>=	
nem egyenlő	jele	<>	

a logikai műveletek

és	jele	AND	
vagy	jele	OR	
negáció	jele	NOT	.

A műveletek elvégzési sorrendjéről a zárójel-, a precedencia- és a balról-jobbra szabályok gondoskodnak.

A string változókra az egymás után fűzés (konkatenáció) mellett a reláció műveletek vannak definiálva. (A konkatenáció jele általában a + jel).

2.2. Parancsok

A BASIC rendszer vezérlésére — a programok kezelésére, szerkesztésére, törlésére, tárolására, betöltésére stb. — szolgálnak a BASIC parancsok. Ilyenek:

AUTO — automatikusan előállítja a soron következő sorszámot.

BREAK — megszakítja a program futását.

CLEAR — helyfoglalás szöveg típusú változók részére, vagy változók nullázása.

CLOAD
LOAD — a programnak mágnesszalagról való beolvasása (idéző jelben tett névvel ellátva, az ilyen nevű program beolvasását jelenti).

CLOAD?
TEST
VERIFY — a mágnesszalagra írt program ellenőrzése (idézőjelben tett névvel ellátva, az ilyen nevű program ellenőrzése).

CLS — törli a képernyőt.

CONT — a megszakított programfutás folytatása. A program onnan indul újra, ahonnan kilépett. Nem funkcionál, ha a programon változtatást idézünk elő.

CSAVE
SAVE — a program kiírása mágnesszalagra. (Idézőjelben tett névvel ellátva, az ilyen nevű program kiírását jelenti. Egyes gépeknél nevet kötelező használni.)

DELETE — programsorok törlése. (A parancsszót a törlendő utasítások sorszámai követik: pl. DELETE 120, DELETE 120—200, DELETE — 200.)

EDIT — egy utasítás javítása. (Az utasítás sorszáma követi a parancsszót.)

LIST — a memóriában lévő program képernyőre listázása. A következő esetek lehetségesek:

LIST	a teljes program,
LIST 20—	a 20. sorszámtól,
LIST —100	a 100. sorszámig,
LIST 20—100	a 20. sortól a 100. sorig való listázás.

NEW — törli a programot a memóriából, nullázva a változók értékét. (Egyes gépeknél a képernyőt is törli.)

RE — újra sorszámozza a programot, a következő értelemben:

RE	a 10. sorszámtól 10-es növekménnyel,
RE 200	a 200. sorszámtól 10-es növekménnyel,
RE 15,5	a 15. sorszámtól 5-ös növekménnyel számoz át.

RUN — a program futtatását teszi lehetővé. Követheti egy sorszám, ekkor a futás ettől a sorszámtól indul. (A program változói egyidejűleg nullázódnak.) (Ha ezt el akarjuk kerülni indítsuk a programot GOTO utasítással. (lásd. később))

SYSTEM — áttérés gépi szintű programozásra. (Az áttérés után — általában — a következő lehetőségek közül választhatunk:
gépi kódú program betöltése szalagról,
gépi kódú program végrehajtása,
BASIC bővítés aktivizálása,
gépi kódú monitor aktivizálása.)

TRON — bekapcsolja a nyomkövető üzemmódot. A program futása során a képernyőre a végrehajtás sorrendjében kiíródnak az utasítások sorszámai. A programok ellenőrzése során alkalmazzuk.

TROFF — kikapcsolja a nyomkövetést.

Végezetül megjegyezzük, hogy a fenti parancsok többsége utsításként is felhasználható, továbbá minden kiadott parancsot egy, a gépen lévő kezelő gombbal kell funkcionáltatni, ennek felirata gépenként változó:

CR
ENTER
NEW LINE
RETURN

billentyűk egyike.

2.3. Utasítások

A BASIC utasítások hatására a gép műveleteket hajt végre. Az utasítás sorokat sorszámozzuk, ezzel a végrehajtási sorrendet írjuk elő. A sorszám bármely pozitív, legfeljebb négyjegyű (gépenként változó) egész szám lehet. A sorszámok közötti növekedés tetszőleges. (Általában 10-esével sorszámozunk, lehetőséget nyújtva arra, hogy két utasítás közé újabbakat szerkeszthessünk.) Egy sorban egy vagy több utasítást írhatunk. Utóbbi esetben ezeket kettősponttal kell elválasztani. Egy utasítássorban maximálisan általában három karaktersor (gépenként különböző számú billentyű leütés: 120, 132, 192 stb.) állhat.

A programutasításokat a következő csoportokban szokás felosztani:

a) műveleti
b) szervező
c) vezérlő
d) ciklus
e) szubrutin
f) deklaráló
g) grafikus

utasítások. Összefoglalásukat mi is e felosztásban adjuk meg. Az utasítások definiálásához és értelmezéséhez felhasználjuk az alábbi egyszerű jelöléseket, rövidítéseket:

NAGYBETŰK — BASIC alapszavak (utasításszavak) változatlanul írandók

kisbetűk	— rövidítések vagy fogalmak, helyükre a megfelelő információ írandó
{ }	— a benne felsoroltak közül kötelező egyet és csak egyet megadni
[ ]	— a benne felsoroltak elhagyhatók
...	— az előző [ ]-ben lévők ismétlődhetnek
s	— sorszám
n	— konstans
sv	— skaláris-változó
v	— változó
r	— relációjel
k	— kifejezés
e	— elem
sz	— szöveg
␣	— szóköz
u	— utasítás

a) Műveleti utasítás

```
[LET] v=k
```

az utasítás hatására a változó értéke a kifejezés értékével lesz egyenlő. A változó és a kifejezés típusának meg kell egyeznie.

```
Pl. 10 A=8
vagy 10 LET A=8
vagy 10 A$ = „BASIC"
```

b) Szervező utasítások

```
INPUT [„sz";] v [, v] ...
```

az utasítás hatására a gép megállítja a program futását, és kérdőjelet ír ki. Ekkor be kell írni az input utasításban szereplő változók értékét, egymástól vesszővel elválasztva. Ha kevesebb adatot írunk be, mint ahány változót az utasításban felsorolunk, a gép ismét kérdőjelet ír ki mindaddig, míg az összes változónak értéket nem adtunk; ha pedig többet írunk be, a gép a feleslegeset nem veszi figyelembe.

Pl.

```
20 INPUT "A HÁROM OLDAL:"; A, B, C
```

vagy

```
10 INPUT X, Y (5)
```

```
READ v [, v] ...
DATA n [, n] ...
```

ez az utasitás (utasításcsoport) egy másik lehetőséget ad arra, hogy változókhoz megadott értékeket rendeljünk. A programban csak együttesen használhatók, így minden READ utasításhoz egyértelműen tartozik egy DATA utasítás (a felírás sorrendjében). Az utasításcsoport hatására a READ utasítás után felsorolt változók rendre felveszik a DATA utasításban szereplő értékeket:

pl.

```
50 READ A, B$, C
60 DATA 5.7, „OLDAL", 1 E—2
```

hatására

```
A=5.7      B$=OLDAL      C=1 E—2
```

lesz.

Az utasításcsoport alkalmazásánál a számítógéprendszer automatikusan hozzárendel a DATA utasításban felsorolt első értékhez egy adatmutatót, amely mindig a következő beolvasandó értéket figyeli, ily módon a következő példa hatása megegyező a fenti példánkéval:

```
50 READ A, B $, C
60 DATA 5.7
70 DATA "OLDAL", 1 E—2
```

Az adat-mutató (pointer) mozgatására a BASIC rendszer a következő utasítást definiálja:

RESTORE [s]

ahol s egy a programban lévő DATA utasítás sorszáma.

Ha a RESTORE után nem írunk sorszámot, akkor hatására az adatmutató a DATA utasitás első adatára áll.

Ha a RESTORE után egy sorszám áll, akkor a gép az adat-mutatót beállítja az ilyen sorszámú DATA utasítás első adatára, ezután az adatbevitel a legközelebbi READ utasításnál itt kezdődik.

Pl.

```
10 DATA 1, 2, 3, 4
20 READ X, Y
30 RESTORE
40 READ Z
50 RESTORE 70
60 READ U, V
70 DATA 2, 3
```

hatására

X=1, Y=2, Z=1, U=2, V=3

lesz.

Ha a programban több adatsor van, az adatok számozása folyamatos. Ha olyan sorszámú adatra hivatkozunk, amely nem létezik, a program futása hibajelzéssel leáll.

PRINT [;,] ... [e] [{;,} e] ... [;,]

az utasítás hatására a PRINT után felsorolt elemek értékei jelennek meg a képernyőn attól a karakterhelytől kezdve, ahol a helyőr (cursor) áll.

A ; és a , elválasztójelek a kiírás formátumát szabályozzák, a kiírás elemei pedig a következők lehetnek:

$$e = \left\{ \begin{array}{l} \text{üres} \\ k \\ \text{TAB (k)} \\ \text{"sz"} \end{array} \right\}$$

Megjegyzések:

1. Ha a PRINT alapszó után nem írunk semmit (a felsorolás üres), hatására egy soremelés történik.
2. Ha a PRINT utasítás a többi elem valamelyike követi, akkor a képernyőn megjelenik az elem értéke, pl.:

   legyen

   A = −3.5, B$ = "BUDA",

   ekkor

   ```
   50 PRINT A
   60 PRINT B$ + „PEST"
   ```

   hatására a képernyőn kiíródik

   ```
   −3.5
   BUDAPEST
   ```

3. Ha a PRINT utasításban a TAB függvényt (tabulátor függvényt) használjuk, akkor az ezt követő elem kiírása az aktuális sorban a kifejezés abszolútértékének megfelelő helyen kezdődik, pl.:

   ```
   50 PRINT TAB (20); A
   ```

utasítás hatására a −3.5 kiírása a 20. karakterhelyen kezdődik el.

4. A kiírás formátumát szabályozhatjuk a ; és , jelekkel. A pontosvessző közvetlen egymásután írást jelent. Ha a kiírandó elem szám, akkor a szám kiírása egy szóközzel zárul, pl.:

   ```
   50 PRINT 1; 2; B$; "PEST"
   ```

   hatására a számok a következőképpen íródnak ki

   ␣ 1 ␣␣ 2 ␣ BUDAPEST

Ha a PRINT utasításban vesszőt használunk, akkor a kiírás egy zónahatár átlépése után kezdődik el. A kiírás tagolhatósága miatt minden sor fel van osztva általában négy részre (pl. C16 gépnél 0—9, 10—19, 20—29, 30—39 karakter-zónákra. A PRIMO gépnél azonban csak három zóna van: 0—15, 16—30, 31—42.)

Pl.:

```
50 PRINT A, B$; PEST"
```

hatására a −3.5 az első, BUDAPEST pedig a második zónában íródik ki.

A kiírás formátumát szabályozó jelek egy PRINT utasításon belül együttesen is használhatók, pl.

```
50 PRINT "A=";A, B$ , TAB (28);"!"
```

(A kiírás formátuma meghatározható még a PRINTUSING utasítással is. A gépi reprezentációk ezt az utasítást annyira eltérően definiálják, hogy erre ezen összefoglalásban nem térünk ki.)

A képernyő egy megadott helyétől való karakteres kiírásra szolgál a

	PRINT ∂ mutató , {k / sz}
vagy	PRINT$ sorszám, oszlopszám, {k / sz}
vagy	CHAR, sorszám, oszlopszám, "sz"

utasítás (az első csak a HT-, a második csak a PRIMO-, a harmadik csak a C—16 gépen használható).

1. A HT gépeken a mutató értéke 0 és 1023 közötti egész szám lehet. Az utasítás hatására az elem értékének kiírása a mutató által megjelölt helytől kezdődik, pl.:

```
50 PRINT ∂2*64+10, "HAT"
```

hatására a HAT szöveg a képernyő 138. karakterhelyétől kezdve íródik ki.

2. A PRIMO gépeken a sorszám értéke 0 és 15 közötti, az oszlopszám értéke 0 és 41 közötti egész szám lehet. Az utasítás hatására az elem értékének kiírása a sor- és oszlopszám által meghatározott helytől kezdődik, pl.:

```
50 PRINT $ 2, 10, "HAT"
```

hatására a HAT szöveg a képernyő 2. sorának 10. oszlopától kezdve íródik ki.

3. A C—16 gépeken a sorszám értéke 0 és 24, az oszlopszám értéke 0 és 39 közötti egész szám lehet. Az utasítás hatására a szöveg kiírása a sor- és oszlopszám által meghatározott helytől kezdődik, pl.:

```
50 CHAR, 2, 10, "HAT"
```

```
REM sz
```

A REM ark (megjegyzés) a program listát olvasónak szól. Magyarázó szövegek és megjegyzések elhelyezésére szolgál. Bárhol elhelyezhető a program listán.

```
STOP "sz"
```

utasítás hatására a program végrehajtása megszakad, a program változóinak értéke megmarad, CONT paranccsal a program a STOP után következő utasítással folytatódik.

```
END
```

utasítás hatására a program végrehajtása befejeződik, a gép szerkesztőmódba áll vissza. Az END utasítás a program logikai végét jelzi.

c) Vezérlő utasítások

GOTO s

utasítás a feltétel nélküli vezérlés átadás, hatására a program a GOTO utasításban megadott számú sor végrehajtásával folytatódik. Használjuk arra is, hogy egy programot megadott sorszámú utasítástól kezdve újra futtassunk le ezzel a paranccsal, ekkor ugyanis a változók értékei a parancs hatására nem változnak meg.

IF logikai k THEN $\left\{\begin{matrix} s \\ u \end{matrix}\right\}$ $\left[\right.$ ELSE $\left\{\begin{matrix} s \\ u \end{matrix}\right\}$ $\left.\right]$

utasítás a feltételes vezérlés átadás, hatására, ha a logikai kifejezés értéke igaz, akkor a THEN utáni rész hajtódik végre, (ill. a THEN után megadott számú sor végrehajtásával folytatódik a program), míg ha hamis, akkor az ELSE utáni rész, (ill. az itt megadott számú sorra adódik át a vezérlés). Ha az utasítás ELSE részt nem tartalmaz, akkor a feltétel nem teljesülése esetén a program a soronkövetkező utasítással folytatódik. A feltételes vezérlést átadó utasítások egymásba is ágyazhatók. (A C—16 gépen az ELSE részt kettőspont előzi meg.)

Példák helyesen irt feltételes vezérlés átadása:

```
IF B↑2−4 * A * C > 0 THEN 20
IF A>0 THEN A=A ELSE A==A
IF A$=„I" OR A$ = "IGEN" THEN 200
IF X<YOR X<Z THEN IF Y<Z THEN PRINT Z
ELSE PRINT Y ELSE PRINT Z
```

ON k GOTO s [,s] ...

utasítás a többirányú elágazást teszi lehetővé, hatására a szám típusú kifejezés értékétől függően adódik át a GOTO részben szereplő valame-

lyik sorszámú utasításra. A rendszer a kifejezés értékének egészrészét képzi, ha ez

— negatív vagy 255-nél nagyobb, ekkor futási hiba jön létre,
— ha 1 és 255 közé eső szám, akkor a GOTO utáni ennyiediknek felsorolt sorszámon folytatódik a program,
— ha nulla, vagy nagyobb mint ahány sorszámot megadtunk, akkor a végrehajtás a soronkövetkező utasítással folytatódik,

pl.:

```
ON K GOTO 100, 200, 300, 400, 500, 600
```

hatására, ha

```
K=1 akkor 100,
K=2 akkor 200,
K=3 akkor 300,
K=4 akkor 400,
K=5 akkor 500,
K=6 akkor 600
```

számú soron folytatódik a program végrehajtása (ezektől eltérő K érték esetén a megjegyzésben foglaltak állnak elő).

d) Ciklus utasítás

```
FOR sv = k TO k [STEP k]
[u]
:
NEXT sv
```

Egy vagy több utasítás ismételt végrehajtását ciklusnak nevezzük. A FOR ... NEXT utasításpárral szervezhetjük meg az ismétlődő utasítások többszöri végrehajtását. A FOR és NEXT utasítások csak együtt alkalmazhatók. Összetartozó FOR, NEXT utasításpárnak azonos a ska-

lárisváltozója. Ciklust megkezdeni csak FOR urasítással lehet. Megengedett a ciklusok tetszőleges mélységű (számuk gépenként más és más) egymásba skatulyázása. A FOR utasításban szereplő kifejezések ceak szám típusúak lehetnek.

A FOR utáni skalárisváltozót ciklusváltozónak, az egyenlőségjel utáni kifejezést kezdőértéknek, a TO utáni kifejezést végértéknek és a STEP utáni kifejezést növekménynek (lépésnek) nevezzük. Ha a növekmény +1, akkor a STEP rész elhagyható.

A ciklus utasítás a következő módon hajtódik végre: a ciklusváltozó felveszi a kezdőértéket és a ciklus magjában lévő utasítások végrehajtódnak. A NEXT utasítással visszaadódik a vezérlés a FOR utasításra, ekkor a ciklusváltozó értéke megnő a növekménnyel és a rendszer ellenőrzi, hogy a ciklusváltozó értéke túlhaladta-e a végértéket. Amennyiben nem, akkor a ciklusmagban szereplő utasítások újra végrehajtódnak, míg ha igen, akkor a vezérlés a NEXT után soronkövetkező utasításra adódik (kilépünk a ciklusból). Vegyük észre, hogy a FOR utasításban szereplő kifejezések értékeitől függetlenül a rendszer legalább egyszer végrehajtja a ciklust! A ciklus magjában a ciklusváltozó értékét ne változtassuk meg, és a ciklusból kiugrani sem ajánlatos. Megengedett (de nem ajánljuk) a NEXT utasításban a ciklusváltozót elhagyni. (A C—16 BASIC-ben feltételes ciklus utasítás is definiált:

DO [{UNTIL / WHILE} logikai kifejezés]: [u] ...: LOOP)

Példák:

```
1. 100 FOR I = 1  TO 100
   110 PRINT I * I
   120 NEXT I
2. 100 FOR X = 1  TO 2  STEP .1
   110 PRINT X, X↑2, X↑3
   120 NEXT X
```

3.
```
100 FOR T=1  TO  3000
110 NEXT  T
120 REM  Időhúzás
```

4.
```
100 FOR I=1  TO  10
110 FOR J=1  TO  10
120 INPUT A (I,  J)
130 NEXT  J
140 NEXT  I
```

e) Szubrutin utasítások

GOSUB s

utasítás az s sorszámú utasítással kezdődő szubrutin felhívó utasítása, hatására a főprogramból a vezérlés átadódik a szubrutin első utasítására. A szubrutin utasításainak végrehajtása után a

RETURN

utasítással adódik vissza a vezérlés a főprogram GOSUB után soronkövetkező utasítására. A mai korszerű programozástechnikában elengedhetetlen ezen utasításpár használata: áttekinthetővé, jól tagolttá teszi a főprogramokat.

Megengedett a szubrutinok egymásba skatulyázása, a többirányú szubrutin hívás is. pl.:

1.
```
100 GOSUB  200
110
  :
200 REM 1.  SZUBRUTIN
210 GOSUB  300
  :
```

```
290 RETURN
300 REM 2. SZUBRUTIN
  :
390 RETURN
2. 100 ON K  GOSUB 200, 300, 400
```

f) Deklaráló utasítások

A tömbök méreteit deklarációs utasítással kell megadnunk a programban való használatuk előtt, e helyfoglalásra szolgál a

```
DIM e [, e] ...
```

utasítás, ahol

$$e = \begin{cases} \text{változónév (t)} \\ \text{változónév (t [, t] ...)} \end{cases}$$

alakú lehet és a t index értéke 0 és 255 közötti egész szám (a mátrixok elemeinek maximális száma gépenként változó). Példa:

```
10 DIM V(9), N$(35), M(24, 19)
```

utasítással helyet foglalunk egy V nevű 10 elemű szám típusú, egy N nevű 36 elemű szöveg típusú vektor és egy M nevű 25×20 elemű szám típusú mátrix részére.

Az egyes gépirendszerek a tömbök méretére alapértelmezés szerinti helyfoglalást is biztosítanak (általában $0 \leqq t \leqq 10$). A tömbök méreteit a program futása során nem lehet újra definiálni.

Egyváltozós függvények definiálására és programban való felhasználására szolgál (csak a C—16) gépen a

```
DEFFN változónév (sv) = k
```

utasítás, ahol k csak szám típusú kifejezés lehet.

```
DEFFN F (X) = X↑2 + ABS (X)
```

utasítás az $f(x) = x^2 + |x|$ függvényt definiálja a számítógép számára. Ezen deklaráló utasítás után a programban a függvényt a következő módon használhatjuk:

```
Y = FNF(—3.5)
```

g) Grafikus utasítások

A számítógépek karakterkészletében lévő grafikus jelek lehetőséget adnak arra, hogy felhasználásukkal ábrákat, alakzatokat rajzolhassunk a képernyő tetszőleges helyére.

A BASIC nyelv ezeken kívül grafikus utasításokat is definiál a képernyőre való rajzolás elősegítésére. Ezek az utasítások szintaktikailag a HT és PRIMO gépeken megegyezőek:

```
SET (oszlopszám, sorszám)
RESET (oszlopszám, sorszám)
```

A SET utasítás a képernyő adott sor- és oszlopszámú helyét kivilágítja, míg a RESET utasítás a kivilágítást törli. A képernyő grafikus szempontból a következő módon van felosztva:

HT gépen:

$0 \leqq$ sorszám $\leqq 47$
$0 \leqq$ oszlopszám $\leqq 127$,

PRIMO gépen:

$0 \leqq$ sorszám $\leqq 191$
$0 \leqq$ oszlopszám $\leqq 255$.

Példa:

```
50 FOR Y = 0 TO 255
60 SET (Y, 96)
70 NEXT Y
```

utasítások a végrehajtás során a PRIMO képernyőjének 96. sorában (közepén) minden pontot kivilágít (vízszintes egyenest rajzol), míg az

```
50 FOR X = 0 TO 47
60 SET (64, X)
70 NEXT X
```

utasítások hatására a HT gép képernyőjének 64. oszlopában minden pont megvilágítódik (függőleges egyenest rajzol).

A C—16 gépek grafikus lehetőségei sokkal szélesebbek, de bizonyos hátránnyal is járnak. Így a grafikus utasításoknak csak legegyszerűbb alakjait soroljuk fel. Mindenekelőtt megjegyezzük, hogy a képernyő a következő módon van felolsztva:

$$0 \leqq \text{sorszám} \leqq 199$$
$$\text{és} \quad 0 \leqq \text{oszlopszám} \leqq 319,$$

azaz 200×320 pont kivilágítására van lehetőség.

A grafikus területet a

GRAPHIC n

paranccsal, illetve utasítással lehet kijelölni, ahol n értéke 0, 1, 2, 3, 4 lehet. Ha

n = 0 akkor normál színű, karakteres,
n = 1 akkor normál szinű, grafikus,
n = 2 akkor normál szinű, grafikus és karakteres,
n = 3 akkor többszinű, grafikus
n = 4 akkor többszinű, grafikus és karakteres

képernyő jön létre. A grafikus és karakteres képernyő esetén a képernyő alsó öt sorában lehet a megjelenítés karakteres (természetesen ekkor $0 \leqq$ sorszám $\leqq 159$ lehet).

A grafikus terület kijelölése után lehetőség van pontok, szakaszok, téglalapok és ellipszisek rajzolására, ezek kitöltésére. Erre szolgálnak a következő utasítások:

```
DRAW [szín], o1, s1 [TO o2, s2] ...
```

ahol a szín a rajzolásra kiválasztott szín típusa, o1 és o2 az oszlopok, s1 és s2 a sorok koordinátái.

Például az

```
50 DRAW, 20, 10, TO 100, 10
```

utasítás hatására a képernyő 10. sorában egy „vízszintes" szakasz rajzolódik ki. (Ha a TO rész elmarad, akkor pl. a 10. sor 20. oszlopában lévő pont világítódik ki.)

```
BOX [szín] , o1, s1, o2, s2
```

utasítás egy téglalapot rajzol úgy, hogy a téglalap bal felső csúcsának, helye az o1, s1 koordinátájú pont, míg o2, s2 a jobb alsó csúcsé.

```
CIRCLE [szín], o, s, k, n
```

utasítás egy ellipszist rajzol az o, s koordinátájú középponttal és k kis- illetve n nagy tengely hosszal. Például az

```
50 CIRCLE, 160, 100, 20, 20
```

utasítás egy kört rajzol a képernyő közepére.

```
PAINT [szín], o, s
```

utasítással egy ábrát, alakzatot festhetünk be (tölthetünk ki), ahol o, s az alakzat egy belső pontjának koordinátái, pl.:

```
50 CIRCLE, 160, 100, 20, 20
PAINT, 160 100
```

utasításokkal a fentiekben megrajzolt kört tölthetjük ki.

Megjegyezzük, hogy a C—16 gép igen jó grafikus lehetőségeit lerontja az a tény, hogy a grafikus terület nagy mértékben lecsökkenti a BASIC munkaterületet.

2.4. Függvények

A programozási gyakorlatban bizonyos függvényeket a rendszerbe épített rutin segítségével lehet kiszámítani nevükre való hivatkozással. A függvényekre való hivatkozás formai szabálya

```
függvény-név ( k [, k] ... )
```

ahol a kifejezések szám-, ill. szöveg típusúak lehetnek. Azok a függvények, melyek értékei szövegtípusúak nevük után a $ jelet is tartalmazzák. A BASIC-interpreter csak ezeknél enged meg több változó használatát, így a többi függvény csak egyváltozós lehet.

A beépített függvények közül felsoroljuk a leggyakrabban használatosakat (csak nevükkel és funkciójukkal):

ABS — abszolút érték függvény

ASC — az argumentumban lévő string kifejezés első karakterének ASCII kódját képzi

CHR$ — képzi az argumentumban megadott kifejezés értékének megfelelő ASCII kódszám karakterét

COS — koszinusz függvény

EXP — exponenciális függvény (e alapú)

FIX — szám típusú kifejezés értékének egész számra csonkítása

INT — szám típusú kifejezés értékének egész részét képzi

LEFT$ — szöveg típusú kifejezés értékének bal oldalát választja le (kétváltozós)

LEN — szöveg típusú kifejezés hosszát képzi

LOG — logaritmus függvény (e alapú)

MID$ — szöveg típusú kifejezés értékének középső részét választja le (háromváltozós)

POINT — meghatározza, hogy a grafikus képernyő egy pontja világít-e (kétváltozós logikai értékű függvény. Ha a pont világít értéke -1, ha nem 0)

RIGHT$ — szöveg típusú kifejezés értékének jobb oldalát választja le kétváltozós

RND — véletlenszám generáló függvény

SGN — előjel függvény

SIN — szinusz függvény

SQR — négyzetgyök függvény

STR$ — szám-típusú kifejezés értékének szöveg típusú alakját képzi

TAB — a kiírási pozíciót meghatározó függvény

TAN — tangenes függvény

VAL — szöveg típusú kifejezés értékének szám típusú alakját képzi

3. SZÁMÍTÓGÉP A TANULÓI MUNKÁBAN

Ebben a fejezetben olyan feladatokat mutatunk be, amelyek alkalmasak arra, hogy általános iskolás gyerekek számítógépes programot írjanak megoldásukra. Hasznos ez olyan szempontból is, hogy látszólag nagyon különböző feladatok megoldásában felfedeztessük a hasonlóságokat, az azonos algoritmusokat.

Legtöbb programot általánosan, paraméterekkel írtuk fel, a gyerekekkel azonban célszerű először konkrét eseteket megbeszélni, adott számokkal írni fel a programot, s ha ez már megalapozott, csak akkor általánosítani.

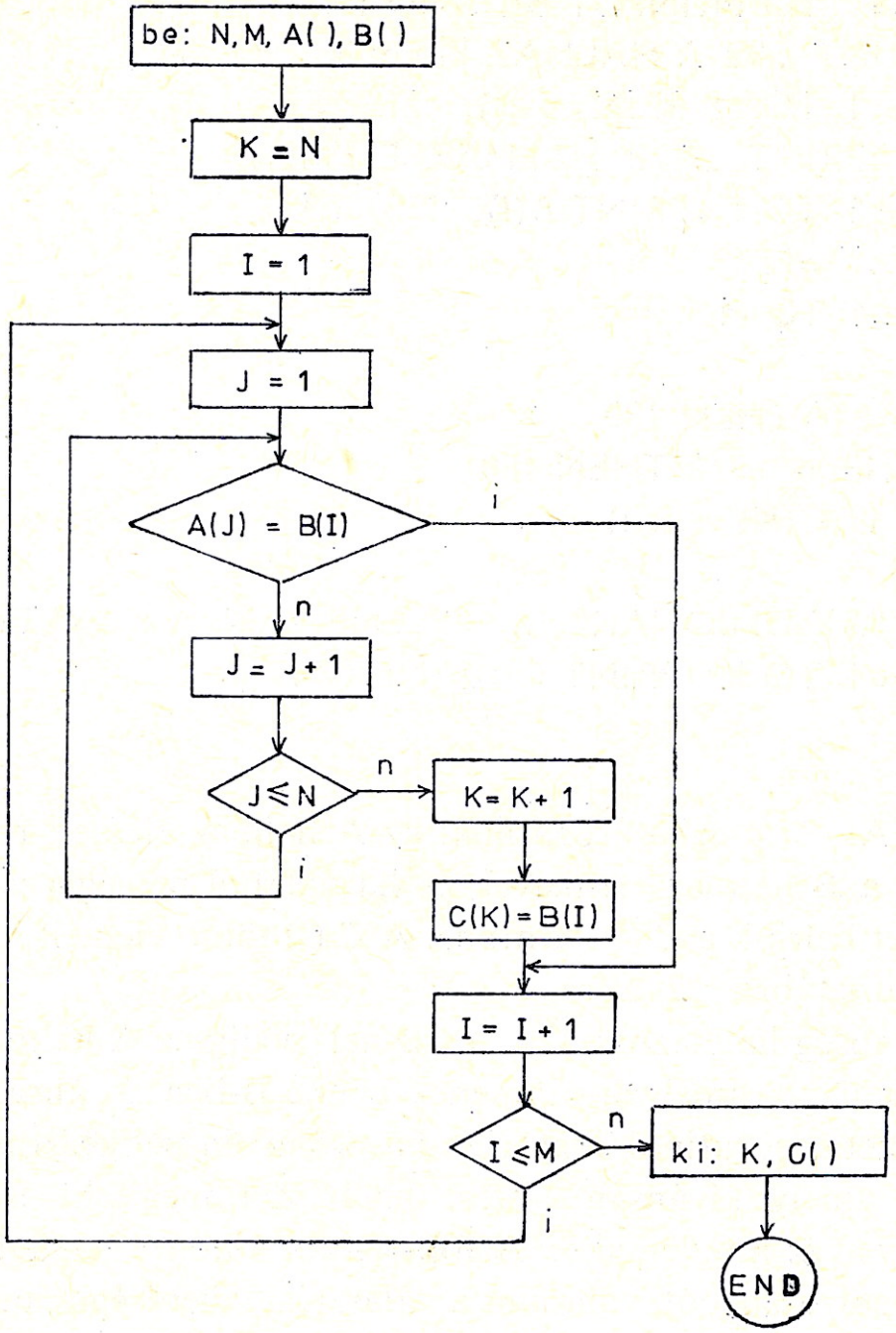
3.1. Halmazelméleti feladatok

1. Határozzuk meg adott véges számhalmazok unióját, metszetét, különbségét! Az egyszerűség kedvéért a halmazok elemei legyenek természetes számok. A halmazok elemeit az A ( ), illetve B ( ) tömbben tároljuk. Az elemek száma N, illetve M. A megfelelő halmazművelet eredményeként kapott elemeket a C ( ) tömbben helyezzük el.

A C=A∪B meghatározásánál az A halmaz elemeit már a beolvasásnál C ( ) tömbben is tároljuk, és ehhez hozzávesszük B halmaz azon elemeit, amelyek az A-nak nem elemei.

Érdemes a képernyőn kiíratni az A és B halmazok elemeit is. Ha az elemek száma nem túl nagy, akkor mindhárom halmaz elemei egyszerre a képernyőre férnek. A C halmaz elemeit meg is számoljuk, s ezt is kiíratjuk.

Folyamatábránk az 1. ábra.



