KANDÓ KÁLMÁN VILLAMOSIPARI MŰSZAKI FŐISKOLA

A BASIC

programnyelv

Budapest
1971

KANDÓ KÁLMÁN VILLAMOSIPARI MŰSZAKI FŐISKOLA

Számítástechnikai füzetek 3.

A BASIC programnyelv

/Kézirat gyanánt/

Irta: Ivanyos Lajos

A program futás inditása, adatok bevitele ugyancsak felhasználói irógépen, ill. adatszalag használata esetén lyukszalag olvasón keresztül történhet.

Az időosztásos rendszerben dolgozó irógépek kezelői /2-16 irógép tartozhat a kisszámitógépes BASIC rendszerhez/ ugy dolgoznak a számukra kijelölt irógépen, vagy display egység mellett, mintha a többiek másik számitógépet használnának.

A FORTRAN nyelvhez képest alapvető eltérés az, hogy minden utasitássort el kell látni cimkével /sorszámmal/ és az utasitások végrehajtása növekvő számértékü cimkék sorrendjében történik, kivéve a programba beépitett vezérlésátadásokat.

1.1 A BASIC számok

A BASIC nem tesz különbséget az egész és valós tipusu számok között. Az adatként szereplő és programba beépitett számértékek:

- előjelet,
- decimális egész számjegyeket,
- tizedespontot,
- decimális törtjegyeket,
- "E" szimbólumot /amely a 10 hatvány alapot jelzi/,
- legfeljebb háromjegyü decimális egész exponenst

tartalmazhatnak.

A mantissza számjegyeinek száma is korlátozott, általában 10-nél több számjegyet nem tartalmazhat.
A felsorolt összetevők használata nem kötelező, pl. helyes számmegadás módok:

2	+2
1.345	-2.16
1078934	.987654
-.0017	1.16E+3
.075E-15	-6.12E1

Fontos megkötés, hogy az E szimbólum előtt legalább egy számjegynek kell állnia:

pl. 1E-12 helyes, E8 nem helyes szám megadás

A BASIC nyomtató utasitás hatására a számoknak legfeljebb hat decimális jegye kerül kinyomtatásra. A 0.01 és 1.000.000 közötti abszolut értékek nyomtatása mindig decimális formában, az ezen határokon kivül eső számok nyomtatása pedig normál alakban történik.
Az egész rész elején álló /értéktelen/ zérusok, valamint a pozitiv előjel nem kerülnek nyomtatásra. Ugyancsak nem kerülnek nyomtatásra a decimális törtrész utolsó zérusai sem./csak az E formátumban/
Példaként 2 hatványainak nyomtatási formáit adjuk meg:

7.812500E-3 0.015625 0.03125 0.0625 0.125
0.25 0.5 1 2 4 8 16 64 128 256

1.2 BASIC azonositók

A BASIC-ben skaláris változók, egydimenziós és kétdimenziós tömbök /vektorok ill. mátrixok/ láthatók el azonositóval.
A skaláris változók azonositója egyetlen betü, vagy egy betü és azt követő egyetlen decimális számjegy lehet.
Példák skaláris azonositókra:

A, A0, A1,.....A9
B, B0, B1,.....B9
.....
Y, Y0, Y1,.....Y9
Z, Z0, Z1,.....Z9

Mivel nem teszünk különbséget valós és egész tipusu számok ábrázolása között, ilyen jellegü megszoritás az azonositókkal kapcsolatban nincs.

A tömbök azonositója mindig egyetlen betü. A tömböket DIM /dimenzionáló/ utasitásban /lásd.1.15/ kell deklarálni, a maximális indexhatár ill. indexhatárok megadásával.
A tömb elemek felhasználásakor a tömb azonositó után, kerek zárójelbe kell az indexet megadni. Két dimenziós tömbök indexeit vessző választja el. Az indexbe kifejezéseket is lehet irni. Arról, hogy az index aktuális értéke ne legyen negativ, ill. egész szám legyen, a programozónak kell gondoskodnia.

A BASIC indexes változóinak indexe 0 is lehet!
Példák tömb elemek megadására:

A(12) B(5,8) C(I+J,K) D(A(I,J),5)

1.3 BASIC kifejezések

A BASIC kifejezések számokból /konstansokból/ azonositókból, függvényekből, müveleti jelekből és zárójelekből épülnek fel. Példák:

5 2.5E-6 A B5
1.2+A 8E-5-A(7.2*SIN(I(K,L))
5.1+LOG(EXP(A(17))-EXP(-A(17)))

1.4 Müveleti jelek

Összeadás:	+
Kivonás:	-
Szorzás:	*
Osztás:	/
Hatványozás:	↑ vagy **
Zárójelek:	(és)

A kezdő és végzárójelek közé tett kifejezés kiszámitása megtörténik mielőtt a zárójelbe tett kifejezés a további müveletekben felhasználásra kerül.

Zárójelet nem tartalmazó kifejezésekben először a hatványozás, majd a szorzás-osztás, végül az összeadás-kivonás a végrehajtási sorrend.

A szorzás és osztási müveletek sorrendjét, a balról-jobbra irányban, előfordulásuk szerint kell érteni.

Pl.

A/B*C=(A/B)*C, A/B/C=A/(B*C)

Ugyanez vonetkozik az összeadásra és kivonásra is:

A-B+C=(A-B)+C, A-B-C=A-(B+C)

1.5 BASIC függvények

A függvények argumentuma tetszőleges kifejezés lehet.

