微型计算机及其应用丛书

微型计算机 BASIC 语言的应用与分析

朱巧生　万加雷　编著

内 容 简 介

本书除介绍一般 BASIC 语言的应用外，主要介绍在其他 BASIC 语言书籍中很少涉及的数据文件输入与输出、汉字和图形的处理、设计通用汉字数据文件管理系统等内容.

全书共分八章.第一至第三章介绍 BASIC 语言的基础知识，字符串的处理和数据文件的输入与输出；第四至第六章介绍如何利用 BASIC 语言开发汉字系统，内容包括汉字系统的配置、用 BASIC 语言实现汉字输出和设计通用汉字数据文件管理系统；第七章介绍一种汉字图形 BASIC——HANBASIC 的功能；第八章以 HANBASIC 为蓝本详细分析了 BASIC 解释系统的结构，着重介绍如何进行语法和词法分析及各语句功能是如何实现的等；书后附有语法表和机内代码对照表. 为了照顾不同程度的读者，书中还附有详细的程序清单和框图.

本书可作为大专院校的教科书或参考书，亦可作为应用 BASIC 语言的工具书.

微型计算机及其应用丛书

微型计算机 BASIC 语言的应用与分析

朱巧生　万加雷　编著

责任编辑　孙月湘　杨　艳

科学出版社出版

北京朝阳门内大街 137 号

中国科学院印刷厂印刷

新华书店北京发行所发行　各地新华书店经售

*

1987 年 7 月第 一 版　开本：787×1092　1/16

1987 年 7 月第一次印刷　印张：13 1/4

印数：0001—9,800　字数：299,000

统一书号：15031·825

本社书号：5015·15—8

定价：3.15 元

序　　言

自一九七一年微处理器问世以来，在短短十三年时间里，微型计算机获得了突飞猛进的发展．在技术性能方面，微型计算机比第一台电子管计算机提高了四至五个数量级；微型计算机的字长由四位、八位、十六位发展到三十二位，覆盖了小型计算机和超级小型计算机的全部领域．多微型机分布结构的系统已经商品化．微型计算机网络技术已日趋成熟，亦有商品问世．多微型机的阵列机或多功能的微型计算机系统已向中、大型计算机系统发起了挑战．另一方面，价廉物美的个人微型计算机日新月异，如雨后春笋般蓬勃发展．总之，微型计算机冲击了计算机科学技术的全部领域，大大扩展了计算机的应用范围，为计算机渗入个人的工作、生活、家庭以及渗入社会的各个方面创造了必要的条件．这样就有力地促进了新的产业革命，加速了信息技术的发展，推动了信息社会向纵深发展．

一九七九年全国计算机委员会确定要大力发展微型计算机，推广微型计算机的应用并且在全国组织试点．四年来，微型计算机的应用扩展了几十倍、上百倍，应用领域遍及工业、农业、科学文化、军事国防等各个方面；应用计算机的人才成百倍地增长，取得了空前丰硕的成果．当前，迎接新的世界技术革命的浪潮澎湃向前，一个突出表现就是席卷全国的“微型计算机热”．毫无疑问，这个热潮必将推动四个现代化的事业向前迅进．

顺应形势，我们编写了这套“微型计算机及其应用丛书”．本丛书贯彻理论和实际相结合的原则，在介绍基本理论的同时引入许多实例．丛书的作者都是该领域的专门家，他们吸收消化了国外的经验，分析了国外的技术和系统，结合我国的情况进行开发、创造，取得了许多可喜的成绩．因此，他们写出的东西是学了即可用得上的．

本丛书并不打算解决所有层次的问题，只想在普及推广方面提供一套基本的系统的材料，使有志于微型计算机应用的人们能有一套参考书．这套丛书不仅包括基本的微型计算机硬件系统、软件系统，还包括了微型计算机的开发技术和实际应用等内容．这些内容中许多都是著者实际工作成果的总结，因此它有一个鲜明的特点：解决实际问题，与四化建设紧密相连．

目前国内外有许多关于微型计算机的著作和资料，不过象本丛书这样全面地讲述还很少见．我们希望这套丛书能为四化作出贡献．

丛书书目

微型计算机硬件系统(上、下)
微型计算机操作系统(上、下)
微型计算机汇编语言的使用与分析
微型计算机 BASIC 语言的应用与分析
微型计算机开发系统
微型计算机的系统设计及性能评价
微型计算机在企业管理中的应用
微型计算机在数控技术中的应用
微型计算机在图象信息处理中的应用

徐正春　张菊年

前　言

BASIC 是英文 Beginner's All-purpose Symbolic Instruction Code 的缩写. 它是一种简单易懂又比较容易掌握的计算机语言. 国内外介绍这种语言入门的书很多，而本书的重点在于介绍 BASIC 语言的应用与分析，书中给出大量的应用实例，目的是向读者提供编制程序的"砖和瓦". 有的例子仅作必要的说明，有的没有说明，目的是让读者自己去理解、实践，不断提高编程技巧. 如第六章中，由于篇幅有限，只给出了 ZDFMS 的全部程序清单，而没有任何解释.

