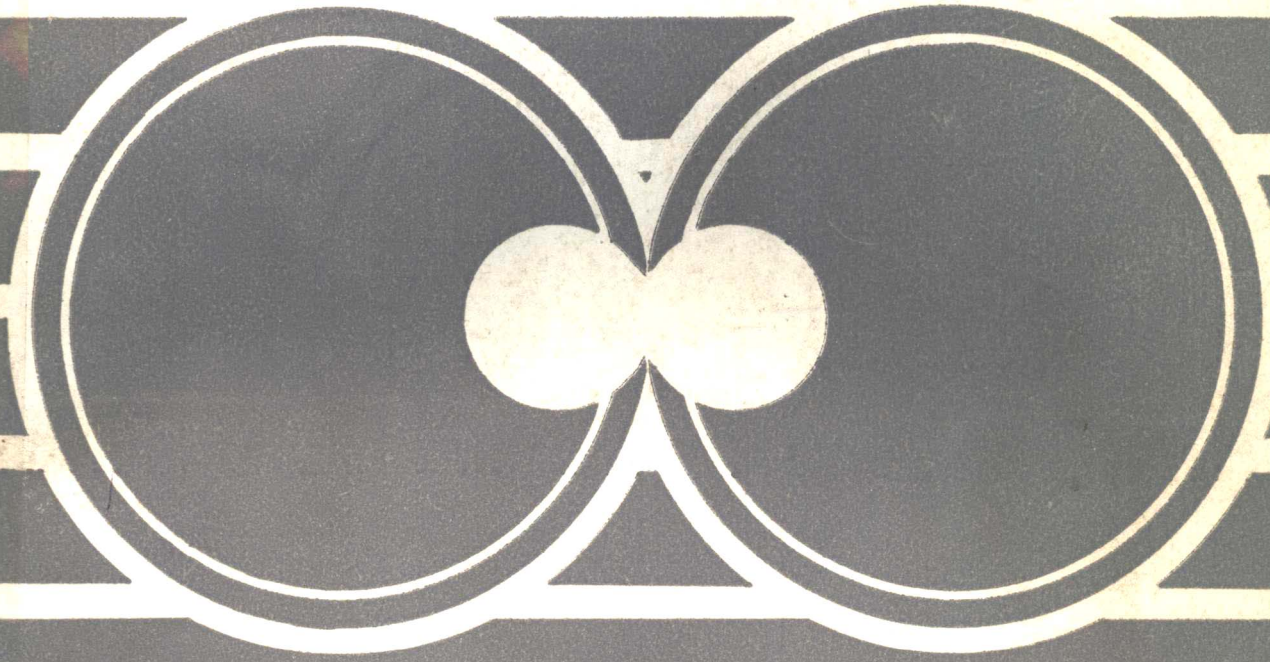
DPS8 BASIC 语言

廖兆衡　黄远利　编



华南工学院出版社

DPS8 BASIC 语言

廖兆衡　黄远利　编

华南工学院出版社

内 容 提 要

本书是编译 BASIC 语言的参考书，包括基本BASIC和高级BASIC。基本 BASIC 包括所有的基本 BASIC 语句，高级 BASIC 包括字符串运算、矩阵运算、数据文件处理和特殊函数等方面的内容。本书除作 DPS 8 BASIC 语言用户手册外，也可供读者学习 BASIC 语言时作参考。

DPS8 BASIC 语言

廖兆衡 黄远利 编

华南工学院出版社出版

（广州 五山）

广东新华书店发行 华南师范大学印刷厂印刷

开本：787×1092 1/16 印张：6.875 字数：170千字

1986年3月第1版 1986年3月第1次印刷

印数：1—10000

书号：15410·004 定价：1.50元

目　录

高级 BASIC 部分

附 录

前　言

从美国霍尼韦尔公司引进的 DPS 8/49大型电子计算机，自投入使用以来，许多用户亟需该机 BASIC 算法语言的参考资料。因此，我们参照该公司《LEVEL 66 TIME SHARING BASIC》一书，从便于国内读者阅读出发，在内容上作了变动和增删，编写成《DPS 8 BASIC 语言》。