SIN(X)	az X kifejezés sinusát
COS(X)	az X kifejezés cosinusát számitja, X számértéke radiánban értendő
ART(X)	az X kifejezés arc tangensét számitja
EXP(X)	az e^X értékét számitja
LOG(X)	az X kifejezés természetes logaritmusát számitja
ABS(X)	az X kifejezés abszolut értékét adja
SQR(X)	az X kifejezés négyzetgyökét számitja
INT(X)	az X kifejezés értékénél nem nagyobb, maximális egész számot adja

RND(X)	a megadott X argumentumtól függetlenül 0 és 1 közötti "véletlen" értéket ad.
SGN(X)	értéke 1, ha X pozitiv 0, ha X nulla -1, ha X negativ
TAB(X)	az irógépsor 0 - 71 számozásu karakterpozicióját közül arra az n.-ik pozicióra lépteti az irófejet, melyre n ≡ X /mod 72/

1.6. BASIC cimkék

Bevezetőben már elmondtuk, hogy a BASIC programok minden utasitássorát sorszámmal /cimkével/ kell ellátni, az utasitások végrehajtása a sorszámok növekvő sorrendjében történik.
A BASIC cimkék 1 és 2047 közé eső számok. A cimke előtt, közben és után tetszőleges számu szóköz helyezhető el. Ugyanez vonatkozik az utasitások, adatok tagolására is. A feldolgozó program a szóköz karaktereket figyelmen kivül hagyja /ignorálja/.

RND(X)	a megadott X argumentumtól függetlenül 0 és 1 közötti "véletlen" értéket ad.
SGN(X)	értéke 1, ha X pozitiv 0, ha X nulla -1, ha X negativ
TAB(X)	az irógépsor 0 - 71 számozásu karakterpoziciója közül arra az n.-ik pozicióra lépteti az irófejet, melyre n = X /mod 72/

1.6. BASIC cimkék

Bevezetőben már elmondtuk, hogy a BASIC programok minden utasitássorát sorszámmal /cimkével/ kell ellátni, az utasitások végrehajtása a sorszámok növekvő sorrendjében történik.
A BASIC cimkék 1 és 2047 közé eső számok. A cimke előtt, közben és után tetszőleges számu szóköz helyezhető el. Ugyanez vonatkozik az utasitások, adatok tagolására is. A feldolgozó program a szóköz karaktereket figyelmen kivül hagyja /ignorálja/.

1.7. Adatbevitel

Irógépről a program számára szükséges adatokat az INPUT utasitással kérhetjük be.
Pl.

```
20 INPUT X
```

utasitás végrehajtásakor a program egy kérdőjelet nyomtat, majd vár, hogy az X azonositónak adjunk az irógépről számértéket. /20 a cimke szerepét tölti be/ A számérték leirásának befejezését vessző, vagy kocsi vissza-soremelés jelekkel jelezhetjük.
A program számára szükséges adatokat előre lyukszalagra lyukaszthatjuk a programnak megfelelő sorrendben. Az INPUT utasitással a Teletype szalagolvasójáról automatikusan olvashatók be az aktuális adatok.
Egy INPUT utasitással több adatot is kérhetünk /ekkor is csak egy kérdőjel jelenik meg/.
Pl. A

```
99 INPUT A,B,C
```

utasitás hatására megjelenik egy kérdőjel. Ekkor kell a megfelelő három számértéket egymástól vesszővel elválasztva megadni.

1.8. Nyomtatás

A BASIC nyelv nyomtató utasitása a PRINT. Segitségével számértékeket és szöveget lehet az irógépen kinyomtatni, ill. a nyomtatási formátumot szabályozni.
Nyomtató formátum utasitások:

PRINT /önmagában/, hatása: kocsi vissza-soremelés

PRINT, /vesszővel/, hatása: kocsi mozgatás a következő nyomtatási oszlop kezdetére

A BASIC-ben 5 nyomtatási oszlop van:

az 1. a lap balszélén a 0.-ik karakterpozíciónál,
a 2. a 15.-ik "
a 3. a 30.-ik "
a 4. a 45.-ik "
az 5. a 60.-ik "

kezdődik.

PRINT; /pontosvesszővel/, hatása: a nyomtatófej egy pozicióval lép jobbra.

Az utasitás /PRINT/ és a sort záró formátum jel /blank, vessző, pontosvessző/ közé BASIC szöveg és BASIC kifejezés irható.
A BASIC szöveg idézőjelek /"/ közé irt ASCII karaktersorozat, mint pl.

"EZxxAZx1.-SZAMUxxPELDA"

A szóközök a BASIC szövegben lényeges szerepet töltenek be. /itt x-el jelöltük/ Az idézőjelbe tett karkatersorozat (a szóközöket is beleértve) formátumjelnek megfelelő kocsi mozgás előtt kinyomtatásra kerül.

A PRINT utasitásban kifejezésként legegyszerübb esetben egy változót adunk meg, melynek számértéke kerül kinyomtatásra.

Pl.

PRINT A hatására A változó aktuális értéke nyomtatódik, majd az irófej a következő sor elejére áll.

PRINT A, hatására A kinyomtatása után az irófej a következő nyomtatási oszlop elejére áll.

PRINT A; hatására A kinyomtatása után az irófej egy pozicióval lép jobbra.

A nyomtató utasitásban összetett kifejezés is szerepelhet, ezt a következő program részletben mutatjuk be.

Feladat: Egy szög fokokban megadott értékének bekérésére, radiánban átszámitott értékének, valamint szinuszának kinyomtatására vonatkozó utasitássorozat. Az utasitásokat a végrehajtási sorrendnek megfelelő növekvő sorszámozással láttuk el.

A 250-es utasitás egyaránt nyomtat szöveget és kiszámitott számértéket, ez utobbit közvetlenül a szöveg után.