1. ábra

```
 5 PRINT "KÉT HALMAZ  UNIÓJÁNAK  MEGHATÁROZÁSA"
10 INPUT "A HALMAZ ELEMEINEK SZÁMA"; N
20 INPUT "B HALMAZ ELEMEINEK SZÁMA"; M
30 DIM A(N),  B(M),  C(N+M)
40 PRINT "AZ A HALMAZ ELEMEIT KÉREM!"
50 FOR I=1 TO N: INPUT A(I):C(I)=A(I):NEXT I
60 PRINT "A B HALMAZ ELEMEIT KÉREM!"
```

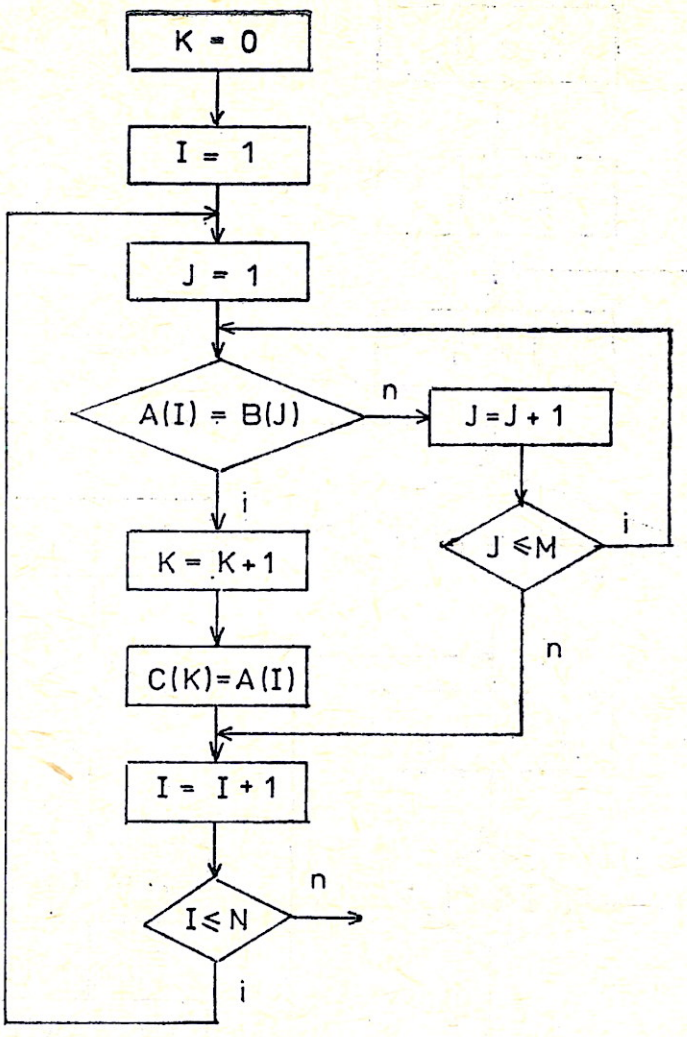
```
 70 FOR I=1 TO M:INPUT B(I):NEXT I
 80 CLS:PRINT „AZ A HALMAZ ELEMEI:"
 90 FOR I=1 TO N:PRINT A(I); :NEXT I
100 PRINT :PRINT „A B HALMAZ ELEMEI:"
110 FOR I=1 TO M:PRINT B(I); :NEXT I
120 PRINT :K=N
130 FOR I=1 TO M
140 J=1
150 IF A(J)=B(I) THEN 180
160 J=J+1:IF J<=N THEN 150
170 K=K+1: C(K) = B(I)
180 NEXT I
190 PRINT "AZ UNIÓNAK" ;K; "ELEME VAN, A KÖVETKEZŐK:"
200 FOR I=1 TO K: PRINT C(I); :NEXT I
210 END
```

A $C=A \cap B$ meghatározásánál az A halmaz elemeit rendre összehasonlítjuk a B halmaz elemeivel, és ha valahol megegyezést találunk, azt az elemet beírjuk a C ( ) tömbbe. A C ( ) tömb elemeit itt is számoljuk. A folyamatábra a 2. ábra.

A különbség halmaznál $C=A \setminus B$-nél pedig az A halmaz azon elemeit tesszük C-be, amelyek nincsenek benne B-ben. 3. ábra.

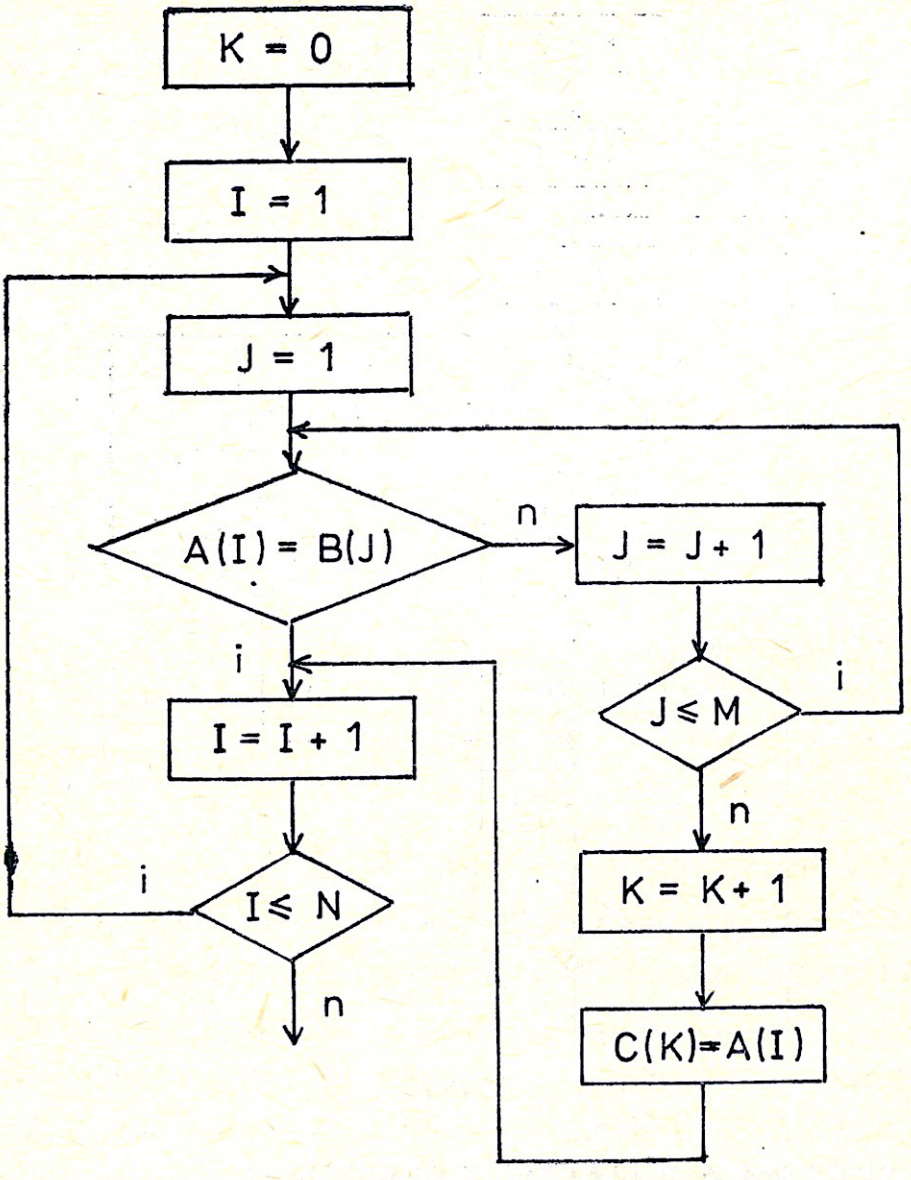
Ha ezeket szubrutinokként írjuk fel, könnyen szerkeszthetünk olyan programot, amely kívánság szerint határozza meg $A \cup B$-t, $A \cap B$-t, $A \setminus B$-t vagy akár mindegyiket. A főprogram álljon a feladat kiírásából, az adatok beolvasásából, valamint a választási lehetőségek megadásából.

```
 10 CLS: PRINT "KÉT HALMAZ UNIÓJÁNAK, METSZETÉNEK,
    KÜLÖNBSÉGÉNEK MEGHATÁROZÁSA (AZ ELEMEK TERMÉSZE-
    TES SZÁMOK.)"
 20 INPUT "AZ EGYIK (A) HALMAZ ELEMEINEK SZÁMA"; N
 30 INPUT "A MÁSIK (B) HALMAZ ELEMEINEK SZÁMA"; M
 40 DIM A(N), B(M), C(N+M)
 50 PRINT "KÉREM AZ A HALMAZ ELEMEIT!"
 60 FOR I=1 TO N: INPUT A(I):NEXT I
```



2. ábra

```
 70 PRINT "KÉREM A B HALMAZ ELEMEIT!"
 80 FOR I=1 TO M:INPUT B(I):NEXT I
 90 CLS: PRINT "MIT HATÁROZZAK MEG?"
100 PRINT "1. A ÉS B UNIÓJÁT?"
110 PRINT "2. A ÉS B METSZETÉT?"
120 PRINT "3. A ÉS B KÜLÖNBSÉGÉT? (ILYEN SORRENDBEN)"
130 PRINT "4. VAGY BEFEJEZZÜK?"
140 PRINT "ÍRD BE A MEGFELELŐ SZÁMOT!"
150 INPUT V
160 IF V<1 OR V>4 THEN 90
170 IF V=4 THEN END
180 CLS:PRINT "AZ A HALMAZ ELEMEI:"
190 FOR I=1 TO N:PRINT A(I): :NEXT I
```



3. ábra

```
200 PRINT  :PRINT  "A B HALMAZ ELEMEI:"
210 FOR I=1 TO M:PRINT B(I); :NEXT I
220 PRINT
230 ON V GOSUB 500, 700, 900
240 PRINT "HA DOLGOZHATUNK TOVÁBB ÉRINTS MEG EGY
 BILLENTYŰT!"
250 IF INKEL$="" THEN 250
260 GOTO 90
500 REM * UNIÓ *
510 FOR I=1 TO N:C(I)=A(I):NEXT I:K=N
```

```
570 FOR  I=1  TO  M
530 J = 1
540 IF  A(J) = B(I)  THEN  570
550 J = J+1:  IF  J< = N  THEN 540
560 K = K+1:C(K) = B(I)
570 NEXT  I
580 PRINT  "AZ UNIÓNAK"; K; "ELEME VAN."
590 FOR  I = 1  TO  K:PRINT  C(I); : NEXT  I
600 PRINT
610 RETURN
700 REM * METSZET *
710 K = 0
720 FOR I = 1  TO  N
730 J = 1
740 IF A(I) = B(J)  THEN K = K+1:C(K) = A(I) : GOTO 760
750 J = J+1 : IF  J< = M  THEN  740
760 NEXT I
770 PRINT  "A METSZETNEK"; K; "ELEME VAN"
775  IF K = 0  THEN 800
780 FOR I = 1  TO K : PRINT  C(I); :NEXT  I
790 PRINT
800 RETURN
900 REM * KÜLÖNBSÉG *
910 K = 0
920 FOR  I = 1  TO  N
930 J = 1
940 IF  A(I) = B(J)  THEN  970
950 J = J+1 : IF  J< = M  THEN 940
960 K = K+1 : C(K) = A(I)
970 NEXT  I
980 PRINT "A ÉS B HALMAZ KÜLÖNBSÉGE"; K; "ELEMBŐL ÁLL."
985 IF  K = 0  THEN 1010
990 FOR  I = 1  TO  K: PRINT  C(I); : NEXT I
1000 PRINT
1010 RETURN
```

A kiíratások előtt a 775 és 985 utasításokkal átugorjuk a kiíratási ciklusokat, ha a halmaznak 0 eleme van. (A BASIC ciklus egyszer mindenképpen végrehajtódik, s a gép kiírná, amit éppen C(1)-ben talál, ha ezekről az átugrásokról nem gondoskodnánk!)

2. Képezhetjük az A és B halmazok Descartes szorzatát, vagyis mindazon rendezett elempárok halmazát, ahol az első komponens A-ból a második B-ből való.

$$A \times B = \{(X, Y) | X \in A \wedge Y \in B\}$$

Vegyük sorba az A halmaz elemeit első komponensnek, s minden A halmazbeli elem esetén a második komponenst futtassuk végig a B halmaz elemein!

A halmazok elemeinek beolvasása hasonló, mint az előbb. Az algoritmus lényegi részét a 4. ábra szemlélteti. Az elempárokat most nem tároljuk, csak kiíratjuk a képernyőre. Az A halmaz N, a B halmaz M elemű.

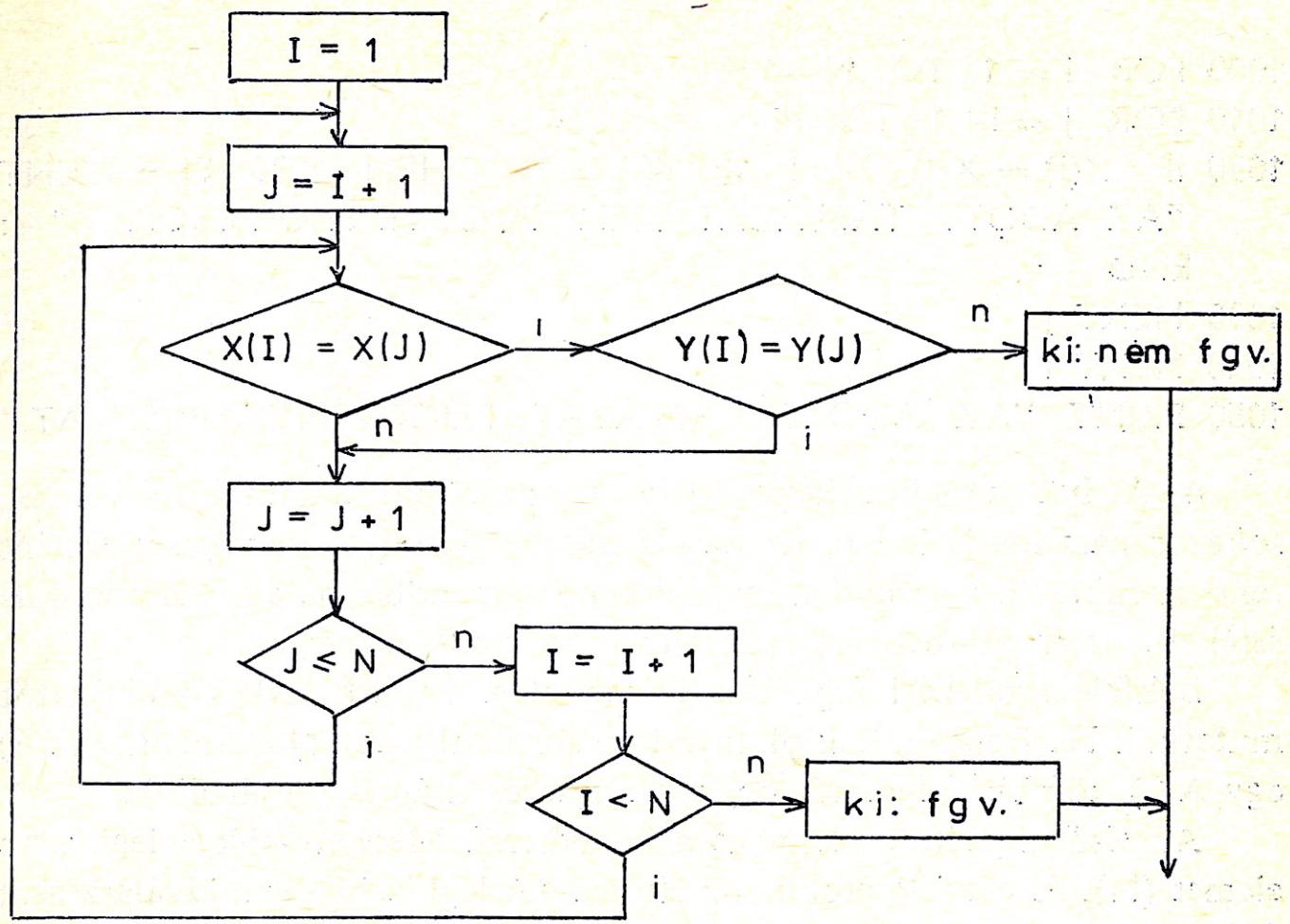
```
1000 FOR I = 1 TO N
1010 FOR J = 1 TO M
1020 PRINT "("; A(I);" , " , B(J);")",
1030 NEXT J
1040 NEXT I
```

Programrészletünket beolvasással és a kiíratás jobb szervezésével (itt zónázott formátum) egészíthetjük ki, vagy más programokban szubrutinként használhatjuk.

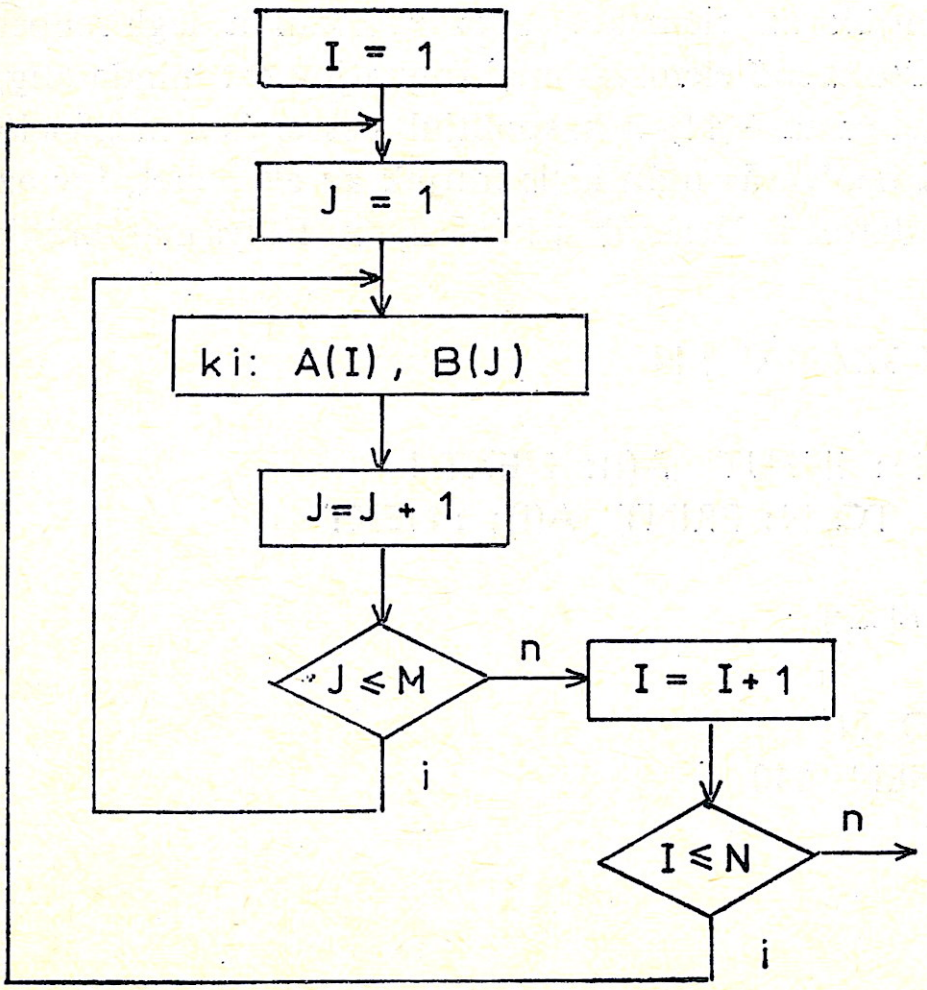
Itt említhetjük meg a következő feladatot.

3. Adott egy táblázat összetartozó (X, Y) értékpárokkal. Állapítsuk meg, kifejezhet-e függvénykapcsolatot a táblázat! Nem kell mást tennünk, mint végigvizsgálni, hogy az egyes X értékek előfordulnak-e a táblázat további elemeként, s ha igen, a hozzárendelt Y érték ugyanaz-e. Ha nem, a táblázat nem fejezhet ki függvénykapcsolatot.

Az X értékeket egy X( ) tömbben az Y értékeket egy Y( ) tömbben tároljuk, az értékpárok összetartozó tagjait azonos indexszel. A beolvasó programrészletet itt sem írjuk fel, mivel teljesen hasonló mint az előző feladatoknál. A folyamatábra az 5. ábra.


I = 1
J = I + 1
X(I) = X(J)
i
Y(I) = Y(J)
n
ki: nem fgv.
n
i
J = J + 1
J ≤ N
n
I = I + 1
i
I < N
n
ki: fgv.
i

4. ábra


I = 1
J = 1
ki: A(I), B(J)
J=J + 1
J ≤ M
n
I = I+1
i
I ≤ N
n
i

5. ábra

```
1000 FOR I = 1 TO N-1
1010 FOR J = I+1 TO N
1020 IF X(I) = X(J) THEN IF Y(I) = Y(J) THEN 1030 ELSE PRINT
     "AZ ADOTT TÁBLÁZAT NEM FÜGGVÉNYKAPCSOLAT." :
     END
1030 NEXT J
1040 NEXT I
1050 PRINT "AZ ADOTT TÁBLÁZAT FÜGGVÉNYKAPCSOLAT."
```

4. A halmazokkal kapcsolatos feladatoknál azonnal felvetődik, sokszor hasznos is lenne, ha az elemeket valamilyen szempont szerint rendeznénk. (Pl. növekvő vagy csökkenő sorrendbe, ha azok numerikus értékek, vagy alfabetikus sorrendbe, ha azok szövegek.)

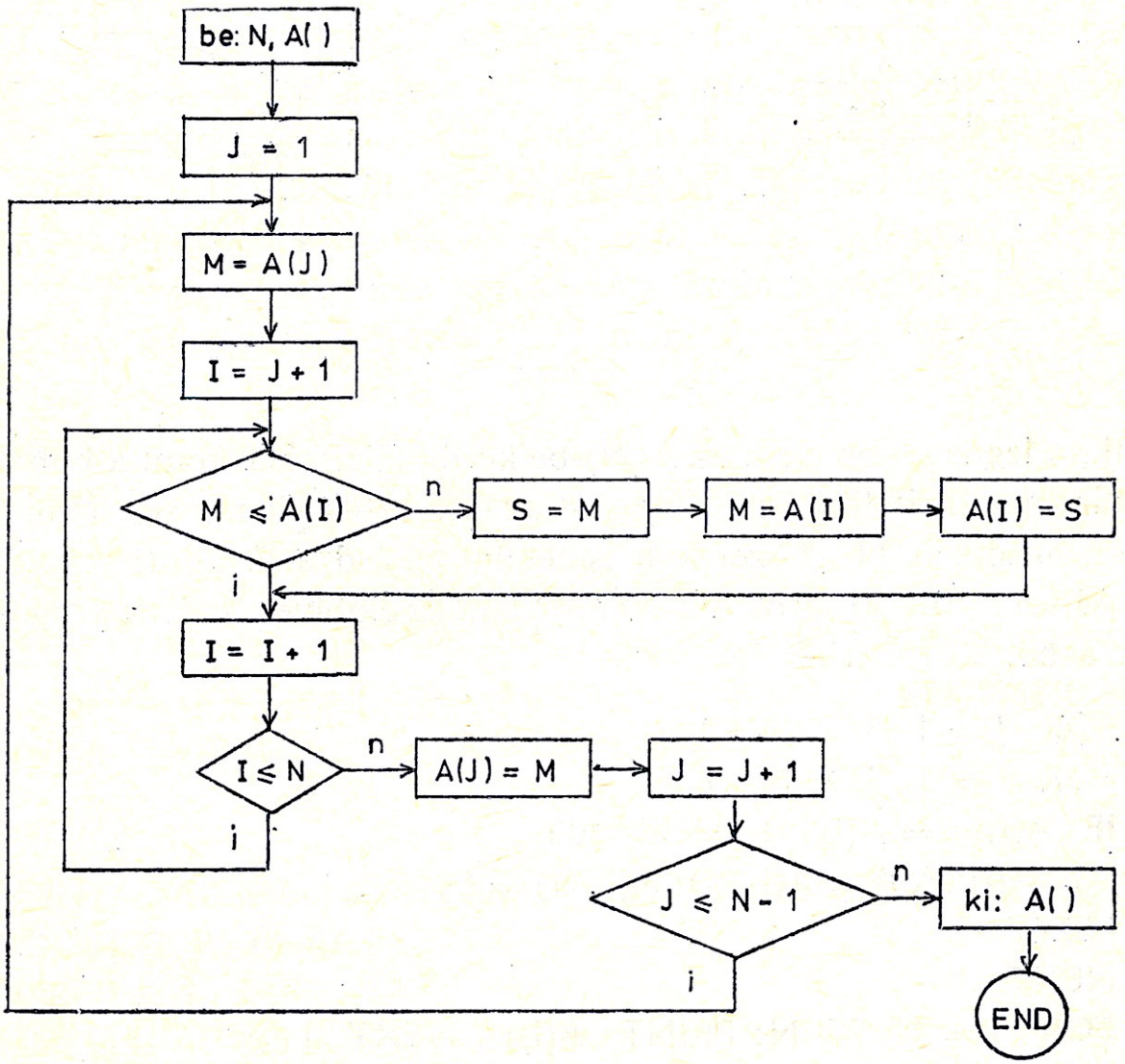
Sokféle rendezési algoritmus ismeretes. Mi itt ezek közül kettőt mutatunk be. Rendezzünk pl. növekvő sorrendbe. Az N elemet (számot) egy A( ) tömbbe olvassuk be, s a rendezés után is itt tároljuk.

Az első algoritmus lényege a következő. Megkeressük a legkisebb elemet (M), és ezt felcseréljük az elsővel. (A legkisebb elem kiválasztását érdemes megelőzően megoldani a gyerekekkel, s ezt fejleszteni tovább rendezéssé.) Ezután ezt megismételjük, csak most N−1 elemre, mivel A(2)-től A(N)-ig tekintjük az elemeket, s ezek közül a legkisebbet A(2)-vel cseréljük fel. Csökkenő elemszámmal folytatjuk ezt mindaddig, míg végül az A(N−1)-et és az A(N)-et hasonlítjuk össze, és a megfelelő sorrendben tároljuk. A beolvasás után ki is íratjuk az elemeket, így az eredeti és a rendezett adatok is láthatók a képernyőn. Folyamatábránk a 6. ábra.

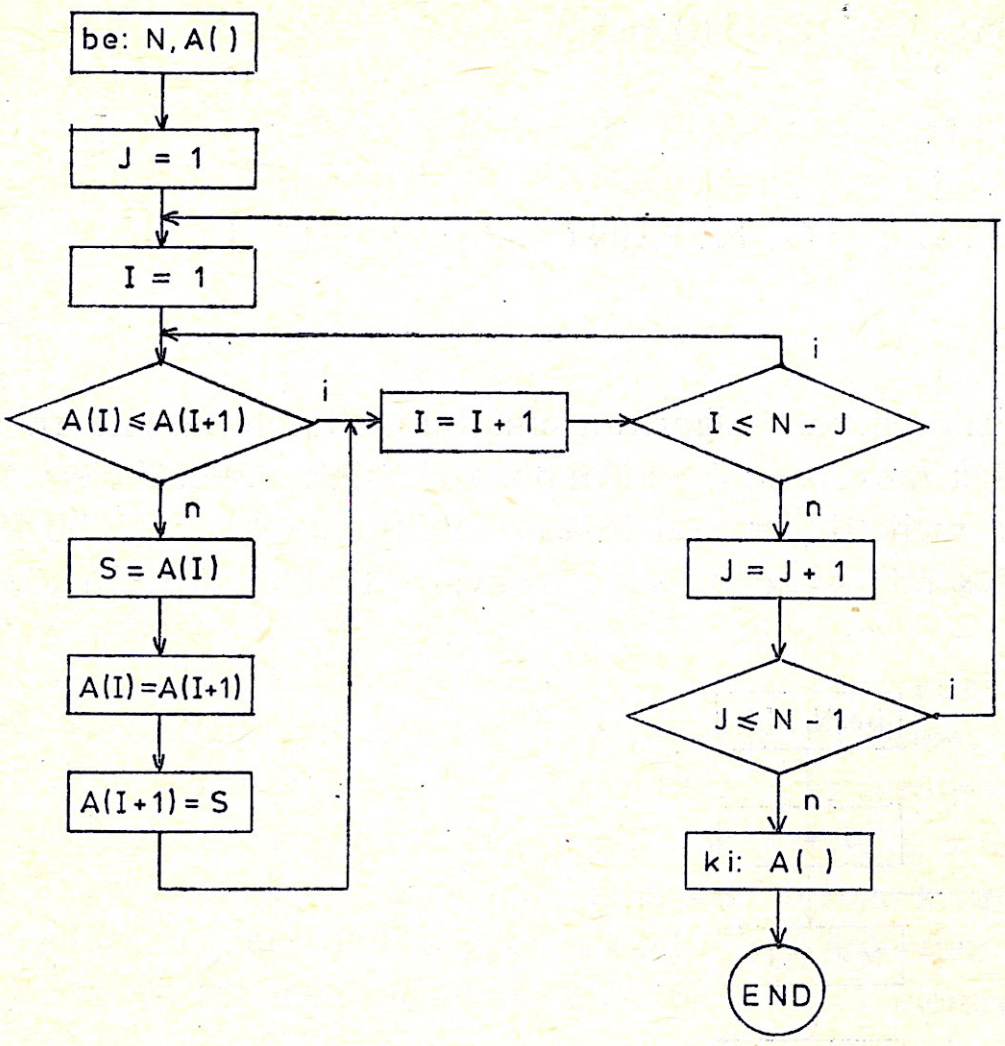
```
10 INPUT "ELEMEK SZÁMA" ; N
20 DIM A(N)
30 FOR I = 1 TO N: INPUT A(I): NEXT I
40 CLS: FOR I = 1 TO N: PRINT A(I); : NEXT I
50 PRINT
60 FOR J = 1 TO N-1
70 M = A(J)
80 FOR I = J+1 TO N
90 IF M< = A(I) THEN 110
```

```
100 S = M: M = A(I): A(I) = S
110 NEXT I
120 A(J) = M
130 NEXT J
140 FOR I = 1 TO N: PRINT A(I); :NEXT I
150 END
```

A másik rendezési algoritmusunk buborék rendezés néven ismeretes. Az elemeket most is az A ( ) tömbben tároljuk. Az első elemtől indulva az egymás melletti elemeket összehasonlítjuk, s ha az előbb lévő a nagyobb, felcseréljük azokat, s így megyünk tovább. Mire a tömb végére



6. ábra



7. ábra

érünk, a legnagyobb elem az A(N)-be kerül. (Felszállt, mint a buborék.) Ezután megismételjük ezt az eljárást az első N−1 elemre stb. (Ennél az algoritmusnál is N−1-szer kell ismételni.) Lásd a 7. ábra!

Mivel a 10—50 sorok ugyanazok mint az előbbiek, ezért a programot csak a 60. sortól írjuk fel.

```
 60 FOR J = 1 TO N-1
 70 FOR I = 1 TO N-J
 80 IF A(I)<=A(I+1) THEN 100
 90 S = A(I): A(I) = A(I+1): A(I+1) = S
100 NEXT I
110 NEXT J
120 FOR I = 1 TO N: PRINT A(I); : NEXT I
130 END
```

Ha névsort akarunk készíteni, a fenti programokon csak annyit kell változtatni, hogy mindazon azonosítók mellé be kell írni a $ jelet, amelyekbe a neveket írjuk. (A$, S$, illetve az előző programnál M$). Ezenkívül gondoskodni kell elegendő tárolóhelyről a szövegek számára. Ezért a programot pl. CLEAR 1000 utasítással célszerű kezdeni.

3.2. Kombinatorikai alkalmazások

Egy másik gazdag témakör ahonnan alkalmas feladatokat választhatnunk a *kombinatorika*. A számítógép lehetőségeit kihasználva általában nemcsak a lehetséges megoldások számát határozzuk meg, hanem kiírjuk a megoldásokat is. Kezdjük a következő egyszerű feladattal.

1. Egy hegy csúcsára öt út vezet. Hányféle útvonal közül választhat egy turista, ha felmászik a hegyre, majd lejön róla? Hány lehetősége van, ha nem akar ugyanazon az úton lejönni, mint amelyiken felment?

Számozzuk meg az utakat (1—5). Így a feladatot úgy fogalmazhatjuk, hogy rendezett számpárokat kell képezni, ahol az első komponens a felfelé, a második a lefelé vezető út sorszáma. Vegyük észre, hogy így a feladatot visszavezettük egy már megoldott problémára, az $A = \{1, 2, 3, 4, 5\}$ halmaznak (utak sorszámainak halmaza) kell képezni az önmagával való Descartes szorzatát.

Ha nem akarunk ugyanazon az úton lejönni, mint amelyiken felmentünk, akkor az azonos komponensű számpárokat el kell hagyni a fenti halmazból. (A géppel nem is képeztetjük.)

Mivel a feladat első részét már megoldottuk, ezért csak a második rész megoldására vonatkozó programot írjuk fel. A következőkben a folyamatábrák rajzolásától már eltekintünk.

```
 2 CLS: S = 0
 5 PRINT "AZ ELSŐ SZÁM A FELFELÉ, A MÁSODIK A LEFELÉ VEZE
   TŐ ÚT SORSZÁMA"
10 FOR I = 1 TO 5
20 FOR J = 1 TO 5
30 IF I = J THEN 50
```

```
40 PRINT "("; I; ":"; J; ")" : S = S+1
50 NEXT J
60 NEXT I
70 PRINT :PRINT "AZ ÖSSZES LEHETŐSÉGEK SZÁMA:"; S$
80 END
```

2. Ugyanez a probléma a következő feladatnál.

Hány szótár kell ahhoz, hogy közvetlenül tudjunk szavakat fordítani az orosz, angol, német, francia, spanyol nyelvek bármelyikéről e nyelvek bármelyikére? (Vagy: az A, B, C, D, E, F, G betűkártyáink vannak. Hányféleképpen tudunk két betűkártyát kiválasztani, ha a sorrend is számít?)

Annyit változtathatunk az előző programon, hogy ne számpárokat írassunk ki, hanem angol—orosz, angol—német stb. szópárokat a szótáraknak megfelelően. Ezért tároljuk A$( ) tömbben a megadott nyelveket.

A$(1)="OROSZ" ... A$(5)="SPANYOL"

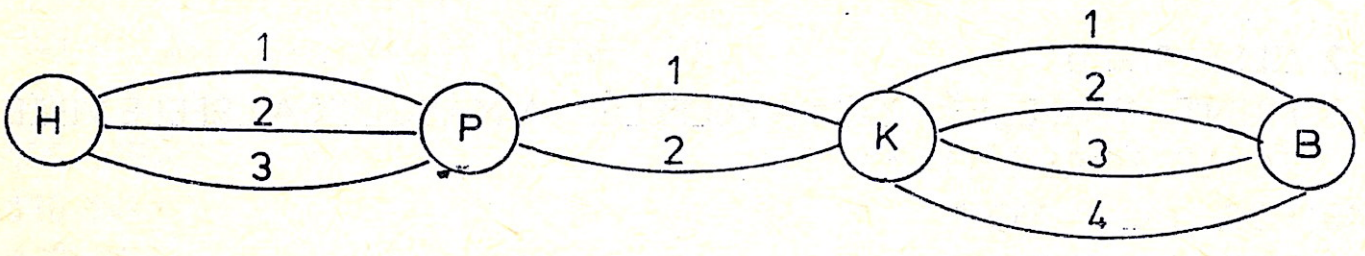
A kiíratásnál pedig I és J helyett A$(I)-t, illetve A$(J)-t írassunk ki.

