KANDÓ KÁLMÁN VILLAMOSIPARI MŰSZAKI FŐISKOLA

Számitástechnikai Tanszék

Számitástechnikai füzetek 6.

EMG 830-10 FOKAL

KANDÓ KÁLMÁN VILLAMOSIPARI MŰSZAKI FŐISKOLA
Számitástechnikai Tanszék

Számitástechnikai füzetek 6.

EMG 830-10 FOKAL

Készitette: Ivanyos Lajos - Nagy András

A FOKAL 830 programnyelv.

A FOKAL /FOrmula KALkulator/ programnyelv a TPA és TPAi magyar gyártmányú kisszámitógépek müszaki - tudományos számitások céljára szolgáló konverzációs nyelve.
Ennek az ugyancsak hazai gyártmányú EMG-830 számitógépre kidolgozott változata a FOKAL 830.
A továbbiakban ennek a számitógépnek a 21 bites, folyamatirányitó változatára 1971-ben kidolgozott gépi reprezentánst ismertetjük.
A FOKAL 830 interpreter programot, valamint az ezt kiszolgáló CSUN nevü vezérlő programot Nagy András, a főiskola Számitástechnikai Tanszékének tudományos munkatársa készitette.

Az interpreter figyelembe veszi a főiskola EMG 830 gépének kevéssé kiépitett voltát:

- lyukszalag bemenet az FS 1500 gyorsolvasón,
- lyukszalag kimenet a FACIT gyorslyukasztón,
- irógépes be és kimenet a vezérlő irógépen keresztül történhet.

A lyukszalag kód ISO 7 szabvány szerinti, ez néhány irásjel kivételével fedi az ASR 33 teletype-ok kódrendszerét, amely kisbetüket nem tartalmaz, szemben az ISO 7 kóddal.

1. FOKAL 830 számok:

Az interpreter nem tesz különbséget az egész és valós tipusú számok között. Adatként és programba beépitve szereplő számok egyaránt tartalmazhatnak:

- előjelet,
- decimális egész számjegyeket,
- tizedespontot,
- decimális törtjegyeket,
- szimbólumot /amely a 10 hatványalapot jelzi/,
- 512-nél kisebb, decimális egész exponenst /512-nél nagyobb exponens megadása sem szintaktikus hiba, de az értelmezés mod. 512 történik/.

A felsorolt összetevők használata nem kötelező, de a szimbólum előtt legalább egy számjegynek kell állnia.

Pl:	1	+3
	2.56	-4.372
	+1675	.976572
	-.056	3.6 &+3
	.0012 &-4	0.12 &5

A nyomtató utasitás hatására az eredmények többféle formában kerülhetnek kiirásra.

A nyomtatási formátum megváltoztatása a TYPE utasitásszó után szereplő % jel szolgál.

Az általános forma:

TYPE⌐%w.d⌐

ahol w az összes számjegyek számát, d pedig a tizedes pont után következő jegyek számát jelenti.

TYPE⌐%w⌐ alakú utasitással normál alakú nyomtatást lehet kérni w jegyre. Ebben az alakban a mantissza 1 és 10 közé eső abszolut értékü, w jegyet tartalmazó, az exponens pedig 3 jegyü szám. A megadott számjegy számának megfelelő kerekitett érték kerül nyomtatásra. A gép 8 decimális számjeggyel számol, ebből általában az első hat tekinthető értékesnek. Hatnál több számjegy nyomtatása esetén az utolsó jegyek bizonytalanok.

TYPE % hatása: áttérés normál alakú nyomtatásra, az előzőleg utoljára megadott összjegy számmal.

Mindkét nyomtatási formánál számolni kell az előjel, a tizedespont ill. ezeket helyettesitő szóköz számára szükséges hellyel, az & formátumnál pedig még a kitevő rész 6 poziciójával.

Ha fixpontos nyomtatásnál túlcsordulás lépne fel, a nyomtatás új sorba, kérdőjel után lebegőpontosan történik.

2. FOKAL 830 azonositók.

A FOKAL nyelv skaláris változók és egydimenziós vektorok /egyindexü változók/ használatát teszi lehetővé.
A skaláris változók azonositója egyetlen betü /kis és nagy betük használata egyaránt megengedett/, vagy egy betü és azt közvetlenül követő egy, vagy két betü vagy, decimális számjegy. A "F" és "f" betük nem szerepelhetnek azonositók első betüjeként. Hosszabb azonositónak csak az első három karakterét veszi figyelembe a gép.

Példák skaláris azonositókra:
A, A0, A1, A2, ... A9, A01, A02, ... A09, ... A99
a, a0, a1, a2, ... a9, a01, a02, ... a09, ... a99
abc, Abc, aBC, AbC, stb
Ugyanazt a változót jelöli pl. az Eotin és Eotal azonositó.
A vektor azonositók hasonló szerkezetüek, az azonositó után kerek zárójelben az index, vagy index kifejezés áll.

Pl: A0(I) Z15(K12(J0(I))) , K08(5)
Az index vagy konkrét szám, vagy kifejezés.
Megjegyezzük, hogy indexelés indexben maximálisan háromszoros mélységben szerepelhet.

Az index kifejezés számértékének meghatározása után /ez nem feltétlenül egész szám/, annak egészre kerekitett értéke tölti be az index szerepét.
Negativ index hibajelzést okoz. A 0 indexü változó az azonos szimbólummal jelölt skaláris változónak felel meg:

Pl: A(0) = A, Z99(0)= Z99.