Az END utasitásról az 1.9 pontban beszélünk.

Közös sorba kerülő nyomtatások PRINT utasitásait össze is lehet vonni. Elválasztó jelként ilyenkor a vessző, vagy pontos vessző a szövegrészek ill. kifejezések közé kerül.

Az előbbi példában összevonhatók, a

12 - 17,

23 - 48,

149 - 180 - 250,

számozásu sorok:

```
10 PRINT
12 PRINT TAB(8); "MINTA PELDA"
23 PRINT "SZOG FOKBAN","RADIANBAN"
60 INPUT S
149 PRINT,S*3.141592/180,"SINUS ERTEK="SIN(S*3.141592/180)
300 END
```

9. BASIC programok befejező utasitásai. /STOP, END/

Ahhoz, hogy a számitógép egy programot végre tudjon hajtani, meg kell jelölnünk a program kezdetét és végét.
A kezdet /első utasitás/ egyértelmüen megadható. BASIC nyelven irott programok mindig a legalacsonyabb sorszámu /cimkéjü/ utasitással kezdődnek.
Egyszerübb programok, amelyekben elágazások nincsenek, általában a legnagyobb sorszámu utasitással fejeződnek be. Ha azonban külön, erre a célra szolgáló utasitással nem jelezzük, hogy több utasitás nem tartozik a programhoz, a számitógép tovább dolgozik és keresi a memóriában a következő utasitást /amit ilyenkor nem talál meg, mert nincs/.

A program végét az END utasitás jelzi, mint előző példánkban mutattuk is.
Minden programban legmagasabb sorszámmal az END utasitásnak kell állnia!

Nem mindig a legmagasabb sorszámu utasitásnak kell utoljára végrehajtásra kerülnie, különösen igy van ez feltételes elágazásokat tartalmazó programokban, melyekben több befejezési pont is lehet. Meg kell tehát különböztetnünk a program fizikai és logikai befejezését, végét.

Az END utasitás a program fizikai végét jelzi, ebből csak egye
len egy van. Logikai befejezés egy programon belül több is leh
ezt a STOP utasitással jelezzük.

1.10. Értékadás

A BASIC nyelv értékadó utasitása a LET, ezt követően azt a szi
bólumot kell irni, melynek értéket akarunk adni, majd egyenlős
jellel azt a kifejezést, melynek értékét a változó felveszi.
Pl.

```
50 LET A'3,5)=(EXP(A(3,5))-EXP(-A(3,5)))/2
```

jelenti, hogy az A(3, 5) tömbelem új értéke legyen a régi érték
kel számitott sinus hyperbolicus függvény értéke.

Az értékadó utasitás használatára példa a következő program is:

```
10 PRINT
15 PRINT TAB(12); "KOR ES GOMB ADATAI-1"
20 LET P=3.141592
25 PRINT "SUGAR CM-BEN";
26 INPUT R
30 LET K=2*R*P
32 LET T=P*R↑2
34 LET F=P*4*R↑2
40 LET V=P*4*R↑3/3
50 PRINT
60 PRINT "KERULET="K; "CM"; TAB(3);
61 PRINT "TERULET="T; "CM↑2"
62 PRINT "FELSZIN="F; "CM↑2"TAB(3);
63 PRINT "TERFOGAT="V; "CM↑3"
70 END
```

a

```
20 IF A<=B THEN 100
```

és

```
20 IF A=<B GOTO 100
```

utasitások egyenértékűek, mindkettő azt jelenti, hogy ha A számértéke nem nagyobb B számértékénél, akkor a 100-as cimkén folytatódik a program, ellenkező esetben pedig a 20-as cimkét nagyság szerint követő cimkén.

Példánk legyen a másodfoku egyenlet megoldása:

```
 10 PRINT
 20 PRINT TAB(12); "MASODFOKU EGYENLET MEGOLDASA"
 30 PRINT "MASODFOKU TAG EGYUTTHATOJA";
 40 INPUT A
 50 PRINT
 60 PRINT "ELSOFOKU TAG EGYUTTHATOJA";
 70 INPUT B
 80 PRINT
 90 PRINT "KONSTANS TAG"
100 INPUT C
110 PRINT
120 IF A<>0 GOTO 200
130 IF B<>0 GOTO 160
140 IF C<>0 GOTO 150
145 PRINT "AZONOSSAG"
148 STOP
150 PRINT "ELLENTMONDAS"
155 STOP
160 PRINT "AZ EGYENLET ELSOFOKU, MEGOLDASA:";-C/B
170 STOP
200 LET D=B↑2-4*A*C
210 IF D>=0 GOTO 400
220 PRINT "KOMPLEX KONJUGALT GYOKOK"
230 PRINT "VALOS RESZ=" -B/(2*A)
240 PRINT "KEPZETES RESZ=", SQR(-D)/2/A, "ES " -SQR(-D)/2/A
390 STOP
400 PRINT "VALOS GYOKOK"
410 PRINT "EGYIK:", (-B+SQR(D))/2/A
420 PRINT "MASIK:", (-B-SQR(D))/2/A
450 END
```