3. Csak kicsit más az alábbi feladat.

Hencidából Piripócsra 3 út vezet. Piripócsról Kukutyinba 2, onnan Boncidába 4. Hányféleképpen juthatunk el Hencidából Boncidába Piripócson és Kukutyinon át? (8. ábra)

Számozzuk meg az egyes helységek közötti utakat! Így rendezett számhármasokat kell képezni, ahol a komponensek más-más halmazból valók.

{1, 2, 3} {1, 2} {1, 2, 3, 4} az utak halmazai.



8. ábra

(Tulajdonképpen a fenti 3 halmaznak a felírás sorrendjében vett Descartes szorzatáról van szó.)
Itt természetesen három egymásba ágyazott ciklusra van szükség. Az 55-ös sorral pedig azt érjük el, hogy a kiírás két oszlopba történjen.

```
 10 CLS:  S = 0:   P = 0
 20 FOR  I = 1  TO  3
 30 FOR  J = 1  TO  2
 40 FOR  K = 1  TO  4
 50 PRINT  TAB(P); : "("; I; ", "; J; ", "; K; ")":  S = S+1
 55 IF  S = 12  THEN PRINT$0, 20,; : P = 20
 60 NEXT  K
 70 NEXT  J
 80 NEXT  I
 90 PRINT  :PRINT  "A KÜLÖNBÖZŐ UTAK SZÁMA:"; S
100 END
```

Teljesen hasonlóan oldhatunk meg más variációs problémákat is.

4. Nem sokat kell változtatni az előző algoritmuson, ha valamilyen halmazból kiválasztok elemeket, de az elemek sorrendje nem számít. (Részhalmazokat képezek — ismétlés nélküli kombinációkat.)

Pl. Egy tíztagú őrsből kéttagú küldöttséget választanak. Hányféleképpen tehetik ezt meg?

Két lehetőség közül választhatunk a feladat megoldásánál aszerint, hogy milyen ismeretanyag birtokában akarjuk a feladatot megoldani.

1— Az egyszerűbb: az őrs tagjait megszámozzuk, és így hivatkozunk rájuk.

2— Az őrs tagjainak nevét egy A$( ) tömbben tároljuk, és ezzel dolgozunk.

A megoldási algoritmus természetesen mindkét esetben ugyanaz. Az első gyerekhez a másodiktól kezdődően választhatunk párt. A második gyereknek a harmadiktól ... a kilencediknek pedig ezek után csak a tizedik lehet a párja. A program felírásánál a második lehetőséget tartottuk szem előtt.

```
10 DIM A$(10)
20 FOR I = 1 TO 10: INPUT A$(I) : NEXT I
30 FOR I = 1 TO 9
40 FOR J = I+1 TO 10
50 PRINT A$(I);" "; A$(J)
60 NEXT J
70 NEXT I
80 END
```

5. Más jellegű problémát vet fel a most ismertetendő feladat. Hányféleképpen olvashatjuk le az ábráról a BALAMBÉR szót, ha csak jobbra vagy lefelé haladhatunk?

```
B A L A M B
A L A M B É
L A M B É R
```

Összesen 7-szer döntünk, hogy jobbra vagy lefelé haladjunk. A 7 alkalomból 2-szer kell a lefelé utat, 5-ször a jobbra haladást választani. Minden olvasásnak megfeleltethetünk egy öt 0-t és két 1-est tartalmazó sorozatot, ahol az 1-es a lefelé, a 0 a jobbra haladást reprezentálja. Először tegyük az egyik 1-est az első helyre, a másikat pedig sorban a 2., azután a 3. ... 7. helyre. (Ez lesz a belső ciklus.) Ezután az első 1-est tegyük a 2. helyre és a másik 1-est helyezzük a 3. ... 7. helyre stb. Addig folytatjuk, amíg az első 1-es a 6. helyen lesz, míg a másik a 7. helyre kerül. (Az első 1-es helyének megváltoztatását végzi a külső ciklus.) A kiíratásnál azt kell figyelni, hogy a sorozatban hol van 1-es, mert akkor egy sorral lejjebb kell folytatni, és nem kell jobra lépni.

```
 5 CLS
10 A$ = "BALAMBÉR": S = 0: PRINT CHR$(2)
20 FOR I = 1 TO 7: M(I) = 0: NEXT I
30 FOR J = 1 TO 6
40 M(J) = 1
50 FOR I = J+1 TO 7
60 M(I) = 1: E = 0
70 FOR K = 1 TO 7
```

```
 80 IF  M(K) = 1  THEN PRINT  MID$(A$, K, 1):  E =E+1: PRI
    NT  TAB(K-E); ELSE  PRINT MID$(A$, K, 1);
 90 NEXT  K
100 PRINT  RIGHT$(A$, 1)
110 M(I) = 0: PRINT  :  S = S+1
120 FOR  L = 1  TO 500:  NEXT  L
130 NEXT  I
140 M(J) = 0
150 NEXT  J
160 PRINT  "AZ ÖSSZES LEHETŐSÉGEK SZÁMA:";  S
170 GOTO  170
```

A szöveg megfelelő karakterének leválasztása a MID$ és a RIGHT$ függvényekkel történik. A CHR$(2) kóddal nyújtott karakteres írásmódot alkalmazunk, így az egyes kiíratások jobban olvashatók a képernyőn. A 120-as sorral szünetet tartunk.

6. Ugyanezt az algoritmust alkalmazhatjuk a következő feladat megoldásánál.

Ezek a számkártyáink vannak: 1, 1, 1, 1, 1, 2, 2. Hány különböző hétjegyű számot tudunk kirakni?

Az M tömbbe nem 0-kat és 1-eseket kell tennünk, hanem öt 1-est és két 2-est, és kiíratni nem A$-t kell, hanem az M tömb elemeit, és ezeket egymás mellé.

3.3. Számelméleti problémák

Nagy választási lehetőségünk van a *számelmélet* problémakörében is. Nézzünk először a prímszámokkal kapcsolatos néhány alapvető feladatot!

1. Keressük meg az első N (pl. 100) természetes szám közül a prímszámokat az Eratoszthenészi szita segítségével!

Képzeljük el felsorolva 1-től N-ig a természetes számokat! Fedjük le a 2 többszöröseit! (A 2-t nem!) Ezután a 3 többszöröseit. (A 3-t nem!) A 4 többszörösei már lefedettek. Ezután lefedjük az 5 többszöröseit.

A következő le nem fedett a 7. Ezt folytatjuk $\sqrt{N}$-ig. Amelyik ezek után nincs lefedve, az 1-nek és önmagának többszöröse, tehát prímszám.

A természetes számokat egy A( ) tömbben tároljuk. A lefedésnek feleltessük meg a megfelelő tömbelem nullázását.

Az algoritmust két egymásba ágyazott ciklussal valósíthatjuk meg. A külső ciklus veszi sorban (a második elemtől kezdve) az A( ) tömb elemeit $\sqrt{N}$-ig. (Ha a gyerekekkel dolgozunk, más, ennél nagyobb végértéket is választhatunk.) Ha valamelyik A(I) elem nem 0 (nincs lefedve), akkor indul a belső ciklus, amely kinulláz minden I-edik elemet az A(I) elem után. Végül megvizsgáljuk az így kapott A( ) tömböt, és a nem 0 elemeket (az 1 kivételével) kiíratjuk.

Tulajdonképpen felesleges az A( ) tömbben a természetes számokat tárolni, hiszen ezek pontosan a tömbelemek indexeivel egyenlőek. Elegendő lenne, ha minden elemnek 1 értéket adnánk, és a kiíratásnál, ha a tömbelem 1, akkor az indexét kellen kiíratni. Ez azonban semmivel sem kevesebb munka, ellenben az előző a gyerekek számára szemléletesebb. (Rövidíthetünk viszont, ha az eredeti állapotú nullázott A( ) tömbbel indulunk, s a lefedésnek az 1-est feleltetjük meg. Ez azonban kevésbé természetes.)

```
  5 CLS
 10 INPUT "MEDDIG ÍRJAM KI A PRÍMSZÁMOKAT?"; N
 20 DIM A(N): V = SQR(N)
 30 FOR I = 1 TO N: A(I) = I: NEXT I
 40 FOR I = 2 TO V
 50 IF A(I) = 0 THEN 90
 60 FOR J = 2*I TO N STEP I
 70 A(J) = 0
 80 NEXT J
 90 NEXT I
100 FOR I = 2 TO N
110 IF A(I) = 0 THEN 130
120 PRINT A(I);
130 NEXT I
140 END
```

2. Hasonlóan oldhatunk meg egy érdekes általános iskolai feladatot is. Rámutathatunk itt arra is, hogy a számítógépnek lehetősége van „eljátszani" a feladatot.

Egy szultán 100 cellába bezárat egy-egy rabot. A cellákon kétállású zárak vannak, forgatással felváltva nyílnak, illetve záródnak. A rabok nem veszik észre, ha nyitják vagy zárják a cellákat. A száz rab bezárását követően a szultán meggondolja magát, és végigszalaszt egy őrt, hogy minden záron fordítson egyet, majd újra meggondolja magát, és elküld egy másik őrt, hogy minden második záron forditson egyet, majd egy harmadik őrt, hogy minden harmadik záron fordítson egyet, és így tovább. A századik őrt azzal a paranccsal küldi, hogy a századik záron fordítson egyet. Ezután elrendeli, hogy akinek a cellája nyitva van, azt bocsássák szabadon. Hány rab szabadul ki, és mely cellákból?

A cellákat reprezentálja egy C(100) tömb. A zárt állapotnak feleljen meg az, hogy a tömbelem 1, a nyitottnak pedig az, hogy a tömbelem -1. A záron való fordítást így -1-gyel való szorzással valósíthatjuk meg. Itt is két ciklussal oldhatjuk meg az algoritmust. A külső ciklus mutatja, hogy hányadik őr van úton, a belső pedig, hogy éppen melyik cellának a zárján fordít egyet. Egy egyszerű, de ügyes képernyőre írással szemléletessé is tehetjük. A cellák sorszámait 10×10-es elrendezésben írassuk ki a képernyőre. (Így pl. a 26. cella a képernyőn a 3. sor 6. oszlopában van.) Ha a cella be van zárva, írjuk felül a sorszámot valamilyen karakterrel. (Az alábbi programban >< jellel.) Így a képernyőn a számok a nyitott cellák sorszámait jelentik. Egy lehetséges megvalósítás a következő program:

```
10 CLS: DIM C(100): PRINT CHR$(6)
20 FOR J = 1 TO 100: C(J)= 1: GOSU 100: NEXT J
30 FOR I = 1 TO 100
35 PRINT$12, 14, I; ". ŐR";
40 FOR J = 1 TO 100 STEP I
50 C(J) = -C(J) : GOSUB 100
60 NEXT J
70 NEXT I
80 GOTO 80
```

```
100 REM * KIÍRÁS *
110 S = INT((J-1)/10)+1
120 O = J-(S-1)*10
130 PRINT$S, (O-1)*4,;
140 IF C(J) = -1 THEN PRINTUSING "# # # #"; J; : GOTO 160
150 PRINT " >< "
160 RETURN
```

A CHR$(6) kódot előtörlésre használjuk, ezzel akadályozzuk meg az egymásra írást.

A PRINT utasításban USING "# # # #" formátummal írjuk le, hogy a számjegyektől függetlenül 4 pozícióra történik a kiírás, és a felesleges pozíciókra szóközök kerülnek. Így a számok és a >< jelek egymás alatt lesznek a képernyőn.

A cella sorszámából (J) az ábrán levő helyét a programban felírt képletekkel határozhatjuk meg. A sorszám megegyezik az aktuális képernyő sorszámmal, míg azt, hogy a képernyőn melyik oszlopnak felel meg az (O-1)*4 képlettel kapjuk. (Minden cella kiírása 4 pozíció!) A 35. sorral kiíratjuk, hogy melyik őr van úton.

Ugyanezt a szemléltetést valósíthatjuk meg az Eratoszthenészi szitánál is, ha pl. 100-ig akarjuk a prímszámokat meghatározni, és az algoritmust is nyomon akarjuk követni. Ez a módosított program a következő lehet:

```
  5 CLS: PRINT CHR$(6);
 10 PRINT "PRÍMSZÁMOK 100-IG": N = 100
 20 DIM A(N): V = SQR(N)
 30 FOR J = 1 TO N: A(J) = J: GOSUB 200: NEXT J
 40 FOR I = 2 TO V
 50 IF A(I) = 0 THEN 90
 60 FOR J = 2*I TO N STEP I
 70 A(J) = 0: GOSUB 200
 80 FOR K = 1 TO 1500: NEXT K
 90 NEXT J
100 NEXT I
```

```
110 END
200 REM * KIÍRÁS *
210 S = INT((J-1)/10)+1
220 O = J-(S-1)*10
230 PRINT$S, (O-1)*4, ;
240 IF A(J) = 0 THEN PRINT " ><"; : GOTO 260
250 PRINTUSING "# # # #"; A(J);
260 RETURN
```

3. Nézzünk ezután egy összetettebb feladatot, amelyet természetesen részfeladatokra bontunk. A részfeladatokat szubrutin formában írjuk fel, és ezekből állítjuk össze egy főprogram segítségével a teljes programot, rámutatva a szubrutin-programozás előnyeire. Célunk két (vagy több) természetes szám legnagyobb közös osztójának, illetve legkisebb közös többszörösének meghatározása. A főbb lépések a következők:

— elő kell állítani a számok prímtényezős felbontását.
— ebben szükség van a prímszámokra, a nagyobb (vagy a legnagyobb) számig. — Ezt a programot már felírtuk.
— a prímtényezős felbontásból meg kell határozni a legnagyobb közös osztót, illetve a legkisebb közös többszöröst.

Fel fogjuk használni a prímszámok előállítására felírt programot, de a képernyőre való kiíratás helyett egy P( ) tömbben tároljuk a prímszákat. Ezt figyelembe véve írjuk át programunkat szubrutinná. P1 jelenti az N-ig talált prímszámok számát.

```
1000 REM * PRÍMSZÁMOK N-IG *
1010 DIM A(N), P(N) : V = SQR(N) : P1 = 0
1020 FOR I = 1 TO N: A(I) = I : NEXT I
1025 IF N<4 THEN 1090
1030 FOR I = 2 TO V
1040 IF A(I) = 0 THEN 1080
1050 FOR J = 2*I TO N STEP I
1060 A(J) = 0
1070 NEXT J
```

```
1080 NEXT I
1090 FOR I = 2 TO N
1100 IF A(I) = 0 THEN 1120
1110 P1 = P1+1: P(P1) = A(I)
1120 NEXT I
1130 RETURN
```

Nézzük a törzstényezőkre bontás programrészletét!

Az algoritmus a következő:

— Vegyük a legkisebb prímszámot, a 2-t!

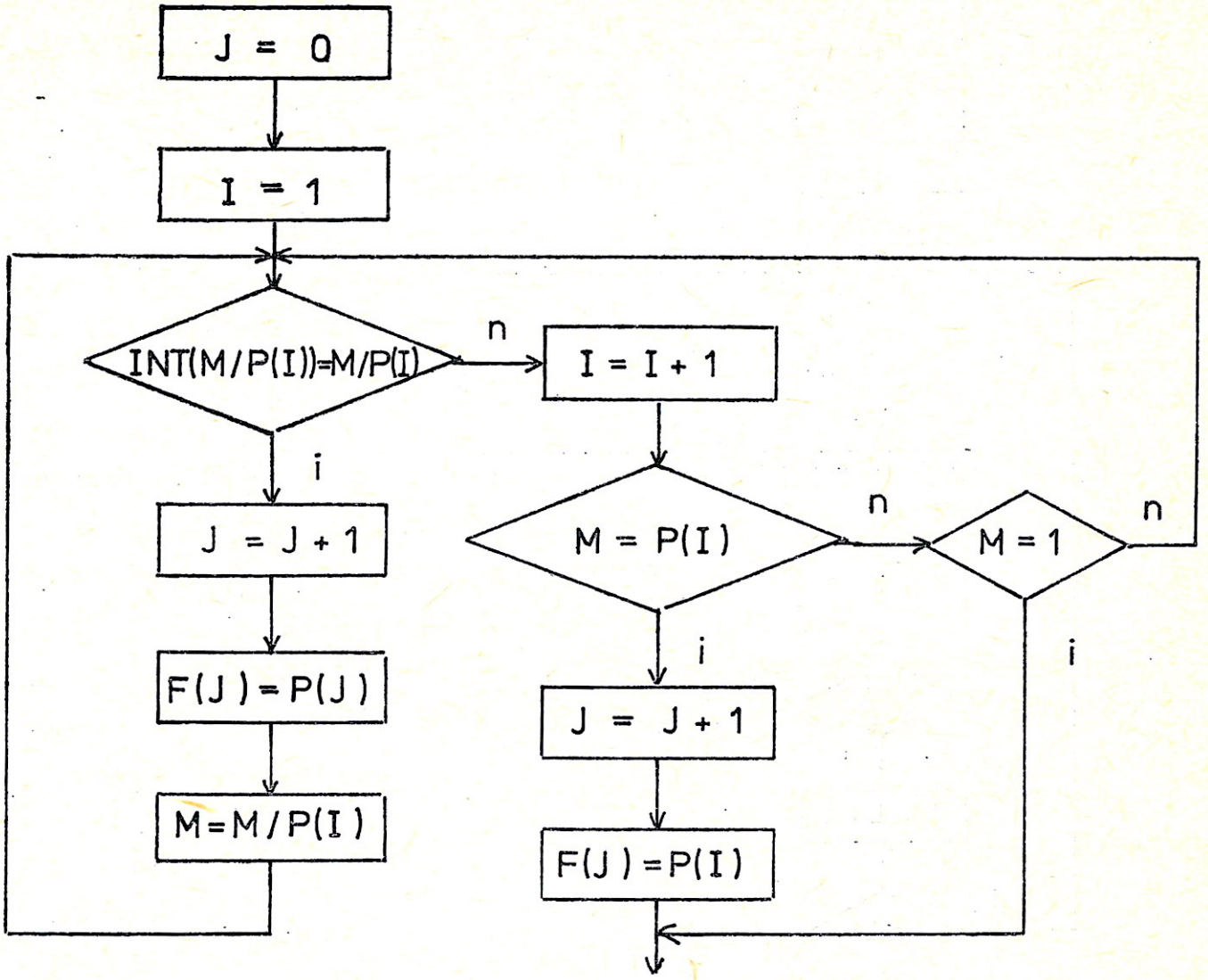
— Nézzük meg, osztható-e a szám vele. (Erre az INT függvényt fogjuk használni. Ha ezt nem akarjuk, ismételt kivonást alkalmazhatunk mindaddig, amíg kisebb számot nem kapunk mint 2, vagy általában amivel való oszthatóságot vizsgálunk. Ha az utolsó kivonásnál kapott szám 0, akkor az oszthatóság teljesül, ha nem 0, akkor az oszthatóság nem teljesül).

— Ha igen, osszuk el, és a hányadosra és ugyanerre a prímszámra alkalmazzuk a második lépést.

— Ha valamelyik lépésben az oszthatóság nem teljesül, vegyük a következő prímet, és ismételjük a 2. lépéstől az algoritmust mindaddig, amíg a hányados prímszám nem lesz.

A feladat nehézségére való tekintettel ismét készítsünk folyamatábrát! (9. ábra)

```
2000 REM * PRÍMTÉNYEZŐKRE BONTÁS *
2010 J = 0 : I = 1
2020 IF INT(M/P(I))<> M/P(I) THEN 2050
2030 J = J+1 : F(J) = P(I)
2040 M = M/P(I) : GOTO 2020
2050 I = I+1
2060 IF M<>P(I) AND M<>1 THEN 2020
2065 IF M = 1 THEN 2080
2070 J = J+1 : F(J) = P(I)
2080 RETURN
```

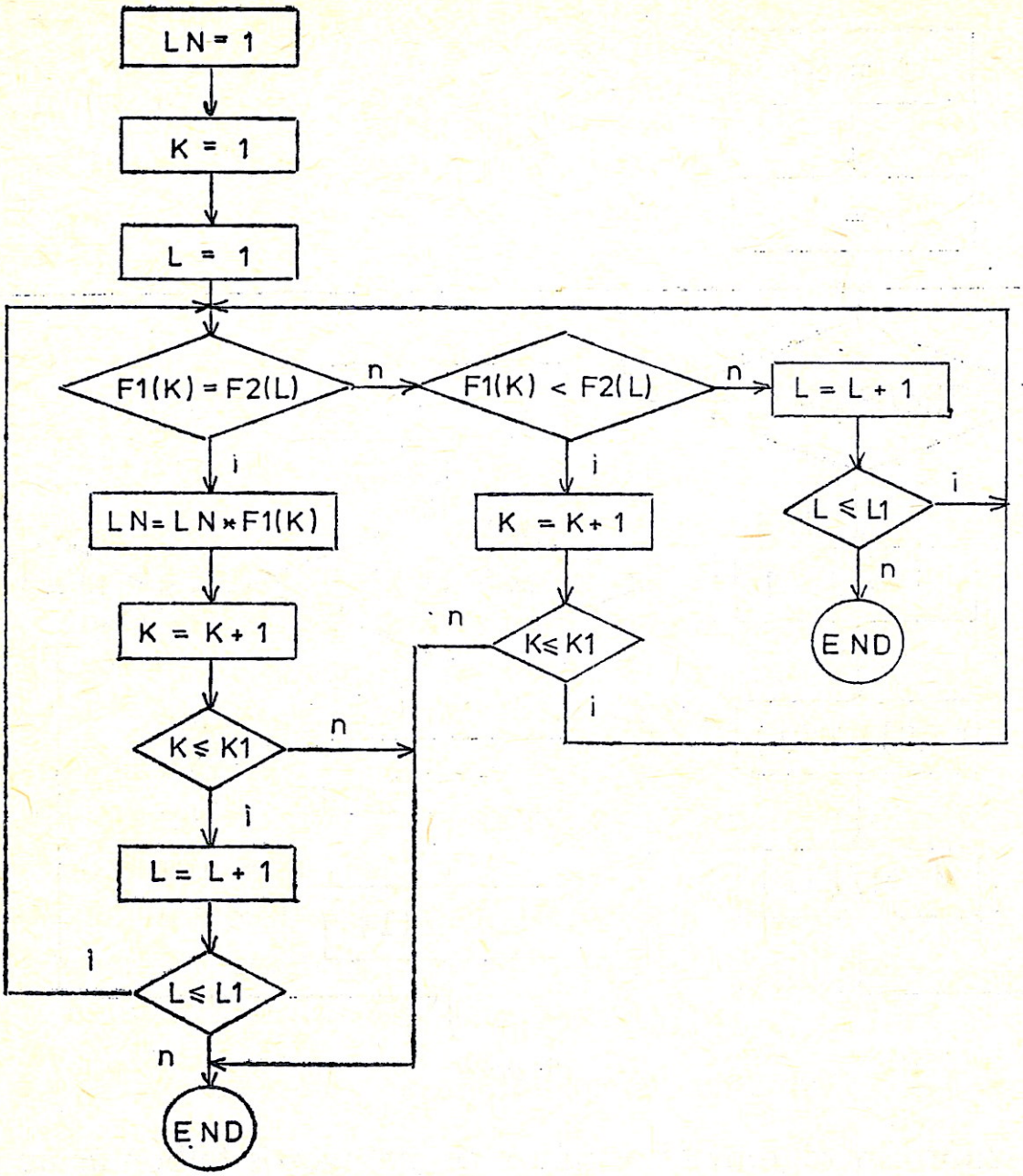


9. ábra

A szubrutin végrehajtása után a prímtényezős felbontás az F(1,) ... ..., F(J) tömbelemekben van.

Most nézzük a legnagyobb közös osztó meghatározásának algoritmusát és folyamatábráját! (Két egyidejűleg nem zéró természetes szám esetén.)

A prímtényezős felbontásokat az F1( ), illetve F2( ) tömbökben tároljuk, növekvő sorrendben. Rendre összehasonlítjuk a felbontások prímtényezőit, s ha megegyezőt találunk, ezzel szorozzuk a legnagyobb közös osztót (LN), és mindkét felbontásban továbblépünk. Ha nem egyenlő prímtényezőt találunk, csak abban a tömbben (felbontásban) kell továbblépnünk, ahol a kisebb szám van. Ezt mindaddig ismételjük, míg valamelyik felbontás végére nem érünk. (10. ábra)



10. ábra

```
3000 REM * LEGNAGYOBB KÖZÖS OSZTÓ *
3010 LN = 1:  K = 1:  L = 1
3015 IF  K1 = 0  OR  L1 = 0  THEN 3080
3020 IF  F1(K) = F2(L)  THEN 3050
3030 IF  F1(K)<F2(L)  THEN  K = K+1 :  IF  K< = K1  THEN
     3020  ELSE 3080
3040 L = L+1 :  IF  L< = L1  THEN 3020  ELSE 3080
3050 LN = LN*F1(K) :  K = K+1
3060 IF  K>K1  THEN  3080
3070 L = L+1 :  IF  L< = L1  THEN  3020
3080 RETURN
```

Ezután a legkisebb közös többszöröst legegyszerűbben úgy határozhatjuk meg, hogy a két szám szorzatát elosztjuk a legnagyobb közös osztóval. A főprogram felírása után bemutatunk egy lehetőséget a legkisebb közös többszörösnek a prímtényezős felbontásokból való meghatározására. Ez lényegében hasonló a legnagyobb közös osztó meghatározásához, de annál jóval bonyolultabb. Érdemes tehát a legkisebb közös többszöröst a legnagyobb közös osztó felhasználásával meghatározni, még akkor is, ha azt egyébként nem számítanánk ki.

```
10 CLS: DIM F1(100), F2(100), F(100)
20 PRINT "ADD MEG A KÉT SZÁMOT!"
30 INPUT A, B
35 IF A<1 OR B<1 THEN PRINT "ROSSZ ADAT!": GOTO 20
40 IF A>B THEN N = A ELSE N = B
45 IF N = 1 THEN LN = 1 : GOTO 110
50 GOSUB 1000 : REM * PRÍMSZÁMOK ELŐÁLLÍTÁSA *
60 M = A : GOSUB 2000 : REM * A PRÍMTÉNYEZŐKRE
   BONTÁSA *
70 K1 = J : FOR I = 1 TO K1 : F1(I) = F(I) : NEXT I
80 M = B : GOSUB 2000 : REM * B PRÍMTÉNYEZŐKRE BONTÁ
   SA *
90 L1 = J : FOR I = 1 TO L1 : F2(I) = F(I) : NEXT I
100 GOSUB 3000 : REM * A LNKO MEGHATÁROZÁSA *
110 LK%= A*B/LN
120 PRINT A; "ÉS" ; B
130 PRINT "LEGNAGYOBB KÖZÖS OSZTÓJA:"; LN
140 PRINT "LEGKISEBB KÖZÖS TÖBBSZÖRÖSE" : LK%
150 END
```

Az LK-nál a % jelet az osztás miatt írtuk, az esetleges gépi pontatlanság miatt.

Nézzük ezután a legkisebb közös többszörösnek a meghatározását a legnagyobb közös osztó ismerete nélkül! (Ezt csak a legjobb tanulóknak ajánljuk, kellő előkészítés után!)

Itt is rendre összehasonlítjuk a prímtényezős felbontások egyes elemeit.

— Ha megegyezőt találunk, ezzel megszorozzuk a legkisebb közös többszöröst LK-t, és mindkét felbontásban továbblépünk. Ha valamelyikben már nincs több elem, a másik összes további prímtényezőjével szorozzuk LK-t.
— Ha különbözőt találunk, akkor a kisebbel megszorozzuk LK-t, és csak itt lépünk tovább. Ha a továbblépésre nincs lehetőség, akkor a másik felbontás összes további prímtényezőivel szorozzuk LK-t.

Ezt mindaddig ismételjük, míg valamelyik (vagy mindkét) felbontás végére nem érünk. (11. ábra)

Az L=L−1 blokkra azért van szükség, mert olyan pontra csatlakozunk vissza, amely után L-et 1-gyel megnöveljük, akármelyik irányban is haladunk tovább. A fenti helyről való csatlakozásnál pedig az L értékét nem kell megváltoztatni.

```
4000 REM * LKKT *
4005 IF L1 = 0 OR K1 = 0 THEN LK = A*B
4010 LK = 1 : K = 1 : L = 1
4020 IF F1(K) = F2(L) THEN 4050
4030 IF F1(K)<F2(L)) THEN L = L-1 : GOTO 4050
4040 LK = LK*F2(L) : GOTO 4080
4050 LK = LK*F1(K) : K = K+1
4060 IF K<=K1 THEN 4080
4070 L = L+1 : IF L<=L1 THEN LK = LK*F2(L): GOTO 4070
     ELSE 4110
4080 L = L+1 : IF L<=L1 THEN 4020
4090 LK = LK*F1(K) : K = K+1
4100 IF K<=K1 THEN 4090
4110 RETURN
```

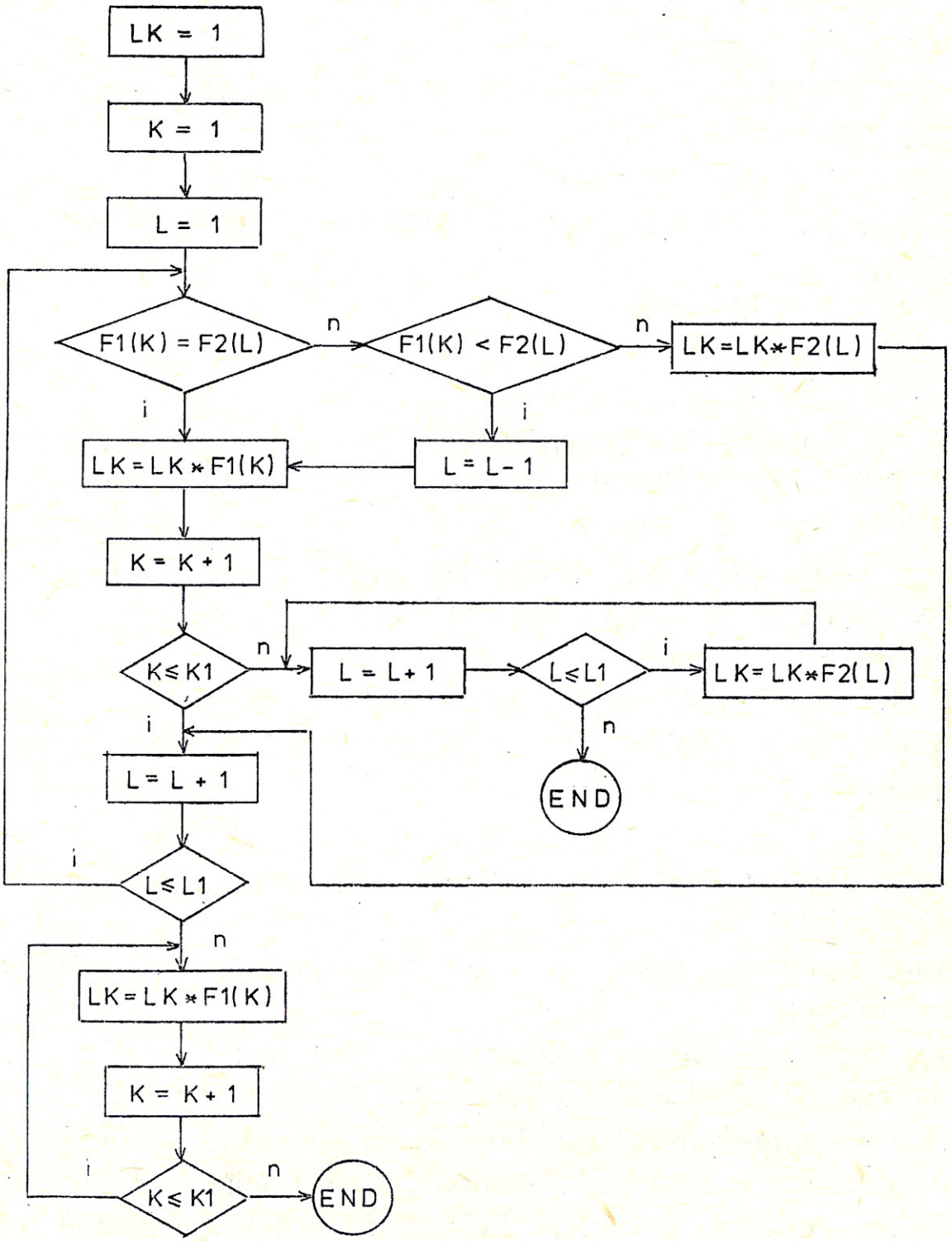
Ha az előző főprogramban ezt a szubrutint akarjuk használni, akkor

```
110 GOSUB 4000
```

sorral futtatjuk a programot, és a 140-es sorban az LK% helyett LK-t írunk.

4. A prímszámokkal kapcsolatban könnyen megoldható feladat még pl. a következő.

Az első N (paraméter) természetes számból keressük ki az ikerprímeket! (Olyan prímek, amelyek különbsége 2.)



11. ábra

A feladatot úgy is általánosíthatjuk, hogy adott [M, N] intervallumban keressük meg az ikerprímeket.

Felhasználjuk az előzőekben felírt prímszámelőállító szubrutint. 1-től N-ig előállítjuk a prímszámokat, azután az [M, N] intervallumba esők közül kiválasztjuk azokat, amelyeknek különbsége 2.

```
 10 CLS
 20 PRINT "MELYIK INTERVALLUMBA ESŐ IKERPRÍMEKET HATÁ-
    ROZZAM MEG?"
 30 INPUT "A VÉGPONTOK" ; M, N : S = 0
 40 IF N<=M THEN PRINT "ROSSZ INTERVALLUM!" : GO
    TO 20
 50 IF N<3 THEN 150
 60 GOSUB 1000
 70 I = 1
 80 IF P(I)<M AND P(I)<P(P1) THEN I = I+1 : GOTO 80
 90 IF P(I) = P(P1) THEN 150
100 FOR J = I TO P1-1
110 IF P(J+1)-P(J)<>2 THEN 130
120 S = S+1 : PRINT P(J) ; P(J+1)
130 NEXT J
140 IF S<>0 THEN 160
150 PRINT "AZ ADOTT INTERVALLUMBAN NINCS IKERPRÍM"
160 END
```

5. Az előzőek felhasználásával már könnyen megoldhatjuk a következő feladatot.

Adott egy természetes számokból álló számhalmaz. Keressük ki a relatív prímeket!

