

TRS—80 MODEL I/Ⅲ 微型计算机

编译 BASIC 手册



沈阳机电学院

科技情报研究室

TRS-80 MODEL I/III

微 型 计 算 机 編譯BASIC手册

下 册

译者: 王一公 刘春梅 龚宇等

校对: 林应文 王 伟 刘存等

沈阳机电学院科技情报室

一九八四年五月

第 四 部 分

RSBASIC程序設計

这部分阐述了如何用RSBASIC写程序

I / III型专用程序设计提示

屏幕显示编程

I / III型屏幕显示有两种方式：滚动方式和图形方式。除用图形符号之外，BASIC用滚动方式将全部打印到显示输出。请参阅在滚动方式中程序设计的打印部分资料。及在图形方式中程序设计的CRTG部分资料。（PRINT和CRTG在关键字章）

编译BASIC提供RSBASIC语言用于写程序。RSBASIC是BASIC的格式，在这个手册中，我们把它看作BASIC。这一部分涉及你使用RSBASIC时的资料。

我们假定你已经熟悉BASIC。如果你是一个初学者，有许多很好的BASIC教课书，我们推荐一些：

COMPUTER PROGRAMMING IN BASIC FOR EVERYONE, Thomas Dwyer and Michael Kaufman, Radio Shack Catalog Number 62—2015.

BASIC AND THE PERSONAL COMPUTER, Thomas Dwyer and Margot Critchfield, Addison—Wesley Publishing Company, 1978.

BASIC FROM THE GROUND UP, David E. Simon, Hayden Book company,
1978.

ILLUSTRATING BASIC, Donald Alcock, Cambridge University Press, 1977.

下 册 目 录

第四部分 RSBASIC程序设计

第一章 (原书第三章)	
RSBASIC概念.....	1—21
解释BASIC如何处理和使用数据	
第二章 (原书第四章)	
建立数据文件.....	22—45
说明如何产生并存贮数据文件	
第三章 (原书第五章)	
程序分段.....	46—57
说明如何分隔一个长程序为短程序和子程序	
第四章 (原书第六章)	
BASIC关键字.....	58—175
包含每个关键字的按字母顺序的项目	

第一章 BASIC的概念

引言

这一章阐明了 BASIC 处理数据的方法，这部分对编写更有效地处理数据的程序来说是很有帮助的。

第一章 提纲

BASIC概念

I. 概述——程序的成份

- A. 程序
- B. 语句
- C. 表达式
- D. 判别式

II. BASIC处理数据的方法

A. 数据的表示方法

- 1. 常量
- 2. 变量
 - a. 变量名
 - b. 保留字
 - c. 简单变量和下标变量

B. BASIC存贮数据的方法

1. 数值数据

- a. 整型数
- b. 实型数

2. 字符串数据

C. BASIC标识常量的方法

D. BASIC标识变量的方法

E. BASIC转换数据类型的方法

- 1. 实型数转换成整型数
- 2. 整型数转换成实型数
- 3. 非法转换

III. BASIC进行数据操作的方法

A. 运算符

1. 数字型

- a. 加法
- b. 减法
- c. 乘法
- d. 除法
- e. 整数除法
- f. 指数
- g. 模数运算

2. 字符串型

3. 判断运算符

- a. 关系运算符
- b. 逻辑运算符

B. 函数

IV. 表达式语法

A. 简单表达式

B. 复合表达式

C. 函数

I、概述——程序的成份

A、程 序

程序是由一个或多个带行号的行组成。每行包括一个或多个语句。行号可以是0—65535中的任何数字。一个程序最多可有2048行。

每个程序行最多可有255个字符（不包括行号）。在这个程序行中，可有二个或多个语句，语句之间用冒号分隔。

这里举一个程序的例子：

行号	BASIC 语句	语句间的冒号	BASIC语句
100	PRLNT;	PRLNT "THIS IS THE FIRST PRLNT LFNE"	
110	FORI=1 TO 1000;	NEXTI;	'DELAY LOOP'
120	PRINT STRINGS(28, "—");		
130	PRINT "THIS IS THE NEXT		

当BASIC执行程序时，从开始到结束，每次处理一行。有些语句例如：GOTO,ONGOTO,GOSHB要改变程序的执行顺序。

B 语 句

一个语句是BASIC的一条完整的指令，告诉计算机进行某些操作。例如：

GOTO100

告诉计算机进行如下的操作：（1）控制转向100行。（2）执行100语句。

STOP

本语句告诉计算机停止程序的执行。

有许多语句要求对数据进行操作。如：在语句，

PRINT "SEPTEMBER REPORT"