1.12. Feltétel nélküli vezérlés átadás

Sokszor fordul elő, hogy bár a számitás menete valamilyen feltétel szerint elágazik, mégis célszerü visszatérni mindegyik ágból ugyanarra a programrészre, függetlenül a feltételektől. Előfordulhat az is, hogy programunkhoz olyan kiegészitő részt kivánunk csatolni, amely már nem fér el a szomszédos cimkék között megmaradt helyen.
Ilyen esetekben alkalmazhatjuk a GOTO utasitást.
Pl.

```
101  GOTO 1050
```

utasitás azt jelenti, hogy a program nem a 101-et nagyság szerint követő, hanem az 1050-es cimkénél folytatódik,

```
2000  GOTO 110
```

utasitás pedig azt, hogy a 2000-es cimke után a 110-es következik

Példaként nézzünk olyan program részletet, amely 10 üres sort hagy at irógépen. / 10 db PRINT utasitásnál kevesebb helyet foglal!/

```
.....
1000 LET I=1
1010 PRINT
1020 LET I=I+1
1030 IF I<=10 GOTO 1010
1040 GOTO.....
```

1.13. Emlékeztető és magyarázó szövegek beépitése

Bármilyen egyértelmü is a jól megirt program, mégis azok számára akik először, vagy hosszú szünet után vesznek kezükbe nem is tulságosan bonyolult programot,fejtörést okozhat, hogy egy-egy utasitásnak, program részletnek milyen szerepe lehet.
Ezért a szimbólikus programnyelvek biztositani szokták emlékeztető megjegyzések /REMARK, COMMENT/ beépitési lehetőségét úgy, hogy ezek a program tényleges futását ne zavarják.

A BASIC nyelv ilyen utasitása a REM, ez a nem végrehajtásra szolgáló utasitások közé tartozik, a REM betükkel kezdődő utasitássort a feldolgozó program nem vizsgálja tovább.

Előbbi példánkat ilyen megjegyzésekkel egészitjük ki:

```
990 REM ITT 10-SZER ISMETELJUNK EGY UTASITAST
991 REM AZ ISMETLESEK SZAMAT EGY
993 REM VALTOZO FELTETEL VIZSGALATTAL
994 REM ELLENORIZZUK
995 REM AZ ILYEN PROGAMRESZT CIKLUSNAK
996 REM NEVEZZUK
1000 LET I=1
1001 REM I A CIKLUSVALTOZO, ENNEK KEZDO
1002 REM ERTEKET ADUNK
1010 PRINT
1011 REM EZ VOLT AZ ISMETLENDO UTASITAS
1012 REM AZ ISMETLENDO RESZ A CIKLUSMAG
1020 LET I=I+1
1021 REM ITT LEPTETJUK A CIKLUSVALTOZOT
1030 IF I<=10 GOTO 1010
1031 REM A RELACIO TARTALMAZZA A CIKLUS
1032 REM VALTOZO VEGERTEKET
1040 GOTO....
1041 REM ITT KILEPUNK A CIKLUSBOL
.............
```

1.14. A BASIC ciklus utasitása

A BASIC nyelvben a ciklusmagot két utasitással határolhatjuk. Elé a FOR utasitást, végére a NEXT utasitást tesszük.
A FOR utasitásban kell megadni a ciklusváltozót, annak kezdő és végértékét, valamint a lépésközt.

Pl.

```
100 FOR X1=5 TO 100 STEP 0.1
```

jelenti, hogy a ciklusváltozó X1
a kezdőérték 5
a végérték 100
a lépésköz 0.1

Az X1 ciklusváltozóval kezdett ciklus záró utasitása:

```
.....NEXT X1
```

A záró utasitás hatására végzi el a számitógép a ciklusváltozó léptetését, majd annak ellenőrzését, hogy a ciklusváltozó túllépett-e már a megfelelő FOR utasitásban adott végértéken. Ha igen, akkor a NEXT után következő utasitásra adja a vezérlést, ha nem, akkor újra végrehajtja a ciklusmagot.

Az 1.12 pont példája ciklus utasitással:

```
1000  FOR I=1 TO 10 STEP 1
1010  PRINT
1030  NEXT I
1040  GOTO .........
```

Megjegyezzük, hogy +1 lépésközt nem kell kiirni, igy az 1000 cimkéjü utasitás,

```
1000  FOR I=1 TO 10
```

formában is helyes.

A ciklusváltozó nem csak növekvő, hanem csökkenő értelemben is változhat:

```
1000  FOR I=10 TO 1 STEP-1
.....
```

A ciklusváltozó értékhatárait és a lépésközt változókkal, sőt kifejezésekkel is megadhatjuk, pl.

```
150 FOR K=LOG(N) TO LOG(N1) STEP LOG(Q)
.............
200 NEXT K
```

Jól jegyezzük meg: a ciklusváltozó kezdő és végértékének eltérésétől függetlenül, a ciklusmag egyszer végrehajtásra kerül, csak ezután következik az ellenőrzés. Az ellenőrzés a lépésköz előjelétől függően a nem nagyobb /pozitiv lépésköz/ ill. a nem kisebb /negativ lépésköz/ relációk szerint történik.

Egy programban több ciklus is szerepelhet. Egy-egy cikluson belül is lehetnek további ciklusok. Fontos szabály, hogy ha egy ciklus egy másik cikluson belül kezdődik, akkor azon belül kell befejező NEXT utasitásnak is lennie.

Természetesen feltételes, vagy feltétel nélküli GOTO utasitással szabad a ciklusból kilépni. Másik fontos szabály, hogy cikluson kivülről csak a ciklus kezdő utasitásra szabad vezérlést adni.