在 DPS 8/49 机上使用的是多用户分时 BASIC（可同时供60多个用户用机），它包括所有的基本 BASIC 语句，但高级 BASIC 部分与一般微机使用的扩展 BASIC 有些不同。在字符串运算、矩阵运算、数据文件处理方面，它有较强的功能。由于 DPS 8/49 机内存容量大、运算速度快（平均速度110多万次/秒），采用编译 BASIC，因此，适宜作大、中型的数值计算和事务处理。

全书共十章，前四章属基本 BASIC，其后五章属高级 BASIC，第十章是“上机操作步骤”，此外还有三个附录，提供了“语句一览表”和“出错信息表”等。

本书除可作 DPS 8 BASIC 语言用户的手册外，也可供读者学习基本 BASIC 语言时作参考。

由于编者水平有限，书中错误在所难免，请读者批评指正。

编　者

一九八五年四月

第一章 基本概念

§1.1 概 述

BASIC 这个词，是英语 Beginner's ALL-purpose Symbolic Instruction Code（初学者通用符号指令代码）的缩写。BASIC 语言是面向问题的代数编程语言，它具有简单易懂、使用方便和会话性的特点，既适用于微型计算机，又适用于分时多用户的计算机系统，因此，BASIC 语言得到了广泛的使用。

DPS8/49计算机使用的是分时 BASIC。这种 BASIC 语言与一般的 BASIC 语言一样，具有人机对话的功能。用户只要将编好的程序输入计算机，并键入执行命令，计算机就会按顺序编译和执行。如果程序有错，计算机就会在终端上显示出错信息，指出错误的位置和类型。用户就可以在终端键盘上进行修改。程序修改后，再键入执行命令，计算机就会重新编译和执行程序，如果还有错，就再次修改。这样，直至将程序调试好，最后得到满意的结果为止。

§1.2 基本符号

DPS8 BASIC 语言（以下简称为 BASIC）有一套特定的基本符号，编写程序时，一定要按照 BASIC 语言的规则，用下面的基本符号来编写，不允许使用其他符号。

1．英文大写字母（26个）：A、 B、C…… Z

2．英文小写字母（26个）：a、 b、c …… z

3．数字（10个）：0、1、2、3、4、5、6、7、8、9

4．标点符号（6个）： .（小数点）、 ,（逗号）、 ;（分号）、 ((左园括号）、)（右园括号）、 "（双引号）、 :（冒号）。

5．算术运算符：

+（加）、 -（减）、 *（乘）、 /（除）、 ^或**（乘幂）

6．数关系运算符： <（或LT）（小于） >（或GT）（大于） =（或EQ）（等于） <=（或LE）（小于等于） >=（或GE）（大于等于） ><或<>（或NE）（不等于）

7．字符串关系运算符： <（或LT）（居前） >（或GT） （居后） =（或EQ）（等于） <=（或LE）（前于等于） >=（或GE）（后于等于） ><或<>（或NE）（不等于）

8．输出控制专用符： EOT 4（传输结束符） BELL 7（响铃符） LF 10（跳行符） CR 13（回车符）

9．其他符号：$（字符串类型说明符，货币号）　&（字符串联结号，和号）／（倒斜杠，语句分隔符）　?（问号）　%（百分号）　!（叹号）　[（左方括号）　]（右方括号）　#（#符）　␣（空格符）　'（单引号）

说明：1．小写字母与大写字母在计算机表示的意义是不同的，可以在终端屏幕上输出，但是，在行打印机输出时，一律以大写字母的形式输出。

2．空格符是为了书写方便，在输出时并无此符号，只是代表一个空白位置。

3．在输出控制专用符后面的数字，是该字符的 ASCII 十进制代码。

§1.3 常　量

常量是具有固定值的量。在 BASIC 语言中，常量是指数值常量和字符串常量（串常量在第四章介绍）。

数值常量一律采用十进制形式，其表示方法与日常习惯基本相似但稍有区别。数的书写形式分为定点表示和浮点表示。定点表示只包括数字、小数点和正负号（正号可以任选），如：0.01，2，－3.675，123456789，＋56.78，－．987654321，483.4156 等。浮点数是在定点数基础上增加指数部分E±e，表示 10 的±e次方，e 是一位或两位数，其正号可以省略。例如：0.123456789E－2（表示0.123456789×10^{-2}），12.3456789E－4（表示12.3456789×10^{-4}），19.67E＋2（表示19.67×10^{2}），1.967E3（表示1.967×10^{3}），1E7（表示10^{7}）。应注意不能单独用指数部分表示一个数。例如：用E7表示10000000是错误的，应该用 1E7 表示。

