

清

科学资料·

JS-Z1 BASIC 说明书

中山大学 · 贵州大学
中国科学院数学研究所计算站
一九七五年二月

7221
0

目 录

前	言	1
一	引 子	2
二	語法介绍	2
三	键盘操作	13
四	使用方法	15
五	例 题	17
六	附 录	30
附录1	語法公式	30
附录2	标准函数表	34
附录3	键盘操作表	36
附录4	开关表	39
附录5	錯誤性質表	40

前 言

这里的BASIC语言是由中大、贵大和数学所三单位合作，为DJS-21机配的，吉林大学参加了框图阶段的工作。1974年9月调出了第一稿，经过将近半年的试用，几经改进现又调出第二稿，正式交付试用。

BASIC语言比起ALGOL 60，其显著优点是简单易学，便于广大工程技术人员和工农兵群众掌握。它是一种会话型语言，可以人机对话，根据这个特点，我们决定采用解释系统，并配了一些动态调试手段，因此使用起来方便灵活，特别适合并解释一些中、小型题目。其缺点是解释系统速度稍慢，为此采取了一些措施，如换码时实行部份优化，以及把主要的执行语句——赋值语句改为编译型。此外增加了一些通常的BASIC语言中没有的功能，如数组的动态分配。

在研制工作中，得到清华大学电子系林行良同志的热心帮助，为我们讲解了NOVA机BASIC系统的部份实现思想。我们工作中的主要参考资料之一是中国科学院物理研究所编印的“NOVA机BASIC使用说明”。

本系统在DJS-21机上运行时间不久，可能还存在不少问题，请使用的同志把发现的问题及时告诉我们，以便改进。

“DJS-21 BASIC 框图”也即将印刷出来。

“BASIC 研制组”

1975年2月

一、引子

BASIC程序由一系列语句组成，每个语句占一行，每行左端有一个标号，右端以电传打字机的“回车”（记为 $\downarrow$ ）为结尾。语句的标号可以是0至90000的任一整数。在编写程序时，各语句的标号不一定是连续（往往为了进行修改，增添语句的方便，而留有一些间隔），也不要求按大小顺序排列。执行时，解释系统自动按标号大小顺序执行各语句。

例：已知电压 $U=220$ 伏，电阻 $R=1100$ 欧，计算功率 P 等于多少瓦？

计算公式： $P = \frac{U^2}{R}$

对应的DJS-21 BASIC程序为：

```

1 DATA 220; 1100  $\downarrow$  (数据220和1100)。
3 READ U; R  $\downarrow$  (读入数据220和1100分别赋给对应的变量U和R)。
5 LET P = U * U / R  $\downarrow$  (计算  $\frac{U^2}{R} = \frac{220^2}{1100} = 44$ ，赋给变量P)。
7 PRINT P  $\downarrow$  (打印变量P的数值)。
9 END  $\downarrow$  (计算结束)。

```

下面详细介绍DJS-21 BASIC 程序编写方法。

二 语法介绍

DJS-21 BASIC的语法在附录1中给出了严格定义，这里作一些介绍和说明。

I 词法

1 数的表示

①输入时：数可为整数和实数。

整数的范围为： $|I| < 2^{34} - 1 \approx 10^{10}$ ，最多有八位有效数字。例：1432；+57；-188；18432513。

实数的范围为： $10^{-19} < |R| < 10^{19}$ ，最多也只有八位有效数字。例：0.234；+.52₁₀；-3；-25；-201

· 23 ; -13 · 25 10 4 ; 153 · 2 10 - 15 ; 12 · 53 2
67 2 10 + 5 。

② 程序执行时：只进行实数运算。

③ 输出时：若为上述整数范围内的整数，则按整数形式打印，否则，按实数规格化形式打印。整、实数范围同上。

2 变量

变量包括简单变量和下标变量。

① 简单变量（简称为简变）

简单变量的名字用字母 A~Z 中任一字母或一个字母后跟一个数字所组成。例如：A；B8；W；Z3 等可以作为简单变量的名字。而 3C；DE；A56 等则不能作为简单变量的名字。

② 下标变量

只允许一维和二维数组。数组的名字用一个字母表示。用下标变量表示数组元素，其名字由数组名字和括在圆括号中的下标组成，下标规定一律由零开始计数。一维数组的元素只有一个下标。二维数组的元素有两个下标，两个下标之间用逗号分开，第一个下标表示行数，第二个下标表示列数，在内存中元素按行排列。

