LEM—S

程序语言初级使用说明书

（1973.5 第四版）

中国科学院原子能研究所 201室多道组译

1974·1·

目　　录

Ⅰ. 一般说明
Ⅱ. 变量
Ⅲ. 标示运标
Ⅳ. 常数
Ⅴ. 函数
Ⅵ. 表达式
Ⅶ. 赋值标符
Ⅷ. 可变址变量特殊法则
Ⅸ. 转移和条件
Ⅹ. 输入——输出
　Ⅹ. 1. PUNT
　Ⅹ. 2. RED
　Ⅹ. 3. 输出标识符
　　Ⅹ. 3. 1 @
　　Ⅹ. 3. 2 ↑
　Ⅹ. 4. 输出格式
　Ⅹ. 4. 1. 显式
　　Ⅹ. 4. 2. 隐式
　　Ⅹ. 4. 3. 输出字符串
　　Ⅹ. 4. 4. 字符的格式
　Ⅹ. 5. 输入
　　Ⅹ. 5. 1. 自由格式
　　Ⅹ. 5. 2. 穿孔带输入格式
　　Ⅹ. 5. 3. 字符串的输入
Ⅺ. HLT
Ⅻ. 子程序指令
XIII. 循环指令
XIV. 注释

XV LEM-S 和 SL30 间的关联
XV.1. SRT
XV.2. EXT
XV.3. LNS
XVI 程序编制调整
XVI.1. 输入符号
XVI.2. 即时命令(操作系统)
XVI.2.1. # ZERO
XVI.2.2 # CLEAR
XVI.2.3. # n
XVI.2.4. # GOL
XVI.2.5. # STEP
XVI.2.6. # PERF
XVI.2.7. # LIST
XVI.2.8. # DSPV
XVII 程序的校正
XVII.1. 错误仪息
XVII.1.1. 直接错误
XVII.1.2. 逻辑错误
XVII.1.3. 数学错误
XVII.2. 即时校正
XVII.2.1. ←
XVII.2.2. #
XVIII 程序举例
XVIII.1. CPM
XVIII.2. 阶乘
XVIII.3. 平均
附录I. 错误仪息
附录II. 装入和开工

附录III 操作系统的使用

I —— 一般说明

在写程序和使用计算机之前，请阅读整个说明书。

LEM—S 是直接介释的符号语言

LEM—S 程序由一系列行构成。每行的开始用数字或标号，这些行通过电传打字机或通过光的纸带读数因送入 MULTI—8。

这些行可以按照任何顺序送入 MULTI—8（象在 BASIC 中一样）每行终了时回车，该行将作为 ASCII—8 代码所构成的符号系列存储在内存中，两个 8 位的字节放一个标号，然后每个字节放一个字符

数字数据以浮点格式（尾数和阶码）存贮，每个数据块占4个字节。

符号程序和数据共同使用一内存区，在有 8K 内存的 Multi—8 的配置文本中，用于符号程序的可用内存区为 4096 个字节，介释程序、数学子程序库和编辑程序用另外 4096 个字节。在程序执行过程中，每一行在进入下一行前执行。当一个字符或一组字符读入时，介释程序的一个子程序即加以处理。（这就是所谓的直接介释程序按照标号增加的顺序执行。

例如： 标号 行

标号	行
1	PNT 'A'
3	PNT 'C'
5	PNT 'E'

2 PNT 'B'
4 PNT 'D'

这个程序将按ABCDE的顺序打印。(假定命令PNT意味着打印单引号之间的内容。)称为操作系统的程序的一个部门,分为操作者提供如下的方便:

——符号程序表格的打印输出。
——行的修改和清除。
——变量当前值的打印输出。
——程序的逐步执行。
——从任何指定的行上开始执行
——编辑符号程序穿孔纸带,把它保留起来供以后在任何时间重新装入
印出仅息警告被介释程序查出错误的用户,按一般的法则,并指出有错的一行用它的标号数来表示。

三个专用指令把Multi-8和Multimat1系统中的SL30联接起来,因此能到在线处理从SL30来的数据。

用ASCII代码(8单位)穿在纸带上供使用的数据的简化和处理能到离线操作。

语言的描述和操作系统的使用在本手册的后面给出。

II —— 变量

定义两类变量:

a) A, B, C, T, N, X, Y, Z, W;

这些变量的数值可以由程序或由SL30的计数给出。在后种情况下他们对应于:

N——样品号

T——计数的时间

A, B, C } 道A、B、C中的原始计数。

X, y　在E_1和E_2中外部的标准计数。

Z, W　在E_1和E_2中的取样计数。

句带变址的变量MI和变址I本身都被认为一个变量。

Ⅳ——运算符

算术运算符：给出的是 +, -, *, /，分别标作加减乘除运算实现。

Ⅴ——常数

常数最多为十位，包括符号和小数点。没有符号的常数被认为是正数。如果没有指出小数点，它被认为跟在最后一个数字的后面。

例：
.05
0.05
-.52
5.35
1234567.
-23.4568　是允许的
1234567890　是不允许的

V. ——函数

定义如下函数

FRC —— 平方根
FSI —— 正弦(角度以弧度给出)
FCO —— 余弦(角度以弧度给出)
FLO —— 自然对数
FEX —— 指数
FIN —— 整数部分。