Az azonositók értékét a SET utasitással állithatjuk be, a TYPE utasitással irhatjuk ki:

3. Müveleti jelek:

Összeadás:	+
Kivonás:	-
Szorzás:	*
Osztás:	/
Hatványozás:	' /felső vessző/
Zárójelek:	/és/

A müveletek sorrendjére vonatkozóan megjegyezzük, hogy a törtvonalat követő szorzás a nevező szorzását eredményezi:

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

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

Hatványozásnál kitevőként összetett kifejezéseket is, lehet használni, ha azok hatványozási müveletet nem tartalmaznak.

4. FOCAL 830 kifejezések:

A kifejezések számokból /konstansokból/, azonositókból, függvényekből, müveleti jelekből és zárójelekből épülnek fel.
Az előbb már emlitett két megkötés:

- indexes változók indexként legfeljebb háromszoros mélységben fordulhatnak elő
- kitevőben hatványozási művelet nem szerepelhet

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, az osztás, végül az összeadás-kivonás a sorrend.
Aritmetikai kifejezéseket lezáró karakterként szerepelhet az egyenlőség jel is. Egyetlen felhasználási területe:
a TYPE? A + B + C = ? ↵ tipusú utasitásokban.
SET és TYPE utasitások felhasználásával kifejezések értékét direkt módon is számolhatjuk.
A sorok elején a * azt jelenti, hogy a gép utasitás fogadására alkalmas állapotban van, az utasitássor végén ↵ leütése jelzi a gépnek, hogy az utasitássort befejeztük.
A F ill. f betüvel kezdődő szimbólumok a FOKAL 830 könyvtári függvényeit jelentik, ezeket az 5. pontban ismertetjük.

5. FOKAL 830 függvények

FSIN /argumentum/ a radiánban értett argumentum sinus-át

FCOS /argumentum/ a radiánban értett argumnetum cosinus-át

FTAN /argumentum/ a radiánban értett argumentum tangens-ét

FATG /argumentum/ az argumentum arcus tangensének főértékét adja radiánban

FEXP /argumentum/ az "e" alapú hatványfüggvényt

FLOG /argumentum/ az "e" alapú logaritmust

FSQT /argumentum/ az argumentum négyzetgyökét

FABS /argumentum/ az argumentum abszolut értékét

FINT /argumentum/ az argumentumnál nem nagyobb szomszédos egészre kerekit

FMOD /arg 1/, /arg 2/ FMOD x, y = x - y* FINT(x/y)

FRAN /argumentum/ értéke 0 és az argumentum közé eső pszeudó-véletlenszám

Megjegyezzük, hogy a függvények nevében a kis és nagy betük egyaránt használhatók.

FSIN(X) = fSIN(X)= fsIN(X)= FsIn(x) stb

6. FOKAL 830 cimkék

A nem direkt végrehajtásra szóló utasitás sorokat /több sorból álló utasitássorozat minden sorát/ cimkével kell ellátni. A cimkével ellátott utasitássorok karakterformában tárolásra kerülnek. A tárolt utasitások végrehajtása csak akkor kezdődik meg, ha erre direkt formában adunk GO, vagy DO utasitást.

A cimkék két részből állnak, a két részt pont választja el.
Mindkét rész egy vagy kétjegyü decimális szám lehet.
Az első részt csoportszámnak, a másodikat sorszámnak nevezzük.

Az	1	és	01	
	2	és	02	
	---	---	---	
	9	és	09	formájú megadás egyenértékü.

A cimke jegyei közé szóközt leütni nem szabad, a cimke előtt és után azonban szabad szóközöket is irni.

A legkisebb értékü cimke 01.01
a lagnagyobb " " 99.99

Az utasitások végrehajtása DO, vagy GO utasitás hatására a memóriában tárolt legalacsonyabb értékü cimkéjü utasitással kezdődik. Ha vezérlésátadó utasitás nem szerepel, akkor a már végrehajtott utasitás cimkéjét követő nagyobb értékü cimkével folytatódik a végrehajtás.

7. Adatbevitel /ASK utasitás/

A számitások során szükséges adatokat vagy beépitjük a programba, azaz SET utasitással a megfelelő azonositónak értéket adunk, vagy a programfutás közben irógépről vagy lyukszalagról kérjük be ASK utasitással.

Az ASK utasitás szót szóköz követi, ezután lista következik. A lista elemek között az értéket kapó változó vagy változók szimbólumán kivül szerepelhetnek

- kinyomtatásra kerülő karakterek
- soremelési jelek,
- visszaléptető jelek,
- vesszó /,/ /hatástalan/
- kérdőjel / lásd a 8. pontnál/

A kinyomtatásra kerülő karaktereket idézőjelek közé kell tenni, ez a "Szöveg" sorrendi elhelyezésének megfelelően, változtatás nélkül nyomtatódik.
Soremelési jel a nem idézőjelek között szerplő felkiáltó jel: !, visszaléptető jel a kettős kereszt:#
Egy-egy ilyen jel hatására egy soremelés /ujsor kezdés/, ill. egy pozicióval visszalépés következik be.

8. Eredmények nyomtatása /TYPE utasitás/

A FOKAL nyelv nyomtató utasitása a TYPE. Segitségével számértékeket és szövegeket lehet az irógépen kinyomtatni ill. a nyomtatási formátumot lehet szabályozni.