1．程序中使用的数表示范围：绝对值在10^{-38}至10^{+38}之间。若这个数的绝对值大于10^{38}之时，计算机将置一个不确定的值，输出错误信息；若这个数的绝对值小于10^{-38}时，计算机将置一个 0 值，输出错误信息，然后程序继续执行，这样可以一次检测出整个程序运行的出错情况，以方便调试和节省时间。

2．向计算机输入数据时，有效数位可以达到 9 位。不大于134217727时，数可以精确表示；大于此值时，数只能近似表示。因此，超过 8 位数字后，数的精度就不太可靠了。

3．计算机输出数据时，若这个数取绝对值为X，当$10^{-4}\leqslant X<10^{8}$时，用定点表示印出，最多有 7 位有效数字。数位不超过 7 个时，数能精确表示；数位多于 7 个时，第 8 位数四舍五入，数只能近似表示。

当$10^{8}\leqslant X$或$X<10^{-4}$时，用浮点表示印出，最多有 6 位有效数字，在打印时，小数点固定在第一位有效数字的后面。不管定点还是浮点表示印出，数的负号必定印出，而正号不打印，但留一个空格。例如：

表示的数	输出格式
－3.14159	－3.14159
3.1415926	3.141593（第 8 位数字四舍五入）
0.000123456	.0001235（第 8 位数字四舍五入）
1234567.567	1234568（第 8 位数字四舍五入）
9999999	9999999

0.0001	.0001
0.00001	1.00000E－05
10000000	1.00000E␣07
123456789	1.23457E␣08
0.00001234567	1.23457E－05

§1.4 变量、数组、下标变量、串下标变量

§1.4.1 变量 BASIC 中的变量是指在程序执行时其值可以变化的量。**赋给变量**一个指定的值叫做赋值。变量一经赋值，在程序执行期间保持不变，除非变量再一次被赋给新的值。变量有四种形式：简单变量和下标变量，串变量和串下标变量，为了便于识别和引用，就需要分别给各个变量起名字，下面先介绍简单变量和串变量的取名规则：

1．简单变量：一个字母或者一个字母后跟一个数字。例如，A，Z，K6，M1是正确的变量名，而AR，Z12，6K，22则是错误的。

2．串变量：简单变量名后跟一个字符$。

例如：A$，B5$等。

§1.4.2 数组 在数学上，我们使用的向量和矩阵，都是由有一定规则的一组数组成的。例如：一维向量$X=(x_1, x_2, x_3\cdots, x_n)$

nxm 矩阵A $\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & & & \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix}$

在 BASIC 语言中，把按一定规则排列而成的一组数或变量称为数组。数组名的取名规则与变量名相同，在同一程序中，数组名与变量名可以相同。

§1.4.3 下标变量、串下标变量

1．下标变量。组成数组的那些变量，在数组中各自都有对应的位置，每个位置用下标说明。在 BASIC 中，这种带有下标的变量称为下标变量。下标变量由数组名后跟带下标的圆括号组成，其格式如下：格式 〈数组名〉(下标)或 〈数组名〉(下标，下标)

有一个下标的称为一维下标变量，有二个下标的称为二维下标变量。下标可由整数、变量或算术表达式表示，如(5)，(10)，(A)，(B5)，(1+K)或(A(3，7))等。下标的取值范围从1至120000。每个维下标的最大值称为下标上界值。下标上界超过10时，要用 DIM 语句说明，否则，将给出错误信息(DIM 语句将在§2.8节叙述)。