A megoldás fő lépései a következők lehetnek:

— A halmazból két elemű részhalmazokat képezünk. (Olyan számpárokat, ahol a komponensek különbözőek, és a sorrend nem számít.) Ezt valósitottuk meg a 3.2. rész 4. feladatának megoldásakor. (10 elem helyett N-nel dolgozva s kiíratás helyett tárolást végrehajtva.)

— Ezután meghatározhatjuk az egyes párok legnagyob közös osztóját

az előzőekben felírt program felhasználásával, s ha ez 1, akkor a számpárt kiíratjuk, mint relatív prímeket.

6. A közös osztók és így a legnagyobb közös osztó meghatározására más, kevésbé „szép", de rövidebb (időben nem biztos!) algoritmus is adódik.

Legyen a két szám A és B. Válasszuk ki a kisebbet! Jelöljük ezt K-val, a nagyobbat N-nel. Vegyük 2-től K-ig a természetes számokat, és nézzük meg, melyik osztója K-nak. (Az INT függvény segítségével vagy a már említett ismételt kivonással.) Ha találunk K-nak egy osztóját, nézzük meg, osztója-e N-nek. Ha igen, tároljuk (vagy írassuk ki), mint közös osztót. Így a legnagyobb közös osztó (LN) az utoljára talált közös osztó.

```
 10 INPUT A, B : CLS
 20 LN = 1 : PRINT A; "ÉS"; B; "KÖZÖS OSZTÓI :"
 30 IF A<B THEN K = A : N = B ELSE K = B: N = A
 40 FOR I = 1 TO K
 50 IF INT(K/I)<>K/I THEN 80
 60 IF INT(N/I)<>N/I THEN 80
 70 LN = I : PRINT I;
 80 NEXT I
 90 PRINT : PRINT "A LEGNAGYOBB KÖZÖS OSZTÓ:"; LN
100 END
```

Azért is célszerű ezt az algoritmust is megbeszélni a gyerekekkel, mert egy teljesen más módszert ismerhetnek meg a legnagyobb közös osztó meghatározására. (Számunkra nem nagyon használható algoritmus, de a gép gyors műveletvégzése miatt — nem túl nagy természetes szám esetén — viszonylag gyorsan elvégzi.)

7. Végül megemlítünk még egy feladatot a prímszámokkal kapcsolatban, melynek megoldására érdemes programot írni.

Goldbach (1690—1764) sejtése szerint minden 2-nél nagyobb páros szám felbontható két prímszám összegére.

(Pl. $4=2+2$, $6=3+3$, $8=3+5$, $10=5+5=3+7$ ...)

Egy [M, N] intervallumba eső páros számokra végeztessük el a géppel ezt a felbontást. Ismét szükségünk van a prímszámokat előállító szubrutinra.

— Képezzük a prímszámokat N-ig.

— Ezután vesszük az [M, N] intervallumba eső páros számokat (jelöljük A-val), és az első prímszámtól kezdve (2) képezzük A—P(I)-t.

— Amennyiben ez prímszám, (megtaláljuk prímszámaink között A-ig) kiíratjuk a felbontást, s a következő prímszámmal próbálunk másik felbontást keresni.

— Ha a különbség nem prím, a kiíratás elmarad, s úgy lépünk tovább.

— Új felbontás keresésénél a prímeket csak A/2-ig vesszük. (Ezután már csak a tagok sorrendje lenne más.)

```
 10 CLS
 20 PRINT "MELYIK INTERVALLUMBA ESŐ PÁROS SZÁMOKAT
    AKAROD FELBONTANI PRÍMSZÁMOK ÖSSZEGÉRE?"
 30 INPUT "A VÉGPONTOK"; M, N
 40 IF M<0 OR N<=M OR N<4 THEN PRINT "ROSSZ AZ
    INTERVALLUM" : GOTO 20
 50 GOSUB 1000 : REM * PRÍMSZÁMOK ELŐÁLLÍTÁSA *
 60 IF M/2 = INT(M/2) THEN A = M ELSE A = M+1
 70 I = 1
 80 B = A-P(I)
 90 J = 1
100 IF B = P(J) THEN 130
110 J = J+1
120 IF P(J)>B OR J>P1 THEN 140 ELSE 100
130 PRINT A; " = " ; P(I); "+"; P(J)
140 I = I+1
150 IF P(I)<=A/2 THEN 80
160 A = A+2
170 IF A<=N THEN 70
180 END
```

8. Gyakran előforduló feladat: egy tízes számrendszerbeli természetes számot (N) írjunk át egy A alapú (1<A<10) számrendszerbe!

Mivel ez a probléma (ha nem is ilyen megfogalmazásban) már az alsó tagozatban ismert, célszerűnek látjuk a programot az INT függvény alkalmazása nélkül is felírni.

Az A alapú számrendszerben kapott számjegyeket az A( ) tömbben tároljuk. Az N-ből annyiszor levonjuk A-t, míg a kapott érték kisebb nem lesz, mint A. Ez az érték az A alapú számrendszerben a szám egyesek helyén álló jegye. A levonások számát S-ben tároljuk. Ezután S értékével kell megismételni azt, amit az előbb N-nel tettünk. Ekkor a következő helyiértékű számjegyet kapjuk. Az eljárás akkor ér véget, amikor S értéke 0 lesz.

```
 10 DIM  A(50) :  CLS
 20 INPUT  "AZ ALAPSZÁM";  A
 30 INPUT  "AZ ÁTÍRANDÓ SZÁM" ;  N
 40 I = 1
 50 S = 0
 60 IF  N<A  THEN  80
 70 N = N-A :  S = S+1 :  GOTO 60
 80 A(I) = N
 90 IF  S = 0  THEN 110
100 N = S :  I = I+1 :  GOTO 50
110 FOR  J = I  TO 1  STEP -1
120 PRINT  A(J);
130 NEXT  J
140 END
```

Az INT függvény alkalmazásával pedig:

```
 10 DIM  A(50) :  CLS
 20 INPUT  "AZ ALAPSZÁM";  A
 30 INPUT  "AZ ÁTÍRANDÓ SZÁM" ;  N
 40 I = 1
 50 H = INT(N/A) : M = N-H*A : A(I) = M
 60 IF  H = 0  THEN 80
```

```
 70 N = H :  I = I+1 :  GOTO 50
 80 FOR  J = I  TO 1  STEP  - 1
 90 PRINT  A(J);
100 NEXT  J
110 END
```

9. A másik ehhez szorosan kapcsolódó feladat, amikor egy A(1<A<10) alapú számrendszerben adott természetes számot akarunk átírni tízes számrendszerbe.

Az egyszerűség kedvéért a szám jegyeit külön olvassuk be, és tároljuk egy A( ) tömbben. (Az egyeseket A(0)-ban, ..., az A^N helyiértékű jegyet A(N)-ben.) Így csupán az átírás annyi, hogy a jegyeket megszorozzuk a megfelelő A hatványokkal, és ezeket összegezzük.

$$M = A(0) \cdot A^0 + A(1) \cdot A^1 + A(2) \cdot A^2 + \ldots + A(N) \cdot A^N$$

```
 10 DIM  A(50)
 20 INPUT  "AZ ALAPSZÁM"; A
 30 INPUT  "A JEGYEK SZÁMA"; S: N = S-1
 40 PRINT  "A SZÁM JEGYEI CSÖKKENŐ HELYIÉRTÉKEK SZERINT"
 50 FOR  I = N  TO 0 STEP  -1
 60 INPUT  A(I)
 70 NEXT  I:  CLS
 80 PRINT  "A SZÁM EREDETI ALAKJA:"
 90 FOR  I = N  TO 0  STEP  -1
100 PRINT  A(I);
110 NEXT  I : PRINT
120 H = 1 :  M = 0
130 FOR  I = 0  TO N
140 A(I) = A(I)*H :  H = A*H :  M = M+A(I)
150 NEXT  I
160 PRINT  "A SZÁM A TÍZES SZÁMRENDSZERBEN:";  M
170 END
```

Megmutathatjuk a következő felbontást is:

$$M = (\ldots((A(N) \cdot A + A(N-1)) \cdot A + A(N-2)) \cdot A + \ldots + A(1))A + A(0)$$

Ennek felhasználásával a beolvasást követő programsorok így alakulnak:

```
120 M = A(N)
130 FOR I = N-1 TO 0 STEP -1
140 M = M*A+A(I)
150 NEXT I
```

A kiírás ugyanaz, mint fent.

10. Az előző algoritmusban és számos más feladatnál arra van szükségünk, hogy egy szám számjegyeit külön kezeljük. Egy természetes szám számjegyekre való bontásának két különböző módját mutatjuk be.

Az első egy algebrai módszer. Jelölje N a természetes számot!

A szám utolsó jegye = N-INT(N/10)-10

A következő jegyet ugyanezen képlettel kapjuk, csak N helyett INT(N/10)-zel kell számolni. Ezt ismételjük mindaddig, míg INT(N/10) 0 nem lesz. A jegyeket A(0), ..., A(I) tömbelemekben tároljuk.

```
 10 DIM A(50) : CLS
 20 INPUT "MELYIK SZÁMOT BONTSAM SZÁMJEGYEKRE"; N
 30 I = 0
 40 S = INT(N/10)
 50 A(I) = N-S*10
 60 IF S = 0 THEN 80
 70 N =S : I = I+1 : GOTO 40
 80 PRINT "A SZÁM JEGYEKRE BONTVA"
 90 FOR J = I TO 0 STEP -1
100 PRINT A(J);
110 NEXT J
120 END
```

A másik megvalósítást a string függvények használata teszi lehetővé. Ezzel a programmal ezeknek a függvényeknek egy érdekes alkalmazását mutathatjuk be.

A számot karaktersorozattá konvertáljuk, majd sorban leválasztjuk az egyes karaktereket, és ezeket számokká konvertáljuk.

```
 10 DIM  A(50) :  CLS
 20 INPUT  "MELYIK SZÁMOT BONTSAM JEGYEIRE" ;  N
 30 N$ = STR $(N) :  H = LEN(N$)
 40 FOR  I = 1  TO  H
 50 M$ = MID$(N$, I, 1)
 60 A(I) = VAL (M$)
 70 NEXT  I
 75 PRINT  "A SZÁM JEGYEKRE BONTVA:"
 80 FOR  I = 2  TO H
 90 PRINT  A(I);
100 NEXT  I
110 END
```

A kiíratást a második elemtől kezdjük, mert az A(1)-ben 0 van, az előjel konverziója. Ha a számjegyeket nem akarjuk kiíratni, hanem csak tároljuk, akkor 80—100-ig elmarad a programból.

Hasonló programokat írhatunk fel tökéletes számok, barátságos számok stb. előállítására.

A három itt ismertetett témakörrel még korántsem merítettük ki a lehetőségeket. Nagyon sok feladatot sorolhattunk volna fel a geometria tárgyköréből is. Az általános iskolában tárgyalt geometriai alakzatok esetén a kerület, terület, felszín, térfogat kiszámításának lényege egy értékadás, ahol egy bizonyos változónak a megfelelő képlettel adunk értéket. Ezek viszont annyira egyszerű feladatok, (INPUT, értékadás, PRINT) hogy bárki, aki valami keveset is foglalkozott a BASIC nyelvvel, pillanatok alatt fel tudja írni programjukat.

Olyan feladatoknak, ahol rajzokat készítenénk, a grafikus lehetőségek nagyfokú eltérése miatt (PRIMO, C—16) nem tartottuk célszerűnek a közlését.

A fejezetben felírt programok egyaránt futtathatók HT, PRIMO, C—16 gépeken, legfeljebb a kiíratást kell módosítani egy-két helyen. (Kivétel: IF THEN ELSE utasításnál C—16-nál az ELSE előtt :!)

4. SZÁMÍTÓGÉP A MATEMATIKATANÁRI MUNKÁBAN

Ebben a fejezetben olyan programokat mutatunk be, amelyeket a matematikatanárok tantárgyuk oktatásában használhatnak. Egyrészt tantervi órákon vagy szakkörön, másrészt korrepetáláson vagy egyéni gyakoroltatás, tanulás vagy számonkérés alkalmával.

A bemutatott példákkal ötleteket, illetve útmutatásokat szeretnénk adni ahhoz, hogy a gyakorló pedagógusok saját maguk is minél több olyan programot készítsenek (vagy készíttessenek), amelyeket hatékonyan tudnak alkalmazni a tanulási-tanítási folyamatban.

Az oktatóprogramok írásánál a géppel való párbeszéd lehetőségét használjuk ki. A feladat egyértelmű megfogalmazása és közlése mellett nagyon fontos, hogy pontosan értesüljön a felhasználó, hogyan, milyen formában kell válaszait megadni. A kiírt szövegek elolvasására, a feladatok megoldásánál a gondolkodásra a szükséges időt biztosítanunk kell. A programba szüneteket többféle módon iktathatunk. Egyik lehetőség az INKEY$ alkalmazása. Pl. 10 A$=INKEY$: IF A$=""THEN 10

Mindaddig a 10-es utasítás hajtódik végre, amíg a billentyűzetről egy jel be nem érkezik.

Másik lehetőség, ha a választ, az adatot INPUT utasítással kérjük be, ilyenkor mindaddig várakozik a gép, ameddig a kellő számú adatot be nem írjuk, és a RETURN billentyűt meg nem érintjük.

Várakozhatunk míg egy „üres" ciklus beiktatásával is.

```
10 FOR I=1 TO N: NEXT N
```

A gép csak számlál 1-től N-ig, így N változtatásával a várakozási idő változtatható.

Elsőnek tekintsünk egy egyszerű és részletesen leírt programot. (A későbbiekben az ilyen nagyfokú részletezéstől eltekintünk.)

4.1. Számsorozatok képzése

A program a képzési szabály felismerésének és újabb tagok felírásának gyakoroltatására alkalmas.

A gép számtani sorozatot képez az

$S(I)=S(I-1)+D$ képlettel $(I>1)$

Az S(1)-et és a D-t az RND függvénnyel állíthatjuk elő.

A sorozat első 3 tagját írassuk ki a képernyőre, s szólítsuk fel a felhasználót a folytatásra. (Pl. a következő 6 tag megadására.)

Ha a programot pl. 4. osztályban akarjuk alkalmazni, arra kell ügyelnünk, hogy a sorozat szóban forgó tagjai (az első 9) között ne legyen negatív.

Ezt pl. $S(1) = RND(400)+200$ $D = RND(40)-20$ értékadásokkal érhetjük el, mivel így

$201 \leqq S(1) \leqq 600$ és $-19 \leqq D \leqq 20$.

Szerkesszük úgy a programot, hogy egyszer javítási lehetőséget kapjon a tanuló. Erre egy jelzőt (F) használunk, amelynek értékét 1-re állítjuk, ha már egyszer hibás válasz érkezett, míg új tag képzése előtt 0-ra állítjuk.

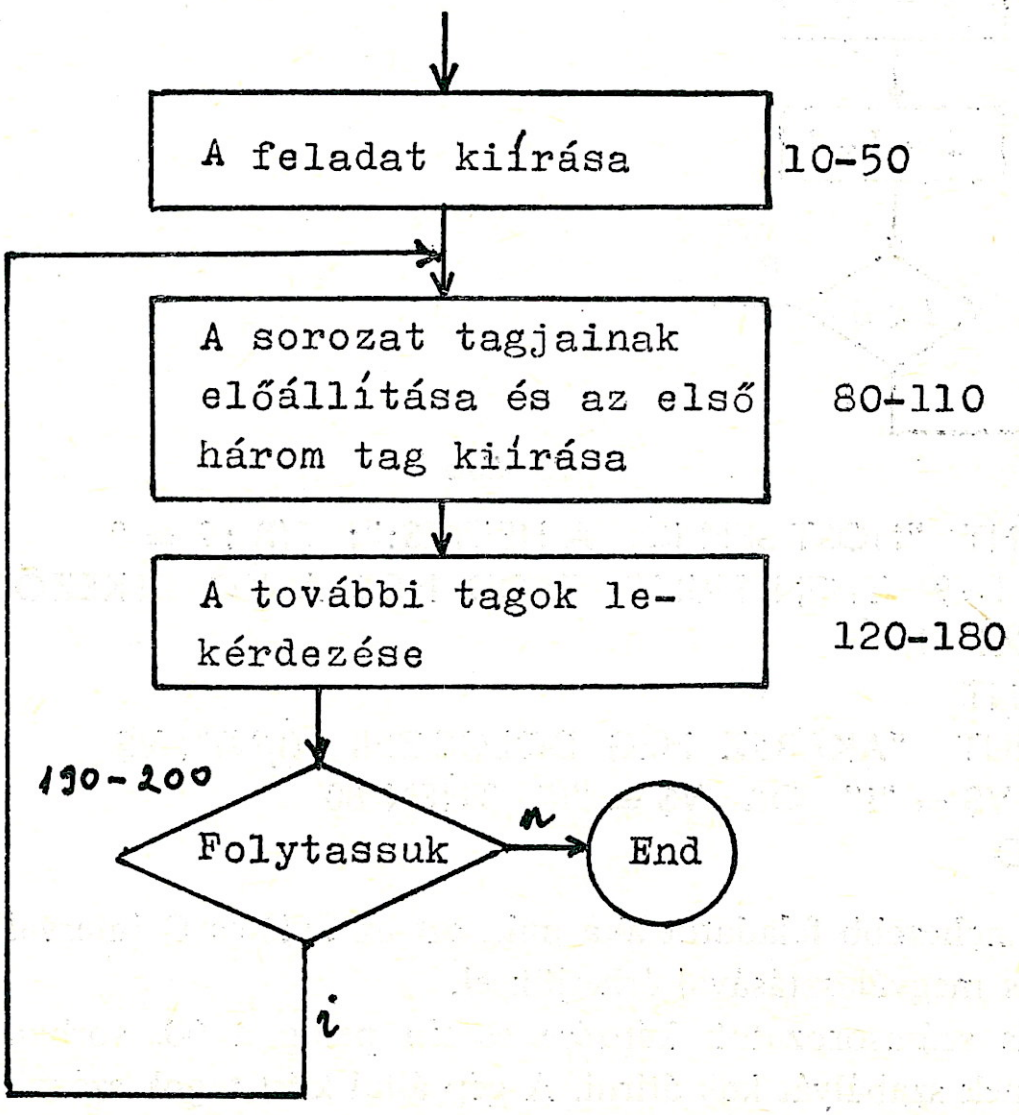
Ennek alapján nézzük a feladat szerkezetét! (12. ábra)

A legtöbb teendőt a 3. rész jelenti. Ezt nézzük részletesen. (13. ábra) A 6 tag lekérdezését ciklussal szervezzük.

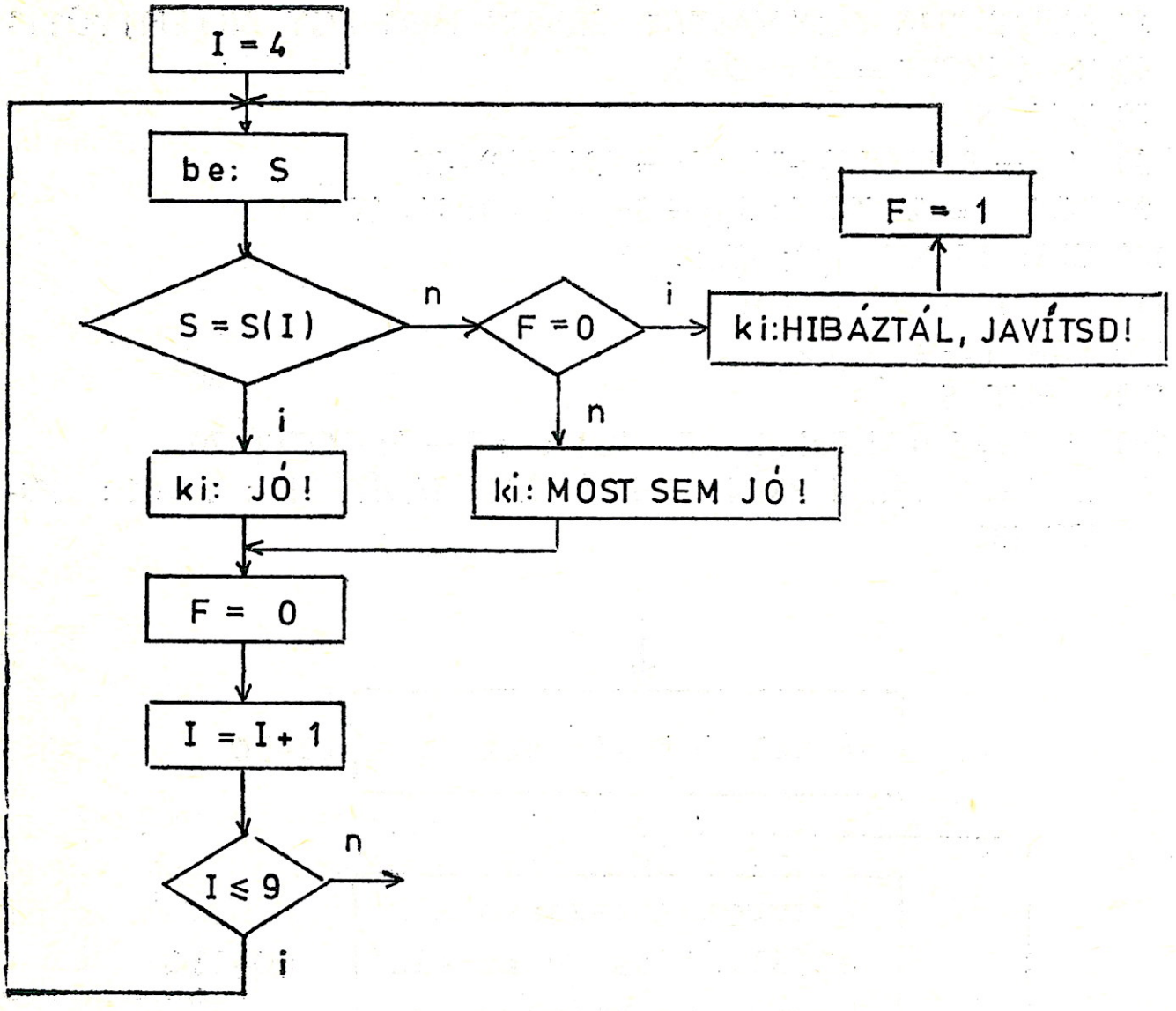
Ezek után programunk a következő lehet:

```
 5 REM  * SZÁMSOROZATOK *
 6 CLS
10 PRINT "SZÁMSOROZATOKAT KELL KÉPEZNED!"
20 PRINT "ÁLLAPÍTSD MEG A SZABÁLYT, MAJD ADD MEG A
   SOROZAT KÖVETKEZŐ 6 TAGJÁT!"
30 PRINT "AZ EGYES TAGOK BEÍRÁSA UTÁN ÉRINTSD MEG A
   RETURN BILLENTYŰT!"
40 PRINT "HA VALAMELYIK VÁLASZOD NEM JÓ, A GÉP JELZI ÉS
   EGYSZER JAVÍTHATSZ."
```

```
 50 PRINT "HA ELOLVASTAD ÉRINTS MEG EGY BILLENTYŰT!"
 60 IF  INKEY$ = ""THEN 60
 70 DIM  S(9)
 80 S(1) = RND(400)+200: D = RND(40)-20
 90 FOR  I = 2  TO 9: S(I) = S(I-1)+D: NEXT  I
100 CLS: PRINT  S(1); S(2); S(3)
110 PRINT  "FOLYTASD!"
120 FOR  I = 4  TO 9
130 INPUT  S
140 IF  S = S(I) THEN PRINT  "JÓ!";  : F = 0: GOTO 170
150 IF  F = 0  THEN PRINT  "HIBÁZTÁL, JAVÍTSD!" :  F = 1 :  GO-
    TO 130
```



12. ábra



13. ábra

```
160 PRINT "MOST SEM JÓ, A HELYES:"; S(I) : F = 0
170 IF I<9 THEN PRINT "ADD MEG A KÖVETKEZŐT!"
180 NEXT I
185 PRINT
190 INPUT "AKARSZ MÉG DOLGOZNI (I/N)"; V$
200 IF V$ = "I" OR V$ = "i" THEN 80
210 END
```

Ha nehezebb feladatot akarunk, ezt az S(1) és D intervallumának alkalmas megválasztásával érhetjük el.

Más számsorozatok képzése esetén pedig a 90. sorban az S(I) képzésének szabályát kell átírni. A gép által kiírt tagok számát is megnövelhetjük.

4.2. Állítások 1.

A program gyakoroltatásra és számonkérésre egyaránt alkalmas, felhasználásakor (futtatáskor) választhatunk a két lehetőség közül.

A gépben állításokat tárolunk, amelyekről a tanulóknak el kell dönteni, hogy helyesek vagy sem. Az állításokat egy A$( ) tömbben tároljuk, s az állításokhoz tartozó logikai értékeket (I=igaz, N=hamis) pedig egy V$( ) tömbben.

A program egyik paramétere, hogy hány állítást vizsgáljon a tanuló. (Ez nyilván kisebb vagy egyenlő, mint a gépben tárolt állítások száma.) A gép véletlenszerűen választ az állítások közül, így annál jobb a programunk, a futtatások annál változatosabbak, minél több állítást tárolunk. Célszerű az állításokat, a hozzájuk tartozó logikai értékeket és magyarázatokat, mint adatokat, szalagon rögzíteni. Ekkor különböző témaköröknek megfelelően, más-más adatszalaggal futtathatjuk a programot. Természetesen nemcsak matematikaórán használhatjuk ezt a programot. Bármilyen tárgykörből választhatjuk az állításokat, ha azokra egyértelmű igennel vagy nemmel lehet válaszolni.

Ha a programot számonkérésre akarjuk használni, akkor az egyes érdemjegyek eléréséhez szükséges helyes válaszok számát is a futtatások alkalmával kell megadnunk. Ekkor a gép érdemjeggyel is értékel, egyébként csak a jó és a rossz válaszok számát írja ki.

Az egyes válaszok után a gép azonnal visszajelzést ad, hogy helyes volt vagy nem. Rossz válasz esetén magyarázatot is kérhetünk. Ezért az egyes állítások igazságának vagy hamisságának indoklására szolgáló magyarázatokat szintén tárolnunk kell. (Ez egy M$( ) tömbben történik.)

A program három fő részből áll:

1. A feladat ismertetése és a paraméterek megadása.
2. Az állítások kiírása és a válaszok értékelése.
3. A tanuló teljesítményének értékelése.

Ezek a fő részek a 3500, 3600 és 4000 sorszámmal kezdődő szubrutinok. A főprogram az adatoknak (az állításoknak, a hozzájuk tartozó logikai értékeknek és a magyarázatoknak) a memóriába való beolvasásából és a három szubrutin hívásából áll. A második szubrutint

ciklikusan hívja a főprogram, ahány állítást kap a tanuló. Annak érdekében, hogy ugyanazt az állítást kétszer ne adja a gép a tanulónak, a már előállított véletlen számokat tároljuk (B( ) tömb), és ellenőrizzük, hogy az újonnan előállított véletlen szám előfordult-e már, s ha igen, új számot állíttatunk elő. (3610—3650)

A programban még szereplő fontosabb változók:

H: helyes válaszok száma

R: helytelen válaszok száma

E ( ) tömb megfelelő elemei az egyes osztályzatokhoz szükséges helyes válaszok száma. (Pl. E(4)-ben tároljuk a 4-es eléréséhez szükséges helyes válaszok számát.)

```
  10 REM * ÁLLÍTÁSOK *
  20 CLEAR 5000
  30 OPEN "ADAT"
  40 INPUT # N
  50 DIM  A$(N), V$(N), M$(N), B$(N), E(5)
  60 FOR  I = 1  TO N : INPUT # A$(I), V$(I), M$(I) : NEXT  I
  65 CLOSE
  70 CLS:  GOSUB 3500: REM * A FELADAT KIÍRÁSA *
  80 FOR  I = 1  TO N1
  90 CLS:  GOSUB 3600:  REM * AZ ÁLLÍTÁSOK KIÍRÁSA, A VÁ
     LASZOK ÉRTÉKELÉSE *
 100 NEXT  I
 110 CLS:  GOSUB 4000: REM * ÉRTÉKELÉS *
 120 END
3500 REM * A FELADAT *
3510 PRINT "ÁLLÍTÁSOKRÓL EL KELL DÖNTENI, HOGY IGAZAK"
3520 PRINT "VAGY NEM.  HA IGAZ, AKKOR AZ I BETŰT, HA"
3530 PRINT "NEM, AKKOR AZ N BETŰT KELL MEGÉRINTENI."
3540 PRINT "ROSSZ VÁLASZ ESETÉN MAGYARÁZATOT LEHET"
3550 PRINT "KÉRNI A GÉPTŐL."
3552 PRINT "HÁNY ÁLLÍTÁST ÍRJON KI A GÉP (MAX "; N;")";
3554 INPUT N1
3560 PRINT "GYAKOROLTATNI AKAR (1), VAGY ÉRDEMJEGGYEL"
```

```
3565 PRINT "AKARJA A TELJESÍTMÉNYT ÉRTÉKELNI (2)";
3570 INPUT K
3572 IF K = 1 OR K = 2 THEN 3574 ELSE PRINT "CSAK 1, VAGY 2
     LEHET A VÁLASZ!  VÁLASZOLJON ÚJRA!" :  GOTO 3570
3574 IF  K = 1  THEN  E = 1 :  GOTO 3590
3576 CLS: P1$ = "HÁNY JÓ VÁLASZ ESETÉN KAP A TANULÓ" :
     P2$ = "ÉRDEM-JEGYET"
3578 FOR  K = 5  TO 2  STEP -1
3580 PRINT  P1$;  K;  P2$; : INPUT  E(K)
3582 NEXT  K
3590 RETURN
3600 REM * AZ ÁLLÍTÁSOK KIÍRÁSA *
3610 J = RND(N)
3620 FOR  K = 1  TO  I-1
3630 IF  J = B(K)  THEN 3610
3640 NEXT  K
3650 B(I) = J
3660 PRINT  A$(J)
3670 W$ = INKEY$ :  IF W$ = ""  THEN 3670
3680 IF  W$ = "I"  OR  W$ = "i"  OR  W$ = "N"  OR  W$ = "n"
     THEN 3690 ELSE PRINT  "CSAK I VAGY N BETŰVEL VÁLASZ
     OLHATSZ! VÁLA-SZOL  ÚJRA!" :  GOTO 3670
3690 IF  W$ = V$(J)  THEN  H = H+1: PRINT "HELYESEN VÁLA-
     SZOLTÁL.": GOTO 3740
3700 R = R+1: PRINT  "HELYTELEN A VÁLASZOD!"
3710 PRINT "AKARSZ MAGYARÁZATOT KAPNI?  (I/N)"
3720 IF  INKEY$ = ""  THEN 3720
3730 IF  INKEY$ = "I"  OR INKEY$ = "i"  THEN PRINT M$(J)
3740 PRINT  "HA FOLYTATHATJUK ÉRINTS MEG EGY BETŰT!"
3750 IF  INKEY$ = ""  THEN 3750
3760 RETURN
4000 REM * ÉRTÉKELÉS *
4010 PRINT N1; "VÁLASZBÓL"; H; "HELYES VOLT"; R; "HIBÁS"
4020 IF  E = 1  THEN 4070
4030 FOR  K = 5  TO 1 STEP -1
```

```
4040 IF H<E(K) THEN 4060
4050 PRINT "ÉRDEMJEGYED"; K : GOTO 4070
4060 NEXT K
4070 RETURN
```

Adatszalag készítése

Az output file-t a CREATE "file név" utasítással nyitjuk meg. (Programunkban az adat-file neve ADAT)

A megnyitott file-ba a PRINT# output lista utasítással írhatjuk az adatokat.

Az adat-file-t a CLOSE utasítással zárjuk le.

Adatszalagunkra mindössze 5 állítást írtunk fel (helyszűke miatt) természetesen ez bővíthető a memória nagyságát figyelembe véve. Az állításoknál és a magyarázatoknál a stringek határát jelző "-et (idézőjelet) karakteresen (CHR$(34)) kivittük a szalagra, ellenkező esetben a szövegben lévő, (vessző)-ket elválasztó jelnek tekinti a gép.

Megjegyzés: az állítások és magyarázatok megfogalmazásánál mindig ügyeljünk arra, hogy max. 255 karakteresek lehetnek, szóközöket is beleértve.