TYPE utasitás szót szóköz követi, ezután lista következik. A lista elemek között:

- ha változtani kivánjuk a szám nyomtatás formátumát, első helyen a "%w.d" alakú nyomtatási formátum beállitó jelek sze-

repelnek, az ezt követő vessző után, vagy ha a nyomtatás formátumát nem változtatjuk, a szóközt követően a többi litaelem következhet.

- kérdőjel: ?
- azonositók
- kifejezések
- soremelési jel: !
- visszaléptető jel: #
- idézőjelbe tett karakterek.
- vessző /hatástalan/

A "?" karakter felhasználása a TYPE utasitásra jellemző, ezért itt ismertetjük.
Hatása: amikor a számitógép ?-et talál a programban, az összes további sorrakerülő karaktert kiirja az irógépen mindaddig, amig ujabb kérdőjelet nem talál.
/lásd a 4. pontnál is/
Ügyeljünk azonban arra, hogy a gép minden végrehajtott utasitásnál az utasitásszó után következő első látható karaktert /tehát nem SP., stb./ kétszer vizsgálja meg! Ezzel magyarázható az alábbi néhány utasitás outputja:

```
*SET A=3;
  TYPE? A ? ; TYPE? A+1=?; TYPE, ?A=?; TYPE ?A=? ↵
  AA  3.0000  AA+1=  4.0000   A=  3.0000  3.0000
```

A legutolsó utasitásban a ? volt az emlitett kétszer vizsgált karakter ezért hatástalan maradt.
További felhasználási területe: nyomkövetés, stb.

A felsorolt további jeleket már ismertettük /2 és 7 pont/, a felhasználási lehetőségeket a következő program szemlélteti:

```
*W@
0].0]ASK ,?A B C D?
0].02SET PI=3.]4]593
0].03TYPE !%8.4 ?A=?!?B=?!?C=?!?D=?!
0].04TYPE "KIFEJEZES:" A+(B'2)*C+D/PI
0].05TYPE ######### "_____"!
0].06TYPE "AZ ALAHUZAST A # JELES VISSZALEPTETESSEL A GEP VEGEZTE"

99.99
```

A program futtatása során kapott nyomtatási kép:

```
*GO@
A :].8   B :3  C : -2   D:6.283
A=     ].8000
B=     3.0000
C=    -2.0000
D=     6.2830
KIFEJEZES:  -]4.200]
AZ ALAHUZAST A # JELES VISSZALEPTETESSEL A GEP VEGEZTE
*
```

TYPE utasítással készíthetünk listát a memóriában tárolt azonosítók aktuális értékéről, ennek formátuma:

TYPE $

Az azonosítók egymás alatt, mindig új sorban jelennek meg, a számnyomtatás tizedest tartalmazó mantisszával történik. A 2 pontban mondottaknak megfelelően a 0 értékű indexes azonosító index nélküli azonosítónak felel meg.

```
*TYPE $@

SYMBOLS:

A= 1.800000&+000,
B= 3.000000&+000,
C=-2.000000&+000,
D= 6.283000&+000,
P= 3.141593&+000,
I= 6.000000&+000,
X  (0)=-2.000000&+000,
X   (1)=-1.000000&+000,
X   (2)= 0.000000&-001,
X   (3)= 1.000000&+000,
X   (4)= 2.000000&+000,
X   (5)= 3.000000&+000,
Y = 8.500000&+001,

*
```

9. Program befejező utasitás /QUIT/

Előfordul, hogy a program nem a legmagasabb cimkéjü utasitásnál fejeződik be. Annak megakadályozására, hogy a program befejező utasitása után a gép tovább, most már értelmetlen munkát végezzen, a QUIT utasitást alkalmazzuk. Megjegyezzük, hogy egy sorba több utasitás is irható, és egy cimkézett sor több tipográfiai sorba is széttördelhető. Az utasitásokat pontosvesszővel kell elválasztani. A QUIT utasitás a befejező utasitássor végére is irható.

10. Értékadás /SET utasitás/

A SET utasitást előző példáinkban arra használtuk fel, hogy egy - egy azonositónak konkrét számértéket adjunk. A SET utasitás általános formája:

SET¬azonositó = kifejezés

Az utasitásszót szóköz követi, ezután azt az azonositót kell irni, amelynek értéket kivánunk adni.
Az azonositó után közvetlenül egyenlőségjelet kell irni.
Az egyenlőségjel jobboldalára kifejezés /lásd 4. pont/ irható, ebben az azonositók és müveleti jelek között ill. előttük és utánuk tetszőleges számú szóköz, soremelés, TAB helyezhető el.

```
*W 2@
02.0]TYPE ! "KUP ADATAINAK SZAMITASA"
02.02SET PI=3.]41593
02.03ASK !"ALAPKOR SUGARA" R
02.04ASK !"MAGASSAG       " M
02.05SET A=fSQT(R'2+M'2)
02.06SET AT=PI*R'2
02.07SET PF=R*PI*A
02.08SET V=AT*M/3
02.09TYPE !"ALKOTO  =" A !"FELSZIN =" AT+PF !"TERFOGAT=" V;QUIT

*GO@

KUP ADATAINAK SZAMITASA
ALAPKOR SUGARA:2.8
MAGASSAG       :5.]81
ALKOTO  =  5.889207
FELSZIN = 76.434266
TERFOGAT= 42.536]64
*
```

11. Feltételes vezérlés átadás /IF utasitás/

Sok esetben szükséges valamilyen feltétel /reláció/ teljesülésétől, vagy nem teljesülésétől függően más - más utasitássorozatot végrehajtani. A relációk teljesülésének vizsgálatát és a teljesüléstől függő vezérlésátadást az IF utasitás látja el.