在程序中，函数后面必须跟一个称之为自变量的常数或变量。

带有自变量的函数遵循与变量相同的法则

VI —— 表达式

类似于：X ⊕ X' 的任何组合。

其中X和X'是变量、常数或带自变量的函数。如果⊕是一个标符，则称其为表达式

例：
```
12.3 + 132.8
  A + B
 12 * C
  X / Y
FSIA / FCOA
FLO12.9
```

VII —— 赋值标符

这个标符用 = 表示。与如FORTRAN和BASIC的"相等"符号有全体的含义。就是赋置模的意思。

在LEM-5中，为前面必须是一个变量

这个标符后面可以跟一个变量，一个常数或一个表达式：

A = 17.2	I = 3 + FSI 23
A = B	T = .5 / FCOB
A = 12 + 4	X = C * FINB
B = 15 / B	Y = FL03 + 15
C = C * 3	Z = C * FINB
N = A + B	B = FSIX, ' FCOX

这些是合法的 LEM-S 指令的例子。

注意 B + C = FSIA

A + B = C + D / X

都是不允许的。

VIII ——带变址的变量的特殊规则

带变址的变量可能出现。

a) 用是变址

例如： M8

M124

M42

b) 用形如 MI 的自变址

例如： I = 10

A = MI 相当于 A = M10

IX ——转移和条件

无条件转移写为： JMPN

N 是标号（引数）。和引到这条指令，程序将转移至标号 N 的行上。

例如： 1 PNT 'A'

2 JMP 6

```
3 PNT 'C'
4 PNT 'D'
5 JMP 8
6 PNT 'B'
7 JMP 3
8 ……
```

执行为：

```
1 PNT 'A'
2 PNT 'B'
3 PNT 'C'
4 PNT 'D'
```

b) 条件转移

用？表示，写作：V? n1, n2, n3

在这个表示式中，V是个变量，n1, n2, n3是转号(不同的或相同的)

格令 V? n1, n2, n3.

也作：JMP n1 如果 V<0

JMP n2 如果 V=0

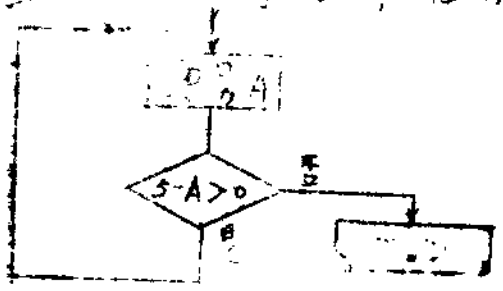
JMP n3 如果 V>0

例如：

```
1 A=A+1
1 B=5-A
3 B? 4, 4, 1
4 PNT 'END'
```

这是一个循环，执行5次，当循环终了时印出的是END，此时A的值是5，B的值为0。

这个画成框图的形式



X 一输入/输出指令

X，1 一打印命令：PNT

这是通世电传打字机打印的命令，它通常写为

PNT $az_1, a'z_2, a''z_3, , ,$

此处 a，a'，a''，可以是数(显格式)或(隐格式)变量。而 z_1，z_2，z_3，可以是变量、标识符或字符串(见下面的定义)。

X，2 一读命令：RED

RED是电传打字机或它的纸带读数机的读入命令。

完整的指令为：

RED $nz_1, n'z_2, n''z_3, \cdots$

此处 n'，n'，n''可以是数(显格式)，变量(隐格式)或符号K(按钮上的自由格式)。z_1，z_2，z_3可以是变量，标识符或字符串(见下面)

在 z_1，z_2、z_3，，，是变量的情况下，n，n'等等可以是1(不用写，表示读带输入)或K(钮盘输入)。

X，3 一输出的标识符

X、3、1 一 @

这个符号表示空格(空白)

例如：

PNT 3@	将打印3个空格	显格式
PNT @	将打印1个空格	
PNTA@		
A=5	将打印5个空格	隐格式

3，2 一 ↑

这个符号表示回車和換行。

PNT↑，@将执行回車和換行

并打印一个空格 }隐格式

A = 3

B = 5

PNT A↑, B @ 将撤销前次的换行，然后打印5个空格 }隐格式

8.4 —输出格式

8.4.1 —显格式

这是一个数，它放在用于输出的变量的指令中来指示输出符号的个数。

例如

假定 A = 123.14

则 PNT 3A

将打印出 1 2 3

而 PNT 5A

将打印出 1 2 3.1

而 PNT 7A

将打印出 1 2 3.1 4 0

必须注意：负号和小数点都算作符号。

例如：A = -12.42

PNT 3A

将印出 -1 2

以及 A = -1 2.4 2

PNT 7A

5 将印出 -1 2.4 2 0

任何与格式不相容的结果在左侧截断，前面印一个(*)号表示有错误。

任何大于 10^{10} 或小于一 $10^{9}-1$ 的数的打印后面跟有E和两位表示指数的数字(浮点计算值)