A ciklus utasitás alkalmazására példeként készitsünk függvénytáblázatot a trigonometrikus függvényekről, úgy, hogy a kezdő és végértéket, valamint lépésközt a felhasználó irógépen adja meg.

```
 10 PRINT
 11 PRINT
 12 PRINT TAB(8); "FUGGVENYEK TABLAZASA"
 20 PRINT "KEZDO ERTEK=";
 21 INPUT K
 30 PRINT, "VEGERTEK=";
 31 INPUT V
 40 PRINT, "LEPES=";
 41 INPUT L
 42 PRINT
 43 PRINT "ARGUMENTUM 1=X", SIN(X), COS(X), TAN(X)
 45 PRINT
 50 REM ITT KEZDJUK A SZAMOLO CIKLUST:
 60 FOR I=K TO V STEP L
 70 PRINT I, SIN(I), COS(I),
 80 IF COS(I)=0 GOTO 90
 85 PRINT SIN(I)/COS(I);
 90 PRINT
 95 NEXT I
 96 REM VEGE A CILUSNAK
100 PRINT
101 GOTO 20
110 END
```

1.15 Tömbök deklarálása

A tömb elemek /egy, vagy két index-el ellátott változók/ elhelyezése a memóriában egymás után történik. Ezért a tömb elemek számára megfelelő helyet kell biztositani, ezt a feladatot látja el a DIM /dimension/ utasitás.

A DIM utasitás szimbolumot követően kell felsorolni a programban felhasznált tömb azonositókat az előforduló legmagasabb indexekkel, vesszővel elválasztva.

Pl.

DIM A(5), B(6,8), X(2,10)

jelenti, hogy az A jelü tömb egydimenziós, maximálisan 6 elemet tartalmaz /a legalacsonyabb index 0!/; a B jelü tömb kétdimenziós: 7 sort és 9 oszlopot tartalmaz, mig X ugyancsak kétdimenziós 3x11-es méretü tömb, ahol a sorindex 0-tól 2-ig, az oszlopindex 0-tól 10-ig változhat.

Egy programban több DIM utasitás is előfordulhat, de a tömb azonositók mindegyike csak egyetlen DIM utasitásban, ott is csak egyszer.

A DIM utasitásra megkötés, hogy előbb kell szerepelnie, mint a benne deklarált tömb bármely elemének.

A program összes DIM utasitását célszerü a program elejére, a végrehajtásra kerülő utasitások elé tenni!

Több egymást követő DIM utasitás összevonható, csak a tömbnevek előfordulási sorrendje számit a helybiztositásnál:

Pl. a

```
10 DIM A(10),B(2,8)
20 DIM K(2,5),C(100)
30 DIM X(10,20),Y(1)
```

utasitássorozat ekvivalens a

```
30 DIM A(10),B(2,8),K(2,5),C(100),X(10,20)Y(1)
```

utasitással.

Tömb elemek felhasználása akár konstans, akár változó index-el lehetséges.

Nagyobb tömbök kezelését célszerü ciklus utasitással végezni.

Példaként tekintsük egy háromismeretlenes egyenletrendszer megoldását Cramer szabállyal, ahol az egyenletrendszer együtthatóit mátrix elemekként olvassuk be:

```
10 DIM A(2,3),X(2)
11 REM AZ ELSO EGYENLET ELSO EGYUTTHATOJA
12 REM AZ "A(0,0)" ELEM!
20 PRINT "HAROMISMERETLENES EGYENLETRENDSZER"
21 PRINT "AZ EGYUTTHATOK SORFOLYTONOSAN:"
30 FOR I=0 TO 2
40 FOR J=0 TO 3
50 INPUT A(J,I)
60 NEXT J
70 NEXT I
80 LET D1=A(0,0)*A(1,1)*A(2,2)+A(0,1)*A(1,2)*A(2,0)+A(0,2)*A(1,0)*A(2,1)
81 LET D2=A(0,0)*A(1,2)*A(2,1)+A(0,1)*A(1,0)*A(2,2)+A(0,2)*A(1,1) *A(2,0)
90 IF D1=D2 GOTO 200
100 LET D=D1-D2
110LET D1=A(0,3)*A(1,1)*A(2,2)+A(0,1)*A(1,2)*A(2,3)+A(0,2)*A(1,3)*A(2,1)
111LET D2=A(0,3)*A(1,2)*A(2,1)+A(0,1)*A(1,3)*A(2,2)+A(0,2)*A(1,1)*A(2,3)
115 LET X(0)=(D1-D2)/D
120LET D1=A(0,0)*A(1,3)*A(2,2)+A(0,3)*A(1,2)*A(2,0)+A(0,2)*A(1,0)*A(2,3)
125LET D2=A(0,0)*A(1,2)*A(2,3)+A(0,3)*A(1,0)*A(2,2)+A(0,2)*A(1,3)*A(2,0)
129 LET X(1)=(D1-D2)/D
132LET D1=A(0,0)*A(1,1)*A(2,3)+A(0,1)*A(1,3)*A(2,0)+A(0,3)*A(1,0)*A(2,1)
136LET D"=A(0,0)*A(1,3)*A(2,1)+A(0,1)*A(1,0)*A(2,3)+A(0,3)*A(1,1)*A(2,0)
140 LET X(2)=(D1-D2)/D
160 PRINT "MEGOLDAS"
161 FOR K=0 TO 2
162 PRINT "X("K;")="X(K),
165 NEXT K
170 PRINT
180 STOP
200 PRINT "DETERMINANSA ZERUS"
210 END
```