A FOKAL 830 IF utasitása a következő relációk vizsgálatát végzi el:

kifejezés aktuális értéke	negativ
kifejezés aktuális értéke	nulla
kifejezés aktuális értéke	pozitiv

Az IF utasitás formái:

a./ IF-kifejezés, cimke

b./ IF-kifejezés, cimke 1, cimke 2

c./ IF-kifejezés, cimke 1, cimke 2, cimke3

Az a./ esetben ha a leirt kifejezés aktuális értéke negativ, akkor a beirt cimkén, ha nulla, vagy pozitiv, akkor a sorrendben következő utasitással folytatódik a program.

A b./ esetben ha a beirt kifejezés aktuális értéke negativ, akkor az elsőnek irt cimkén; ha nulla, akkor a másodiknak irt cimkén; ha pedig pozitiv akkor a sorrendben következő utasitással folytatódik a program.

A c./ esetben a kifejezés pozitiv aktuális értékénél nem a sorrendben következő utasitással, hanem a harmadik helyre irt cimkén folytatódik a program végrehajtása.

```
*W@
01.04TYPE !!!"AZ EGYENLET ALAKJA:"!!"     2"!"  Ax +Bx+C=0"!!%8.4
01.05ASK  ,?A B C ?!;IF A,2,7

02.05SET P=B/A,q=C/A,D=P'2/4-q;IF D,6,5

04.03SET d=fSQT D,x1=-P/2+d,x2=-P/2-d;TYPE ,?x1= ,  x2= ?!;QUIT

05.02SET x2=-P/2;TYPE "x1=" ?x2=?!;quit

06.10SET d=fSQT(-D);TYPE "x1="-P/2"+"d#"i   x2=" -P/2"-"d#"i"!;QUIT

07.04IF B,7.5,7.12
07.05TYPE "AZ EGYENLET ELSOFOKU, x="-C/B!;QUIT
07.12IF  C,7.13,7.14
07.13TYPE "C="#"/0, NINCS MEGOLDAS."!;quit
07.14TYPE "0=0, AZAZ AZONOSSAG!"!;Q

*GO@


AZ EGYENLET ALAKJA:

    2
  Ax +Bx+C=0

A :1  B :0.1  C :2
x1=   -0.0500 +    1.41331i   x2=   -0.0500 -    1.41331

*GO 1.5@
A :1  B :2  C :1
x1=x2=   -1.0000

*g 1.05@
A :1  B :5  C :6
x1=    -2.0000 ,  x2=    -3.0000

*g 1.5@
A :0  B :64 C :16
AZ EGYENLET ELSOFOKU, x=   -0.2500

*g 1.5@
A :0  B :0  C :15
C≠0, NINCS MEGOLDAS.

*
```

12.Feltétel nélküli vezérlésátadás /GO utasitás/

A GO utasitás segitségével a következő utasitás kombinációk hozhatók létre:

a./ GO /direkt formában, cimke nélkül/ önmagában: beinditja a programértelmezés és végrehajtás menetét a legalacsonyabb utasitással.

b./ GO⌐ csoportszám:

- direkt formában megadott csoport legalacsonyabb sorszámú utasitásával indit:
- indirekt formában /cimkézett utasitás sorban/ a megadott csoport legalacsonyabb sorszámú utasitására adja a vezérlést

c./ GO⌐ csoportszám: sorszám

- direkt formában: amegadott csoport megadott sorszámú utasitásával indit:
- indirekt formában: a megadott cimkére adja a vezérlést.

13. Emlékeztető és magyarázó szövegek beépitése /COMMENT és NOOP/

A FOKAL 830 interpreter a "C" és "c" betüvel kezdődő utasitássorokat nem értelmezi, akár cimke nélküli, akár cimkézett sor kezdődik ezzel. Hasonló célra használható az N betüvel kezdődő utasitás. Az ilyen utasitás /nem a teljes sor!/

hatástalan, kivéve, ha kérdőjel szerepel benne: ekkor a kérdőjelek közti része kigépelődik.
Címkézett sorok a memóriában tárolásra kerülnek, igy a program protokolon kiirathatók. Mivel a C kezdetü sorok nem kerülnek végrehajtásra, ilyenekbe tetszőleges magyarázó megjegyzések helyezhetők el, ezek jól használhatók régebben, vagy mások által irt programok megértéséhez szükséges tájékoztatásra.