Adatszalagot pl. a következő programmal készíthetünk:

```
10 REM * ADATSZALAG KÉSZÍTÉSE *
20 CLEAR 5000: N = 5 : DIM A$(5), V$(5), M$(5)
30 DATA "HA EGY SZORZATBAN CSAK NEGATÍV TÉNYEZŐK
       VANNAK, AKKOR A SZORZAT NEGATÍV."
40 DATA "HA EGY SZORZATBAN KÉT NEGATÍV TÉNYEZŐ VAN,
   A TÖBBI POZITÍV, AKKOR A SZORZAT POZITÍV."
50 DATA "HA EGY SZORZATBAN A NEGATÍV TÉNYEZŐK SZÁMA
   PÁROS, AKKOR A SZORZAT POZITÍV."
60 DATA "HA EGY SZORZATBAN A NEGATÍV TÉNYEZŐK SZÁMA
   PÁRATLAN, AKKOR A SZORZAT NEGATÍV."
70 DATA "HA EGY SZORZATBAN A POZITÍV TÉNYEZŐK SZÁMA
   PÁROS, AKKOR A SZORZAT POZITÍV."
80 DATA "N", "I", "I", "I", "N".
```

```
 90 DATA "FIGYELD MEG A KÖVETKEZŐ PÉLDÁT!
    (−2)*(−3)=6"
100 DATA "CSOPORTOSÍTSD KÜLÖN A NEGATÍV ÉS A POZITÍV
    TÉNYEZŐKET! A KÉT NEGATÍV TÉNYEZŐ SZORZATA POZI-
    TÍV, A POZITÍVAK SZORZATA SZINTÉN, ÍGY AZ EREDMÉNY IS
    POZITÍV."
110 DATA "A NEGATÍV TÉNYEZŐKET OSSZAD KETTES CSOPOR-
    TOKBA. AZ EGYES CSOPORTOK SZORZATA POZI-
    TÍV, A POZITÍVAK SZORZATA SZINTÉN, ÍGY AZ EREDMÉNY
    IS AZ."
120 DATA "FIGYELD MEG A KÖVETKEZŐ PÉLDÁT!
    (−2)*(−4)*(−3)*2 = −48"
130 DATA "AZ ELŐJEL A NEGATÍV TÉNYEZŐK SZÁMÁTÓL
    FÜGG!     2*3*(−1) = −6"
140 FOR I = 1 TO N: READ A$(I) : A$(I) = CHR$(34)+A$(I)+
    +CHR$(34) : NEX T I
150 FOR I = 1 TO N: READ V$(I) : NEXT I
160 FOR I = 1 TO N : READ M$(I) : M$(I) = CHR$(34)+M$(I)+
    +CHR$(34) : NEX T I
170 CREATE "ADAT"
180 PRINT# N
190 FOR I = 1 TO 5 : PRINT# A$(I), V$(I), M$(I) : NEXT I
200 CLOSE
210 END
```

Az így elkészített adatszalagot használja a program. OPEN "file név" utasítás hatására (ha a magnó PLAY gombját benyomtuk) a gép megkeresi a megadott nevű adat-file-t, és rááll az első adatra. Az adatokat az INPUT # input lista utasítással olvashatjuk be. Az input-file lezárása a CLOSE utasítással történik. (Ez nem kötelező.)

4.3. Állítások 2.

A következő program önellenőrzést tesz lehetővé. Mindenki saját maga teheti próbára tudását.

Állítások jelennek meg a képernyőn, amelyek közül ki kell választani a helyeset (mindig csak egy helyes). A gép, ha kérjük, azonnal visszajelzést ad, hogy helyes-e a válasz, és ha nem, megadja az igaz állítás sorszámát. Ha úgy rendelkezünk, a visszajelzés elmarad. Helyes válaszaink száma tárolódik, és a feladat végén megjelenik a képernyőn, hogy hány válaszból mennyi volt jó, és a teljesítményt a gép %-osan értékeli.

Így a programot számonkérésre is felhasználhatjuk, ha a %-ot érdemjegyre vagy pontra váltjuk át. Ilyenkor célszerű a válaszok utáni visszajelzést letiltani, mert az esetleg információt szolgáltathat a következő állítás sorozatokra vonatkozóan is.

Az állítás-sorozatok száma (N) és az egyes sorozatokban lévő állítások száma is tetszőleges. Ez utóbbinak egyetlen korlátja, hogy az állítások rá férjenek a képernyőre.

Az állításokat az A$( ) tömbben tároljuk.
S(1), ..., S(N) jelenti az egyes állítás-sorozatokban az állítások számát.
V(1), ..., V(N) pedig a helyes állítások sorszámait tartalmazza.

Az adatokat, állításokat szalagra rögzítettük, hasonló módon mint az előző programnál.

A főprogram a következő részekre tagolható:

1. Az adatok beolvasása szalagról.
2. A feladatot kiíró szubrutin hívása
3. Az állítás sorozatokat kiíró szubrutin ciklikus hívása
4. Végső értékelés.

A feladat kiírását végző szubrutinban (500—630) van elhelyezve a választás, hogy kérünk-e azonnali visszajelzést a válaszokra vagy sem.

Az állításokat kiíró szubrutin (1000—1120) tartalmazza a válasz lekérdezését és értékelését is.

A program alkalmazhatósága éppolyan széles körű, mint az előbb ismertetetté, ha különböző témakörökből a szükséges adatszalagot elkészítjük.

Kis módosítással a programot alkalmassá tehetjük olyan állítássorozatok vizsgálatára, amelyek közül nemcsak egy állítás helyes. Ilyen-

kor az egyes állítás-sorozatok helyes állításainak számát is tárolnunk kell, valamint a helyes állítások sorszámait.

Nézzük ezek után a programot!

```
 10 REM * ÁLLÍTÁSOK *
 20 CLEAR 5000
 30 OPEN "ADAT"
 40 INPUT# N
 50 DIM S(N), V(N)
 60 FOR I = 1 TO N: INPUT# S(I) : NEXT I
 70 FOR I = 1 TO N: INPUT# V(I) : NEXT I
 80 S = 0 : FOR I = 1 TO N : S = S+S(I) : NEXT I
 90 DIM A$(S)
100 FOR I = 1 TO S: INPUT# A$(I) : NEXT I: CLOSE
110 CLS: H1 = 0: K = 0: GOSUB 500 : REM * A FELADAT KIÍRÁ
    SA *
120 FOR I = 1 TO N
130 CLS: GOSUB 1000 : REM * EGY ÁLLÍTÁS SOROZAT KIÍRÁ
    SA *
140 NEXT I
150 CLS: PRINT N; "VÁLASZBÓL"; H1; "VOLT HELYES,"; N-
    H1; "HIBÁS!"
160 PRINT "TELJESÍTMÉNYED"; USING "###.##"; H1/N*
    100; : PRINT "%-OS."
170 END
500 REM * A FELADAT KIÍRÁSA *
510 PRINT "ÁLLÍTÁSOK JELENNEK MEG A KÉPERNYŐN, ME-"
520 PRINT "LYEK KÖZÜL KI KELL VÁLASZTANI A HELYESET!"
530 PRINT "MINDIG CSAK EGY HELYES. A HELYESNEK ÍTÉLT"
540 PRINT "VÁLASZ SORSZÁMÁT KELL BEÍRNI. (PONT NÉL-
    KÜL. )"
550 PRINT "JELEZZE-E A GÉP, HOGY A VÁLASZ HELYES,
    VAGY HELYTELEN? (I/N)"
560 V$ = INKEY$ : IF V$ = "" THEN 560
```

```
570 IF V$ = "I" OR V$ = "i" OR V$ = "N" OR V$ = "n" TH
    EN 600
580 PRINT "CSAK I VAGY N BETŰ LEHET A VÁLASZ!"
590 PRINT "ÚJ VÁLASZT KÉREK!" : GOTO 560
600 IF V$ = "I" OR V$ = "i" THEN V1 = 1
610 PRINT "HA ELOLVASTAD, ÉRINTS MEG EGY BILLENTYŰT!"
620 IF INKEY$ = " " THEN 620
630 RETURN
1000 REM * AZ ÁLLÍTÁSOK KIÍRÁSA *
1010 FOR J = 1 TO S(I)
1020 K = K+1 : PRINT J; ". "; A$(K)
1030 NEXT J
1040 PRINT "HÁNYADIK ÁLLÍTÁST TARTOD HELYESNEK?"
1050 INPUT W
1060 IF W= V(I) THEN H1 = H1+1
1070 IF V1<>1 THEN 1100
1080 IF W = V(I) THEN PRINT "A VÁLASZ HELYES!": GOTO
     1100
1090 PRINT "A VÁLASZ HELYTELEN! AZ IGAZ ÁLLÍTÁS SOR-
          SZÁMA"; V(I);". "
1100 PRINT "HA FOLYTATHATJUK, ÉRINTS MEG EGY BETŰT!"
1110 IF INKEY$ = " " THEN 1110
1120 RETURN
```

Az adatszalagot a következő programmal készítettük:

```
10 REM * ADATSZALAG KÉSZÍTÉSE *
20 CLEAR 5000
30 N = 5
50 DIM S(N), V(N)
50 DATA 6, 3, 4, 3, 3
60 FOR I = 1 TO N: READ S(I):S = S+S(I): NEXT I
70 DATA 4, 2, 3, 2, 3
80 FOR I = 1 TO N : READ V(I) : NEXT I
90 DIM A$(S)
```

```
100 DATA "KÉT KÜLÖNBÖZŐ POZITÍV EGÉSZ SZÁM LEG-
    KISEBB KÖZÖS TÖBBSZÖRÖSE -KISEBB MINDKÉT
    SZÁMNÁL."
110 DATA "-NAGYOBB MINDKÉT SZÁMNÁL."
120 DATA "-KISEBB A NAGYOBB SZÁMNÁL."
130 DATA "-NAGYOBB A KISEBB SZÁMNÁL."
140 DATA "-KISEBB A KÉT SZÁM SZORZATÁNÁL."
150 DATA "-NAGYOBB A KÉT SZÁM SZORZATÁNÁL."
160 DATA "KÉT KÜLÖNBÖZŐ POZITÍV EGÉSZ SZÁM
    LEGNAGYOBB KÖZÖS OSZTÓJA -KIS
    EBB MINDKÉT SZÁMNÁL."
170 DATA "-KISEBB A NAGYOBB SZÁMNÁL."
180 DATA "-KISEBB A KISEBB SZÁMNÁL."
190 DATA "KÉT POZITÍV EGÉSZ SZÁMNAK -MIN
    DIG VAN KÖZÖS VALÓDI OSZTÓJA."
200 DATA "-A LEGNAGYOBB KÖZÖS OSZTÓJA MIND-
    KETTŐNEK VALÓDI OSZTÓJA."
210 DATA "-MINDIG VAN LEGKISEBB KÖZÖS OSZTÓJA."
220 DATA "-MINDIG VAN LEGNAGYOBB KÖZÖS TÖBB-
    SZÖRÖSE."
230 DATA "HA EGY KÉTTÉNYEZŐS SZORZAT OSZTHATÓ
    5-TEL, AKKOR MINDKÉT TÉNYEZŐJE OSZT-  HATÓ
    5-TEL."
240 DATA "HA EGY SZORZAT OSZTHATÓ 7-TEL, AKKOR
    LEGALÁBB AZ EGYIK TÉNYEZŐJE OSZTHATÓ  7-TEL."
250 DATA "HA EGY SZORZAT OSZTHATÓ 20-SZAL, AK-
    KOR LEGALÁBB AZ EGYIK TÉNYEZŐJE  OSZT-
    HATÓ 20-SZAL."
260 DATA "HA EGY KÉTTAGÚ ÖSSZEG OSZTHATÓ EGY
    SZÁMMAL, AKKOR MINDKÉT TAGJA OSZT-  HATÓ
    VELE."
270 DATA "HA EGY ÖSSZEG EGYIK TAGJA NEM OSZT-
    HATÓ 7-TEL, AKKOR AZ ÖSSZEG SEM  OSZTHATÓ
    7-TEL."
280 DATA "HA EGY KÉTTAGÚ ÖSSZEG OSZTHATÓ 3-MAL,
```

```
    ÉS AZ EGYIK TAGJA IS, AKKOR A MÁSIK          TAG IS OSZ-
    THATÓ 3-MAL."
290 FOR I = 1 TO S
300 READ  A$(I) : A$(I) = CHR$(34)+A$(I)+CHR$(34)
310 NEXT  I
320 CREATE  "ADAT"
330 PRINT# N
340 FOR I = 1 TO N : PRINT# S(I) : NEXT  I
350 FOR I = 1 TO N : PRINT# V(I) : NEXT  I
360 FOR I = 1 TO S : PRINT# A$(I) : NEXT  I
370 CLOSE
380 END
```

4.4. Szabályjátékok

Könnyen számítógépre vihető az ismert szabályjáték. A programot akár gyakorlásra, akár számonkérésre is felhasználhatjuk. (Választható.) Ha célunk a gyakoroltatás, akkor a gép csak szöveges értékelést ad, míg ha számonkérésnél alkalmazzuk, úgy érdemjeggyel is minősíti a teljesítményt.

A szabály $Y=A\cdot X+B$ alakú. Az A, B és X véletlenszerűen változik $1\leqq A\leqq 10$, $-9\leqq B\leqq 10$, $-9\leqq X\leqq 10$ intervallumokban. A, B, X egész számok. (Ezen intervallumokat korosztálytól függően a programban megváltoztathatjuk.)

A feladat egy táblázat kitöltése, amely a következő formában jelenik meg a képernyőn. (14. ábra)

	1	2	3	4	5	6	7
X							
Y							

14. ábra

Az első két oszlopot a gép mindig kitölti. A többi oszlopba véletlenszerűen írja vagy az X vagy az Y értéket. (Egy M( ) tömbbe az RND függvénnyel 1-et vagy 2-t írunk. Ha az érték 1, akkor a X-et írja ki, míg ha 2, akkor az Y-t.) Ezután sorban megkérdezi a hiányzó értékeket. Helyes válasz esetén beírja a táblázatba, míg helytelen válasz esetén kiírja a jó megoldást, de a táblázatba ∗∗-ot ír. (Ez a program is tartalmaz ellenőrzést arra vonatkozóan, hogy a táblázatnak ne lehessen két azonos oszlopa.) A táblázat kitöltése után a képernyőn megjelenik a szabály is. Amennyiben a gép osztályzatot ad, az megegyezik a tanuló által adott helyes válaszok számával. (0 jó válasz is elégtelen.)

A program fő részei:

1. A táblázat kirajzolása a képernyőre
2. A véletlen számok előállítása
3. A táblázat kitöltése
4. A teljesítmény értékelése.

Ezeken kívül a programban található a képernyő bizonyos részének letörlését végző és a szüneteket, a várakozást biztosító szubrutin.

```
 10 REM ∗ SZABÁLYJÁTÉK ∗
 15 CLS: PRINT$7, 0, CHR$(2);"          SZABÁLYJÁTÉK" : L1 =
    1500 : GOSUB 4 000
 20 DIM X(7), Y(7), M(7)
 25 PRINT CHR$(1)
 30 CLS: PRINT "VALAMILYEN SZABÁLY ALAPJÁN EGY TÁBLÁZA-
    TOT KELL KITÖLTENI."
 40 PRINT "GYAKORLÁSRA VAGY ÉRTÉKELÉSRE AKARJÁK
    HASZNÁLNI A PROGRAMOT? (1 VAGY 2)"
 50 E$ = INKEY$: IF E$ = " " THEN 50
 60 IF E$ = "1" OR E$ = "2" THEN 50
 70 PRINT "CSAK 1 VAGY 2 LEHET A VÁLASZA! VÁLASZOLJON
    ÚJRA!" : GOTO 50
 80 E1 = VAL(E$)
 90 CLS: GOSUB 1000: REM ∗ TÁBLÁZAT RAJZA ∗
100 GOSUB 1200: REM ∗ A SZÁMOK GENERÁLÁSA ∗
```

```
 110 GOSUB 1400:  REM * A TÁBLÁZAT KITÖLTÉSE *
 120 GOSUB 1600:  REM * ÉRTÉKELÉS *
 130 PRINT "HA AKARSZ MÁSIK FELADATOT, INDÍTSD ÚJRA A
     PROGRAMOT!"
140 END
1000 REM * TÁBLÁZAT RAJZA *
1010 FOR  Y = 185  TO 113 STEP  -24
1020 FOR  X = 3  TO 251:  SET (X, Y) :  NEXT X
1030 NEXT  Y
1040 FOR X = 3 TO 254 STEP 31
1050 FOR Y = 185  TO 113 STEP  -1 :  SET(X, Y):  NEXT Y
1060 NEXT X
1070 PRINT$3, 3 "X"; :  PRINT$5, 3, "Y"; :  S = 1
1080 FOR  I = 1  TO 7
1090 O = I*5+2
1100 PRINT$S, O, ; USING"##"; I;
1110 NEXT I
1120 RETURN
1200 REM * A SZÁMOK GENERÁLÁSA *
1210 A = RND(10) :  B = RND(20)-10:  F$ = "####"
1220 FOR  I = 1  TO 7
1222 X(I) = RND(20)-10: IF  I = 1  THEN 1230
1224 FOR  J = 1  TO I-1
1226 IF  X(J) = X(I)  THEN 1222
1228 NEXT J
1230 Y(I) = A*X(I)+B
1240 NEXT I
1250 FOR  I = 3  TO 7:  M(I) = RND(2) :  NEXT I
1260 FOR  I = 1  TO 7
1270 O = I*5+1
1280 ON  M(I)  GOTO  1300, 1310
1290 PRINT$3, O, ; USING F$; X(I); :  PRINT$5, O, ; USING F$;
     Y(I) ;: GOTO 1320
1300 PRINT$3, O, ; USING F$; X(I); :  GOTO 1320
1310 PRINT$5, O, ; USING F$; Y(I);
```

```
1320 NEXT I
1330 RETURN
1400 REM * A TÁBLÁZAT KITÖLTÉSE *
1410 PRINT$7, 0, "KERESS SZABÁLYT ÉS TÖLTSD KI A TÁBLÁZA
     TOT!";
1420  L1 = 5000 :  GOSUB 4000
1430 H = 42 :  S = 7 :  O = 0 :  GOSUB 5000 :  PRINT$7, 0, ;
1440 T1$ = "TÖLTSD KI A TÁBLÁZAT" :  T2$ = ".  OSZLOPÁT!"
1450 FOR  I = 3  TO 7
1460 PRINT$7, 0, T1$; I; T2$
1470 IF  M(I) = 1  THEN Z = Y(I) :  Z$ = "Y" :  S = 5 ELSE  Z =
     X(I):  Z$ = "X" :  S = 3
1480 PRINT "SZÁMÍTSD KI"; Z$; "ÉRTÉKÉT!"; : INPUT "MENNYIT
     KAPTÁL"; Z1
1490 IF Z = Z1 THEN PRINT "A VÁLASZ HELYES" : H1 = H1+1 :
     PRINT$S, 5* I+1, CHR$(22); USING F$; Z; : L1 = 1000 : GOTO
     1520
1500 PRINT "A VÁLASZ HELYTELEN! A HELYES VÁLASZ "; Z;". "
1510 PRINT$S, 5*I+1,  CHR$(22);"  **"; :  L1 = 5000
1520 GOSUB 4000
1530 H = 168 :  S = 7 :  GOSUB 5000
1540 NEXT I
1550 RETURN
1600 REM * ÉRTÉKELÉS *
1610 PRINT$7, 0, "A SZABÁLY"; CHR$(2); "Y = "; A; CHR$(8);
     "*X"; USING"+ # "#; B; : PRINT CHR$(1);" VOLT."
1620 ON  H1  GOTO 1630, 1660, 1670, 1680, 1690
1630 PRINT  "TELJESÍTMÉNYED  NAGYON  GYENGE!"
1640 IF  E1 = 2  THEN PRINT  "ÉRDEMJEGYED : 1"
1650 GOTO 1710
1660 PRINT  "TELJESÍTMÉNYED  ELÉG  GYENGE!"  :  GOTO  1700
1670 PRINT  "TELJESÍTMÉNYED  KÖZEPES!  FIGYELJ  JOBBAN!"  :
     GOTO 1700
1680 PRINT "ELÉG JÓL DOLGOZTÁL!" : GOTO 1700
1690 PRINT "NAGYON SZÉPEN DOLGOZTÁL!"
```

```
1700 IF  E1 = 2  THEN PRINT  "ÉRDEMJEGYED :"; H1
1710 RETURN
4000 REM ✳ SZÜNET ✳
4010 FOR  L = 1  TO L1 :  NEXT L
4020 RETURN
5000 REM ✳ TÖRLÉS ✳
5010 PRINT$S, O, ;
5020 FOR  J = 1  TO  H
5030 PRINT  CHR$(6);" ";
5040 NEXT J
5050 RETURN
```

4.5. Abszolút értékes táblázat kitöltése

Következő programunk is az egyéni gyakorlást segíti elő. Az alábbi táblázatot kell kitölteni (15. ábra):

Az a és b értékeket a gép véletlenszerűen állítja elő. Különböző nehézségi szinteket érhetünk el, választásunk szerint 1, 2 vagy 3 jegyű

a	b	|a|	|b|	|a| – b	a – |b|	|a|-|b|	|a – b|

15. ábra

számokkal dolgozhatunk. Ezzel differenciáltan foglalkoztathatjuk a tanulókat. A gép ellenőrzést végez, hogy ne lehessen a táblázatnak két azonos sora. (Ekkor új b értéket állít elő.)

A program 6 fő része a következő:

1. A feladat kiírása a képernyőre.
2. A táblázat kirajzolása a képernyőre.
3. A nehézségi szint megválasztása.

4. Az a és b értékek előállítása és beírása a táblázatba.
5. A táblázat kitöltése.
6. A nyújtott teljesítmény értékelése.

A táblázatot soronként kell kitölteni. Minden választ értékel a gép. Ha helyesen adtuk meg az eredményt, ezt a gép beírja a táblázat megfelelő helyére. Ha az eredményünk hibás, a gép ezt közli velünk, kiírja a helyes eredményt, de a táblázatba ∗∗-ot ír. Így a táblázat teljes kitöltése után jól látható, hogy hol hibáztunk. A gép számlálja a helyes válaszokat (J1) a végső értékeléshez.

Mivel a program célja a gyakoroltatás, ezért csak szöveges értékelést ad a gép. (Ez persze nem zárja ki azt, hogy számonkérésre is használjuk, és a kitöltött táblázat alapján osztályozzunk.)
Ha a 18 válaszból

18 helyes — „Gratulálok! Kiváló!"
16—17 helyes — „Jó munka!"
12—15 helyes — „Figyelmesebben!"
0—11 helyes — „Sokat kell még gyakorolni!"
felirat jelenik meg a képernyőn.

Ebben a programban is megtalálható a képernyő bizonyos sorainak törléséről gondoskodó, valamint a szünetet eredményező szubrutin.

Még egy megjegyzés a programmal kapcsolatban. Az abszolút érték jelölésére a függőleges vonalat a CHR$(138)-cal állítottuk elő. A táblázat fejlécében ez viszont nem használható, (1 teljes pozíciót igényel, és ennyi hely nincs!) ezért itt SET utasításokkal rajzoltuk ezeket a megfelelő helyekre. (2070—2120). A kérdéses pozíciókat pedig a 2080-as sorban lévő DATA utasításba írtuk.

Fontosabb jelölések még:

E( ) tömbben találhatók az aktuális kitöltendő sor helyes értékei (3020)

F$( ) tömbben tároltuk a fejléc egyes elemeit, hogy többször és könnyen megjeleníthetők legyenek a képernyőn. (40, 50, 60-as sorban lévő értékadások és 3050-es sorban a felhasználás).

A fentiek alapján programunk a következő:

```
10 REM * ABSZOLÚT ÉRTÉKES TÁBLÁZAT KITÖLTÉSE *
20 DIM F$(6), A(3), B(3), E(6)
30 Z$ = CHR$(138) : F$ = "####"
40 F$(1) = Z$+"a"+Z$ : F$(2) = Z$+"b"+Z$ : F$(3) = Z$+
   +"a"+Z$+"-b"
50 F$(4) = "a-"+Z$+"b"+Z$ : F$(5) = Z$+"a"+Z$+
   "-"+Z$+"b"+Z$
60 F$(6) = Z$+"a-b"+Z$
70 CLS : GOSUB 1000 : REM * A FELADAT KIÍRÁSA *
80 CLS : GOSUB 2000 : REM * A TÁBLÁZAT RAJZA *
90 GOSUB 2300 : REM * FELTÉTELEK KIVÁLASZTÁSA *
100 GOSUB 2500 : REM * A SZÁMOK GENERÁLÁSA *
110 GOSUB 3000 : REM * A TÁBLÁZAT KITÖLTÉSE *
115 GOSUB 4100 : REM * ÉRTÉKELÉS *
120 PRINT "HA AKAR MÁSIK FELADATOT, INDÍTSA ÚJRA A
    PROGRAMOT!"
130 END
1000 REM * A FELADAT ISMERTETÉSE *
1010 PRINT "A PROGRAM EGÉSZ SZÁMOK ABSZOLÚT ÉRTÉKÉ
     NEK"
1020 PRINT "MEGHATÁROZÁSÁNAK ÉS EZZEL VALÓ SZÁMO
     LÁS-"
1030 PRINT "NAK GYAKOROLTATÁSÁRA SZOLGÁL."
1040 PRINT "MEGADHATJUK, HOGY MAX. HÁNY JEGYŰ SZÁM
     OK-"
1050 PRINT "KAL KÍVÁNUNK DOLGOZNI (1, 2, VAGY 3)"
1060 PRINT "A VÁLASZOKAT A GÉP ÉRTÉKELI."
1070 PRINT "HELYES VÁLASZ ESETÉN AZ EREDMÉNYT BEÍRJA"
1080 PRINT "A TÁBLÁZATBA"
1090 PRINT "HELYTELEN VÁLASZ ESETÉN A GÉP KIÍRJA A JÓ"
1100 PRINT "EREDMÉNYT, DE A TÁBLÁZATBA **-OT ÍR."
1110 PRINT "HA ELOLVASTA ÉRINTSEN MEG EGY BETŰT!"
1120 IF INKEY$ = " " THEN 1120
```

```
1130 RETURN
2000 REM * TÁBLÁZAT *
2010 FOR  Y = 185  TO 89  STEP  -24
2020 FOR X = 3  TO 251 :  SET (X, Y) :  NEXT X
2030 NEXT Y
2040 FOR  X = 3  TO 254  STEP 31
2050 FOR Y = 185  TO 89 STEP  -1:  SET (X, Y) :  NEXT  Y
2060 NEXT X
2070 PRINT$1, 2, "a  b  a  b  a-b  a-b  a-b  a-b";
2080 DATA 72, 82, 102, 112, 133, 142, 176, 184, 194, 200, 206, 214, 223,
     244
2090 FOR  I = 1  TO 14
2100 READ X
2110 FOR  Y = 177  TO 169 STEP  -1 :  SET (X, Y) :  NEXT Y
2120 NEXT  I
2130 RETURN
2300 REM * A FELTÉTELEK KIÍRÁSA *
2310 PRINT$9, 0, "LEGFELJEBB HÁNY JEGYŰ SZÁMOKKAL AKAR"
2320 PRINT  "DOLGOZNI  (1, 2, VAGY 3)"; :  H = 126
2330 INPUT Q :  IF  Q> = 1  AND Q< = 3  THEN 2340
2335 PRINT  "CSAK 1, 2,  VAGY 3 LEHET A VÁLASZA! VÁLA-
        SZOLJON ÚJRA!"
2336 FOR  I = 1  TO 1000 :  NEXT  I
2337 PRINT$11, 0,; :  GOSUB 5000 :  PRINT$11, 0,; :  GOTO 2330
2340 PRINT$9, 0, ; :  GOSUB 5000
2350 RETURN
2500 REM * A SZÁMOK GENERÁLÁSA *
2510 FOR  N = 1  TO 6
2520 ON Q GOTO 2530, 2540, 2550
2530 V = RND(19)-10 :  F$ = "###" :  GOTO 2560
2540 V = RND(199)-100 :  GOTO 2560
2550 V = RND(999)-500
2560 S = 2*INT ((N+1)/2)+1 : IF N/2 = INT (N/2) THEN O = 6 :
     B(N/2) = V ELSE 0 = 1 :  A((N+1)/2) = V :  GOTO 2570
2562 IF  N< =  2 THEN 2570
```

```
2564 FOR J = 1 TO N/2-1
2566 IF A(J) = A(N/2) AND B(J) = B(N/2) THEN 2520
2568 NEXT J
2570 PRINT$S, O, ; USING F$; V;
2580 NEXT N
2590 RETURN
3000 REM * A TÁBLÁZAT KITÖLTÉSE *
3010 FOR I = 1 TO 3
3020 E(1) = ABS(A(I)) : E(2) = ABS(B(I)) : E(3) = E(1)-B(I) : E(4) =
     = A(I)-E(2) : E(5) = E(1)-E(2): E(6) = ABS(A(I)-B(I))
3030 PRINT$9, 0, "TÖLTSE KI A TÁBLÁZAT"; I; ". SORÁT!"
3040 FOR K = 1 TO 6
3050 PRINT$10, 0, F$(K); : INPUT W
3060 IF E(K) = W THEN 3090
3070 PRINT "A VÁLASZ HELYTELEN, A HELYES VÁLASZ"; E(K);
3080 S = 2*I+1 : O = (K+1)*5+1 : PRINT$S, O, CHR$(22);"
       **"; : L1 = 1500 : GOSUB 4000 : GOTO 3120
3090 PRINT "A VÁLASZ HELYES. "; : J1 = J1+1
3100 S = 2*I+1 : O = (K+1)*5+1 : PRINT$S, O, CHR$(22);
3110 PRINTUSING F$; W; : L1 = 500 : GOSUB 4000
3120 PRINT$10, 0,; : H = 130 : GOSUB 5000
3130 NEXT K
3140 H = 84 : PRINT$9, 0,; : GOSUB 5000
3150 NEXT I
3160 RETURN
4000 REM * SZÜNET *
4010 FOR L = 1 TO L1 : NEXT L
4020 RETURN
4100 REM * ÉRTÉKELÉS *
4110 PRINT$9, 0, ;
4120 IF J1 = 18 THEN PRINT "GRATULÁLOK! KIVÁLÓ!" : GOTO
     4160
4130 IF J1>= 16 THEN PRINT "JÓ MUNKA!" : GOTO 4160
4140 IF J1>= 12 THEN PRINT "FIGYELMESEBBEN!" : GOTO
     4160
```

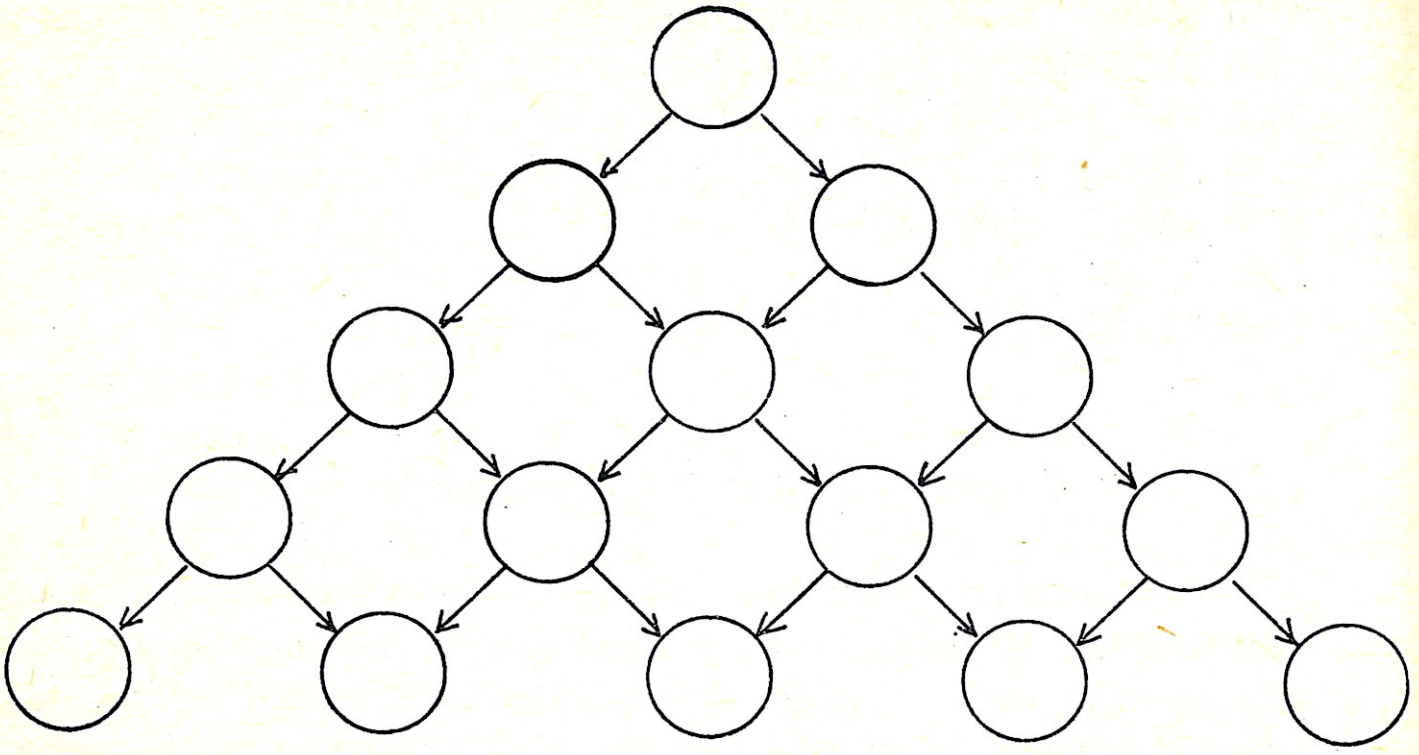
```
4150 PRINT "SOKAT KELL MÉG GYAKOROLNI!"
4160 RETURN
5000 REM * TÖRLÉS *
5010 FOR J = 1 TO H
5020 PRINT CHR$(6);" ";
5030 NEXT J
5050 RETURN
```

4.6. A számítógép mint segítőtárs egy feladat megoldásában

A következő program arra mutat példát, hogy konkrét feladat megoldásánál hogyan használhatjuk a számítógépet. Természetesen nem sok értelme lenne olyan feladatot választanunk, ahol a számítógépet csak arra használnánk, hogy a feladat szövegét a képernyőn megjelenítsük. Olyan feladatoknál, ahol a megoldás több lépésből áll, az egyes részeredmények értékelése, korrigálása, magyarázása segítséget nyújthat a feladat megértésében, további lépéseinek helyes megoldásában. A géppel való dolgozás nagy előnye, hogy akárhányszor újra lehet kezdeni, s a gép mindig ugyanolyan türelmesen válaszol. (Nagy előny ez abból a szempontból is, hogy senki sem jegyzi, még a gép sem, hogy a gyereknek hányszor kellett végignéznie, próbálgatnia a feladat megoldását, amíg azt megértette. Éppen ezért bátrabban is él ezzel a lehetőséggel.)