中数据是SEPTEMBER REPORT。该语句通知计算机打印出引号中的内容。

C. 表达式：

表达式实际上是数据的一般形式，有两种类型的表达式：

1. 数字的表达式，它是由数值数据组成的。

例：

(1+5.2)/3

D

5 * B

3.7682

ABS(X)+RND(6)

SIN(3+E)

2. 字符串表达式，它是由字符串数据组成的。

例：

A\$

"STRING"

"STRING" & "DATA"

& "DATA"

SEGS(A\$, 2, 5)&SEG\$("MAN" , 1, 2) MS&A\$&B\$

3. 函数

函数是自动执行的子程序，大多数BASIC函数是对数进行计算的。有些函数具有特殊的用途，例如控制屏幕的显示。可以像使用数据一样，把函数当作语句的一部分。下面是一些BASIC函数：

INT

ABS

STRING\$

SEG\$

D. 判别式

BASIC可执行两种判别程序，以判别两个或更多的表达式中是否存在的某种关系：

1. 关系判别式：用来判别两个表达式间的等价关系：

例如：

A = 1

A\$ > B\$

2. 逻辑判别式：用来判别关系式间的逻辑关系：

例如：

A\$ = "YES" AND \$ = "NO"

C > 5 OR M < BOR 0 > 2

本章的其余内容，我们将详细介绍BASIC处理数据和数据运算的方法，及如何给程序输入数据。如果你是急于开始使用编译BASIC的话，上面的内容会给你足够的资料。

II BASIC处理数据的方法

这一节介绍如何在BASIC中表示数据及BASIC如何解释、存贮数据。它包括对编写很有效地处理数据的程序所必须的基本资料。

A. 数据的表示方法

BASIC用两种方式标识数。一种是直接的，作为常量。另一种是与内存单元有关的，作为变量。

1. 常量

所有的数据都是作为“常量”编入到程序中的。其值是不改变的。

例如，语句：

```
PRINT "1PLUS1EQUALS" ; 2
```

其中包括一个字符串常量

```
1 PLUS1 EQUACS
```

及一个数值常量

```
2
```

在这个例子里，常量被“读”到PRLNT语句中，告诉PRLNT显示器上打印的数据内容，下面也是常数的例子：

3.14159	"L. O. SMITH"
1.775E+ 3	"0. 123456789ABCDEF"
"NAME TITLE"	-123. 45E-8
57	"AGE"

2. 变量

一个变量在内存中有一块地方。内存好比是一个箱子或是一个文件架。用作存贮数据。与常量不同，变量的值是可以改变的。这就允许在程序中改变量的值。例如在语句：

```
A$ = "OCCUPATION"
```

变量AS现在包含数据OCCUPATION。然而如果以后在程序中出现：

```
A$ = "FINANCE"
```

变量A\$包含OCCUPATION。而是包含数据FINACE。

a. 变量名：

在BASIC中，变量是用名字表示的。变量名必须由字母 A—Z 打头，字母可以是大写，也可以是小写的，并且可以跟 1—5 个字符。（字母或数字总共可有 6 个符。）

例如：

```
AMOUNT    A    A12345    A1    B1AB2    AB
```

都是允许的，均可用作变量名。

变量名可长于6个字符。然而在BASIC中只有前6个字符有意义。

例如:

SUPERN SUPERNUN SUPERNUMFRARY

均被BASIC认为是同一变量。

b. 保留字

BASIC保留某些字作为BASIC函数。不能用这些字符或运算符NOT作为变量名。例如:

ABS SIN LEN ASC

不能被用作变量名。因为它们是BASIC函数。但变量名可以包含保留字。

例如, ABS1及LENGTH可以是变量名。

一个BASIC语句只要它不是开头的语句, 就可以用作变量名。例如:

LET LET=10

是可行的。

但 LET=10

是不允许的。

c. 简单变量与下标变量

上面提到的变量均是简单变量(又称变量)。它们只能是与一个数据项有关。

变量又可以是下标变量, 从而使许多数据可以用一个变量名存贮。这种数据存贮的方法称为数组。例如, 数组A可以包括这些元素(下标变量):

A(0), A(1), A(2), A(3), A(4)

可以用任一元素存贮一独立的数据项, 例:

A(0)=5.3

A(1)=7.2

A(2)=8.3

A(3)=6.8

A(4)=3.7

在本例中, 数组A是一个一维数组, 因为每一元素只包括一个下标。数组也可以是二维的, 每一元素包括两个下标。例如, 二维数组X可以包括这些元素:

X(0, 0)=8.6

x(0, 1)=3.5

x(1, 0)=7.3

x(1, 1)=32.6

编译BASIC不允许出现多于二维的数组。

数组在使用前必须定维, 以便在内存中为其保留空间。用DIM语句定维。上述例子中的数组A可定维如下:

DIM A(4)

给五个下标变量(0, 1, 2, 3, 和4)保留空间。

数组x可定维

DIM x(1, 1)

为 $2 \times 2 = 4$ 个下标保留空间。

注: 关于数组更多的资料请参阅DIM。

B. BASIC存贮数据的方法

BASIC存贮数据的方法决定了数所占的内存量及BASIC处理数据的速度。

1. 数值数据

BASIC把所有的数都用整数或实数存贮起来

a. 整数

(速度快、效率高, 但数值范围有限)

要按整数存贮, 数必须是整数, 并且介于一32768与32767之间。整数在内存中只需要两个字节。当两个操作数均是整数时, 运算速度较快。例如:

1 32700 - 2 500 -12345

可全部按整数存贮。

注: 整数是可以补码表示法存贮的。具体解释在程序员参考部分论述。

b. 实数

(精度较大, 但运算速度较整数慢)

当一个数按实型存贮时, BASIC可以存贮14位的有效数字(输出前6位, 第七位四舍五入)。

实数的范围:

($-1 * 10^{**} -64$, $-1 * 10^{**} 63$) 或 ($1 * 10^{**} -64$, $1 * 10^{**} 63$)

存贮一个实数需8个字节。头一个字节是用来存放指数的。

注: 实数是以十进制编码的二进制形式存贮的具体解释请看程序员参考部分

2. 字符串数据

字符串(字符的序列)在存贮非数字信息, 例如姓名、地址、内容等等时是非常有用的。

可把任一ASCII字符作为字符串存贮。(ASCII字符表参看附录),

例如, 数据常量:

JACK Brown, Age 38

可以作为一个18个字符的字符串存贮起来。每一字符(包括空格)都是用ASCII码存贮的, 需要一个字节。在机器内部, BASIC是以如下方式存贮上述字符串的。

十六进制串码	4A	61	63	6B	20	42	72	6F	77	6E	6C	20	41	67
ASCII字符	J	a	c	K		B	r	O	w	n,		A	g	
65	20	33	38											
e		3	3											

字符串长度可达255。长度为0的字符串称为“零字符串”或“空白字符串”。

C. BASIC标识常量的方法

当BASIC在语句中碰到一个数字常量, 必须要确定数的类型(字符串型, 整型或实型)。

下面是分类规则:

规则 1

若常量是用双引号括起来的, 常量是字符串常量。

例如:

"YES"

"3331 wavERLY WAY"

"1 2 3 4 5 6 7 8 9 0"

带引号的常量自动地划分为字符串型。

规则 2

如果常量前面有一&, 此常量为十六进制的常数。

例如:

&0 &7FC0 &FFFF

都是十六进制的数。十六进制的数实际上是按整数存贮的, 可在特殊情况下例如在EXT语句中使用十六进制的数。

规则 3

不带引号的常量是数值常量。(在用运算符输入数据时。本规则不起作用。参看INPUT, LINE INPUT INKEY\$和INPUT\$)

例如:

1 2 3 0 0 1

1

-7.3214E+ 6

都是数值常量。

规则 4

介于-32768与32767之间的整数是整型数。

例如:

1 2 3 5 0

-12

1 0 0 1 2

均是整型常量。

规则 5

包含十进制小数点的常数和处于整数范围之外的数是实数。如果数中包括字母E的话, 此数也是实数。

注: 指数用带E格式输出。

如：

1 . E+ 7

表示100000000, 或 1×10^8 。

1. E- 8

表示 0.00000001 , 或 1×10^{-8}

D. BASIC标识变量的方法

当BASIC在程序中碰到一变量名时，就按字符串型，整型或实型进行分类。在程序中，只能对变量进行一次分类。不能让BASIC和对某一变量名进行重新分类。

下面是BASIC变量分类的规则

规则 1

如果BASIC没有碰到定义语句（下面规则2中描述的）或类型说明符（下面规则3中描述的）就把所有的变量都划分成实型类，并且用8个字节存贮一个变量。例如：

AB	AMOUNT	XY	L
----	--------	----	---

都是实型变量。如果程序的第一行是:

LP=1.2

BASIC就把LP划分为一实型变量类。

规则 2

如果BASIC碰到一定义语句，就把变量按语句要求进行分类，有三个定义语句：

STRING

INTEGER

REAL

1. STRING语句

STRING要求把所有的变量都划分成字符串。

例如:

SIRNG

要求BASIC把所有的变量都划分成为字符串类。

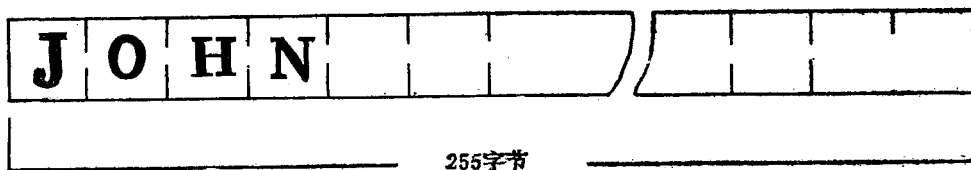
STRINGL

要求BASIC把所有以字母L开头的变量都划分为字符串。

BASIC保证所有的字符串都是用255字节存贮的，尽管变量L的数据只用了四个字节：

L = "JOHN"

BASIC用255个字节存贮这个量。这就使得L包含251个未用的字节。



为了避免浪费内存空间。可在存贮变量时说明要使用的字节数。

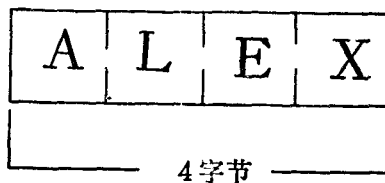
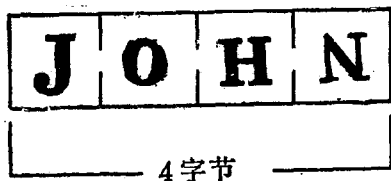
例如：在下面的程序：

```
10 STRING * 4 L
```

```
20 L = "JOHN"
```

```
30 LAST = "ALEXANDER"
```

LAST仅包含四个字节的字符串。



若要不按STRUNG语句的规定存储LAST字符串变量的时候，可用DIM语句。

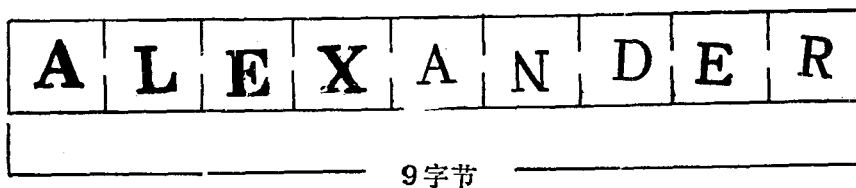
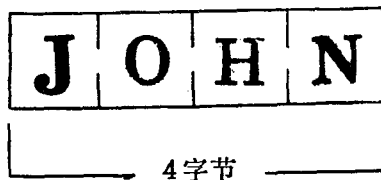
```
10 STRING * 4 L
```

```
20 DIM LAST$ 9
```

```
30 L = "JOHN"
```

```
40 LAST = "ALEXANDER"
```

本程序用四个字节存贮变量L而用9个字节存贮变量LAST。



注：详阅DIM和STRING的内容。

2. INTEGER语句

INTEGER要求BASIC把所有的变量都划分为整型类。

例如：

```
INTEGER A
```

要求BASIC把所有的用字母A开头的变量都划分为整型类。

```
INTEGER
```

要求BASIC把所有的变量都划分为整型。

在当前的BASIC格式中，全部整型变量都是用两个字节存贮的。

3. REAL语句

REAL要求BASIC把用字母表中的字母开头的变量都划分为实型类。

例如，下面程序：

```
10 INTEGER
```

```
20 REALX—Z
```

要求BASIC把所有的变量各除了X，Y和Z开头的均划分为整型类。并把以X，Y和Z开头的变量各均划分为实型类。

在BASIC中，所有的实型变量都是用8个字节存贮的。

定义语句的非法使用

在可执行语句后面不允许出现定义语句，可执行语句是定义语句以外的语句。例如：

```
10 L=10
```

```
20 STRLNG
```

要出现错误因为STRING不能跟在可执行语句L=10之后，但是

```
10 STRING
```

```
20 L=10
```

是正确的。

规则 3

如果在变量名后带有类型说明符，BASIC就会根据说明符对变量加以分类。

\$ 字符串型

% 整型

实型

(但是，不能用说明符对程序中已经分类过的变量各重新分类)