Pl: másodfokú egyenletek megoldó program magyarázó szöveggel kiegészitve.

```
*W@
0].04TYPE !!!"AZ EGYENLET ALAKJA:"!!"    2"!"  Ax +Bx+C=0"!!%8.4
0].05ASK ,?A B C ?!;N EGYUTTHATOK BEOLVASASA, AZUTAN VIZSGALAT
A MASODFOKU TAG EGYÜTTHATOJARA. HA 0, AZ E.LET ELFAJULT;
IF A,2,7

02.04C HA A NEM 0, AKKOR ITT FOLYTATODIK. VIZSGALAT A DISZKRIMINANSRA
02.05SET P=B/A,q=C/A,D=P'2/4-q;IF D,6,5

04.02 C KET VALOS GYOK VAN
04.03SET d=fSQT D,x]=-P/2+d,x2=-P/2-d;TYPE ,?x]= ,  x2= ?!;QUIT

05.0] C D=0, A GYOKOK EGYBEESNEK
05.02SET x2=-P/2;TYPE "x]=" ?x2=?!;quit

06.08 c D KISEBB, MINT 0, KET KOMPLEX GYOK VAN
06.]0SET d=fSQT(-D);TYPE "x]="-P/2"+"d#"1   x2=" -P/2"-"d#"1"!;QUIT

07.04IF B,7.5,7.]2
07.05TYPE "AZ EGYENLET ELSOFOKU, x="-C/B!;QUIT
07.]2 IF C,7.]3,7.]4;c CSAK AKKOR JO, HA C IS 0.
07.]3TYPE "C="#"/0, NINCS MEGOLDAS."!;quit
07.]4TYPE "0=0, VALOBAN!"!;Q
```

14. A FOKAL 830 ciklusutasitása /FOR-REPEAT␣FOR/

A ciklusmagot annak terjedelmétől függően többféle módon határolhatjuk.

A ciklusmag előtt minden esetben a FOR utasitás áll, ebben kell megadni a ciklusváltozót, annak kezdőértékét, a lépésközt és végértékét.

Pl: 25.18 FOR X1L = KEZ,LEP,VEG; ...

Jelenti, hogy a ciklusváltó X1L,

a kezdőérték KEZ,
a végérték VEG,
a lépésköz LEP.

} tetszőleges kifejezések

Amennyiben a lépésköz értéke egy, akkor a lépésközt kiirni nem kell:

25.18 FOR X1L = KEZ,VEG; .....

és 25.18 FOR X1L = KEZ,1,VEG; .....

egyenértékűek.

- A ciklusmag a FOR-t követő utasitással kezdődik, és az utasitások végrehajtásában sorrakerülő első ω jelig, vagy R␣F utasitásig tart.
- Ha az áttekinthetőség érdekében nem akarjuk a ciklus magot egy sorba zsúfolni, vagy ez program technikai okokból nem lehetséges, használjuk a DO szubrutinhivó utasitást. /lásd a 16. pontot/

A teljes utasitássor ezzel:

X.Y FOR CLV = KEZ,LEP,VEG;DO cimke @

ahol a cimke annak az utasitásnak a cimkéje, amelyben a ciklusmagot elhelyeztük. Ezt az utasitás sort is @ -tel kell zárni.

- Amennyiben a ciklusmag leirásához több utasitássorra van szükség akkor a "DO" szubrutinhivó utasitás szócska után csoportszámot kell megadni. A ciklusmag utasitásait ebben az esetben a megadott csoport elején kell elhelyezni, a ciklusmaghoz tarozó utolsó utasitás után pedig a REPEAT FOR utasitás szavakat kell irni.

Példa egy soros ciklus utasitás alkalmazására:

```
*ERASE ALL@

*SET I=15@

*FOR K=1,1,I-5;TYPE "-"@
----------
*@
*FOR K=1,10;TYPE "-";REPEAT FOR;TYPE !;FOR K=1,10;
TYPE "+"@
----------
++++++++++
*@
*FOR K=0,4;TYPE !;FOR I=1,8;TYPE "&"@

&&&&&&&&
&&&&&&&&
&&&&&&&&
&&&&&&&&
&&&&&&&&
*
```

Példa a ciklusmag külön sorba történő elhelyezésére:

```
*WRITE ]3@.
]3.28FOR I=],8;type "#";DO ]3.30
]3.29QUIT
]3.30TYPE %3.0 2'I
*
 GO ]3@
#   2 #   4 #   8 #  ]6 #  32 #  64 # ]28 # 256
*@
*c Pelda elagazasos ciklusmagra:@

*W 25@
25.]0FOR I=],29;IF (-fMOD I,7 ), 25.]2;TYPE "*"
25.]]QUIT
25.]2TYPE "#"
25.]3QUIT
*
 GO 25@
######*######*######*######*#
*
```

Példa külön csoportban elhelyezett ciklus magra:

```
*W 5;T !!;W 7@
05.08FOR I=-3*2,2,7;DO 7
05.09Q

07.83TYPE I
*
 go 5@
  -6   -4   -2    0    2    4    6
*

 W 3];T !!;W 35@
3].]0FOR I=],29;do 35
3].]]q

35.04if -fmod I,7    , 35.08
35.05T "*";RETURN
35.08TYPE "#"
*

 g 3]@
######*######*######*######*#
*
```

Jól jegyezzük meg a következőket:

- Ha a ciklusváltozó kezdőértéke nagyobb a végértéknél, akkor a ciklusmag nem kerül végrehajtásra, hanem a vezérlés a következő sorra adódik.
- A lépésköz csak pozitiv lehet
- Ciklus akkor fejeződik be, amikor a ciklusváltozó határozottan meghaladja a végértéket. Ezzel az értékkel a mag már nem kerül végrehajtásra.
- Ciklusok elvileg tetszőleges mélységig skatulyázhatók egymásba azonban ha egy ciklus egy másik ciklusen belül kezdődik, akkor azon belül kell befejeződnie is.
- Ciklusen kivülről csak a ciklus kezdő utasitásra szabad a vezérlést adni. Belső utasitásra ugrás esetén a ciklusmag úgy viselkedik, mintha közönséges program volna. A REPEAT FOR utasitás akkor hatásos, ha éppen ciklusvégrehajtás van folyamatban.