例如：A(5) 表示一维数组 A 的第 6 个元素，B(0, 2) 表示二维数组 B 的第一行第 3 列的元素。

3 函数

函数包括标准函数和自定义函数。

① 标准函数

在程序中可以直接使用十八种标准函数。

例如：

SIN(X) —— 求 X 的正弦函数。

$TAN(X)$ — 求 X 的正切函数。

$ASN(X)$ — 求 X 的反正弦函数。

$LOG(X)$ — 求 X 的以10为底的对数。

$LON(X)$ — 求 X 的自然对数。

$SQR(X)$ — 求 X 的平方根。

$INT(X)$ — 求与 X 最接近的整数(四舍五入)。

$MAX(X_1, X_2, \dots, X_5)$ — 取 $X_1, X_2, \dots, X_5$ 中最大的数。(最多到 X_5)

使用时,在用圆括号括起来的自变量 $X, X_1, X_2, \dots, X_5$ 的位置上,可为算术表达式(算术表达式下节介绍)。例如: $SIN(5); COS(P); TAN(2 * X + 4 * Y - 6.1); LOG(X * + 2)$ 等。

三角函数的自变量都是以弧度为单位。

显然,对标准函数的自变量的数值要有一些必要的限制: LOG, LON 的自变量必须大于零; SQR 的自变量必须大于或等于零; ASN 的自变量必须绝对值小于或等于1; TAN 的自变量不能太接近 $\frac{\pi}{2}$ 的奇数倍。

标准函数允许递归使用,如: $SIN(SIN(X))$ 。

十八种标准函数详见附录2。

② 自定义函数

使用者可以自行定义函数,称为自定义函数。其名字由 FN 后跟一个字母组成, FN 后的字母不同,就代表不同的自定义函数。在自定义函数的名字后面用圆括号括起来,用逗号分开的实在参数(可为算术表达式)最多不超过五个。例如: $FNA(2 * X + 3, Y)$ 和 $FNK(3 * X, Y, Z + 2, W, V)$ 。

自定义函数允许递归使用,如: $FNA(2 + FNA(X, Y + 2) - Z, SIN(W))$ 。

4 算术表达式 (简称为表达式)

用运算符将数、变量或函数连接成有意义的式子就构成算术表达式。运算符包括 + (加), - (减), * (乘), / (除), ** (乘方)。计算次序与惯例相同。

例如: $X + (3Z / A - C ** 3)$ 表示 $X + (\frac{3Z}{A} - C^3)$;

$(X / 10) ** 2 * SIN(X) + B(I, J) / FNA(K) * W$ 表示 $(\frac{X}{10})^2 * \sin(X) + \frac{B(I, J)}{FNA(K)} * W$ 。

II 句法

1. 数组说明语句

每一个数组的名字、维数和最大下标 (下标一律从零开始计数的, 可以不必指出最小下标) 需先用数组说明语句加以说明, 才可使用该数组及其所包含的下标变量。

其格式以例说明之:

`DIM A(M); B(N, P)` (左端的标号和右端的

的 2 省略, 下同)。

它说明数组 A 为一维数组, 由 $M + 1$ 个元素所组成, 数组 B 为二维数组, 由 $N + 1$ 行、 $P + 1$ 列, 共 $(N + 1) \times (P + 1)$ 个元素所组成。

最大下标可用算术表达式指出, 如: `DIM A(2 * X + 3, Y ** 2)`, 计算出表达式的值, 舍入取整后, 当作最大下标。

若圆括号中无任何算术表达式 (如 `DIM A()`), 表示将此数组撤消。

2. 定义函数语句

对自定义函数进行事先定义。

例如: `DEF FNA(X, Y) = X + SIN(Y ** 2)`

它定义了一个自定义函数 FNA, 有两个形式参数: X 和 Y。

该自定义函数所完成的运算是： $X + \sin(Y^2)$ 。

在圆括号中括起来的、用逗号分开的形式参数（形式与简单变量名字相同）最多不超过五个。

自定义函数不允许递归定义，如： $DEF \quad FNB(X, Y)$
 $= 2 + FNB(X-1, Y)$

是不允许的。

自定义函数的形式参数是局部于定义函数语句的虚变量，在程序中可用作其他名字。

3 赋值语句

用来进行算术运算，并将计算结果赋给某个变量。

例如：

$LET \quad A = 5.6 + X$ 是计算 $5.6 + X$ 的值，并赋给简单变量 A 。

$LET \quad Z(0, 1) = SIN(X) + B(I, J) - FNA(K)$ 是计算 $SIN(X) + B(I, J) - FNA(K)$ 的值，并赋给下标变量 $Z(0, 1)$ 。