2．串下标变量。串下标变量名由串下标变量名后跟一维的下标说明。如：A$(10)，B1$(5)，C$(A(3，7))等。串没有二维下标变量(其详细叙述请看§4.2节)。

§1.5 表达式、运算规则

表达式由数、变量、运算符、圆括号以及函数组成。语句由行号、关键字、表达式和运算符等组成，并且要在一行内写完，一行可写80个字符。单个的数或变量可看成表达式的特殊情形。

表达式运算结果是一个数，运算次序遵守如下规定：

1．先乘方，后乘除，最后加减。

2．括号内的表达式优先，同一级自左至右计算。

3．乘除、加减分别是同一优先级。

例如：计算A＋（B＊C）＊＊D，则首先计算括号内的B＊C，再计算乘方（B＊C）＊＊D，最后加上A，求得结果。

在实际中，括号的使用能令表达式的运算层次清楚，消除可能出现的错误。

第二章　基本语句

§2.1　语句格式和描述

BASIC 语句前面有一个行号，一般以递增次序排列，行号可以是 1 至10^8-1 中的任一整数，行号后面若为数字，则其间至少留一个空格，行号本身不得插入空格。例如：行号75不能写成 7 ␣ 5 。在高级 BASIC 中，一行可以有多个语句。

BASIC 关键字一般位于行号后面，指出语句的类型或操作功能。在书写语句时，为了清晰起见，可以插入适量的空格。例如：

10␣READ␣A,B

其中10是行号，READ 是 BASIC 关键字，A、B 是变量名。

以后，对每个语句的描述，我们将从功能、格式、举例、规则和说明五个方面来说明。

§2.2　赋值语句，LET

功能：计算表达式的值并赋给指定的变量。

格式：LET〈变量〉=〈表达式〉

举例：1．　10　LET　X=X+1

　　2．　10　LET　X(6)=0

说明：1．=是赋值号，它与代数等式中的等号的意义根本不同，X=X+1 在代数等式中是不成立的，但在 BASIC 中，它是有意义的，即计算X+1 的值，然后将其值赋给变量X。

　　2．BASIC 关键字 LET 可以省略，如：　10　X=X+1

§2.3　输入输出语句　INPUT、PRINT、READ

§2.3.1　键盘输入语句　INPUT

功能：在程序运行时，允许从终端键盘上输入数据。

格式：INPUT〈变量或逗号分隔的多个变量〉

举例：10　INPUT　X, Y, Z

说明：当执行 INPUT 语句时，屏幕上会显示　？号，这时就可输入数据。如：

　　？2，3，4

规则：1．INPUT 语句一般放在前面，使用其输入数据的语句应放在后面。

2．输入的数据要与输入名表一一对应，数据之间用逗号分隔。

§2.3.2 打印语句 PRINT

功能：将输出名表的内容打印出来，也就是在终端屏幕上将输出名表的内容显示出来。

格式：PRINX〈输出名表〉

举例：
```
10 LET X= 5
20 PRINT X, SQR(X)
30 PRINT "THE VALUE OF X IS", X
40 PRINT
```

执行语句20，打印数值5和5的平方根。执行语句30，打印字符串常量和数值5，结果是：THE VALUE OF X IS 5。执行语句40，输出一个空行。

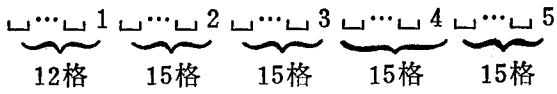
说明：1．输出名表可以是常量（串常量要用双引号括起来）、变量、函数、表达式，它们之间用逗号或分号作分隔符。

2．打印时，可以有下面的格式选择：

（1）逗号，也就是在输出名表中，以逗号作分隔符。BASIC把一行分为5个标准区，输出的数据按五个标准区分配打印，每个区有12个字符位置，区与区之间留3个空格，打印的数据向右对齐。