Példaként tekintsük egy háromismeretlenes egyenletrendszer megoldását Cramer szabállyal, ahol az egyenletrendszer együtthatóit mátrix elemekként olvassuk be:

```
10 DIM A(2,3),X(2)
11 REM AZ ELSO EGYENLET ELSO EGYUTTHATOJA
12 REM AZ "A(0,0)" ELEM!
20 PRINT "HAROMISMERETLENES EGYENLETRENDSZER"
21 PRINT "AZ EGYUTTHATOK SORFOLYTONOSAN:"
30 FOR I=0 TO 2
40 FOR J=0 TO 3
50 INPUT A(J,I)
60 NEXT J
70 NEXT I
80 LET D1=A(0,0)*A(1,1)*A(2,2)+A(0,1)*A(1,2)*A(2,0)+A(0,2)*A(1,0)*A(2,1)
81 LET D2=A(0,0)*A(1,2)*A(2,1)+A(0,1)*A(1,0)*A(2,2)+A(0,2)*A(1,1) *A(2,0)
90 IF D1=D2 GOTO 200
100 LET D=D1-D2
110LET D1=A(0,3)*A(1,1)*A(2,2)+A(0,1)*A(1,2)*A(2,3)+A(0,2)*A(1,3)*A(2,1)
111LET D2=A(0,3)*A(1,2)*A(2,1)+A(0,1)*A(1,3)*A(2,2)+A(0,2)*A(1,1)*A(2,3)
115 LET X(0)=(D1-D2)/D
120LET D1=A(0,0)*A(1,3)*A(2,2)+A(0,3)*A(1,2)*A(2,0)+A(0,2)*A(1,0)*A(2,3)
125LET D2=A(0,0)*A(1,2)*A(2,3)+A(0,3)*A(1,0)*A(2,2)+A(0,2)*A(1,3)*A(2,0)
129 LET X(1)=(D1-D2)/D
132LET D1=A(0,0)*A(1,1)*A(2,3)+A(0,1)*A(1,3)*A(2,0)+A(0,3)*A(1,0)*A(2,1)
136LET D"=A(0,0)*A(1,3)*A(2,1)+A(0,1)*A(1,0)*A(2,3)+A(0,3)*A(1,1)*A(2,0)
140 LET X(2)=(D1-D2)/D
160 PRINT "MEGOLDAS"
161 FOR K=0 TO 2
162 PRINT "X("K;")="X(K),
165 NEXT K
170 PRINT
180 STOP
200 PRINT "DETERMINANSA ZERUS"
210 END
```

1.16 A BASIC függvényutasitása

Programban gyakran előforduló, ismételten kiszámitásra kerülő kifejezéseket függvényként definiálhatunk. A felhasználó által definiált függvény nevének három betüből kell állnia, ezekből az első kettő kötött: FN

Igy a következő nevü függvényeket használhatjuk:

FNA FNB FNC ..... FNX FNY FNZ

A függvény definiáló utasitása: DEF /definition/. Ennek a program elején kell állnia. A definiciónak a DEF utasitássorában el kell férnie.

Pl.

```
105 DEF FNK(A0,A1,A2,X)=A0+A1*X+A2*(X↑2)
```

Az igy definiált függvény felhasználásakor a függvénynév utáni zárójelben, a definició formális paraméterei helyett vesszővel elválasztva olyan BASIC kifejezéseket kell megadni, melyeknek értéke határozott.

A formális paraméterek és az argumentumként megadott kifejezések megfeleltetése a leirás sorrendjének megfelelően történik.

Az előbb definiált függvényt pl. a következő utasitásokkal használva:

```
110 LET A=5
115 LET C=9
120 LET Z=10
150 LET X=2*FNK(5,A,C,Z)
```

az X változó értéke:

```
X=2*(5+A*Z+C*(Z↑2)=1910
```

lesz.

Másik példaként irjunk olyan programot, amely kétdimenziós polár-derékszögü koordináta transzformációt végez, irógépen megadott értékekkel:

```
10 DEF FNX(R,S)=R*COS(S)
15 DEF FNY(R,S)=R*SIN(S)
20 PRINT
21 PRINT "ABSZOLUT ERTEK";
22 INPUT A1
25 PRINT
26 PRINT "SZOG RADIANBAN";
30 INPUT B9
35 PRINT "X="FNX(A1,B9),"Y="FNY(A1,B9)
40 END
```

1.17 Adatkatalógus és kezelése

A BASIC programokban lehetőség van állandó adatok listaszerü beépitésére.
Az adatlista jelző utasitása a DATA. Az utasitás után vesszővel elválasztva a listához tartozó értékeket kell felsorolni. Több DATA sor is szerepelhet, ha azonban az adatlista egy sorban elfér, akkor a sorok összevonhatók.
Pl. a

```
101  DATA 5, 6.8, 1E-9
102  DATA 1, -1, 2.5
```

adatlista ekvivalens a

```
101  DATA 5, 6.8, 1E-9, 1, -1, 2.5
```

adatlistával.

Az adatlistát célszerü közvetlenül a program végét jelző END előtt elhelyezni.

Az adatlista elemeit a READ utasitással, vagy utasitásokkal használhatjuk fel a programban, a listának megfelelő sorrendben.

Pl. a

```
10 PRINT
11 PRINT  "ADATLISTA KINYOMTATAS"
15 READ A,B,C
16 PRINT A,B,C
20 READ X,Y
25 PRINT X,Y
30 DATA 5, -1, 2, 18, 1
32 END
```

program futása utáni "eredmény".

```
ADATLISTA KINYOMTATAS
5              -1             2
18             1
```

A programba beépitett RESTORE utasitással az adatmező pointerét /számlálóját/ a kezdő értékre /azaz a lista első elemére/ állithatjuk vissza; pl. ha az előző programba betoldjuk a

```
17 RESTORE
```

utasitást, akkor X az adatlista első, Y pedig második értékét veszi fel, a nyomtatási kép:

```
ADATLISTA KINYOMTATAS
5              -1             2
5              -1
```

Az adatlista felhasználásának két jelentős szerepét emeljük ki:

- a programba egyszerüen lehet katalógus adatokat beépiteni, ugyanazok a változók ismételt hivás esetén más-más értéket vehetnek fel,
- a sok adattal dolgozó programok adatait program módositó lyukszalagról gyorsabban lehet az adatlistába beépiteni, mint az INPUT utasitásoknál, ezen kivül a program futása is meggyorsul az irógépkezelés lassuságának elmaradása miatt.