等号右端的各变量要求是已被赋值的，否则，解释系统自动用零代替。

4 输入语句和恢复语句

① 可以用赋值语句来实现数据的输入。

例如： $LET \quad X = 3.24$ 它把数据 3.24 赋给变量 X 。

② 用数据语句（DATA）和读入语句（READ）配合使用，输入一组数据。

数据语句可以放在程序的任意位置，流程序输入时，解释系统就把数据依次放入数据区，程序执行时，读入语句依次从数据区取出数据，并赋给对应的变量。数据语句为非执行语句。

例：

```

4    DATA  4.2; 3.2
10   READ  X; Y; Z(0,1); V; U;
.
.
50   READ  X1; Y1; W2
60   DATA  -5, -2; 2; 0.3; 5; 0.2, -1; 0
    
```

上述语句实现：取数据 4.2 赋给变量 X (记为 $4.2 \Rightarrow X$)， $3 \Rightarrow Y$ ， $-5, -2 \Rightarrow Z(0, 1)$ ， $2 \Rightarrow V$ ， $0.3 \Rightarrow U$ ， $5 \Rightarrow X1$ ， $0.2, -1 \Rightarrow Y1$ ， $0 \Rightarrow W$ 。

在程序中可以使用恢复语句

RESTORE $\underline{n}$ (其中 $\underline{n}$ 是由表达式计算后，四舍五入取整得到的值，它表示某一标号，而非字母 $\underline{n}$ ，所以在 $\underline{n}$ 下打一横线，下同) 使其后所遇到的 READ 语句重新从标号为 $\underline{n}$ 的 DATA 语句的第一个数据读起。

RESTORE 是重新从第一个 DATA 语句的第一个数据读起。

RESTORE $\underline{n}, K$ (其中 K 是由表达式计算后，四舍五入取整得到的无符号整数) 是重新从标号为 $\underline{n}$ 的 DATA 语句的第 K 个数据读起。

例：上例中增加语句 20 RESTORE 4

则实现： $4.2 \Rightarrow X$ ， $3 \Rightarrow Y$ ， $-5, -2 \Rightarrow Z(0, 1)$ ， $2 \Rightarrow V$ ， $0.3 \Rightarrow U$ ， $4.2 \Rightarrow X1$ ， $3 \Rightarrow Y1$ ， $-5, -2 \Rightarrow W$ 。

⑤ 用数组读入语句 (AREAD) 读入数组各元素的值。

例如：AREAD B 从数据区顺序读入数据，按行赋

给数组B的各元素。

②在运算中，使用输入数据语句进行人机对话，实现数据的输入。

例：

执行到 10 INPUT X; Y 时

电传输出： 00010 INPUT X; Y

则可电传输入： 0.543, 1; -4.43;

电传输出： IN END 解释系统继续解释执行

其后的语句。

上述可实现：0.543, 1 $\Rightarrow$ X, -4.43 $\Rightarrow$ Y。

例：

执行到 15 AINPUT A 时

电传输出： 00015 AINPUT A

则可电传输入： 1; 2; ; 4.5;

电传输出： IN END 解释系统继续解释执行

其后的语句。

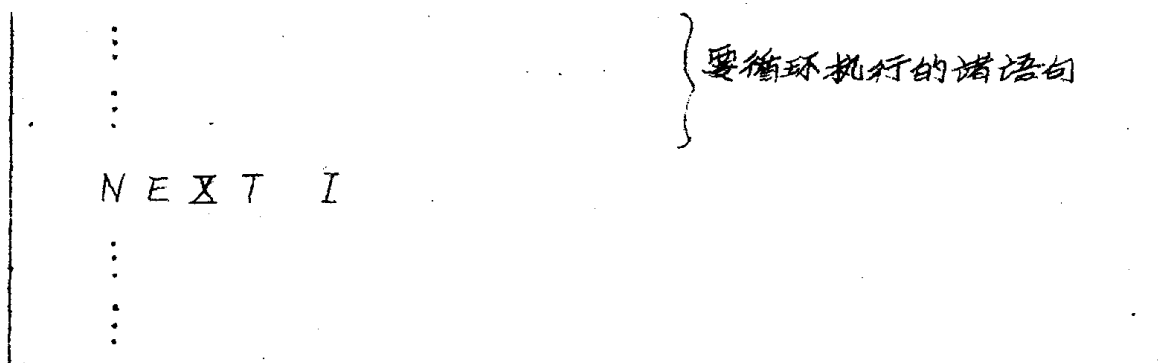
上述可实现：1 $\Rightarrow$ A(0, 0), 2 $\Rightarrow$ A(0, 1),
A(1, 0) 保持不变, 4.5 $\Rightarrow$ A(1, 1)。