15. Indexes változók használata

A 2 pontban az indexes változók formájáról már beszéltünk, felhasználásukhoz a következőket kell szemelőtt tartani:

- a változók táblázata az azonositók első előfordulása sorrendjében kerül tárolásra.
- az azonos nevü változók indexfolytonosan helyezkednek el,
- az index nélküli azonositó ugyanazt a változót jelöli, mint az azonos nevü zérus indexü azonositó.

- a vektorok részére mindig a legnagyobb indexü elemnek megfelelően történik a helybiztositás. Tehát pl., ha MINT azonositó még nem szerepelt, akkor a SET MINT/250/=3.81 ↵ utasitás hatására az interpreter megnyit egy 251 elemü, MIN azonositójú tömböt. Ennek utolsó eleme 3.81 de a többi határozatlan!

A fentiek alapján belátható, hogy bizonyos, pl

- nem zérus indexü elemek érték beállitásával, vagy
- különböző nevü indexes változók értékbeállitásának váltogatásával kezdő és
- a nem indexfolytonos kezdő érték beállitást alkalmazó programoknál problémák léphetnek fel.

Ezért célszerü a programot a felhasznált indexes változók helyének biztositásával kezdeni.

A helybiztositást az adott azonositóra vonatkozó ciklus utasitással végezhetjük:

```
Pl: 1.01FOR¬I=0,15; SET¬A(I)=0
    1.02FOR¬I=0,30; SET¬B(I)=0
```

Ha az elemek száma a felhasznált adatoktól függ, akkor előbb az adatok számát kérjük be:

```
    1.01 ASK¬"N",N,!
    1.02 FOR¬X=0,N; SET¬A(I)=0
    1.03 ASK¬"M",M,!
    1.04 FOR¬I=0,M; SET  B(I)=0
```

Ha nem követelmény a 0 kezdőérték, akkor elegendő a helybiztosítás:

```
1.01 ASK¬"N",N1"M"!;SET¬A(N)=0,B(N)=0
```

Példa:

```
*W ]@
0].0]c MATRIX TRANSZPONALAS
0].02ASK !?N? "sor," ?M? "oszlop" !;SET x(N*M-])=0;T %2.0
0].03FOR I=],N;T !;FOR J=],M;TYPE
"x("#I#","#J#")";ASK x(J*N-N+I-])
0].04T !!!!!;FOR J=],M;T !;FOR I=],N;TYPE %2.0 "x("#J#","#I#")=" %7
x(J*N-N+I-])
0].05QUIT
*

 GO@

N:3  sor,M:4  oszlop

x( ], ]):]  x( ], 2):2  x( ], 3):3  x( ], 4):4
x( 2, ]):5  x( 2, 2):6  x( 2, 3):7  x( 2, 4):8
x( 3, ]):9  x( 3, 2):]0 x( 3, 3):]] x( 3, 4):]2



x( ], ])=    ].000 x( ], 2)=    5.000 x( ], 3)=    9.000
x( 2, ])=    2.000 x( 2, 2)=    6.000 x( 2, 3)=   ]0.000
x( 3, ])=    3.000 x( 3, 2)=    7.000 x( 3, 3)=   ]].000
x( 4, ])=    4.000 x( 4, 2)=    8.000 x( 4, 3)=   ]2.000
*
```

16. FOKAL szubrutinok /ciklusón kivüli/ DO, RETURN

Az azonos csoportba irt utasitások sorozata / a cimkék csoportszáma azonos/ a program tetszőleges pontjáról a DO utasitásszó segitségével szubrutinszerüen hivható.

Hasonlóan az egysorba irt utasitás, vagy utasitások szubrutin szerü hivása is a DO-val lehetséges

A hivás formája:

DO⌐ csoportszám ↵

vagy DO⌐ cimke ↵

A hivás hatása:

A hivott csoportba tartozó utasitások sorozata, ill. a hivott cimkéjü utasitássor egyszer végrehajtásra kerül, ezt követően a DO utasitást sorrendben követő utasitás végrehajtása következik. A szubrutin kilépési pontja a sor, ill. csoport vége; vagy a sorrakerülő első RETURN utasitás.

Megjegyzés: 1. ezzel a DO utasitás ciklusban történő alkalmazásának szerepe is magyarázható.

2. Ne keverjük össze a RETURN és a REPEAT FOR utasitásokat! Ha a program szubrutinban van, akkor a R⌐F; ha ciklusban a RETURN utasitás hatástalan. Természetesen, a ciklusok és szubrutinok kölcsönösen, gyakorlatilag tetszőleges mélységben egymásba skatulyázhatók.

17. Programok irása, módositása és listázása

A FOKAL 830 változat elsősorban online müszaki számitások gyakorlására ill. feladatmegoldására teszi alkalmassá az EMG 830 számitógépet.

A felhasználó a gépkezelői irógépen keresztül adja utasitásait ill. tölti be programját, ugyanez az eredmények nyomtatásának eszköze is.