例1：
```
10 PRINT X, Y, Z, A, B
```

如果X，Y，Z，A，B的值分别为1，2，3，4，5，那么执行该语句时，将在屏幕上显示如下结果：



例2：
```
10 PRINT X, Y,
```

执行该语句时，将把X，Y的值打印在第一和第二个标准区，由于变量Y后面有个逗号，那么将从第三个标准区起打印下一个打印语句输出名表的内容。

（2）分号，也就是在输出名表之间，用分号作分隔符，那么输出的内容将以紧凑格式打印。

如果打印的数据在10^{-4}至$10^{8}-1$之间，定点输出，打印时遵从下面规则：

a．如果打印的数据只有1至4个数位，将分配12个区（每个区6列）打印输出。

b．如果打印的数据只有5至7个数位，将分配8个区（每个区9列）打印输出。

c．如果打印的数据超过7个数位，将分配6个区（每个区12列）打印输出。

如果打印的数据小于10^{-4}或大于$10^{8}-1$时，浮点输出，将分配4个区（每个区16列）打印输出。

（3）可以用特殊函数TAB(X)和SPC(X)来选择打印格式。请参阅§3.3节和§6.9.1节。

§2.3.3 读数语句 READ

功能：从 DATA 数据区读出数据，赋给指定的变量。

格式：READ〈变量或逗号分隔的多个变量〉

举例：
```
10 READ A, B, X, Y, Z
    ⋮
100 DATA 1, 2, 7, 3, -167.921
```

规则：1．READ语句必须位于前面，使用其输入数据的语句应位于后面。

2．READ 语句中的变量必须与 DATA 语句中的数一一对应。如上例，1赋给A，2赋给B，7赋给X，3赋给Y，-167.921赋给Z。

说明：1．READ 语句和 DATA 语句总是一起出现的。在读数时，如果DATA数据区的数据不足，则程序结束，并显示出 OUT OF DATA 的信息。

2．如果一个程序有多个READ读语句或一个READ读语句被重复执行多次，将从DATA数据区未读过的第一个数据开始，依次读出数据，按READ读语句出现的先后次序，给相应的变量赋值。

§2.4 数据语句 DATA 和恢复数据区语句 RESTORE

§2.4.1 DATA 语句

功能：将数据送入数据区，为 READ 读语句提供数据。

格式：DATA〈数据或逗号隔开的多个数据〉

举例：见上节例。

规则：1．DATA 语句使用十进制数，定点或浮点表示均可以。

2．DATA 语句中的数据必须与 READ 语句的变量一一对应。

3．数据可以放在一个或多个 DATA 语句中，上一节的 DATA 语句可写成五个语句。

```
100 DATA 1
110 DATA 2
120 DATA 7
130 DATA 3
140 DATA -167.921
```

也就是说，以上五个DATA语句等价于 100 DATA 1，2，7，3，-167.921。但数据的顺序应保持不变。

说明：1．DATA语句总是和 READ 语句一起出现的。

2．程序中全部的数据语句的数据的集合称为程序的数据区。

3．DATA语句的位置是任意的，一般来说，将全部 DATA 语句集中放在程序的最后较好。

§2.4.2 RESTORE 语句

功能：使数据区恢复到初始状态，READ 读语句可以重新从数据区第一个数据开始读出数据。

格式：RESTORE

举例：
```
100 READ A, B, C
 ⋮
160 RESTORE
170 READ X, Y, Z
 ⋮
200 DATA 1, 3, 5, 7, 9, 11, 13, 15
210 END
```

执行100号语句时，数值1赋给A，3赋给B，5赋给C。执行160号语句时，使数据恢复到起始位置，所以执行170号语句时，数值1赋给X，3赋给Y，5赋给Z。若无160恢复数据区语句时，则数值7赋给X，9赋给Y，11赋给Z。

§2.5 控制转移语句 GOTO、IF … THEN （或GOTO）

§2.5.1 GOTO 语句

功能：无条件转移到语句号指定的语句执行。

格式：GOTO〈语句号〉

举例：
```
50 GOTO 20
```

说明：GOTO 语句用于改变程序的执行顺序，语句号是程序中的一个行号。

§2.5.2 IF…THEN 或 IF…GOTO 语句

功能：有条件地转移到语句号指定的语句执行，否则按顺序执行。

格式：IF〈表达式〉关系运算符〈表达式〉 $\left\{\begin{matrix}\text{THEN}\\ \text{GOTO}\end{matrix}\right\}$ 〈语句号〉