例如，如果变量S，MON，FINANCE和CHART还没有在程序中使用过：

```
S$ MON$ FINANCE$ CHART$
```

将全部被分类成字符串变量名，而不管事先已对字母S、M、F和C作出的定义。

如果变量名I、LM、NUM、COUNTER还没有在程序中使用过：

```
I% LM% NUM% COUNTER%
```

将全部被划分成整型变量名，而不管事先已对字母I、L、N和C所做出的定义。

如果变量LR，ER，MP235和LINE还没有在程序中使用过：

```
LR# ER# MP235# LINE#
```

将全部被划分成实型变量，而不管事先已对字母L—E和M所作出的定义。

例如，，在下面的程序：

```
10 STRINGA
```

```
20 AB= "NEW"
```

语句：

```
30 AB%= 1
```

就会出现错误，因为AB已经被划分成字符串变量，不能再重新划分。但是：

```
30 AR%= 1
```

是正确的。因为类型说明符(%)取消了STRINGA语句的作用。

一旦用类型说明符对变量分了类,就不必在程序中使用说明符了。例如,执行完下一语句后:

B\$ = "DATA"

就可简单地把字符串变量B\$记作B。在执行程序的其它部分时B一直是字符串变量。

(即使仅在引入变量名时需要用说明符,我们仍建议大家在每次使用变量名时也用说明符,从而使得程序始终如一,易于修改)

E、BASIC转换数值数据的方法

在程序中经常要求BASIC把整型常数赋值给实型变量,例如:

A = 5

或是把实型常量赋值给整型变量,例如:

B% = 5.2

为了做到这一点,BASIC必须首先转换数据常数。方法如下:

1. 实型转换成整型

BASIC把小数部分截掉,舍掉后的值必须介于一32768与32767之间。

例如:

A% = -10.5

把-10赋值给A%。

A% = 32767.9

把32767赋值给A%。

A% = 2.5E+3

把2500赋值给A%

A% = 60000

出现整数溢出警告,并把32767赋值给A%(32767是整型数中的最大值)。

2. 整型数转换成实型

把整型转换成实型,其值不变,但存储空间是原来的四倍。(整型数占2个字节,实型数与8个字节)。例如:

A = 1

在A中存贮的是1.0000000000000。

3. 非法的转换

BASIC不能自动地把数值数据转换成字符串反之亦然。

例如,语句:

A\$ = 1234

A% = "1234"

是非法的。(这种转换要用STR\$和VAL)

III BASIC进行数据操作的方法

这部分介绍了要求计算机处理或判断数据的方法。可以使用运算符和函数两种工具。

A、运算符

运算符是简单的运算符号，它是表示对一个或两个已知值的操作数所进行的处理。

一般来说，运算符的使用格式是：

操作数1 运算符 操作数2

操作数1 和操作数2 可以是表达式

有些运算只需要一个操作数，使用格式是：

运算符 操作数

这是一元操作的形式，

例：

$6 + 2$

加法运算符“+”把两个操作数6和2加在一起，得到结果8。

-5

取负运算符“-”对单个运算数5进行处理，得到结果负5。

不管是 $6 + 2$ 还是 -5 都不能单独出现，必须在语句中才对BASIC起作用。例如：

$A = 6 + 2$

PRINT -5

运算符可分为三类：

数字 Numeric

字符串 String

判别 TeSt

分类的根据是所需要的操作数的类型和产生的结果。

数字运算符

数字运算符用于数值表达式。其操作数必须是数字，得出的结果是一个数值。

在下面的叙述中，我们使用术语整型和实型运算。整型运算包含两个字节的操作数，实型运算包含八个字节的操作数。实型运算速度较慢，因为它们包括较多的字节。

有9种不同的数字运算符。符号“+”和符号“-”是二种一元运算符，因为它们只有一个操作数。符号运算对操作数的精度无任何影响，

例如，在语句：

PRINT -77 , $+77$

符号运算符“-”和“+”分别得出负77和正77。

注：若一个数字前无任何运算符，则其符号为十。

其它的任何运算符都是二元制的，因为它们都需要二个操作数。