Gyakorlatban gépkezelő könnyen tévesst, ezért a hibás leütések javitását egyszerü módon biztositottuk: a ⇐ jelü /visszaléptető/ billentyü lenyomása az irófej visszaléptetésével a kérdéses pozicióra billentyüzött jelet törli a memóriában. Ismételt visszaléptetés visszamenőleg megfelelő számú jelet töröl. A helyesbités igy a hibás szöveg "tetejére" irható. Teljes sor törlése a ¢ jel leütésével érhető el, hatására az irófej a következő sor elejére áll be, és * -gal jelentkezik a FOKAL.

A felhasználónak először a program legalacsonyabb értékü cimkéjét kell gépelnie. A cimke után az utasitássor következik, ennek gépeléskor vigyázni kell a felhasznált utasitás formai szabályaira.

Itt jegyezzük meg, hogy nem szükséges az utasitásszavak teljes kiirása, elegendő azok kezdőbetüjét és azután szóközt billentyüzni. Az utasitásoknál a kis és nagybetük szerepe azonos /vigyázat a változók nevében nem!/

Az utasitássort ↵ jellel kell befejezni.
Ennek hatására az irófej a következő sor elejére áll és kinyomtatja a következő cimkét.
Ha programunkban ez a cimke szerepel, itt irhatjuk a megfelelő utasitássort.
Ha a kinyomtatott cimkét nem kivánjuk felhasználni, akkor a kocsi vissza(⏎)majd ezt követően az ↵ jelet kell bebillentyüznünk.
Az uj sorban most nem automatikus a cimke nyomtatás, a felhasználónak kell a kivánt cimkét billentyüznie. A cimke előtt álló LF, SP, ↵ karakterek hatástalanok.

Utólag észrevett hibák kijavitása a MODIFY direktiva segitségével lehetséges.

MOD⌐cimke ↵

alakú utasitással cimke szerint kiválasztjuk a javitandó sort, majd a _ jel megjelenése után a sort ujra irjuk, vagyis a " _ " karakter után gépelhetünk:

a./ szöveget. Ez bekerül a javitott sorba.
A szöveget az b./, c./, d./ módok valamelyikével zárhatjuk.

b./ _ ↵-t. Ekkor a sor hátralévő teljes tartalma kigépelődik és változatlanul megmarad.

c./ ↵-et. Ekkor a sor azon a helyen lezáródik

d./ _ X ↵ -et, ahol X tetszőleges karakter. /Lehetőleg a hibás rész utolsó karaktere/
Ekkor a sor hátralévő része az első sorrakerülő X ka-

rakterig kigépelődik. Ezután a javitás az a./, b./, c./, d./, módok bermelyikével folytatható.

A hibás részeket a visszaléptető /BS/ billenytüvel törölhetjük, hatása ugyanolyan, mint programbeirásnál. /lásd ott/

A memóriában tárolt utasitásokról a WRITE direktivával kérhetünk listát:

Az utasitás formái:

WRITE ⌐cimke ⌙

hatása az adott cimkéjü utasitássor kinyomtatásra kerül.

WRITE ⌐csoportszám ⌙

hatása az adott csoport összes utasitása kinyomtatásra kerül.

WRITE ⌙

hatása: a memóriában tárolt össze utasitás kinyomtatásra kerül

WRITE ALL ⌙

hatása: az azonositó táblázat beállitott értékei és az összes tárolt utasitás kinyomtatásra kerül.

Törlés:

Az ERASE direktiva segitségével a memória tartalom, vagy annak egyes részei törölhetők:

ERASE␣cimke ↵ ---------- sor törlését

ERASE␣csoportszám↵-------- csoport törlését

ERASE ↵ ---------- a tárolt változók törlését

ERASE␣ALL ↵ ---------- az összes FOKAL információ törlését végzi.

ERASE␣PUFFERS ---------- lásd a 20 pontnál

Megjegyezzük, hogy mielőtt egy-egy program beviteléhez hozzákezdünk, célszerű az ERASE ALL direktivával az előző felhasználók által otthagyott töredékeket és adatokat kitörölni.

18. Programfutás inditása, hibajelzések

A tárolóba bevitt cimkézett utasitássorozat értelmezése és végrehajtása cimke nélküli /direkt/ GO utasitásra kezdődik meg.

Az értelmezés során kiderülő hibákat /szintaktikus hibákat/ az interpreter észleli és hibaüzenetet ad.

A hibaüzenet formája:

ERROR < hibakód > LINE <cimke>

LAST CH's␣ .....

A hibakódokat lásd a 30. oldalon.

<cimke> : a sor, ahol a hibát a gép észlelte

LAST CH' : a hiba észlelése előtti /de legfeljebb 9/ karakter.

Indexes változók bizonyos nem megengedett használatánál, általában a nem határozott kifejezésekből adódó túl nagy, túl kicsi értékek felhasználásánál, 0-val osztásnál

"flp error"

formájú hibaüzenetet kapunk. Ez utóbbi után @jelet billentyüzve az általános hibaüzenet jelenik meg. 0 hibakóddal, 00.00 cimke jelzéssel. /Itt az utolsó karakter - last CH'8: után álló üzenetrész - nem utal közvetlenül a hiba helyére/ Mindig 0 hibakódú üzenetet kapunk akkor, ha a programfutást a 7 és a DO feliratú kezelőgombok egyidejü lenyomásával szakitjuk meg.