用人机对话方式输入数据时，在每个数据（包括最后一个数据）后面要跟以分号“;”。在输入数据过程中（包括数据内部）可任加回车键、换行键或空格键。

5 循环语句

例如：

```
FOR I=X+1 TO 4*Y STEP 3*Z
```



其中简单变量 I 为循环变量，表达式 $X + 1$ 为循环变量的初值，表达式 $4 * Y$ 为其终值， $3 * Z$ 为其步长。步长可为正数或负数，当为 1 时，可略去不写“STEP 1”。

FOR 语句与 NEXT 语句之间是要循环执行的诸语句。

循环允许嵌套，但不能超过八重，内外循环层次要分清，不能交叉。

允许有非正常出口，但不允许从循环外面不经 FOR 语句直接转入循环内部。

在循环内部可以改变循环变量值（见五·例 8），但不能改变初值、终值和步长。

6 转向语句

用来改变程序执行的顺序，转到给定标号的语句继续执行。

格式：`GOTO n` 则转到标号为 n 的语句去执行。

7 条件语句

用来实现程序的分支。

例如：

```
IF X + 3 > 5 / Y THEN 40
```

它实现：当由关系运算符 $>$ 连接表达式 $X + 3$ 与表达式 $5 / Y$ 所形成的关系式成立时，则转标号为 40 的语句继续执行；否则，继续执行下一语句。

关系运算符包括：

=	等于
{ =	小于等于
} =	大于等于
{	小于
}	大于
{ }	不等于

8 打印语句

用来在电传打字机或宽行打印机或快速打字机上（见附录4开关5、6、7的功能）打印表达式或字符串或数组各元素的值。

打印表达式或字符串时，用 `PRINT` 语句。打印数组时用 `APRINT` 语句。

各表达式或字符串之间，若用分号隔开，则以紧凑格式打印（表达式的值后面出两个空格，字符串后面不出空格）。若用逗号隔开，则以表格形式打印（电传纸每行分四级，宽行纸每行分五级），利用这种格式可以打印整齐的表格（见五·例8）。若先写一个表达式，接着写一个冒号，然后再写字符串或表达式，其后也以分号或逗号隔开，则由冒号前的表达式的值确定电传纸（或宽行纸）的列数，并从此列开始打印出冒号后面写的字符串或表达式的值。（紧跟在 `PRINT` 后的“1:”可省略），利用这一功能可以打印各种复杂的图形（见五·例9）。

`PRINT 2` 表示空推一行。

若用宽行打印数组时，数组各元素皆按表格形式打印，并在每个数组后打印出“`OUT END`”的字样。

当用快速打印机打印数组时，在每个数组后面空推几行。

例：开关 5、6 未按下，开关 7 按下，执行下述语句：

```
10 PRINT 'X='; X; 'Y='; Y
12 PRINT 'Z='; Z; 'W='; W
14 PRINT
16 PRINT 32: 'I'; 44: 'O'
18 PRINT A; B
```

则在电传纸上打印出：

```
X = .4310 1   Y = .1210 2
Z =           .1510 1   W =           .1310 2
                                     I           O
```

在宽行纸上打印出：

```
.110 1   .2   .23   .45   .51
.12   .41
OUT END
.1610 1   .7610 2
OUT END
```