例：1 2 3 4 5 6 7 8 9 1 E 0 2 =
1 2 3 4 5 6 7 8 9 1 0 0
- 1 2 3 4 5 6 7 8 9 E 0 2 = -
1 2 3 4 5 6 7 8 9 0 0

8、4、2 隐格式

这种格式对应于一个变量，它的在打印时告数于这个变量的整数值。在隐格式中的0被解释为1

例：A = 5
B = 2 3 、 4 5 5
P N T A B

将印出 2 3 、 4 5

例2：
9 B = 3 、 1 4 1 5
10 A = 0
20 A = A + 1
30 P N T 1 , A B
40 J M P 20

将打印如下
3
3 .
3 . 1
3 . 1 4
3 . 1 4 1
3 . 1 4 1 5
3 . 1 4 1 5 0
、 、 、 、 、 、 、 、 、
、 、 、 、 、 、 、

(注：对於在电信打字机上画出谱是有用的)

8.4.3 字符串输出

字符串由包含在单引号之间的一系列字母一数字符拼成。

例如：'THIS IS A MESSAGE'

'YEAR 1971, MESSAGE'

执行 PNT 'STRING' 指令将印出 STRING

例如：PNT "THIS IS A MESSAGE

将打印 THIS IS A MESSAGE

注意：除"QUOTE"(31号)外 X-ON; X-OFF, TAPE, ~~TAPE~~, Bell 之类的任何"控制符号"都可包含在字符串中。这一功能允许在打印输出的同时穿出纸带，以便按用这样产生的纸带作进一步的处理。

8.4.4 字符串允许的特殊格式

每格式中 1 号字符串中唯一允许的格式

例如 PNT 2 'TRY' 是不对的

PNT 'TRY' 是允许的

相反，隐格式(变量)是允许的

例如：
```
10 A = 0
20 A = A + 1
30 PNT # A '*'
40 JMP 20
```
将打印：
```
*
* *
* * *
* * * *
. . . . .
. . . . . .
```

8.5

8.5.1 一自由格式K

这个格式用于通过键盘输入数据，并能输入用空格(空白)表示结束的数据。当执行诸如 R E D K V 的指令时(其中V是一个变量)，电传打字机等待操作者在键盘上打字符的回答。打一个空格(空白)表示数据结束。然后将该数据存储在变量 V 中。

例如：
```
10 REDK A, KB, KC
20 PNT !, 2@, 'A=', 3A, 2@'B=',
   =B, 2@'C=', 3C
```

将有如下的对话发生，最后得到的打印：
(b表示空格)

TELETYPE	OPERATOR TYPES
... 等待 ...	·5b
... wait ...	10.5b
... wait ...	-42b

```
RUN
bbA=.5bbB=10.bbC=-42
```
[这里b表示一个空格(空白)]

8.5.2 穿孔带的输入格式

a) 变量：

如果指令 RED 没有超越格式，计算机将认为数据在穿孔带上，并将启动读数口，当第一个空格(空白)输入时就认为数据已经结束。

例如：REDA, B, C

将命令读出口数到(如果读数启动能的位置

在"STOP"位置)，并且将第一个数值装入变量A中，将第二个数值装入变量B中，第三个装入变量C中。

如果读数机的控制键在"FREE"上，操作者必须将键向前推一下以启动读数。

b)由于带读出机的机械惯性的原因，在穿孔带上的任何数据的前面要有至少三个空格(空白)或没有意义的0

例如：一个包含 | b b b1bbb25bbb328 | 的带此处b表示空格(空白)下面的指令将正确地读入该纸带

REDA，B，C

执行终了时A将等于1，B将等于25，C将等于328。

用同样的方法

| bbb1b00036b0010 |

将被指令REDA，B，C正确地读入，结果形成A=1，B=36，C=10。相反，1b2.35bb32.8不能用REDA，B，C正确地读入且A，B和C的值以不可知的方法隔断。

C)输入时跳过数据

C.1 —输入标识符@

在指令RED中的标识符@，将表示一个数据块(即纸带上两个空格(空白)之间)将被跳过

例如：一条纸带包含有

| bbb128.3bbb-12.bbb134 |

以指令

REDA，@，B 读入

结果形成把值128.3送到A，把值134送到B。
当同样的带被指令 RED@，A，B读入时结果将是把值-12送至A；把值134送至B。
标识符@前面还可有跟格式或星格式。
如带的内容为

bbb145bbb@31bbb17bbb-15

用 REDA，2@，B读入时
结果形成 A=145
B=-15

C12 一输入指令中的标识符↑
读数据带的时候，指令RED中的标识符↑
可跳过n个数据。它将跳过所有数据直到
下一个新的换行被输入为止。