Az itt bemutatásra kerülő feladat a következő:
A 16. ábra felső köréből kiindulva és a nyíllal jelölt irányban haladva, hányféleképpen juthatunk el az egyes körökhöz? A lehetséges utak számát írd be a körökbe!

A gép felrajzolja az ábrát, és kiírja a feladatot. (Írásra a képernyő alsó 6 sora van fenntartva.) A válaszadásnál mindig arra a körre vonatkozó számot kell beírni, ahol egy * villog. Ha nem jó a válaszunk, a gép ezt kiírja a képernyőre. A megfelelő magyarázat után (nyíllal megjelöli, melyek azok a helyek, vagy melyik az a hely, ahonnan a kérdéses körbe juthatunk) a helyes választ beírja a szóban forgó körbe, s azután továbbhalad. Ha jó választ adunk, beírja a megfelelő helyre, és áttér a következő körre.



16. ábra

A nagyon rövid főprogram mindössze a következőkből áll:

— Az adatok elhelyezése. (A$( ) tömbben tároljuk a feladat megoldását jelentő konstansokat. Ezeket a 30-as DATA utasításba írjuk, és 40—55-ig olvassuk be a tömbbe.)

— Három szubrutin hívás

A szubrutinok a következő teendőket végzik:

1. A feladat kiírása.
2. A rajz készítése.
3. A válaszok értékelése és a számok beírása a körökbe.

A rajzoló szubrutin még kört és egyenest rajzoló szubrutint használ. Ennél is, mint minden rajzot készítő programnál, nagyon fontos az ábra pontos megtervezése a képernyőn. 5 sort kell rajzolni, minden sorba annyi kört, ahányadik sorról van szó. (Így ez két egymásba ágyazott ciklus.) A legfelső kör középpontja (126, 185). Egy sorral lejjebb lépés az X, Y koordinátákat 24, 24-gyel csökkenti, míg az egy szinten lévő körök középpontjai közötti távolság (X koordináta) 48. Ezeket figyelembe véve készült a rajzoló szubrutin. Nem kevesebb tervezést igényel a számoknak

a megfelelő körökbe való írása. (Az egyes körsorok között a távolság 2 betűsor, míg az egy sorban lévő körök között 8 pozíció.)

Programunkban még megtalálható a bizonyos sorok törlését végző és a szünetet biztosító szubrutin.

```
 10 CLS
 20 DIM A$(5, 5)
 30 DATA "1", "1", "1", "1", "2", "1", "1", "3", "3", "1", "1",
    "4", "6", "4", "1"
 40 FOR J = 1 TO 5
 45 FOR I = 1 TO J
 50 READ A$(J, I)
 55 NEXT I : NEXT J
 58 GOSUB 500 : REM * A FELADAT KIÍRÁSA *
 60 GOSUB 1000 : REM * RAJZ *
 70 T1 = 210
 80 GOSUB 2000 : REM * SZÁMOK BEÍRÁSA *
100 END
500 REM * A FELADAT KIÍRÁSA *
510 PRINT$10, 0, "AZ ÁBRA FELSŐ KÖRÉBŐL KIINDULVA ÉS A
    NYÍL-LAL JELÖLT IRÁNYBA HALADVA HÁNYFÉLEKÉPPEN
    JUTHATUNK EL AZ EGYES KÖRÖKBE?"
520 PRINT "AMELYIK KÖRBEN VILLOG A *, AZ ARRA VONAT-
    KOZÓ SZÁMOT ÍRJAD BE!"
530 RETURN
1000 REM * RAJZ *
1010 FOR J = 1 TO 5
1020 FOR I = 1 TO J
1030 O1 = 126-(J-1)*24+(I-1)*48: O2 = 185-(J-1)*24
1040 GOSUB 1200 : REM * KÖR RAJZA *
1050 IF J = 5 THEN 1080
1060 X = O1-6 : Y = O2-6 : GOSUB 1400 : REM *NYÍL RAJZA*
1080 NEXT I
1090 NEXT J
1100 RETURN
```

```
1200 REM * KÖR RAJZA *
1210 R = 7
1220 FOR K = 3. 15*R TO -3. 15*R STEP -1
1230 F = K/R : X = O1+R*COS(F) : Y = O2+R*SIN(F)
1240 SET (X, Y)
1250 NEXT K
1260 RETURN
1400 REM * NYÍL RAJZA *
1410 FOR L = 0 TO 13 : SET(X-L, Y-L) : SET(X+12+L, Y-L) :
     NEXT L
1420 X1 = X-13 : Y1 = Y-13 : X2 = X1+38
1430 FOR L = 1 TO 3
1440 SET (X1, Y1+L) : SET(X1+L, Y1) : SET(X2, Y1+L) : SET(X2-
     L, Y1)
1450 NEXT L
1460 RETURN
2000 REM * SZÁMOK BEÍRÁSA *
2010 FOR J = 2 TO 5
2020 S = 2*(J-1) : O = 20-(J-1)*4
2030 FOR I = 1 TO J
2035 F$ = " ": O1 = 0: S1 = 10: GOSUB 4000
2040 PRINT$ S, O+(I-1)*8, "*";
2050 H1 = 100 : GOSUB 5000
2060 PRINT CHR$(8); : GOSUB 5000
2070 IF F$ = " " THEN 2040
2075 IF F$ = A$(J, I) THEN PRINT$10, 0, "HELYES!"; : T1 = 40 :
     H1 = 250 ELSE 2090
2080 PRINT$S, O+(I-1)*8, A$(J, I); : GOSUB 5000 : GOTO 2200
2090 PRINT$10, 0, "NEM JÓL VÁLASZOLTÁL! NÉZD MEG HONNAN
     JUT-     HATSZ EBBE A KÖRBE!";
2095 H1 = 1500 : GOSUB 5000 : T1 = 80 : S1 = 10: O1 = 0 : GO
     SUB 4000
2100 IF J = 2 THEN PRINT$10, 0, "CSAK EGY ÚTON ÉRKEZHETSZ!" :
     H1 = 800 : T1 = 40 : GOTO 2080
2105 J1 = J-1 : I1 = I-1 : I2 = I
```

```
2110 IF I = 1 OR I = J THEN 2120 ELSE 2150
2120 S1 = 2*(J1-1)-1 : IF I = 1 THEN O1 = 20-(J1-1)*4+
     (I2-1)*8 ELSE O1 = 20-(J1-1)*4+(I1-1)*8
2125 GOSUB 3000
2130 PRINT$10, 0, "CSAK A MEGJELÖLT HELYRŐL JÖHETÜNK EBBE
     A KÖRBE ÍGY IDE IS UGYANANNYI ÚT VEZET MINT AZ ELŐ-
     ZŐBE."
2135 H1 = 2500 : GOSUB 5000 : T1 = 1 : GOSUB 4000
2140 T1 = 200 : H1 = 200 : GOTO 2080
2150 S1 = 2*(J1-1)-1: O1 = 20-(J1-1)*4+(I1-1)*8 : GOSUB
     3000 : O1 = 20-(J1-1)*4+(I2-1)*8 : GOSUB 3000
2160 PRINT$10, 0, "EBBE A KÖRBE A MEGJELÖLT KÉT HELYRŐL
     JÖHE- TÜNK MIVEL A FENTI KÖRÖKBE ";A$(J-1, I-1);"
     ILLETVE ";A$(J-1, I);" Ú- TON ÉRKEZHETÜNK, ÍGY EB
     BE A KÖRBE ";
2170 PRINT A$(J-1, I-1); "+"; A$(J-1, I); "="; A$(J, I);" ÚT VE
     ZET. ";
2175 H1 = 2500 : GOSUB 5000 : T1 = 1 : GOSUB 4000 : O1 = O1
     -8 : GOSUB 4000
2180 T1 = 200 : H1 = 200 : GOTO 2080
2190 PRINT$J-1, I, CHR$(6);" ";
2200 NEXT I
2210 NEXT J
2220 RETURN
3000 REM * NYÍL RAJZOLÁSA *
3010 PRINT$S1, O1, CHR$(133);
3020 RETURN
4000 REM * TÖRLÉS *
4005 PRINT$S1, O1,;
4010 FOR T = 1 TO T1
4020 PRINT CHR$(6);" ";
4030 NEXT T
4040 RETURN
5000 REM * SZÜNET *
5010 FOR H = 1 TO H1
```

```
5015 IF F$ = " " THEN F$ = INKEY$
5020 NEXT H
5030 RETURN
```

4.7. A sík egybevágósági transzformációi közötti kapcsolat bemutatása

A számítógépet felhasználhatjuk mint demonstrációs eszközt is. Az előre elkészített élettelen ábrákkal szemben a számítógép rajzai előttünk készülnek, dinamikusak. Jól hasznosíthatjuk ezt különösen a geometria oktatásában vagy a függvényekkel kapcsolatos témakörök tanításakor, illetve minden olyan esetben, amikor fontos a szemléltetés, és ez számítógéppel megoldható. Példaként tekintsünk egy, a sík egybevágósági transzformációinak néhány kapcsolatát bemutató programot. Akár összefoglaló órán használhatjuk, — ekkor a teljes programot — akár részekre bontva, egyes részeit külön bemutatva, új anyag feldolgozásánál. (Az eltolás, a forgatás vagy középpontos tükrözés oktatásánál.)

A program a bemutatandó transzformációs kapcsolatok kiírásával kezdődik. Ezek a következők:

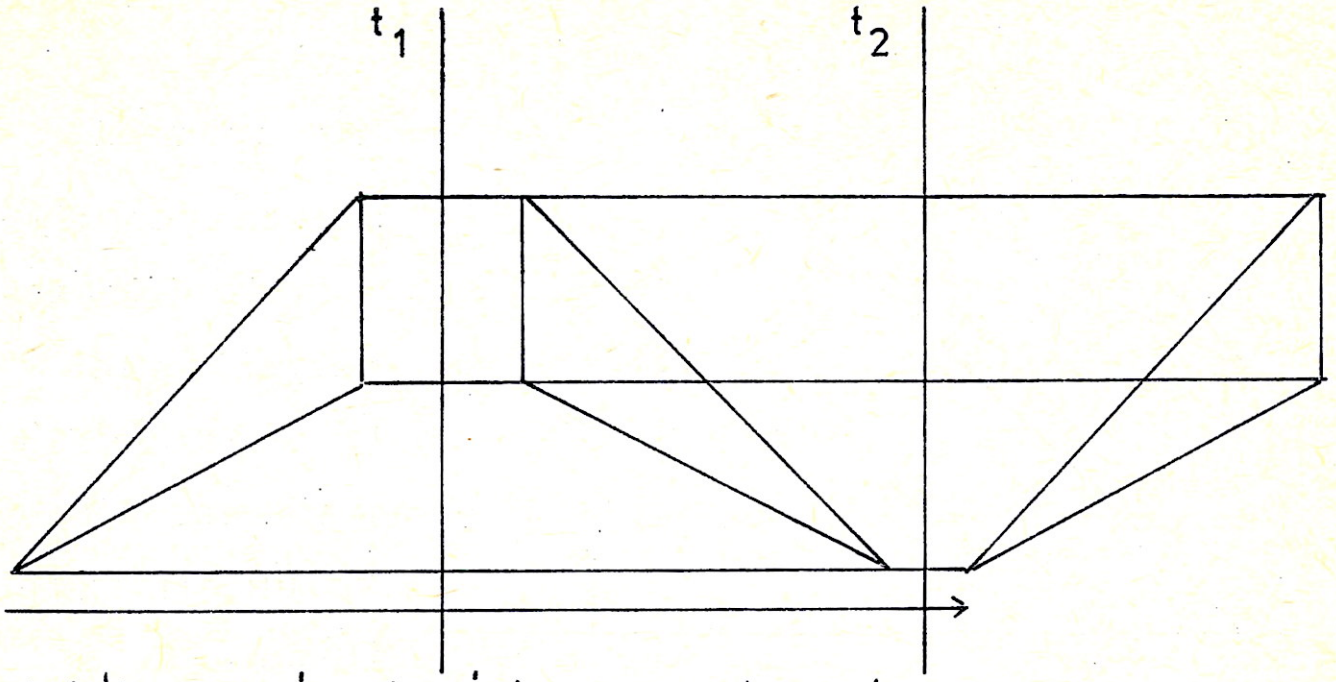
1. Két egymással párhuzamos tengelyre való tükrözés = a tengelyekre merőleges irányú eltolás.
2. Két egymást metsző tengelyre való tükrözés = a tengelyek metszéspontja körüli elforgatás.
3. Ha a két egymást metsző tengely merőleges, akkor a tengelyekre való tükrözés = a tengelyek metszéspontjára való középpontos tükrözés.

Ezek közül választhatjuk ki, hogy melyiket akarjuk megnézni.

A program szerkezete is ezt a tagolást követi. Az előbb felsorolt lehetőségek megvalósítása rendre az 1000, 2500, és 1500. sorral kezdődő szubrutinokkal történik.

A képernyőre az alábbi ábrák kerülnek felrajzolásra a fenti sorrendet figyelembevéve (17, 18, 19. ábra)

A rajzolást a következő szubrutinok végzik.



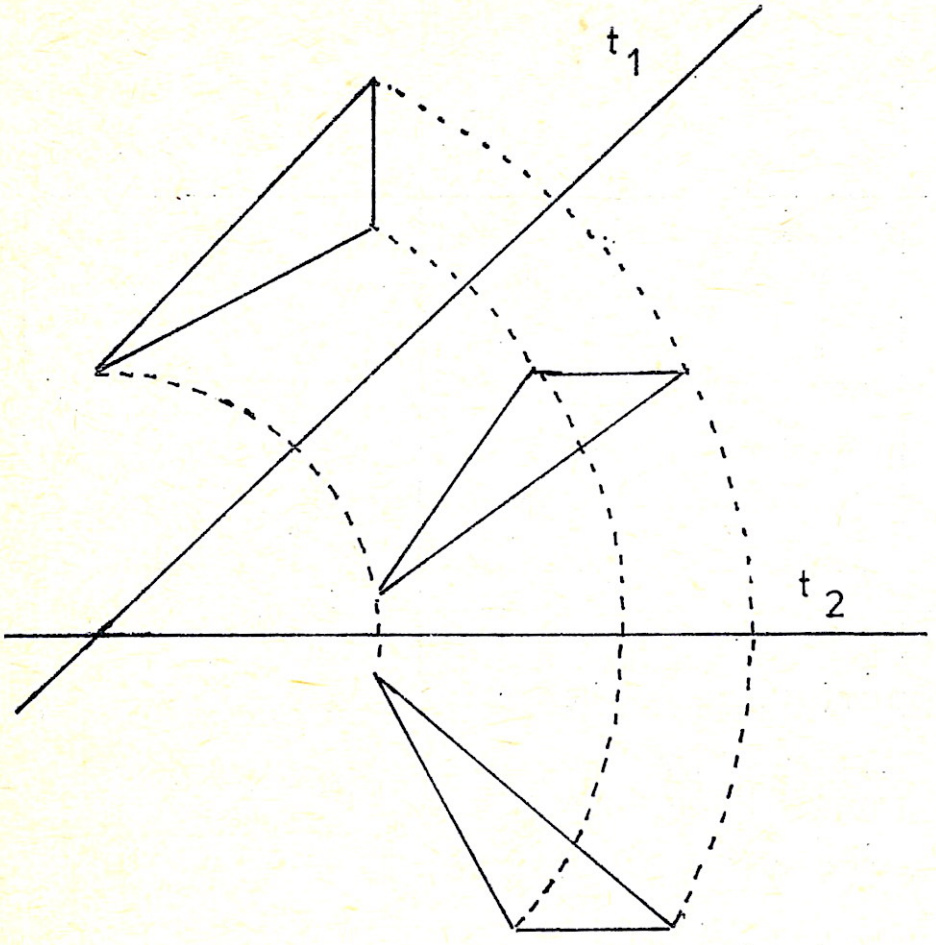
17. ábra

Az 5000. sorral kezdődő szubrutin végpontjaival adott szakaszt rajzol. Ezzel a szubrutinnal valósítjuk meg a tengelyek és a háromszögek oldalainak rajzát. A végpontok koordinátáit DATA utasításokban helyeztük el. (A 17. ábrához tartozó adatok 400—450, a 18.-hoz az adatok 510—570, míg a 19. ábra adatai a 460—500 sorok DATA utasításaiban vannak.

A megfelelő végpontok beolvasása a 2000. sorral kezdődő szubrutinnal történik, és ez a szubrutin végzi a szakaszt rajzoló szubrutin paramétereinek beállítását.

A 6000-től induló szubrutin körívet rajzol a középpontos forgatáshoz.

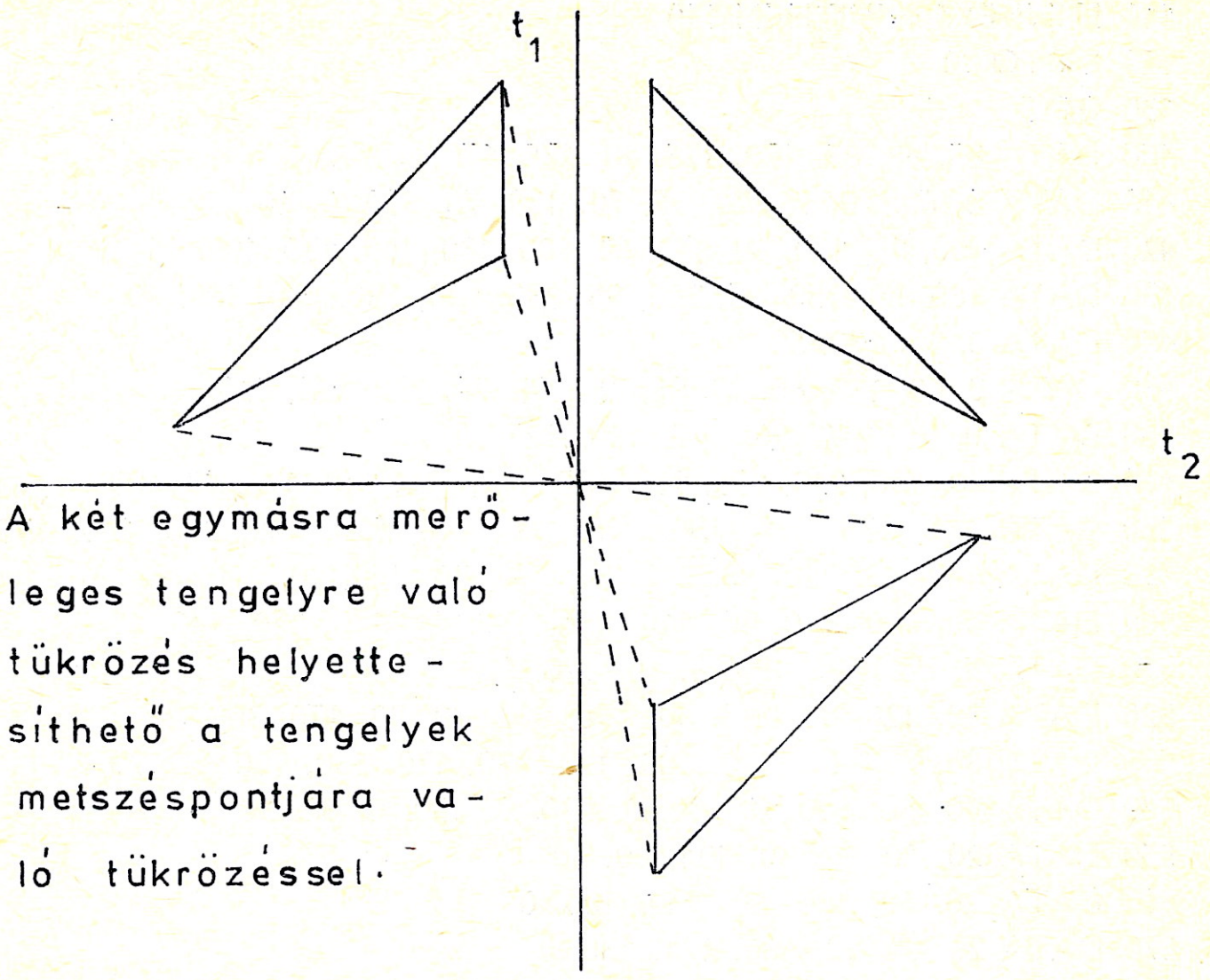
Szüneteket a 3000-es szubrutinnal biztosítunk.



Két egymást metsző tengelyre való tükrözés helyettesíthető a tengelyek metszéspontja körüli elforgatással.

18. ábra

```
10 REM *EGYBEVÁGÓSÁGI TRANSZFORMÁCIÓK KAPCSOLATA*
20 CLS : PRINT$6, 0, CHR$(2); "EGYBEVÁGÓSÁGI TRANSZ-
   FORMÁCIÓK NÉHÁNY KAP- CSOLATÁNAK BEMUTATÁSA";
   CHR$(1)
25 H1 = 5000 : GOSUB 3000
30 CLS : PRINT "1. KÉT EGYMÁSSAL PÁRHUZAMOS TENGELYRE
   VA- LÓ TÜKRÖZÉS = A TENGELYEKRE MERŐLEGES I-
       RÁNYÚ ELTOLÁS."
40 PRINT "2. KÉT EGYMÁST METSZŐ TENGELYRE VALÓ TÜK-
   RÖZÉS = A TENGELYEK METSZÉSPONTJA KÖRÜLI ELFOR
   GATÁS."
50 PRINT "3. HA A KÉT EGYMÁST METSZŐ TENGELY MERŐ-
   LEGES, AKKOR A TENGELYEKRE VALÓ TÜKRÖ-
```



19. ábra

```
      ZÉS = A TENGELYEK METSZÉSPONTJÁRA VALÓ KÖZÉP
      PONTOS TÜKRÖZÉS. "
   60 PRINT "MELYIKET AKAROD MEGNÉZNI (1, 2, 3), VAGY BE
      AKAROD FEJEZNI (4)";
   70 INPUT V
   80 IF V<1 OR V>4 THEN PRINT "A VÁLASZOD ROSSZ! VÁLA
      SZOLJ ISMÉT!"; : GOTO 70
   90 IF V = 4 THEN END
   95 CLS: PRINT "HA AZ ÁBRÁT MÁR NEM AKAROD NÉZNI,
      ÉRINTS MEG EGY BETŰT!"
   96 H1 = 3000 : GOSUB 3000
  100 ON V GOSUB 1000, 2500, 1500
```

```
110 IF INKEY$ ="" "THEN 110
120 GOTO 30
130 GOTO 130
400 DATA 85, 48, 85, 170, 175, 48, 175, 170
410 DATA 0, 60, 70, 95, 70, 95, 70, 130, 70, 130, 0, 60
420 DATA 100, 95, 170, 60, 170, 60, 100, 130, 100, 130, 100, 95
430 DATA 180, 60, 250, 95, 250, 95, 250, 130, 250, 130, 180, 60
440 DATA 0, 50, 180, 50
450 DATA 0, 60, 180, 60, 70, 95, 250, 95, 70, 130, 250, 130
460 DATA 0, 90, 240, 90, 125, 180, 125, 5
470 DATA 40, 100, 110, 135, 110, 135, 110, 170, 110, 170, 40, 100
480 DATA 140, 135, 210, 100, 210, 100, 140, 170, 140, 170, 140, 135
490 DATA 140, 45, 210, 80, 210, 80, 140, 10, 140, 10, 140, 45
500 DATA 40, 100, 210, 80, 110, 135, 140, 45, 110, 170, 140, 10
510 DATA 5, 45, 140, 180, 5, 60, 170, 60
520 DATA 30, 120, 80, 140, 80, 140, 80, 170, 80, 170, 30, 120
530 DATA 80, 70, 100, 120, 100, 120, 130, 120, 130, 120, 80, 70
540 DATA 80, 50, 100, 0, 100, 0, 130, 0, 130, 0, 80, 50
550 DATA 20, 60, 62, 30, 121, 80, 50
560 DATA 20, 60, 100, 80, 140, 100, 0
570 DATA 20, 60, 126, 80, 171, 130, 0
1000 REM *KÉT PÁRHUZAMOS TENGELYRE VALÓ TÜKRÖZÉS*
1010 CLS : RESTORE
1040 FOR I = 1 TO 11
1050 L1 = 1 : GOSUB 2000
1060 NEXT I
1070 PRINT$1, 11, "t"; CHR$(16); "1"; CHR$(17); TAB(26); "t";
     CHR$(16); "2"; CHR$(17);
1080 PRINT$13, 0, "A KÉT, EGYMÁSSAL PÁRHUZAMOS TENGELYRE
     VALÓ-   TÜKRÖZÉS HELYETTESÍTHETŐ EGY, A TENGE-
     LYEKRE MERŐLEGS IRÁNYÚ ELTOLÁSSAL. ";
1100 GOSUB 2000 : SET(X2-2, Y2-2) : SET(X2-2, Y2+2) : SET
   (X2-1, Y2-1) : SET(X2-1, Y2+1)
1110 L1 = 1
1120 FOR I = 1 TO 3
```

```
1130 GOSUB 2000
1140 NEXT I
1150 RETURN
1500 REM * KÉT MERŐLEGES TENGELYRE VALÓ TÜKRÖZÉS *
1510 CLS : RESTORE 460
1520 FOR I = 1 TO 11
1530 L1 = 1 : GOSUB 2000
1540 NEXT I
1550 PRINT$0, 18, "t"; CHR$(16); "1"; CHR$(17);
1560 PRINT$7, 39, "t"; CHR$(16); "2"; CHR$(17);
1570 PRINT$9, 0, "A KÉT EGYMÁSRA MERŐ-";
1580 PRINT$10, 0, "LEGES TENGELYRE VALÓ";
1590 PRINT$11, 0, "TÜKRÖZÉS HELYETTE-";
1600 PRINT$12, 0, "SÍTHETŐ A TENGELYEK";
1610 PRINT$13, 0, "METSZÉSPONTJÁRA VA-";
1620 PRINT$14, 0, "LÓ TÜKRÖZÉSSEL. ";
1630 L1 = 3
1640 FOR I = 1 TO 3: GOSUB 2000 : NEXT I
1650 RETURN
2000 REM * A SZAKASZ VÉGPONTJAI *
2010 READ X1, Y1, X2, Y2 : SET(X1, Y1) : SET(X2, Y2)
2030 IF X1 = X2 THEN 2050
2040 IF ABS((Y2-Y1)/(X2-X1))<1 THEN W = 1 : P1 = X1 : P2 =
     X2 : Q1 = Y1 : Q2 = Y2 : GOTO 2060
2050 W = 2 : P1 = Y1 : P2 = Y2 : Q1 = X1 : Q2 = X2
2060 GOSUB 5000
2070 RETURN
2500 REM * EGYMÁST METSZŐ TENGELYEK *
2510 CLS : RESTORE 510
2520 FOR I = 1 TO 11
2530 L1 = 1 : GOSUB 2000
2540 NEXT I
2550 PRINT$1, 19, "t"; CHR$(16); "1": CHR$(17);
2560 PRINT$10, 25, "t"; CHR$(16); "2"; CHR$(17);
2570 PRINT$1, 27, "KÉT EGYMÁST";
```

```
2580 PRINT$2, 27, "METSZŐ TENGELY-";
2590 PRINT$3, 27, "RE VALÓ TÜKRÖ-";
2600 PRINT$4, 27, "ZÉS HELYETTE-";
2610 PRINT$5, 27, "SÍTHETŐ A TEN-";
2620 PRINT$6, 27, "GELYEK METSZÉS-";
2630 PRINT$7, 27, "PONTJA KÖRÜLI";
2640 PRINT$8, 27, "ELFORGATÁSSAL."
2650 FOR I = 1 TO 3 : GOSUB 6000 : NEXT I
2660 RETURN
3000 REM * SZÜNET *
3010 FOR H = 1 TO H1 : NEXT H
3020 RETURN
5000 REM * RAJZ *
5010 IF P1<P2 THEN L = L1 ELSE L = - L1
5020 FOR P = P1 TO P2 STEP L
5030 Q = (Q2-Q1)/(P2-P1)*(P-P1)+Q1
5040 IF W = 1 THEN SET(P, Q) ELSET SET(Q, P)
5050 NEXT P
5060 RETURN
6000 REM * KÖR *
6010 READ O1, O2, R, X1, Y1, X2, Y2
6020 DEFSNG K, F
6030 FOR K = 1. 57*R TO -1. 57*R STEP -4
6040 F = K/R
6050 X = O1+R*COS(F) : Y = O2+R*SIN(F)
6060 IF X<X1 THEN 6090
6070 IF Y>Y1 OR Y<Y2 THEN 6090
6080 SET (X, Y)
6090 NEXT K
6100 RETURN
```

4.8. Nyitott mondatok grafikus megoldása

A differenciált foglalkoztatás megvalósításának egy kiváló eszköze a számítógép. Mindenki önállóan dolgozhat, képességének megfelelő gyorsasággal. Rossz megoldás esetén ugyanolyan nehézségű feladatot

ad a gép. Csak akkor haladhat tovább, ha a feladatot jól oldotta meg. Így fokozatosan nehezebb és nehezebb feladatokat kell megoldani, s közben, ahol erre szükség van, gyakorolhat. A géppel való dolgoztatás nagy előnye, hogy a tanár több gyermeket is könnyebben irányíthat, mint hagyományos módon. (Nem ő ellenőrzi a megoldás helyességét, nem ő ad új feladatot, a gyerek folyamatosan dolgozhat, nem kell megvárnia, amíg a tanár eljut hozzá, hogy az értékelést elvégezze, és a további feladatot kijelölje.) A következő programmal ilyen differenciált foglalkoztatást valósíthatunk meg. A témakör, ami feldolgozásra került, a nyitott mondatok grafikus megoldása.

A nehézségi fokozatok a következők:

1. Egyenlet — mindkét oldal elsőfokú.
2. Egyenlőtlenség — mindkét oldal elsőfokú.
3. Egyenlet — a bal oldal másodfokú, a jobb oldal elsőfokú.
4. Egyenlőtlenség — a bal oldal másodfokú, a jobb oldal elsőfokú.

(Ez több fokozattá bővíthető, ha 5.-nek olyan egyenletet engedünk meg, ahol mindkét oldal másodfokú, és a 6. fokozat az ugyanilyen típusú egyenlőtlenség lehet.)

A másodfokú kifejezésben elsőfokú tag nem szerepel. Az 1., 2., illetve 3., 4. nehézségi fokozatra 10—10 feladat van a gépbe tárolva, s ezeket $=$ illetve $<$ vagy $>$ jelekkel adja a gép. (A 2. és a 4. nehézségi fokozatnál a $<$ illetve $>$ jeleket véletlenszerűen választja a gép.) A 10—10 feladat közül is véletlenszerűen választ a gép, de mindaddig jegyzi a kapott feladat sorszámát, amíg az elsőfokú feladatokról a másodfokúra át nem térünk. Így lehet kiküszöbölni, hogy ugyanazt a feladatot kétszer kaphassa a tanuló. (Kivéve, ha már az adott típusból mindegyik feladatot megoldotta.) Ha a megoldással elkészült, beírja az eredményét, amit a gép értékel. Ha jó a megoldás, kaphatja a nehezebb feladatot, de be is fejezheti, ez esetben a gép kiírja, hogy melyik fokozatot teljesítette. (A következő alkalommal innen indulhat.) Ha rosszul oldotta meg a feladatot, a gép kirajzolja a képernyőre a megoldás lépéseit, és kiírja a megoldást. Ekkor az új feladat nehézségi foka azonos lesz a régivel. (A kirajzolást egyébként jó megoldás esetén is kérhetjük a géptől.)

A program a következő fő részekből, szubrutinokból áll:

— A feldolgozott témakör megjelölése. (1000—)
— A nehézségi fok kiírása, kiválasztása. (1300—)
— A feladat kijelölése (RND függvénnyel) a nehézségi fok alapján. (1400—)
— A feladat kiírása a képernyőre. (2000—)
— A feladat megoldása. (2300—)
— A válasz (a megoldás) megadási módjának közlése. (2600—)
— A válasz lekérdezése. (2800—)
— A válasz értékelése (3300—)
— Az értékelés eredményének kiírása. (7800—)

A főprogram a felsorolt szubrutinok hívásán kívül megszervezi a továbbhaladást (nehezebb vagy azonos szintű feladattal) vagy a megállást. (130—220)

Ezeken kívül még kisebb szubrutinok vannak a programban, ilyenek pl. az ábrázolást végző szubrutin által hívott koordináta tengelyt vagy egyenest, illetve parabolát rajzoló szubrutinok. (Itt is megtalálhatók a szüneteket és a törlést biztosító szubrutinok.)

A válasz lekérdezését végző szubrutin egyszerűbb formában is felírható, amennyiben a válaszadás módját jobban korlátozzuk. (Pl. az X betűt mindig bal oldalra kell írni, szóközöket nem lehet írni, vagy másodfokú egyenlet esetén a gyököket növekvő sorrendben kell megadni stb.) Ebben az esetben kényelmetlen lehet a program felhasználása a sok megkötés miatt. Ha viszont több szabadságot engedünk meg, a programunkat kell mindezekre felkészíteni. Ilyen pl. a szóközök kiszűrését végző szubrutin (3100—).

Általánosan is igaz, hogy egy program felhasználása, futtatása kényelmesebb legyen az alkalmazó számára, ahhoz a program készítőjének kell nehezebb feladatot megoldani.