举例：
```
1．10  IF A = 0 THEN 65
       10  IF A = 0 GOTO 65
2．20  IF SIN(X) = M THEN 80
       20  IF SIN(X) = M GOTO 80
```

规则：BASIC提供的六种关系运算符（见§1.2节）均适用于IF语句。

说明：〈表达式〉关系运算符〈表达式〉叫做数关系式，当数关系式成立时，程序就转移到指定的行号去执行，否则顺序执行。

§2.6 循环语句 FOR … TO 和 NEXT

功能：FOR…TO语句是循环程序的开始语句，它规定循环控制变量的初值、终值和步长值。NEXT语句是循环程序的结尾语句，它将判断循环是继续执行还是转去执行NEXT之后的语句。

格式：FOR〈变量〉=〈表达式1〉TO〈表达式2〉STEP〈表达式3〉

⋮

NEXT〈变量〉

〈变量〉是循环控制变量，是数的简单变量。表达式1置变量的初值，表达式2置变量的终值。表达式3的值是步长值，它规定了循环控制变量的增量，若此项缺省，则增量为1。步长为正值时，循环重复至变量值等于或大于终值时为止；步长为负值时，循环重复至变量值等于或小于终值时为止。

例1：
```
30   FOR X4 = 1 TO 25
⋮
80   NEXT X4
```

例2：
```
120  FOR X=(17+COS(Z)/3) TO 3*SQR(10) STEP N*Z
⋮
235  NEXT X
```

例3：
```
240  FOR Z=8 TO 1 STEP -1
⋮
300  NEXT Z
```

例4：以10分为一个分数段，统计一个学生班在各个分数段的人数及全班的平均成绩。L是累计学生的人数，R是累计总分，AVERAGE表示平均成绩，数据－1是用来控制输入数据的结束标志。编出程序如下：

```
10 DIM A(11)
20 READ P\IF P<0 GOTO 60
25 L=L+1\R=R+P
30 H=P/10+1
40 A(H)=A(H)+1
50 GOTO 20
60 FOR I=1 TO 10
70 PRINT 10*(I-1); "--------"; I*10-1; A(I)
80 NEXT I
85 PRINT "          --------   100"; A(11)
90 PRINT "     AVERAGE=", R/L
```

```
95 DATA 51, 72, 68, 100, 87, 67, 75, 99, 65, 87
100 DATA 71, 87, 54, 76, 65, 82, 95, 100, 85, 42
110 DATA 87, 87, 91, 82, 69, 76, 65, 77, 98, 100, -1
120 END
* BRN
```

程序运行后输出如下结果：

```
0————————    9          0
10————————   19         0
20————————   29         0
30————————   39         0
40————————   49         1
50————————   59         2
60————————   69         6
70————————   79         6
80————————   89         8
90————————   99         4
  ————————   100        3
AVERAGE =    78.66667
```

注：倒斜杠＼是本BASIC语言的语句分隔符，请参阅§6.10节。

规则：1．如果步长值超过终值时，循环体仍要执行一次；如果初值大于终值，而步长为正值时，则循环体仅按初值被执行一次。

2．FOR…TO和NEXT必须配对出现，并且要有相同的循环控制变量。

3．NEXT语句的循环控制变量不能省写。

说明：1．循环体是指在FOR语句与NEXT语句之间的程序语句（不包括FOR语句和NEXT语句）。

2．循环语句允许嵌套，即是一个循环语句可以包含一个或者多个循环语句，循环语句的层次要清楚，不能交叉。执行时，从内到外，逐层循环，直至最外面一层循环执行完为止。

例5：

```
FOR   X
  ⋮
  FOR   Y
    ⋮
  NEXT  Y
  ⋮
NEXT  X
```

在这个例子中，外循环X每执行一次，内循环Y要执行完规定的次数。下面是多重循环的例子：

例6：对输入一批数据，由小到大排队。我们采取互换选择法，先找出A数组的最小元素，并与A数组的第一元素互换位置；再找出A数组除第一元素外的最小元素，并与A数组的第二元素互换位