1.18 BASIC szubrutinok

BASIC programokba zárt szubrutinokat is be lehet épiteni. A szurutinok hivása a szubrutin első utasitásának cimkéje segitségével, a GOSUB utasitással történik.
A szubrutin záró utasitása RETURN, ennek hatására a GOSUB-ot cimkesorrend szerint követő utasitással folytatódik a program.
Egy szubrutinban több RETURN is elhelyezhető.
A BASIC szubrutinok nem önálló szegmensek, mint a FORTRAN-ban, hanem a programba beépitett, a program változóival dolgozó utasitássorozatok.

Példa: Adatmezőben elhelyezett komplex számpárok összeadása, kivonása, szorzása és osztása /a DATA utasitásokban valós rész, képzetes rész, valós rész, képzetes rész ..... legyen a sorrend/. Az első DATA utasitásban a számpárok számát adjuk meg.

```
10 PRINT
15 PRINT TAB(8); "KOMPLEX MUVELETEK SZUBRUTINOKKAL"
20 READ N
25 FOR I=1 TO N
30 READ R1,I1,R2,I2
35 PRINT "R1="R1, "I1="I1, "R2="R2, "I2="I2
40 GOSUB 70
45 GOSUB 80
50 GOSUB 90
55 GOSUB 120
56 PRINT
60 NEXT I
65 STOP
70 PRINT "OSSZEG", "R="R1+R2, "I="I1+I2
75 RETURN
80 PRINT "KULONBSEG", "R="R1-R2, "I="I1-I2
85 RETURN
90 GOSUB 150
95 LET A=A1*A2
100 LET S=S1+S2
105 GOSUB 210
110 PRINT "SZORZAT", "R="R, "I="I
115 RETURN
120 GOSUB 150
125 LET A=A1/A2
130 LET S=S1-S2
135 GOSUB210
140 PRINT "HANYADOS", "R="R, "I="I
145 RETURN
150 LET A1=SQT(R1↑2+I1↑2)
155 LET A2=SQT(R2↑2+I2↑2)
160 IF R1=0 GOTO 185
165 LET S1=ATN(I1/R1)
170 IF R2=0 GOTO 195
175 LET S2=ATN(I2/R2)
180 RETURN
185 LET S1=3.141592/2
190 GOTO 170
195 LET S2=3.141592/2
200 RETURN
210 LET R=A*COS(S)
215 LET I=A*SIN(S)
220 RETURN
230 DATA.....
235 DATA....
.........
300 END
```

1.19 BASIC programok irása és módositása

A programokat számitógéphez csatlakozó /online/ iróberendezésen keresztül közvetlenül a tárba irjuk be, vagy ASCII kódu előkészitő berendezésen lyukszalagra lyukasztjuk, és a lyukszalagot olvastatjuk a számitógéppel.

A számitógép iróberendezésén megjelenő READY szöveg jelzi, hogy a számitógép programot, programszalagot, vagy módositásokat képes fogadni.
Teljes program bevitele előtt a SCRATCH szöveg legépelésével, majd a RETURN billentyü lenyomásával törölhetjük a memória program területét.

Ha a bevitt programnak csak néhány sorát kivánjuk törölni, akkor a DELETE szó után a törlendő sor sorszámát kell leirni, majd a RETURN billentyüt kell lenyomni. Több egymást követő sor törlése esetén a DELETE szó után először az első, majd vesszővel elválasztva az utolsó törlendő sor sorszámát irjuk.
Nem egymást követő sorszámu sorok törlését több DELETE direktivával végezhetjük.

Sortörlést végezhetünk úgy is, hogy a kérdéses sorszámot leirjuk, majd ezt követően a RETURN billentyüt lenyomjuk.

Sor módositást úgy végezhetünk, hogy azonos cimke sorszámmal az új szöveget billentyüzzük.

Billentyüzés közben a hibás leütéseket a ROBOUT billentyü lenyomásával hatástalanithatjuk. Egy-egy leütés törlését a ← jel megjelenése jelzi. Többszöri ROBOUT leütéssel annyi karaktert törlünk visszamenőleg, ahány ← megjelenik.

A memóriában tárolt programokat a LIST direktivával /utána RETURN/ irathatjuk ki, ill. ha egyidejüleg a szalaglyukasztót bekapcsoljuk, rögzithetjük lyukszalagon.

Direktivák törlése az ALTMODE billentyü lenyomásával érhető el. Megjegyezzük még, hogy a felhasznált számitógép tipusától, ill. kiépitettségétől függően egyéb direktivákat is szoktak használni, /pl. program beolvasás gyors szalagolvasóról, háttértárból, program lyukasztása gyors lyukasztón, program elhelyezése háttértárban stb/ ezek részletezésével itt nem foglalkozunk.

1.20 BASIC programok futtatása

Az iró berendezésen megjelenő READY felirat jelzi, hogy a számitógép az előzőleg megadott direktivát /sorszámozás nélküli kezelési utasitást/ végrehajtotta.
A program bevitele /és kijavitása/ után RUN direktivával indithatjuk el a program futását.
Ha ezt követően a fordító program nem tud értelmezni valamit a programban, vagy a futás során rendellenesség jelentkezik /zerussal osztás, kevés adat a DATA utasitásban stb/ az iróberendezésen hibajelzés iródik ki.