A feladatok adatait DATA utasításokban helyeztük el (500—), de természetesen azokat itt is olvashattuk volna szalagról is.

```
 5 CLEAR 1000
10 REM * DIFERENCIÁLT FOGLALKOZTATÁS A NYITOTT MON-
   DATOK TÉMAKÖRÉBŐL. *
20 DEFINT D, X : DIM VS(10) : FS = 1 : RANDOM
30 CLS : GOSUB 1000 : REM * A FELADAT KIÍRÁSA *
40 Z = 12 000 : GOSUB 5000 : CLS
50 GOSUB 1300
60 GOSUB 1400 : REM * FELADAT KIJELÖLÉSE *
65 GOSUB 500 : M$(1, 1) = " " : M$(1, 2) = " " : K1$ = " " :
   K2$ = " " : N1$ = " " : N2$ = " "
70 CLS : PRINT "OLDD MEG GRAFIKUSAN A KÖVETKEZŐ NYI-
   TOTT MONDATOT ! " : GOSUB 2000 : REM * A FELADAT
   KIÍRÁSA A KÉPERNYŐRE *
80 GOSUB 2300 : REM * A FELADAT MEGOLDÁSA *
90 GOSUB 2600 : REM * A VÁLASZ MEGADÁSÁNAK MÓDJA *
100 GOSUB 2800 : REM * A VÁLASZ LEKÉRDEZÉSE *
110 GOSUB 3300 : REM * ÉRTÉKELÉS *
120 CLS : GOSUB 7800 : REM * AZ ÉRTÉKELÉS EREDMÉNYÉN
   EK KIÍRÁSA *
130 CLS : IF JO<2 THEN 60
140 IF NF = 4 THEN 200
150 PRINT "MEGÁLLSZ ENNÉL A NEHÉZSÉGI FOKNÁL (1),
   VAGY DOLGOZHATUNK TOVÁBB (2)";
160 INPUT Z : IF Z = 1 THEN 190
170 NF = NF+1 : IF NF = 3 THEN FOR I = 1 TO 10 : VS(I) = 0 :
   NEXT I : FS = 1
180 CLS : GOTO 60
185 IF NF = 1 THEN PRINT "AZ"; ELSE PRINT "A";
190 PRINT NF; CHR$(8); ". NEHÉZSÉGI FOKOT TELJESÍTETTED. "
200 IF NF = 4 THEN PRINT "GRATULÁLOK!" : GOTO 220
210 PRINT "LEGKÖZELEBB INNEN FOLYTATHATOD!"
220 END
500 REM * ADATOK *
510 DATA 3, 1, 1, 3, 4, 0, 6, 2, 2, 5, -1, -1, 2, 1, 1, 2, -2, 8, 1, 2
```

```
520 DATA 2, −3, −1, 3, −4, 0, 2, −6, −2, 0, 1, −3, 2, 0, 3, 1, −4,
    −1, 2, 5
530 DATA 1, −1, 3, −4, 1, 2, 0, 1, 2, 2, 1, 4, −2, 1, 0
540 DATA −2, −1, 0, −3, −1, −1, −1, −1, 1, −5, −1, 1, −5,
    1, 1
550 IF S< = 10 THEN RESTORE 510 : GOTO 560 ELSE RESTORE
    530 : GOTO 580
560 A1 = 0 : FOR I = 1 TO S : READ B1, C1, B2, C2 : NEXT I
570 GOTO 600
580 B1 = 0 : A1 = 1
590 FOR I = 1 TO S−10 : READ C1, B2, C2 : NEXT I
600 RETURN
1000 REM * A FELADAT KIÍRÁSA *
1010 PRINT CHR$(2); "DIFFERENCIÁLT FOGLAL−
        KOZTATÁS A NYITOTT                    MONDA
    TOK TÉMAKÖRÉBŐL" :  PRINT : PRINT CHR$(1)
1020 PRINT "KÜLÖNBÖZŐ NEHÉZSÉGŰ NYITOTT MONDATOKAT
        KELL GRAFIKUSAN MEGOLDANI."
1030 PRINT "A KEZDŐ NEHÉZSÉGI FOKOT MEGVÁLASZTHATOD."
1040 PRINT "HA NEM JÓL OLDOD MEG, ISMÉT UGYANOLYAN
    NE−    HÉZSÉGŰ FELADATOT KAPSZ."
1050 PRINT "CSAK AKKOR KAPHATSZ NEHEZEBB FELADATOT,
    HA A KÖNNYEBBET JÓL OLDOTTAD MEG."
1060 RETURN
1300 REM *NEHÉZSÉGI FOK KIÍRÁSA*
1310 PRINT "NEHÉZSÉGI FOK :"
1320 PRINT "1. ELSŐFOKÚ EGYENLETET KELL MEGOLDANI."
1330 PRINT "2. ELSŐFOKÚ EGYENLŐTLENSÉGET KELL MEGOL
    DA−  NI."
1340 PRINT "3. MÁSODFOKÚ EGYENLETET KELL MEGOLDANI."
1350. PRINT "4. MÁSODFOKÚ EGYENLŐTLENSÉGET KELL MEGOL
      −DANI."
1360 INPUT "MILYEN NEHÉZSÉGI FOKKAL INDULSZ"; NF
1370 IF  NF> = 1  AND  NF< = 4  THEN 1390
1380 PRINT "NEM JÓL VÁLASZOLTÁL!  VÁLASZOLJ ÚJRA!" :  GO
    TO 1360
```

```
1390 RETURN
1400 REM * A FELADAT KIJELÖLÉSE *
1410 ON NF GOTO 1420, 1430, 1440, 1450
1420 T = 1 : S = RND(10) : GOTO 1460
1430 T = RND(2)+1 : S = RND(10) : GOTO 1460
1440 T = 1 : S = RND(10)+10 : GOTO 1460
1450 T = RND(2)+1 : S = RND(10)+10
1460 I = 1
1470 IF S = VS(I) THEN 1410
1480 I = I+1 : IF I< = FS-1 THEN 1470
1490 VS(FS) = S : IF T = 1 THEN J$ = " = "ELSE IF T = 2 THEN
     J$ = "<" ELSE J$ = ">"
1500 RETURN
2000 REM * A FELADAT KIÍRÁSA KÉPERNYŐRE *
2010 IF S>10 THEN 2080
2020 PRINT B1; "*X"; : IF C1 = 0 THEN PRINT " "; : GOTO 2040
2030 PRINTUSING "+##"; C1;
2040 PRINT J$; : IF B2 = 1 THEN PRINT " "; : GOTO 2060
2050 IF B2 = -1 THEN PRINT "-"; : GOTO 2060 ELSE PRINT
    B2;"*";
2060 PRINT "X"; USING "+##"; C2
2070 GOTO 2120
2080 PRINT "X"; CHR$(17); "2"; CHR$(16); : IF C1 = 0 THEN 2100
2090 PRINTUSING "+##"; C1;
2100 PRINT J$; : IF B2<>1 THEN PRINT "-";
2110 PRINT "X"; : IF C2<>0 THEN PRINT USING "+##"; C2 ELSE
     PRINT
2120 RETURN
2300 REM *A FELADAT MEGOLDÁSA*
2310 A = A1 : B = B1-B2 : C = C1-C2 : M$(1, 1) = " " : M$(1, 2)=
     = " "
2320 IF A = 0 THEN 2400
2330 D = SQR(B*B-4*A*C)
2340 X1 = (-B+D)/(2*A) : X2 = (-B-D)/(2*A)
```

```
2350 IF X1<X2 THEN M$(1, 1) = STR$(X1) : M$(1, 2) = STR$(X2) :
     GOTO 2365
2360 M$(1, 1) = STR$(X2) : M$(1, 2) = STR$(X1)
2365 FOR I = 1 TO 2
2366 IF ASC(M$(1, I)) = 32 THEN M$(1, I) = RIGHT$(M$(1, I), LEN
     (M$(1, I))-1)
2367 NEXT I
2370 IF J$ ="=" THEN 2470
2380 IF J$ = "<" THEN K2$ = M$(1, 1) : N2$ = M$(1, 2) : GOTO
     2470
2390 N2$ = M$(1, 1) : K2$ = M$(1, 2) : GOTO 2470
2400 X = -C/B : M$(1, 1) = STR$(X) : IF ASC(M$(1, 1)) = 32 THEN
     M$(1, 1) = RIGHT$(M$(1, 1), LEN M$(1, 1))-1)
2410 IF J$ = " = " THEN 2470
2420 IF B1>B2 THEN 2450
2430 IF J$ = "<" THEN K2$ = M$(1, 1) : GOTO 2470
2440 N2$ = M$(1, 1) : GOTO 2470
2450 IF J$ = "<" THEN N2$ = M$(1, 1) : GOTO 2470
2460 K2$ = M$(1, 1)
2470 RETURN
2600 REM * A VÁLASZ MEGADÁSÁNAK MÓDJA *
2610 PRINT "ÍRD BE A MEGOLDÁST!                    AZ EGY
     ENLŐSÉGET  PL. X = 2 ALAKBAN."
2620 PRINT "HA KÉT MEGOLDÁS VAN, VAGY A MEGOLDÁST"
2630 PRINT "TÖBB EGYENLŐTLENSÉGGEL TUDOD KIFEJEZNI,"
2640 PRINT "A RETURN GOMBOT ÉRINTSD MEG KÖZÖTTÜK!"
2650 PRINT "HA MÁR NINCS TÖBB MEGOLDÁS, A GÉP KÉRDŐ-"
2660 PRINT "JELÉRE CSAK A RETURN-T ÉRINTSD MEG! A GÉP"
2670 PRINT "EBBŐL TUDJA, HOGY A VÁLASZADÁST BEFEJEZ-"
2680 PRINT "TED.  HA A MEGOLDÁS PL. 2<X ÉS X<3, AKKOR"
2690 PRINT "EZT 2<X<3 ALAKBAN IS FELÍRHATOD."
2700 RETURN
2800 REM * A VÁLASZ LEKÉRDEZÉSE *
2810 E1$ = " " : E2$ = " " : N1$ = " " : K1$ = " " : J1 = 0 : J1$ =
     " " : J2$ = " ": S1 = 0 : S2 = 0 : J5 = 0
```

```
2820 V$ = " " : INPUT V$
2830 IF V$ = " " THEN 3040
2840 H = LEN(V$) : J5 = 1
2850 GOSUB 3110 : REM * SZÓKÖZÖK ELHAGYÁSA*
2860 I = 1
2870 IF MID$(V$, I, 1) = " = " THEN S0 = I : GOTO 3000
2880 I = I+1 : IF I< = H THEN 2870
2890 REM *EGYENLŐTLENSÉG*
2900 J1 = 0 : J1$ = " " : J2 = " ": S(1) = 0 : S(2) = 0
2910 FOR I = 1 TO H
2920 W$ = MID$(V$, I, 1)
2930 IF W$ = "<" OR W$ = ">" THEN J1 = J1+1 : S(J1) = I:
     J$(J1) = W$
2940 NEXT I
2950 IF J1 = 1 THEN H1 = H: W$ = V$: S1 = S(1): GOSUB 3200:
      GOTO 2820
2960 IF J1>2 OR J$(1)<>J$(2) THEN PRINT "ILYEN FORMÁBAN
     NEM ADHATOD MEG AZ EREDMÉNYT! ADD MEG ÚJRA!" :
     GOTO 2810
2970 V1$ = LEFT$(V$, S(2)-1) : V2$ = RIGHT$(V$, H-S(1))
2980 H1 = S(2)-1: W$ = V1$: S1 = S(1): GOSUB 3200
2990 H1 = H-S(1): W$ = V2$: S1 = S(2)-S(1): GOSUB 3200:
     GOTO 2820
3000 REM * EGYENLŐSÉG *
3010 IF ASC(MID$(V$, S0+1, 1))<58 THEN W$ = RIGHT$(V$, H - S0)
     ELSE W$ = LEFT$(V$, S0-1)
3020 IF E1$ = "" THEN E1$ = W$ ELSE E2$ = W$
3030 GOTO 2820
3040 RETURN
3100 REM * SZÓKÖZ ELHAGYÁS *
3110 J = 0 : V1$ = " "
3120 FOR I = 1 TO H
3130 W$ = MID$(V$, I, 1)
3140 IF ASC(W$) = 32 THEN 3160
3150 J = J+1 : V1$ = V1$+W$
```

```
3160 NEXT I
3170 H = J :  V$ = V1$
3180 RETURN
3200 REM * LEVÁLASZTÁS *
3210 IF ASC(MID$(W$, S1-1, 1))>64 THEN 3240
3220 W1$ = LEFT$(W$, S1-1)
3230 IF J$(1) = "<" THEN K1$ = W1$ ELSE N1$ = W1$ : GOTO
     3260
3240 W1$ = RIGHT$(W$, H1-S1)
3250 IF J$(1) = "<" THEN N1$ = W1$ ELSE K1$ = W1$
3260 RETURN
3300 REM *ÉRTÉKELÉS*
3310 IF T<>1 THEN 3360
3320 REM *EGYENLŐSÉG*
3330 IF E1$ = M$(1, 1) THEN IF E2$ = M$(1, 2) THEN JO = 2 : GOTO
     3450 ELSE JO = 1 : GOTO 3450
3340 IF E2$ = M$(1, 1) THEN IF E1$ = M$(1, 2) THEN JO = 2 : GOTO
      3450 ELSE JO = 1 : GOTO 3450
3350 IF M$(1, 2) = E1$ OR M$(1, 2) = E2$ THEN JO = 1 : GOTO 3450 :
     ELSO JO = 0 : GOTO 3450
3360 REM *EGYENLŐTLENSÉG*
3370 JO = 0
3380 IF K1$ = K2$ THEN JO = JO+1
3390 IF N1$ = N2$ THEN JO = JO+1
3400 RETURN
5000 REM *SZÜNET*
5010 FOR K = 1 TO Z : NEXT K
5020 RETURN
5200 REM *TÖRLÉS*
5210 PRINT$Q1, 0,;
5220 FOR K = 1 TO Q2
5230 PRINT CHR$(6); "  " ;
5240 NEXT K
5250 RETURN
6000 REM * KOORD, RNDSZ. RAJZA *
```

```
6010 U = 123 : V = 113
6020 Y = V
6030 FOR X = 22 TO 226 : SET(X, Y) : NEXT X
6040 SET (225, V−1) : SET(225, V+1) : SET(224, V−2) : SET(224, V+2)
6050 X = U
6060 FOR Y = 35 TO 191 : SET(X, Y) : NEXT Y
6070 SET(U−1, 190) : SET(U−2, 189) : SET(U+2, 189) : SET(U+1, 190)
6080 FOR I = −8 TO 8
6090 X = 123+1*12
6100 FOR Y = 111 TO 115 : SET(X, Y) : NEXT Y
6110 NEXT I
6120 FOR I = −6 TO 6
6130 Y = 113+I*12
6140 FOR X = 121 TO 125 : SET(X, Y) : NEXT X
6150 NEXT I
6160 PRINT$7, 3, "−8−7−6−5−4−3−2−1      1 2 3 4 5 6 7 8";:
     PRINT$ 7,20, "0" ;
6170 FOR  I = 0  TO 5
6180 PRINT$I, 17, 6−I;
6190 NEXT I
6200 FOR  I = 1  TO 6
6210 SS = 7+I
6220 PRINT$SS−1, 17, −I;
6230 NEXT I
6240 FOR  Y = 41  TO 185  STEP 12
6250 FOR  X = 27  TO 222 STEP 6:  SET(X, Y):  NEXT X
6260 NEXT Y
6270 FOR  X = 27  TO 222  STEP 12
6280 FOR  Y = 185  TO 41 STEP −6:  SET(X, Y) :  NEXT Y
6290 NEXT X
6300 RETURN
6500 REM * EGYENES RAJZA Y = M9*X+B9 *
6505 L9 = 1/12
6510 IF  ABS(M9)>1  THEN 6570
6520 FOR  P = −8  TO  8 STEP L9
```

```
6530 Q = M9*P+B9 :  X = P*12+123 :  Y = Q*12+113
6540 IF  Y> = 41  AND  Y< = 185  THEN SET(X, Y)
6550 NEXT P
6560 GOTO 6610
6570 FOR  Q = -6  TO 6 STEP L9
6580 P = (Q-B9)/M9:  X = P*12+123:  Y = Q*12+113
6590 IF  X> = 27  AND X<222  THEN SET(X, Y)
6600 NEXT Q
6610 RETURN
7000 REM * PARABOLA RAJZA Y = A9*X↑2+C9 *
7010 FOR P1 = -8 TO 8 STEP 1/12
7020 Q1 = A9*P1*P1+C9 :  IF Q1< = 6 THEN 7040
7030 NEXT P1
7040 FOR  P = P1 TO 8 STEP 0. 01
7050 Q = A9*P*P+C9 :  IF Q>6 THEN 7080
7060 X = P*12+123 :  Y = Q*12+113 :  SET(X, Y)
7070 NEXT P
7080 RETURN
7240 REM * ÁBRÁZOLÁS *
7250 GOSUB 6000 :  REM KOORD.  RNDSZ. RAJZ*
7260 PRINT$14,0,; :  GOSUB 2000
7270 PRINT$15, 0, "A BAL OLDALON LEVŐ FÜGGVÉNY RAJZA.";
7280 IF  S>10  THEN 7300
7290 M9 = B1 :  B9 = C1 :  GOSUB 6500 :  GOTO 7310
7300 A9 = A1 :  C9 = C1 :  GOSUB 7000
7310 Q1 = 15 :  Q2 = 40 :  GOSUB 5200
7320 PRINT$15, 0, "A JOBB OLDALON LEVŐ FÜGGVÉNY RAJZA.";
7330 M9 = B2 :  B9 = C2 :  GOSUB 6500
7340 Q1 = 15 :  Q2 = 40 :  GOSUB 5200
7350 PRINT$15, 0, "A MEGOLDÁS : ";
7360 ON NF GOTO 7370, 7380, 7400, 7410
7370 PRINT "X=" ;  M$(1, 1); : GOTO 7430
7380 IF K2$ = " " THEN PRINT "X<"; N2$; ELSE PRINT "X>";
     K2$;
7390 GOTO 7430
```

```
7400 PRINT "X = "; M$(1, 1);" "; "X = "; M$(1, 2); : GOTO 7430
7410 IF T = 2 THEN PRINT M$(1, 1); "<X<"; M$(1, 2); : GOTO 7430
7420 PRINT  "X<"; M$(1, 1);"   X>"; M$(1, 2);
7430 Z = 5000 :  GOSUB 5000
7440 RETURN
7800 REM * AZ ÉRTÉKELÉS EREDMÉNYÉNEK KIÍRÁSA *
7805 IF J5 = 0 THEN PRINT "NEM ADTÁL VÁLASZT!":  GOTO 7830
7815 ON  JO+1  GOTO 7820, 7890, 7900
7825 PRINT "A MEGOLDÁS NEM JÓ!"
7835 PRINT "NÉZD MEG AZ ÁBRÁZOLÁS MENETÉT ÉS AZ ERED
        MÉNYT!"
7840 Z = 2000 : GOSUB 5000
7850 CLS : GOSUB 7240 : Z = 4000 : GOSUB 5000
7860 FS = FS+1:  IF FS>10 THEN FOR I = 1 TO 10:  VS(I) = 0:
     NEXT I
7870 CLS : PRINT "ISMÉT HASONLÓ NEHÉZSÉGŰ FELADATOT
     KAPSZ."
7880 Z = 3000:  GOSUB 5000:  GOTO 7930
7890 IF  T = 1  THEN 7820 ELSE PRINT  "MEGOLDÁSOD CSAK
     RÉSZBEN JÓ!" :  GOTO 7830
7900 PRINT "JÓ A MEGOLDÁSOD!"
7910 PRINT "FELRAJZOLJA-E A GÉP A MEGOLDÁST (I/N)";
7920 INPUT Z$:  IF Z$ = "I" OR Z$="(i)" THEN CLS:  GOSUB 7240
7930 RETURN
```

4.9. Függvény transzformáció bemutatása

A program két fő logikai részből áll:

1. Megismertet az elemi transzformációkkal.
2. Az $X \mapsto C(X+A)^2+B$, illetve az $X \mapsto C|X+A|+B$

(A, B, C paraméter) függvényeket ábrázoltathatjuk elemi transzformációk segítségével.

Az 1. részt, a függvény transzformációnak, mint új anyagnak a tanításánál tudjuk jól felhasználni. Két függvény, az $X \mapsto X^2$ és az $X \mapsto |X|$

függvények transzformációit követhetjük végig. (A kettő közül választani lehet.) Mindkét tengely irányában az eltolást, a zsugorítást, illetve a nyújtást, valamint a tükrözést követhetjük nyomon. Ezen elemi transzformációs lépések rögzített A, B, C értékekkel, de mindegyik több példán keresztül kerülnek bemutatásra. Ezek megtekintését akárhányszor megismételhetjük.

A 2. részben általunk megadott A, B, C értékek ($|A| \leqq 6$, $|B| \leqq 6$, $1/6 \leqq |C| \leqq 6$ illetve $C=0$ lehet!) esetén ábrázolja a gép a függvényt, úgy, hogy az ábrázolást lebontja elemi transzformációs lépésekre, s azok bemutatásán keresztül jutunk végül a kívánt függvény ábrázolásához. (Az adatok megadásának módja és a paraméterekre vonatkozó korlátozás megjelenik a képernyőn.) Pl. Ábrázoltatni akarjuk az $X \mapsto 3(X-2)^2+4$ függvényt. Az ábrázolás menete a következő:

1. $X \mapsto X^2$
2. $X \mapsto (X-2)^2$
3. $X \mapsto 3(X-2)^2$
4. $X \mapsto 3(X-2)^2+4$

A gép felrajzolja az 1. függvényt, azután a 2-at, s letörli a képernyőről az elsőt. Ezután váltakozva megvillantja az elsőt, (ekkor a második nem látszik) majd a másodikat, (ekkor az első nincs a képernyőn), s ezt háromszor megismétli. Ez a mozgás, villogás a tekintett transzformációs lépés lényegét emeli ki. A képernyőn ezután csak a 2. függvény marad. Ezután a 3. függvényt a 2-ból ugyanolyan módon kapjuk, mint a 1-ből a 2-at. A transzformációs lépésekhez tartozó képleteket a gép akkor írja ki, amikor a tekintett függvényt elkezdi rajzolni, de ez nem is törlődik a képernyőről.

Ez a második rész különösen a függvény transzformáció elmélyítésénél, begyakorlásánál lehet hasznos. Egyéni tanulás, korrepetálás alkalmával a gyerek annyiszor nézi végig, ahányszor csak szükségesnek tartja.

A villogás (2900-as szubrutin), illetve az 1. részben a koordinátarendszer és a kiindulási függvény ($X \mapsto X^2$ ill. $X \mapsto |X|$) gyors kirajzolása megvalósításának részletezésére itt nem térünk ki. Megjegyezzük,

hogy ez egy rövid gépi kódú program segítségével, (30, 40. sorban az adatok) ennek kihívásával (CALL) valósítható meg, illetve azt használtuk ki, hogy a PRIMO gép memóriájában van egy úgynevezett másodlagos képernyő terület.

A koordinátarendszer rajzolását, a függvények ábrázolását, a képletek kiírását, a transzformációs lépések meghatározását mind külön szubrutinok valósítják meg.

```
 10 P$ = "X"+CHR$(132)+"  C*(X+A)"+CHR$(17)+"2"+
    CHR$(16)+"+B"
 12 A$ = "X"+CHR$(132)+"  C*"+CHR$(138)+"X+A"+CHR$
    (138)+"+B"
 20 DEFINT G, H :  DIM G(6),  H(6)
 30 DATA 33, 4584, -14 336, 1, -4840, -13 904
 40 DATA 33, 4552, -6144, 1, -4840, -13 904
 50 FOR  I = 1  TO 6 :  READ G(I) :  NEXT I
 60 FOR  I = 1  TO 6 :  READ H(I) :  NEXT I
100 CLS : PRINT "A GÉP FÜGGVÉNY TRANSZFORMÁCIÓT MUTAT
    BE."
110 PRINT "1. MEGISMERTET AZ ELEMI TRANSZFORMÁCIÓKKAL";
120 PRINT "2.  ";P$;" ILLETVE ";  A$
130 PRINT "FÜGGVÉNYEKET ÁBRÁZOLJA ELEMI TRANSZFORMÁ-"
135 PRINT "CIÓK SEGÍTSÉGÉVEL."
140 PRINT "MIT AKAR EZEK KÖZÜL? 1 VAGY 2"
150 INPUT  V9
160 ON  V9 GOTO 490, 600
490 REM ** ELEMI TRANSZFORM. BEMUTATÁSA **
492 CLS
493 PRINT$1, 0, "X"; CHR$(132); " X"; CHR$(17); "2"; CHR$(16);"
    VAGY ";" X"; CHR$(132);" ";CHR$(138); "X"; CHR$(138)
494 PRINT "FÜGGVÉNY TRASZFORMÁCIÓJÁT AKARJA"
495 PRINT "MEGNÉZNI?"
496 INPUT "1 VAGY 2"; W :  CLS :  GOSUB 2500
497 FOR  I = 1  TO 2000 :  NEXT I
498 GOTO 550
```

```
550 CLS: PRINT "AKARJA-E VALAMELYIK FÜGGVÉNY TRANSZ
    FOR-"
552 PRINT  "MÁCIÓJÁT MEGNÉZNI?"
560 INPUT  "IGEN (I),  NEM (N)"; V$
570 IF  V$ = "N"  THEN 600
580 PRINT  "X"; CHR$(132);  "  X"; CHR$(17);  "2"; CHR$(16);"
          VAGY  ";"  X"; CHR$(132);"  ";CHR$(138);  "X"; CHR
    $(138)
582 PRINT "FÜGGVÉNY TRANSZFORMÁCIÓJÁT AKARJA"
583 PRINT "MEGNÉZNI?"
590 INPUT "1 VAGY 2"; W :  CLS :  GOSUB 2500
595 FOR  I = 1  TO 2000 :  NEXT I
596 GOTO 550
600 CLS
605 PRINT$0, 0, "TEKINTSÜK AZ ";A$;" ILLETVE ";P$"; FÜGGVÉ-
    NYEKET!"
606 PRINT "A   C, A, B ÉRTÉKEKET VÁLTOZTATNI LEHET."
607 PRINT CHR$(138); "A"  ; CHR$(138);  "< = 6,";  CHR$(138);
    "B";  CHR$(138);"< = 6,  1/6< = ";  CHR$(138);  "C";  CHR$
    (138); "< = 6 ILL. C = 0 LEHET.";
610 PRINT "A   C,  A,  B ÉRTÉKEKET VALÓDI TÖRT ALAKBAN"
615 PRINT "KELL MEGADNI, HA VALAMELYIK EGÉSZ SZÁM,"
620 PRINT "(MÉG HA 0 IS), 1 NEVEZŐT ÍRJUNK!"
630 PRINT "NEGATÍV SZÁMNÁL A SZÁMLÁLÓ LEGYEN NEGA
    TÍV"
640 PRINT "ÉS A NEVEZŐ POZITÍV!"
650 PRINT "A SZÁMLÁLÓT ÉS A NEVEZŐT VESSZŐVEL ELVÁ-"
660 PRINT "LASZTVA KELL MEGADNI!"
670 PRINT "MIT SZERETNE?"
680 PRINT "1. A ";P$;" FÜGGVÉNYT ÁBRÁZOLTATNI?"
690 PRINT "2. A ";A$"; FÜGGVÉNYT ÁBRÁZOLTATNI?"
700 PRINT "3. BEFEJEZNI?"
702 INPUT V9 : CLS
703 ON V9 GOTO 704, 705, 706
704 V = 1 :  GOTO 710
```

```
705 W = 2 :  GOTO 710
706 END
710 INPUT "C = P/Q"; C1, C2
720 INPUT "A = P/Q"; A1, A2
730 INPUT "B = P/Q"; B1, B2
740 CLS : GOSUB 2700
750 PRINT$14, 0, "A MEGADOTT"; PRINT$15, 0, "FÜGGVÉNY";
760 FOR  I = 1  TO 3000 :  NEXT I
770 CLS:  GOTO 670
900 REM * AZ ÁBRÁZOLÁS MENETE *
910 C = P1/Q1 :  A = P2/Q2 :  B = P3/Q3
920 IF T = 0 THEN 940
925 PRINT$L, 0,;
930 GOSUB 2100 :  REM * FÜGGVÉNY KIÍRÁSA *
940 GOSUB 2000 :  REM * FÜGGVÉNY RAJZA *
950 IF  B9 = 1  THEN  I1 = 2500 ELSE I1 = 500
955 FOR  I = 1  TO I1 :  NEXT I
960 RETURN
1000 REM ** KOORDINÁTARENDSZER RAJZA **
1010 U = 147 : V = 89
1020 Y = V
1030 FOR  X = 49  TO 255  STEP 2
1040 SET (X, Y)
1050 NEXT X
1051 SET (254, V-1):  SET (254, V+1):  SET(253, V-2):  SET(253, V+2)
1060 X = U
1070 FOR Y = 0 TO 191 STEP 2
1080 SET (X, Y)
1090 NEXT Y
1091 SET(U-1, 190):  SET(U+1, 190):  SET(U-2, 189):  SET(U+2,
     189):  SET (U, 191)
1100 FOR  I = -8  TO 8
1110 X = 147+I*12
1120 FOR  Y = 87  TO 91
1125 SET(X, Y)
```

```
1130 NEXT Y
1140 NEXT I
1150 FOR  I = −7  TO 7
1160 Y = 89+I*12
1170 FOR  X = 145  TO 149
1180 SET(X, Y)
1190 NEXT X
1200 NEXT I
1210 PRINT$9, 7,;
1220 PRINT" −8, −7, −6, −5, −4, −3, −2, −1";"      1 2 3 4 5 6
     7 8"
1230 FOR I = 0 TO 6
1240 PRINT$I+1, 21,;
1250 PRINT 7−I
1260 NEXT I
1270 FOR I = 1 TO 7
1280 S = 8+I
1290 PRINT$S,21,;
1300 PRINT −I;
1310 NEXT  I
1320 RETURN
2000 REM  A  C*(X+1)↑2+B  PARABOLA  RAJZA
2001 ON SGN(C)+2 GOTO 2010, 2002, 2020
2002 Y2 = B*12+89 : Y1 = Y2
2003 FOR  X = −8  TO 8 STEP 0. 05
2004 X2 = X*12+147
2005 IF  T = 1  THEN SET(X2, Y2)  ELSE IF  INT(X) = 147 OR INT(Y2)
     = 89 THEN 2006 ELSE RESET (X2, Y2)
2006 X1 = X2
2007 NEXT X
2008 GOTO 2054
2010 D1 = (−7−B)/C : IF  W = 1  THEN  D1 = SQR(D1)
2011 IF  −D1−A<−8 THEN  X3 = −8 ELSE  X3 = −D1−A
2012 IF  D1−A>8  THEN  X4 = 8  ELSE  X4 = D1−A
2013 GOTO 2025
```

```
2020 D2 = (7−B)/C : IF W = 1 THEN D2 = SQR(D2)
2021 IF −D2−A<−8 THEN X3 = −8 ELSE X3 = −D2−A
2022 IF D2−A>8 THEN X4 = 8 ELSE X4 = D2−A
2025 IF ABS(C)>1 THEN LE = ABS(C) ELSE LE = 1
2026 FOR X = X3 TO X4 STEP 0. 05/LE
2030 IF W = 1 THEN Y = C*(X+A)↑2+B ELSE Y = C*ABS(X+
     A)+B
2040 X2 = X*12+147 : Y2 = Y*12+89
2041 IF T = 1 THEN SET(X2, Y2) ELSE IF INT(X2) = 147 OR INT(Y2)
     = 89 THEN 2045 ELSE RESET (X2, Y2)
2045 X1 = X2 : Y1 = Y2
2050 NEXT X
2054 IF B9 = 0 THEN 2060
2055 X1 = INT(X1/6)−1 : Y1 = 15−INT(Y1/12)
2056 PRINT$Y1, X1,; L;
2060 RETURN
2100 REM A FÜGGVÉNY KIÍRÁSA
2110 REM * PARAMÉTEREK *
2120 REM ** C = P1/Q1 **
2130 REM ** A = P2/Q2 **
2140 REM ** B = P3/Q3 **
2145 PRINT "X"; CHR$(132);" ";
2150 IF C = 1 THEN 2200
2155 IF C<0 THEN PRINT "—";
2160 IF Q1 = 1 THEN 2190
2170 F$ = "#/#" : PRINTUSING F$; ABS(P1); Q1
2180 GOTO 2200
2190 IF ABS(P1) = 1 THEN 2200 ELSE F$ = "#" : PRINTUSING F$;
     ABS(P1);
2200 IF A = 0 THEN IF W = 1 THEN PRINT "X"; : GOTO 2270 ELSE
     PRINT CHR$(138); "X"; CHR$(138); : GOTO 2270
2210 IF Q2 = 1 THEN 2250
2220 IF W = 1 THEN F$ = "(X+#/#)" ELSE F$ = CHR$(138)+
     +"X+#/#"+CHR$(138)
2230 PRINTUSING F$; P2; Q2
```

```
2240 GOTO 2270
2250 IF W = 1 THEN F$ = "(X+#)" ELSE F$ = CHR$(138)+"X+
     +#"+CHR$(138)
2260 PRINTUSING F$; P2;
2270 IF W = 1 THEN PRINT CHR$(17); "2"; CHR$(16);
2280 IF B = 0 THEN 2340
2290 IF Q3 = 1 THEN 2330
2300 F$ = "+#/#"
2310 PRINTUSING F$; P3; Q3;
2320 GOTO 2340
2330 F$ = "+#" : PRINTUSING F$, P3;
2340 RETURN
2500 REM *A TRANSZFORM. BEMUTATÁSA*
2510 L = 1 : T = 1 : B9 = 1
2520 GOSUB 1000: REM *KOORD. RNDSZ. RAJZA*
2530 P1 = 1 : Q1 = 1 : P2 = 0 : Q2 = 1 : P3 = 0 : Q3 = 1
2540 GOSUB 900
2541 Z = CALL (VARPTR(G(1)))
2550 P3 = 3 : L = L+1 : GOSUB 900
2551 PRINT$14,0, "2. TENGELY IRÁNYÚ"; : PRINT$15, 0, "ELTOLÁS";
2560 P3 = -5 : L = L+1 : GOSUB 900
2561 CLS : L = 1 : Z = CALL(VARPTR(H(1)))
2570 P2 = 3 : P3 = 0 : L = L+1 : GOSUB 900
2571 PRINT$14, 0, "1. TENGELY IRÁNYÚ"; : PRINT$15, 0, "ELTOLÁS";
2580 P2 = -5 : L = L+1 : GOSUB 900
2581 CLS : L = 1 : Z = CALL(VARPTR(H(1)))
2590 P1 = 2: P2 = 0 : L = L+1 : GOSUB 900
2591 PRINT$14,0, "2. TENGELY IRÁNYÚ"; : PRINT$15, 0, "NYÚJ-
     TÁS";
2592 P1 = 3 : L = L+1 : GOSUB 900
2593 CLS : L = 1 : Z = CALL(VARPTR(H(1)))
2600 P1 = 1 : Q1 = 2 : L = L+1 : GOSUB 900
2601 PRINT$14, 0, "2. TENGELY IRÁNYÚ"; : PRINT$15, 0, "ZSUGO-
     RÍTÁS";
2602 Q1 = 3 : L = L+1 : GOSUB 900
```