Ha flp error után * @ -et gépelünk, a gép megkisérli a számitás folytatását, de az redmény valószinüleg hibás lesz. 70-nél nagyobb szám fEXP-függvénye és negativ szám négyzetgyöke, logaritmusa esetén S.ERROR a hibaüzenet.

A hibaüzenetekből a hiba tipusa az alábbi táblázat és az utasitássorok használatára vonatkozó szabályok alapján, a hiba helye pedig az üzenetben kinyomtatott cimke /sorszám/ és legutóbb vizsgált karakterek alapján megállapitható.

Hibakód táblázat

0 Beavatkozás pultról, külön számmal nem jelölt hibák /flp error, S.ERROR után/

1 Cimben vagy FORMAT-ban több "." /pont/

2 Túl nagy sorszám

3 Túlcsordult az F tároló mező

4 Túl sok index egymásba skatulyázása

5 Sor-, vagy csoportszám 0

6 Index hiba; azonositóban "-" karakter; túlcsordult az R tároló mező /kb. 600-nál több sor beirása esetén/

7 Illegális utasitás

8 Hivatkozás nem létező sorra /ERASE nem váltja ki/

9 Illegális kifejezés

10 Nyitva maradt zárójel

11 Felesleges záró zárójel

12 Illegális format /%/ megadás

13 Hivatkozás nemlétező változóra /ismeretlen azonositó/

14 " /túl nagy index/

15 9-nél többszörösen összetett függvény

16 Nemlétező /hibás/ függvény

17 Hibás jel SET utasitásban

18 Hibás jel IF-ben

19 Hibás jel ASK-ban

20 Illegális numerikus input /ASK-ban/

21 Hibás jel FOR-ban

22 Hibás periféria hivás

19. Programok rögzitése lyukszalagon, lyukszalagon rögzitett programok bevitele /KEEP és HELP direktivák/

KEEP ↵

direktivával a tárolt /cimkézett/ FOKAL 830 utasitásokból álló program gyorslyukasztón keresztül lyukszalagon kiperforálódik ISO 7 kódban.

Használata: KEEP cimke ↵
KEEP csoportszám ↵ } ugyanúgy, mint WRITE-nél
KEEP ↵

HELP ↵

direktiva a KEEP-pel előállitott lyukszalag visszaolvasására szolgál.

A sikeres beolvasást két /új sorba kerülő/ * nyugtázza, ezután a FOKAL inputra vár.

20. A gyorsolvasó és gyorslyukasztó felhasználása programfutás közben input és output céljára:

A FOKAL 830 alapállapotban /beolvasás után/ a consol irógépet használja input és output perifériaként. Ezt az állapotot a következő utasitásokkal lehet megváltoztatni:

PERIPHIRIAL ⌐ INPUT ⌐ TELEX ↵ röviden: P ⌐ I ⌐ T ↵

a gyorsolvasóba helyezett és "rátöltött" telex adatszalag-

ról olvassa az ASK utasitás hatására az input adatokat.

A telex adatszalagon előjel; decimális számjegyek; tizedespont; "ˢ" /felsővessző mint tizhatvány alapjel/, egész kitevő, valamint CR,LF,SP mint terminátor jel fordulhatnak elő. A kocsi vissza és soremelés jelek ignorálódnak. Fontos, hogy két adat között legalább 3 semleges karakter /SP, stb/ legyen.

PERIPHERIAL⌐INPUT⌐ISO◌ röviden P⌐I⌐I◌

a gyorsolvasóba helyezett és rátöltött 8 csatornás ISO kódú adatszalagról olvassa ASK utasitás hatására az adatokat.

Az ISO adatszalagon előjel, decimális számjegyek; tizedespont; & /mint tizhatványalap/ egész kitevő, valamint SP, NL /mint terminátor jel/ fordulhatnak elő.

PERIPHÉRIAL⌐INPUT⌐CONSOL◌, röviden P⌐I⌐C◌

input perifériaként visszaállitja a consol irógépet.

PERIPHÉRIAL⌐OUTPUT⌐ISO◌ , röviden P⌐O⌐I◌

output perifériaként a gyorslyukasztót kapcsolja. A TYPE utasitásokban előirt ill. számértékek ISO 7 kódban kerülnek perforálásra.

PERIPHERIÁL⌐OUTPUT⌐CONSOL◌ , röviden P⌐O⌐C◌

output perifériaként visszaállitja a consol irógépet.

A felsorolt utasitások a periféria pufferek tartalmával operálnak. A puffer felhasználás előtti tartalmát célszerü törölni, igy elkerülhető a hibás adatbevitel és kiirás. A periféria pufferek törlésére szolgál as

ERASE¬PUFFER@, röviden E¬ P @ utasitás

Példa telex input használatra:

```
 *WRITE 3@
03.05f i=1,9;a !x;t  %10.3 x %7 x
*P I TELEX@

*E P@

*DO 3@

:        1.000   1.000000&+000
:      234.150   2.341500&+002
:
? 9.999999940&+066   1.000000&+067
:
?-8.3000000]1&+008 -8.300000&+008
:    10000.010   1.00000]&+004
:-1000000.998 -1.00000]&+006
:     -123.456 -1.234560&+002
:      456.789  4.567890&+002
:      789.012  7.890120&+002
*
```

Ára:1.80.-Ft.