Ha a hiba miatt a program futása leáll, akkor a READY felirat is megjelenik, ha nem, akkor a CTRL és C jelü irógépbillentyük egyidejü lenyomásával a programfutás megállitható.
A hibakódok jelentését a számitógépekhez tartozó leirások tartalmazzák.

1.21. BASIC utasitások összefoglalása

Az összehasonlitást az 1.1 táblázat tartalmazza. Itt szimbólikusan ismertetjük a BASIC utasitások szerkezetét, és a felhasználásra vonatkozó esetleges korlátozásokat.
A BASIC vagy FORTRAN nyelven irt programok másik nyelvre átirásához, emlékeztetőként feltüntettük a megfelelő FORTRAN utasitásokat is.

1.1 TÁBLÁZAT

BASIC utasitások

Jelölések:

v	változó
k	kifejezés
n,m	egész szám
s	szám
c	cimke
f	formulák /szöveg, kifejezés, tabuláció/
r	reláció
x	tetszőleges betü

BASIC	FORTRAN megfelelő
LET /értékadás/ LET v = k k értéke kiszámitásra kerül, ezt az értéket v felveszi	v = k
READ /beolvasás az adatmezőből/ READ V1, V2,....vn a felsorolt változók rendre felveszik a DATA utasitásban soronkövetkező szám értékeket	DATA változólistája

BASIC	FORTRAN megfelelő
DATA /adatmező megadás/ DATA s1, s2, s3 .... sn a felsorolt számértékeket READ utasitással lehet változóknak adni	DATA értéklistája
RESTORE /visszaállitás/ RESTORE az adatmező számlálóját az első DATA értékre állitja	DATA értéklista ismétlés
PRINT /nyomtatás/ PRINT f1, f2, f3 .... fn a felsorolt formulák kinyomtatásra kerülnek a kiiró berendezésen	WRITE /n,m/ PRINT /n/
INPUT /adatbevitel/ INPUT v1, v2, v3 .... vn a felsorolt változók értékűket rendre az input billentyüzetről /lyukszalagolvasóról/ kapják	READ /n,m/

BASIC	FORTRAN megfelelő
FOR - NEXT /ciklus utasitás/ FOR v = K1 TO K2 STEP K3 ..... NEXT a FOR és NEXT utasitások közötti programrész ismétlése mig a v ciklusváltozó túl nem lép a K2 határon. /Az egymásba skatulyázható ciklusok száma max. 8 /	DO n I = K1, K2, K3 ..... n CONTINUE
GOTO /vezérlés átadás/ GOTO C következőként a C cimkéjü utasitást kell végrehajtani	GOTO n
IF /feltételes vezérlés átadás/ IF K1 r K2 GOTO c ha a K1 és K2 kifejezések közötti reláció igaz, akkor következőként a c cimkéjü utasitást kell végrehajtani IF K1 r K2 THEN c ekvivalens az előbbivel	IF K1 r K2 GOTO n /logikai IF/

BASIC	FORTRAN megfelelő
REM /emlékeztető/ REM f az f szöveg a program végrehjtása szempontjából közömbös, az utasitás-sor nem kerül végrehajtásra	C /comment/
STOP STOP a program logikai befejezése	STOP
END END a program fizikai befejezése	END
DIM DIM v1,/n1,m1/, v2/n2,m2/.... indexes változók /tömbök/ deklaráló utasitása	DIMENSION v1/n1,m1/..
DEF DEF FNX/v1,v2.../=k, a v1, v2 ... változóktól k kifejezés szerint függő FNX függvény definiciója/	FNX/v1,v2../= K /utasitás függvény/

BASIC	FORTRAN megfelelő
GOSUB - RETURN C1 GOSUB C szubrutin hivás ..... C .... szubrutin C2 RETURN A szubrutin hivással a vezérlés a C címkére adódik, úgy hogy az ezt követő RETURN hatására a C1 címkét követő címkére kerül vissza a vezérlés. Szubrutin hivások 40-es mélységben skatulyázhatók egymásba.	CALL SUBROUTINE.... ..... SUBROUTINE..... .... RETURN

1.2 TÁBLÁZAT

BASIC direktivák

/szerkesztő és vezérlő utasitások/

COMPILE	COM név a BASIC leforditja a felhasználó memóriába beirt programját, és a megadott névvel tárolja a disc-en.
DELETE	DEL n vagy DEL n, m törli az n cimkéjü utasitássort, ill. az n cimkéjü sortól kezdve, az m cimkéjüvel bezárólag az utasitássorokat.
KEY	KEY billentyüzet és lassu lyukszalagolvasó üzemmódjainak visszaállitása.
LIST	LIS a memóriában karakter formában tárolt programról lista kiiratás LIS n az n cimkéjü BASIC utasitássor kiiratása LIS n, m az n cimkéjü sortól kezdve, az m cimkéjüvel bezárólag kiiratás

OLD	OLD disken tárolt program lehivásának előkészitése
REPLACE	REP diskről lehivott program visszahelyezése azonos program névvel
RUN	RUN a karakterformában tárolt program forditásának és futtatásának beinditása
SAVE	SAV a megadott névvel tárolja a disken a felhasználó karakter formájában levő programját
SCRATCH	SCR törli a memóriában levő felhasználói programot
TAPE	TAP átkapcsolás lyukszalagos üzemmódra /a szalagolvasóról beolvasott programról egyidejüleg nem készül lista/
UNSAVE	UNS név, név ... törli a disken tárolt megadott nevü programokat

K.K.V.M.F. Sokszorosító 1971.