9 注解语句

在程序中用来加以注释说明，为非执行语句。

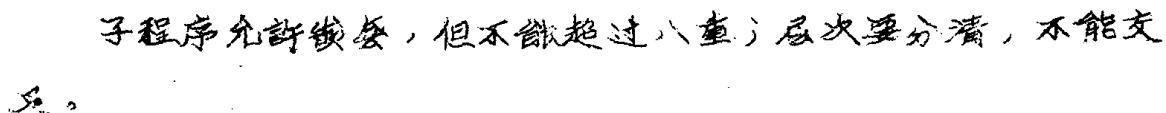
例： REM 121 BASIC EXAMPLE --1

10 空语句

格式

也是非执行语句。

例：



12 暫付證句

用来使运款暂付。

格式: STOP

运行时，遇到该语句则程序暂停执行，并且电话打印出当前

語句标号、“STOP”和即将要执行的語句标号，等待程序员进行鍵盤操作。

1.3 终止語句

表示程序执行完毕。

格式：END

运行时，遇此語句则程序行止执行，并且电传打印出 C O M P U T E E N D。

三 鍵盤操作

用鍵盤操作（鍵盤命令或鍵盤运算）来实现控制程序的运行、修改流程图、打印流程图、动态调试流程图和台式计算机等功能。详见附录3和附录4。

除错码  的功能有二：

①在光电或鍵盤输入（或增添或修改）流程图时，将其前一字符（包括回车、换行、空格等，但不包括数字键和字母键）进行除错。例如：“RE B C   A D X”，输入后为“RE A D X”。

②鍵盤操作时，在打回车键之前，发现有誤操作，则可打入一个除错码 ，将整个鍵盤操作作废，回到等待鍵盤操作状态，可重新打入正确的鍵盤操作。但没有对鍵盤操作中单个字符的除错功能。

鍵盤操作的使用见下节。

四 使用方法

I 解释程序的调入和等待鍵盤操作状态

①顺序按动“Q n Q J”、“Q n G、D、G R”、“Q n D C、D Y”、“D r o r Q”、“Q n G、D、G R”

按鍵。

② “J S Z D M”置0020,按动“Z J S Z”和“Q1”。
則將解釋程序從磁鼓中調入內存。

③將“D C S C”扳向上。

則解釋程序處於等待鍵盤操作狀態（控制台J S Z氖燈呈現2000與2001燈亮狀態），可以開始進行各種鍵盤操作。

II 流程序的輸入、修改及啟動解釋執行。

1 准备工作：

①把流程序紙帶裝在光電輸入機上（紙帶前面要留一米空白，末尾要穿五個以上字母鑽）。

②打命令 N 2，則為輸入新的流程序作好準備。

2 輸入流程序

(一)光電輸入：

①將控制台上的開關“光輸入”扳向上，“X u S R”扳向下。

②打入命令 R 2

(二)電傳輸入：

直接在鍵盤上打入各語句。也可同時穿孔輸出。

流程序輸入時，解釋程序只取同一標號的最後一次出現，其餘皆舍棄掉。

當開關3（見附錄4）未按下時，輸入後立即進行換碼；
當開關3按下時，只輸入，不進行換碼，需要打入命令 T 2 後才進行換碼。

3 檢查和修改流程序

(一)檢查流程序：

①解釋程序輸入流程序後，在換碼時對流程序進行語法檢

查，把错误语句改为空语句，并且当把“DCSC”扳下时，则将形式为“ERR 错误性质编号；出错语句标号；”的静态错误信息打印出来，查附录5“错误性质表”，则可知错误原因。

②若源程序纸上写的该语句不存在此种错误，则可打入命令 $L\ n\ 2$ 就把输入计算机中的第 n 句打出来，看是否在穿孔时穿错了，以致产生此种错误。

还可用 $L\ m,\ n\ 2$ 将第 m 到 n 句都打印出来。

使用 $L\ 2$ 则打印源程序清单，以便检查。

使用上述三者，可同时用电传机穿孔输出。

(二) 修改源程序：

① 在电传机上将修正或增添的语句打进去。

也可使用 $n\ 2$ 删除第 n 句（使之变为空语句）。

②再按一和二检查和修改源程序的静态错误；

4 启动程序解释执行

①打入命令 02 ，即可启动程序并解释执行。

②解释系统在解释执行程序过程中，还对程序进行进一步的语法检查，并把动态错误信息打印出来，以便按3进行检查和修改之。

③若为局部性错误，修改后，可打入命令 02 ，继续解释执行。

若为全局性错误，则打入命令 $R\ E\ S\ E\ T\ 2$ ，使之恢复到可重新启动的状态（初次启动程序时或打入命令 $N\ 2$ 后源程序执行到 $E\ N\ D$ 语句而结束时，可不再打入此命令），先改正错误，然后又可按①重新启动程序进行计算。