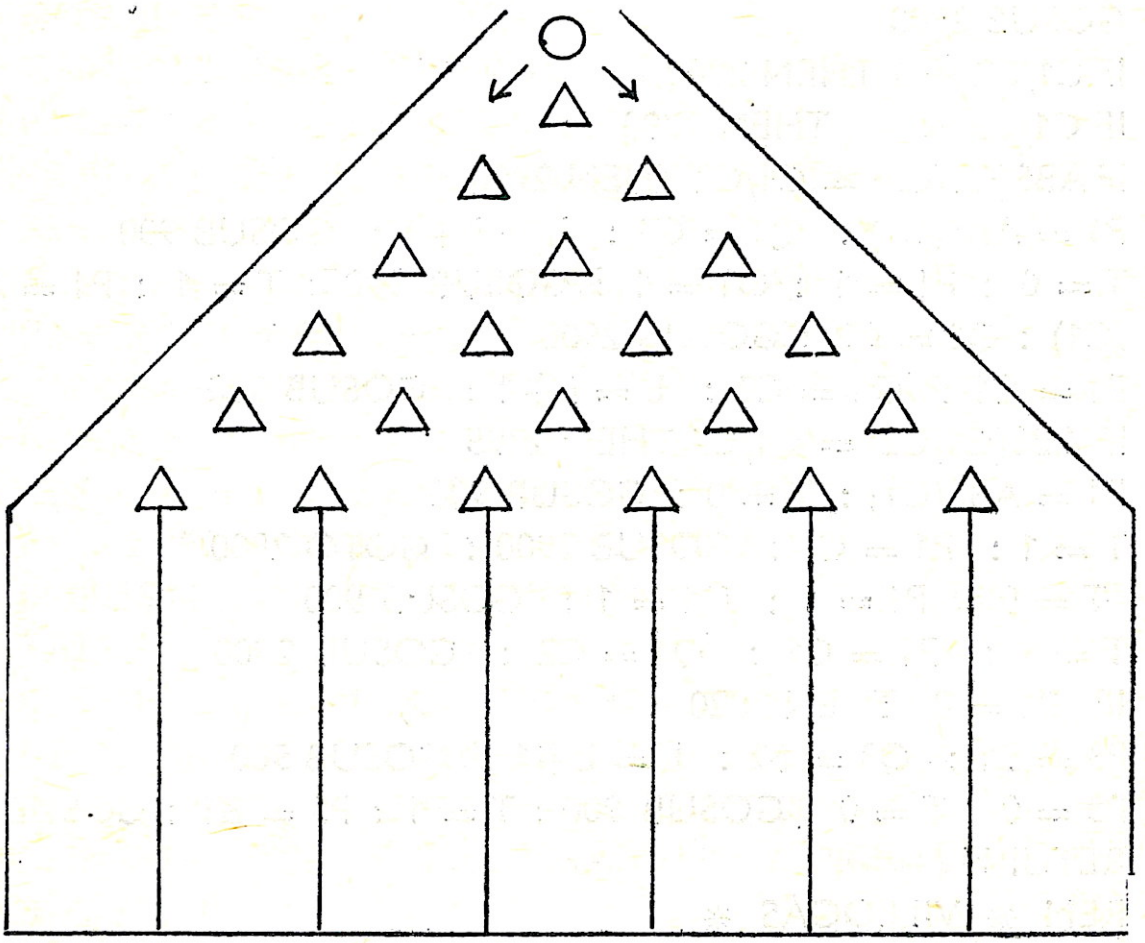
```
2603 CLS : L = 1 : Z = CALL (VARPTR(H(1)))
2610 P1 = −1 : Q1 = 1 : L = L+1 : GOSUB 900
2611 PRINT$14, 0, "1. TENGELYRE"; : PRINT$15, 0, "VONATKOZÓ
     TÜKRÖZÉS";
2612 FOR I = 1 TO 3000 : NEXT I
2620 RETURN
2700 REM * TETSZ. FGV. LÉPÉSENKÉNTI ÁBR. *
2705 B9 = 0
2710 GOSUB 1000
2720 P1 = 1 : Q1 = 1 : P2 = 0 : Q2 = 1 : P3 = 0 : Q3 = 1 :
     T = 1
2730 L = 1 : GOSUB 900
2731 Z = CALL(VARPTR(G(1))
2740 IF A1 = 0 THEN 2760
2750 P2 = A1 : Q2 = A2 : L = L+1 : GOSUB 900
2755 T = 0 : P2 = 0 : GOSUB 900 : P2 = A1 : T = 1
2756 GOSUB 2900
2760 IF C1/C2 = 1 THEN 2800
2765 IF C1/C2 = −1 THEN 2790
2770 IF ABS(C1/C2) = C1/C2 THEN 2790
2780 P1 = ABS(C1) : Q1 = C2 : L = L+1 : GOSUB 900
2785 T = 0 : P1 = 1 : Q1 = 1 : GOSUB 900 : T = 1 : P1 = ABS
     (C1) : Q1 = C2 : GOSUB 2900
2790 P1 = C1 : Q1 = C2 : L = L+1 : GOSUB 900
2791 IF ABS(C1/C2) = C1/C2 THEN 2795
2792 P1 = ABS(C1) : T = 0 : GOSUB 900
2793 T = 1 : P1 = C1 : GOSUB 2900 : GOTO 2800
2795 T = 0 : P1 = 1 : Q1 = 1 : GOSUB 900
2796 T = 1 : P1 = C1 : Q1 = C2 : GOSUB 2900
2800 IF B1 = 0 THEN 2820
2810 P3 = B1 : Q3 = B2 : L = L+1 : GOSUB 900
2811 P3 = 0 : T = 0 : GOSUB 900 : T = 1 : P3 = B1 : GOSUB 2900
2820 RETURN
2900 REM * VILLOGÁS *
```

```
2910 FOR J = 1 TO 3: OUT 6, 240: FOR I = 1 TO 500: NEXT I :
     OUT 6, 248 : FOR I = 1 TO 500 : NEXT I : NEXT J
2920 Z = CALL(VARPTR(G(1)))
2930 RETURN
```

4.10. Galton deszka szimulációja

Végül bemutatunk egy programot, amely számítógépes szimulációt valósít meg. A gép a Galton deszkát szimulálja. A szintek száma és a golyók száma paraméter. (A szintek száma max. 14 lehet.) Ezen értékek megadása után a képernyőn megjelenik a Galton deszka rajza. (20. ábra)

A golyók fentről indulva, az ékekbe ütközve véletlenszerűen térnek el jobbra vagy balra. A golyók útját végigkövethetjük, a tartályba pottyanva, egy-egy vízszintes vonal reprezentálja azokat. Így szemünk



20. ábra

láttára alakul a hisztogram. (Az egyes tartályokba milyen gyakoriságokkal jutnak a golyók.) A képernyő jobb felső sarkában a mozgásban levő golyó sorszáma látható.

Rámutathatunk ezen példa kapcsán a valószínűségszámítás, a matematikai statisztika egy alapvető kérdésére, a kísérletek számának (golyók számának) a jelentőségére. Kapcsolatot teremthetünk a már bemutatott 4.6. feladattal. (Általánosíthatunk — Pascal háromszög.)

Ezután a gép kiírja a képernyőre a „menűt", amiből választhatunk, mit akarunk látni.

1. A relatív gyakoriságok számértékét
2. Az eloszlás rajzát
3. Az eloszlásfüggvény rajzát
4. Újra akarjuk indítani a szimulációt
5. Be akarjuk fejezni

(Természetesen a kísérlet alapján kapott tapasztalati eloszlásról és eloszlásfüggvényről van szó.) Bármelyik megtekintése után újra választhatunk a fenti lehetőségek közül. Egy-egy ábra addig látható, amíg egy billentyűt meg nem érintünk.

Az ékek és golyók rajzolását, valamint a golyók régi helyzetének törlését szubrutin végzi a kezdő koordináták megadása után. A tartályok, a koordinátarendszer és az eloszlás, valamint az eloszlás függvény rajzolását is szubrutinnal oldottuk meg.

Egy kísérlet számítógéppel való megvalósításának több előnyét figyelhetjük meg ennél a példánál is. Nem kell bonyolult eszközöket használni, akárhányszor megismételhetjük, és jól megfigyelhetjük a jelenséget. A kísérlet mindannyiszor jól látható formában jelenik meg, mely azonnal számokká, grafikonokká változtatható.

Hasonlóan készíthetünk programokat az egyszerű pénz- vagy kockadobás modellezésére, vagy az egyenesen, illetve a síkon való bolyongás problémájára.

```
 5 CLS
10 PRINT "A GÉP A GALTON DESZKÁT SZIMULÁLJA."
20 PRINT "A SZINTEK SZÁMA ÉS A GOLYÓK SZÁMA PARAMÉ-"
```

```
 30 PRINT "TER. A SZINTEK SZÁMA MAX. 14 LEHET."
 40 PRINT "A GÉP EZEN KÍVÜL MEGHATÁROZZA, HOGY AZ"
 45 PRINT "EGYES DOBOZOKBA MILYEN RELATÍV GYAKORI
    SÁG-";
 46 PRINT "GAL ESNEK A GOLYÓK, FELRAJZOLJA A GOLYÓK"
 47 PRINT "TAPASZTALATI ELOSZLÁSÁT ÉS ELOSZLÁSFGV-ÉT.";
 50 PRINT "AZ EGYES ÁBRÁK FELRAJZOLÁSA UTÁN A GÉP"
 55 PRINT "MEGÁLL ÉS TETSZŐLEGES BETŰ MEGÉRINTÉSÉ-
 56 PRINT "VEL TOVÁBBINDÍTHATÓ."
 60 PRINT "A PROGRAM IS EGY BETŰ MEGÉRINTÉSÉVEL IN-"
 65 PRINT "DÍTHATÓ".
 70 IF INKEY$ = " " THEN 70
 80 CLS
 90 INPUT "SZINTEK SZÁMA"; M
 91 DIM D(M+1), P(M+1)
100 INPUT "GOLYÓK SZÁMA"; C
105 CLS : W = 0
110 FOR Y = 1 TO M
120 N = J : GOSUB 1900
130 NEXT J
210 FOR I = 1 TO 10+(N-1)*8
220 SET (119-I, 189-I) : SET (132+I, (189-I)
225 SET(120-I, 189-I) : SET(131+I, 189-I)
230 NEXT I
240 REM TARTÁLYOK RAJZA
250 FOR X = 125-(N-1)*8-16 TO 125+(N-1)*8+16 STEP 16
260 FOR Y = 176-(N-1)*8 TO 108-(N-1)*8 STEP -1
270 SET(X, Y) : SET(X+1, Y)
280 NEXT Y
290 NEXT X
291 Y = 107-(N-1)*8
292 FOR X = 125-(N-1)*8-16 TO 125+(N-1)*8+16 STEP 2
293 SET(X, Y)
294 NEXT X
300 REM GOLYÓ INDUL
```

```
310 FOR J = 1 TO G
315 PRINT CHR$(6);
316 PRINT$1, 35, J;
317 PRINT CHR$(32)
320 A = 125 : B = 186 : A1 = 125: B1 = 186 : DJ = 0
330 GOSUB 2000 : FOR K = 1 TO 100 : NEXT K
340 FOR I = 1 TO N
350 V = RND(2)-1 : DJ = DJ+V
360 A = A1 : B = B1 : W = 1 : GOSUB 2000
370 A = A1—8 : A = A+16*V : B = B1-8
380 W = 0: GOSUB 2000 : FOR K = 1 TO 100 : NEXT K
390 A1 = A : B1 = B
400 NEXT I
410 D(DJ+1) = D(DJ+1)+1
420 W = 1 : GOSUB 2000 : W = 0
430 Y = 107-(N-1)*8+D(DJ+1)
431 X1 = 127-(N-1)*8+(DJ-1)*16
432 FOR X = X1 TO X1+13
433 SET(X, Y)
434 NEXT X
450 NEXT J
500 REM RELATÍV GYAK. SZÁMÍTÁS
510 FOR I = 1 TO M+1
520 P(I) = D(I)/G
530 NEXT I
540 IF  INKEY$ = " " THEN 540
550 CLS
560 PRINT "MIT AKAR LÁTNI?"
570 PRINT  "1.  A  RELATÍV  GYAKORISÁGOK  SZÁMÉRTÉKÉT?"
580 PRINT  "2.  AZ ELOSZLÁS RAJZÁT?"
590 PRINT  "3.  AZ ELOSZLÁSFGV. RAJZÁT?"
600 PRINT  "4.  ÚJRA AKARJA INDÍTANI?"
610 PRINT  "5. BE AKARJA FEJEZNI?"
620 PRINT "ÍRJA BE A MEGFELELŐ SZÁMOT!"
630 INPUT K
```

```
640 ON K GOTO 650, 710, 750, 800, 820
650 REM REL. GYAK. KIÍRÁSA
660 CLS : PRINT "DOBOZOK SORSZÁMA";"      REL. GYAK."
670 FOR I = 1 TO M+1
680 PRINT I; TAB(20); P(I)
690 NEXT I
700 IF INKEY$ = "" THEN 700 ELSE 550
710 REM ELOSZLÁS RAJZA
720 CLS : PRINT "AZ ELOSZLÁS" : GOSUB 2200
730 GOSUB 2500
740 IF INKEY$ ="" THEN 740 ELSE 550
750 REM ELOSZLÁSFGV. RAJZA
760 CLS : PRINT "AZ ELOSZLÁSFÜGGVÉNY"
770 GOSUB 2200
780 GOSUB 2600
790 IF INKEY$ = "" THEN 790 ELSE 550
800 REM ÚJRAINDÍTÁS
810 CLEAR : GOTO 90
820 END
1900 REM N-EDIK SZINT RAJZA
1910 B = 178-(N-1)*8
1920 FOR A = 125-(N-1)*8 TO 125+(N-1)*8 STEP 16
1930 GOSUB 2100
1940 NEXT A
1950 RETURN
2000 REM A, B KOORDINÁTÁVAL INDUL
2010 IF W = 0 THEN SET(A, B) : SET(A+1, B) ELSE RESET(A, B) :
     RESET(A+1,B)
2020 FOR Y = B+1 TO B+2
2030 FOR X = A-1 TO A+2
2040 IF W = 0 THEN SET(X, Y) ELSE RESET(X, Y)
2050 NEXT X
2060 NEXT Y
2070 IF W = 0 THEN SET(A, B+3): SET(A+1, B+3) ELSE RESET
     (A, B+3) : RESET (A+1, B+3)
```

```
2080 RETURN
2100 REM ÉK RAJZA
2110 FOR  Y = B  TO B+1
2120 FOR  X = A-2  TO A+3
2130 SET(X, Y)
2140 NEXT X
2150 NEXT Y
2160 Y = B+2
2170 FOR X = A-1 TO A+2
2180 SET(X, Y)
2190 NEXT X
2192 SET(A, B+3) : SET(A+1, B+3)
2195 RETURN
2200 REM KOORD. RNDSZ. RAJZA
2210 FOR  X = 3  TO 255 STEP 2
2220 SET(X, 20)
2230 NEXT X
2240 FOR  Y = 20  TO 180 STEP 2
2250 SET (3, Y)
2260 NEXT Y
2270 PRINT$15, 0, "0";
2280 PRINT$15, 19, "10";
2290 PRINT$15, 29, "15";
2310 FOR  I = 1  TO 15
2320 FOR  Y = 18  TO 22
2330 SET(3+I*12, Y)
2340 NEXT Y
2350 NEXT I
2360 FOR  X = 1  TO  5 : SET(X, 180) : NEXT X
2370 PRINT$1, 1, "1"
2380 RETURN
2500 REM ELOSZLÁS RAJZA
2510 FOR  I = 1  TO M+1
2520 X1 = P(I)*159+20
2530 FOR  Y = 20  TO X1
```

```
2540 SET(3+12*I, Y)
2550 NEXT Y
2560 NEXT I
2570 RETURN
2600 REM  ELOSZLÁSFGV.  RAJZA
2610 Y = 20
2620 FOR  I = 1  TO N+2
2630 Y = Y+P(I-1)*159
2640 FOR  X = 3+(I—1)*12  TO  3+I*12
2650 SET(X, Y)
2660 NEXT X
2670 NEXT I
2680 FOR  X = 3+(I-1)*12  TO 200
2690 SET(X, Y)
2700 NEXT X
2710 RETURN
```

5. LEHETŐSÉGEK MÁS TÁRGYAK TANÍTÁSI-TANULÁSI FOLYAMATÁBAN

Ebben a fejezetben kitekintést teszünk az általános iskolában folyó tantárgyi oktatás számítógépes alkalmazásaira. Kitekintésünk lényegesen szerényebb, mint eddigi fejezeteinkben leírt mondanivalónk, hisz e területeken nem vagyunk már szakemberek, a tárgyak tanítási-tanulási folyamatát az általános iskolai szaktanárok ismerik a legjobban, de egy-egy példával igyekszünk segítséget nyújtani a számítógép alkalmazási lehetőségeire.

Összeállításunkban elsősorban az általános iskolákban folyó kísérletekre, újítások, pályázatok eredményeire, tapasztalataira támaszkodunk, de hivatkozunk hazai és nemzetközi konferenciák előadásaira, elhangzott tapasztalataira és nem utolsó sorban a szakirodalmakban található igen bő ajánlott és kínált lehetőségekre.

Mondandónkat az 1985. májusában Magyarországon megrendezett „Mikroscience '85" nemzetközi konferencia egyik megállapításával indítjuk: minden algoritmikusan megfogalmazható tevékenység alkalmas arra, hogy számítógéppel követhető legyen, ezért a számítógép az oktatásban is alkalmazható, tudatosan alkalmaznunk kell tehát az iskolai tantermekben — a kisiskoláskortól az érettségiig — a tantárgyak tanítási-tanulási folyamatában. Ezeket az eszközöket alkalmassá tehetjük arra, hogy — akár individuális oktatási formában is — új ismeretekhez juttathassuk tanítványainkat, taníthassunk, szemléltethessünk, gyakoroltathassunk velük.

5.1. Természettudományos tárgyak

Az általános iskolában oktatott természettudományos tárgyak — fizika, kémia, biológia, technika, környezetismeret, földrajz — bő lehetőséget szolgáltatnak algoritmizálható anyagrészekre. Ily módon e szaktanárok a lehetőségek széles skálája közül választhatnak. A számításos feladatokon, anyagrészeken kívül a mikro-világ (-fizika, -kémia, -biológia) a szimulációs felhasználásra nyújt igen jó alkalmat.

A szaktanárok figyelmét elsősorban a tárgyuk tanítási lapjaira (Fizika-, Kémia- stb. Tanítása) hívjuk fel. Ezekben a lapokban igen jó ötletek számítógépes megvalósítása található, de hasznosan forgatható a μC Programkatalógus, melynek Oktatási fejezete is sok programot, programcsomagot kínál.

Kiemelve a számítógép adta szimulációs lehetőségeket, mi itt egy három populáció (róka, nyúl, fű) együttéléséből adódó biológiai konfliktust szimuláló programot mutatunk be. Ez az úgynevezett élet-játék program egy olyan összetett rendszer viselkedését írja le, amelyekben az egyes fajok populációi (egy fajhoz tartozó egymással szoros kapcsolatban lévő egyedek csoportja) egymással kölcsönhatásban élnek: a nyulak füvet esznek, a rókák nyulakat stb.

E kölcsönhatást a matematika differenciálegyenletek segítségével írja le.

10×10-es négyzet alakú elrendezésben nyulak, rókák és fű van. A programban az egyes egyedek számát véletlenszerűen határozzuk meg.

A játék lényege: fűből nyúl lesz, nyúlból róka, valamilyen általunk megadott valószínűséggel. Rókából pedig fű lesz (elpusztul), ha nincs nyúl a közelében (a vele szomszédos 4 mező valamelyikében). A gép a vizsgálandó mezőt is véletlenszerűen választja ki. Ezeket a változásokat a képernyőn követhetjük nyomon: A nyúl &, a róka 0, a fű jele pedig,.

```
10 PRINT CHR$(1)
20 PRINT "ADD MEG A VALÓSZÍNŰSÉGEKET!"
30 INPUT "FŰBŐL NYÚL"; E1
40 INPUT "NYÚLBÓL RÓKA"; E2
50 DIM A(11, 11) : CLS : N = 0 : N1 = 0 : F = 0 : F1 = 0
```

```
 60 PRINT CHR$(2)
 70 FOR I = 1 TO 10
 80 FOR J = 1 TO 10
 90 K = RND((3))
100 ON K GOTO 110, 120, 130
110 A(I, J) = 1 : PRINT $ I, 2 * J,"&"; : GOTO 90
120 A(I, J) = 2 : PRINT $ I, 2 * J, "0"; : GOTO 90
130 A(I, J) = 3 : PRINT $ I, 2 * J, "; ";
140 NEXT J
150 NEXT I
160 PRINT CHR$(6)
170 PRINT $ 13, 2, "NYÚL : &"
180 PRINT $ 14, 2, "RÓKA : 0"
190 PRINT $ 15, 2 "FŰ : , ";
200 I = RND(10) : J = RND(10)
210 ON A(I, J) GOTO 220, 260, 310
220 REM * NYÚL *
230 N = N+1 : N2 = INT(N*E2) : IF N1>N2 THEN 250
240 N1 = N1+1 : A(I, J) = 2 : PRINT$ I, 2*J, "0";
250 GOTO 200
260 REM * RÓKA *
270 IF A(I-1, J) = 1 OR A(I+1, J) = 1 THEN 210
280 IF A(I -1,J) = 1 OR A(I +1,J) = 1 THEN 210
290 A(I, J) = 3 : PRINT $ I, 2*J, ", ";
300 GOTO 200
310 REM * FŰ *
320 F = F+1:F2 = INT (F*E1) : IF F1>F2 THEN 340
330 F1 = F1+1 : A(I, J) = 1 : PRINT $ I, 2*J," &";
340 GOTO 200
350 END
```

5.2. Humán tárgyak

Nehezebb, de nem reménytelen a helyzet a nyelvek, a történelem, a rajz és az ének oktatása során. Egy történelemtanár írja: „A történelem mint tárgy — ismeretanyaga jellegéből következően — nem tartja számon módszerei között a számítástechnikát, legtöbb területén nincsenek, nem is lehetnek algoritmizálható részek. Mindez nem zárja ki annak lehetőségét, hogy — ahol a feltételek adottak — az oktatás folyamatában, annak hatékony növelése érdekében megfelelő időben és helyen ne lehetne a számítástechnika elemeit, eszközeit is felhasználni", és nem sokkal később — szintén a Pedagógiai Szemlében — már cikk jelenik meg „Az iskolaszámítógép alkalmazása a történelem érettségire történő felkészítésében" címmel. De utalhatnánk itt is a Tanítás-i lapokra és a már említett μC Programkatalógusra. A ma már szép számban megjelenő számítástechnikai lapok (Számítástechnika, μMagazin, Ötlet stb.) bővelkednek olyan programokban, melyek a számítógép grafikai és szín lehetőségeit (rajz, rajzolás), hangeffektusait (zene, ének) használják ki, és mutatnak jó példát is a rajz és az ének (zene) oktatásban való felhasználásokra.

Mi most itt példaként a helyesírás gyakoroltatására (pontosabban a szótagolásra) mutatunk be egy példát. Megjegyezzük azonban, hogy az alábbi algoritmus más kérdés feltevéssel, a szótár szavainak átírásával, más helyesírási feladat gyakoroltatására is alkalmas. (Lásd μMagazin 1984/4. 6—7. oldal.)

A szótagolás gyakorlásának algoritmusa most a következő:

1. A szótár szavainak és az ellenőrző szavaknak beolvasása,
2. a kérdés feltevése, és a válasz(-ok) értékelése,
3. végül a tanulói teljesítmény értékelése.

A fentiekben leírt algoritmus egy lehetséges programja az alábbi lehet:

```
10 REM SZÓTAGOLÁS
20 REM A SZÓTÁR SZAVAINAK BEOLVASÁSA (SZÁMUK N)
30 INPUT N
40 DIM S$(N), T$(N)
```

```
 50 FOR I = 1 TO N
 60 READ S$(I), T$(I)
 70 NEXT I
 80 REM A SZÓTÁR SZAVAI
 90 DATA "DIÓI", "DI-Ó-I", "LEÁNYAI", "LE-Á-NYA-I", "GALLYAT",
     "GALY-LYAT","..."
100 REM A KÉRDÉS FELTEVÉSE, A VÁLASZ ÉRTÉKELÉSE
110 FOR  K = 1  TO N
120 PRINT "HOGYAN SZÓTAGOLJUK AZT A SZÓT HOGY"; S$(K)
130 INPUT V$
140 IF V$<>T$(K)  THEN 160
150 PRINT "HELYES" : J = J+1 : GOTO 180
160 PRINT "NEM JÓ" : R = R+1
170 PRINT "HELYESEN : "; T$(K)
180 FOR  L = 1  TO 3000 : NEXT L
190 NEXT K
200 REM A TANULÓI TELJESÍTMÉNY
210 PRINT "A JÓ VÁLASZOK SZÁMA: "; J
220 PRINT "A ROSSZ VÁLASZOK SZÁMA : "; R
230 END
```

5.3. Iskolai adminisztráció

A személyi számítógépek elterjedésével az iskolai adminisztráció, statisztikák készítése is korszerűbbé válhat. Iskolai látogatásaink során meggyőződhettünk e munkák széles körű elterjedéséről is. A munkák fokozottabb elterjedését nehezíti, hogy célszerűen csak ott alkalmazható (igényesen), ahol a személyi számítógépek sornyomtatóval is el vannak látva. Az iskolák vezetésének kell törekedni e lehetőség megteremtéséért.

Az osztályfőnökök, szaktanárok a félévi, év végi statisztikák elkészítéséhez felhasználhatják az alapkonfigurációt is, legfeljebb a statisztikákat a képernyőről rögzítik papírra vagy a kívánt statisztikai lapra.

Mi most egy ilyen lehetőséget mutatunk be: tantárgyi osztályzatok statisztikájáról. Bemenő adatok: az osztály jele, a tanulók száma, a

tárgy neve és a tanulói osztályzatok, míg eredményül a számítógép kiírja az osztályzatok megoszlását és a tanulói átlagot (az osztályzatok összeszámlálására egy szubrutint készítettünk).

```
 10 REM TANTÁRGYI OSZTÁLYZAT STATISZTIKA
 20 REM AZ OSZTÁLY JELE
 30 INPUT O$
 40 REM A TANULÓK SZÁMA
 50 INPUT N
 60 REM A TÁRGY NEVE
 70 INPUT T$
 80 REM OSZTÁLYZATOK
 90 GOSUB 1000
100 REM EREDMÉNYEK KIÍRÁSA
110 PRINT O$
120 PRINT T$
130 FOR O = 5 TO 1 STEP -1
140 PRINT O; " : ", O(O);" TANULÓ"
150 NEXT O
160 PRINT "AZ ÁTLAG : ", (5*O(5)+4*O(4)+3*O(3)+2*O
    (3)+2*O(2)+1*O(1))/N
170 END

1000 REM AZ OSZTÁLYZATOK ÖSSZESZÁMLÁLÁSA
1010 DIM O(5)
1020 FOR I = 1 TO 5
1030 O(I) = 0
1040 NEXT I
1050 FOR T = 1 TO N
1060 INPUT O
1070 O(O) = O(O)+1
1080 NEXT T
1090 RETURN
```

FÜGGELÉK

a BASIC szavak jelentése, kiejtése

Alapszó	Jelentése	Kiejtés	Megjegyzés
LET	legyen	let	
INPUT	bevitel	input	
READ	olvasás	rid	
DATA	adatok	déjtő	
RESTORE	helyreállít	risztor	
PRINT	nyomtatás	print	
CHAR	betű, jel	ker	character (keriktör) röv.
REM	megjegyzés, megjegyez	rim	remark (rimák) röv.
STOP	megállás	stop	
END	vég, befejezés	end	
GOTO	menj ...hez	gótu	go to-ból
IF	ha	if	
THEN	akkor	den	
ELSE	különben	elsz	
TO	-ig	tu	
FOR	részére, számára	for	
NEXT	következő	nekszt	
STEP	lépés	sztep	
DO	tesz, elvégez	du	
UNTIL	amíg, ameddig (egy biz. ideig)	ántil	
WHILE	míg, mialatt	vájl	
RETURN	visszatérés	ritörn	
GOSUB	menj... szubrutinhoz	gószub	go subroutine-ból

DIM	dimenzió, méret	dim	dimension röv.
DEFFN	fgv. definiálás	def fn	definition function röv.
SET	tesz, helyez	szet	
RESET	vissza-	riszet	
GRAPHIC	rajzolt	grefik	
DRAW	rajzol	dro	
BOX	doboz	boksz	
CIRCLE	kör	szörkl	
PAINT	fest	péjnt	
MID	középső	midl	middle röv.
RIGHT	jobb	rájt	
LEFT	bal	left	
AUTO	önműködő, automatikus	autö	automatic röv.
BREAK	megszakítás	bréjk	
CLEAR	tisztít	klir	
LOAD	betölt	lód	
TEST	megvizsgál	teszt	
VERIFY	átvizsgál	verifáj	
CLS	képernyő törlés	klirszkrin	clear screen-ből
CONT	folytat	kánt	continue röv.
SAVE	megment	széjv	
DELETE	töröl	dilit	
EDIT	szerkesztés	edit	
LIST	lista, jegyzék	liszt	
NEW	új	nyu	
RE	újra	ri	
RUN	futás, fut	rán	
SYSTEM	rendszer	szisztim	
TRON	nyomkövetés bekapcsolása	trejsz on	trace on-ból
TROFF	nyomkövetés kikapcsolása	trejsz off	trace off-ból

Ajánlott és felhasznált irodalom

[1] *Bartha Árpád:* A játék, a szimuláció, az esettanulmány a pedagógiában, Pedagógiai Szemle, XXXVI. évf. 1. sz. 34—43, 1986.

[2] *Farkas Ottóné:* PRIMO az általános iskolai fizikaórán. Fizika Tanítása XXIV. évf. 6. sz. 170—174, 1985.

[3] *Hámori Miklós:* Tanulás és tanítás számítógéppel, Tankönyvkiadó, Budapest, 1983.

[4] *Kovács Győző:* Még egyszer az informatika oktatásáról, μM 1985. évi 5. sz. 2. o.

[5] *Körösné:* Mikroszámítógépek a tudományok oktatásában. Pedagógiai Szemle XXXV. évf. 12. sz. 1260—1262, 1985.

[6] *Kunstár Jánosné dr.:* Feladatgyűjtemény Módsz. Közl. Könyvtár 6. Szeged, 1977.

[7] *Mágoriné Huhn Ágnes:* Néhány általános iskolai feladat számítógépes megoldásáról, Módsz. Közl. 24. évf. 3. sz. 150—158. o. 1984.

[8] Matematika 5—8 (Általános iskolai tankönyvek) Tankönyvkiadó, Budapest 1978—81.

[9] *Nagy Sándor:* A számítógépes oktatás pedagógiai problematikája (előadás) Budapest, 1984.

[10] *Nagy Sándor—Szűcs Pál:* Mikroszámítógépek az iskolában, Pedagógiai Szemle, XXXV. évf. 7—8. sz. 653—669., 1985.

[11] *Páris György:* A középszintű számítástechnikai képzés feladatai μM 1983. évi 1. sz. 4. o.

[12] *Puskás Albert:* Algoritmus-folyamatábra-program, Módsz. Közl. 24. évf. 1. sz. 29—34. o. 1984.

[13] Számítógépek kézikönyvei (C—16, HT—1080Z, PRIMO)

[14] *Szűcs Barna—Zsáry Piroska:* Az általános iskolai kísérleti számítástechnikai programról. μM 1986. évi 1. sz.

[15] *Szűcs Pál:* Az audiovizuális oktatás hatékonysága, Tankönyvkiadó, Budapest, 1984.

[16] *Ury László:* COMMODORE C—16. LSI, Budapest, 1985.

[17] *Votisky Zsuzsa:* Etűdök személyi számítógépekre. Gondolat, Budapest, 1984.

[18] *Wiedemann László:* Számítástechnika az oktatásban μM 1984. 1. sz. 4—6. o.

[19] *Zátonyi Sándor:* C—16 programok az általános iskolai fizikatanításhoz, Fizika Tanítása XXV. évf. 1. sz. 16—21. o. 1986.

Kiadja a Juhász Gyula Tanárképző Főiskola
A kiadást gondozza: a JGYTF
Szakszervezeti Bizottsága
A kiadásért felelős: Riesz Béla
86-3929 — Szegedi Nyomda
Felelős vezető: Surányi Tibor igazgató
Megjelent: 8,5 A/5 ív terjedelemben 7000 példányban
Készült: monószedéssel, íves magasnyomással az MSZ 560—159
és az MSZ 5602—55 szabvány szerint

Ára: 60,— Ft