从 BASIC 语言问世以来，它还没有一个统一的文本. 随着机器的不同，BASIC 版本亦不相同，因此要介绍所有 BASIC 版本是不可能的. 本书主要以 CROMEMCO 扩展 BASIC 为基础，介绍 BASIC 语言的应用与分析. 经适当修改后，这些例子也适用于其他版本的 BASIC.

为满足初学者的要求，本书在前三章里，讨论了 BASIC 基本指令的用法，但重点还是放在应用上，即如何应用这些指令，并指出容易出错的地方. 其中第一章介绍基本 BASIC 指令，第二章介绍字符串；第三章介绍数据文件的读写.

第四至第六章着重介绍 BASIC 的应用，主要内容是怎样用 BASIC 语言来实现汉字的输入与输出. 由于全部采用 BASIC 的语句来实现，所以具有较大的可移植性. 其中第四章介绍汉字系统的配置，为下面几章作准备，第五章介绍用扩展 BASIC 实现汉字输出，第六章介绍一个应用程序包：通用汉字数据文件管理系统 ZDFMS.

第七章介绍 HANBASIC 的补充功能. HANBASIC 是在扩展 BASIC 的基础上，增加了汉字字符串的概念以及有关汉字显示、作图的语句和函数后，而形成的汉字、图形 BASIC 的最新版本. 第八章介绍 HANBASIC 的分析，以帮助读者了解解释程序工作原理及深入掌握各种语句的功能，更好地使用 BASIC 语言.

第四章至第八章的大部分内容取自 KSJ 微型机汉字系统. 该系统是笔者近几年的研究成果，在全国得到广泛的应用. 由于水平有限，书中的例子虽然实用可行，但未必最佳，错误之处在所难免，敬请读者批评指正.

陆汝铃同志对本书的编写不仅给出原则性的指导，而且在百忙中审阅了本书初稿. 本书编写过程中得到陈平、张寿云、弓惠生、朱敏慧、陈教芳、赵立平等同志的大力支持，在此一并致谢.

一九八三年十二月于北京

作者

目　　录

第一章　基本 BASIC

1.1　常量、变量及其类型

由于 BASIC 版本的不同，它能处理的常量和变量的类型也不同. 例如控制 BASIC (亦称 IBASIC) 只能处理整型数及整型变量. 而一般 BASIC 能处理整型、短浮点型和长浮点型数值常量及其相应变量. 在扩展 BASIC 中引进了一种新的类型——字符串及字符串变量. 本书最后第七、八章介绍的汉字 BASIC 中又增加了另一种新的类型，即汉字字符串及汉字字符串变量. 本章将讨论一般 BASIC 的数值常量及其相应变量.

1.1.1　数值常量及其运算

一般 BASIC 中的常量有以下三种:

(1) 整型数　一个小于 10000 而不带小数点的常数称为整型数. 例如 6，−16，9999.

(2) 浮点数　一个带小数点或指数(E)的常数称为浮点数，而一个大于等于 10000 而不带小数点的数亦定义为浮点数.

在一个浮点数中，若数字个数不大于 6 个，则称为短浮点数. 例如 7.6，−16.32，2.45E-20，16.

在一个浮点数中，若数字个数大于 6 个，则称为长浮点数. 例如 6.00000000，1.234569085E + 6.

(3) 十六进制数　两边用 % (百分号)括起来的数称为十六进制数. 例如 %6%，%FAB3%.

一个常量的类型由此常量的表示形式所确定，而与此常量在数学概念上的值无关. 例如 6，6.，6.00000000，%6% 是四个不同类型的常量，但从数学概念来看是同一个值.

有些在数学上完全等价的计算式子，由于涉及常量的类型而结果完全不一样. 例如:

```
6/7                 =0
6./7                =0.857143
6.00000000/7        =0.857142857142857
2.*3/4              =1.5
3/4*2.              =0
```

1.1.2　数值变量及算术表达式

每个人都有一个名字，以示与他人区别. 变量也一样，在一个程序中每个变量都有自己的名字，以在执行过程中区别对待. 一般 BASIC 规定一个变量名最多只能由两个字符组成，而且第一个必须为英文字母，第二个(可以没有)必须是数字. 例如 A，D1，Z9，B0.

数值变量可有三种类型：

(1) 整型变量　　在机器内占两个字节，表示范围：－32768～＋32767。

(2) 短浮点型变量　　在机器内占四个字节，表示范围：±9.99E＋62 ～ ±1.00E－65，可有六个数字。

(3) 长浮点型变量　　在机器内占八个字节，表示范围：±9.99E＋62 ～ ±1.00E－65，可有十四个数字。

用算术运算符＋，－，＊，/，↑(或**)及括号()把一些常量、变量及函数连接起来的有意义的式子称为算术表达式。例如：

Y = G * H/2	$y = \frac{1}{2}gh$
Z = SQR(COS(X) ↑ 2 + SIN(Y) * *2)	$z = \sqrt{\cos^2(x) + \sin^2(y)}$
X = A + B/(C + D/(E + F/G))	$x = a + \cfrac{b}{c + \cfrac{d}{e + \cfrac{f}{g}}}$

一般 BASIC 都配备一些内部函数，例如 SIN(X)，COS(X) 等。读者可以查阅有关使用手册。

1.1.3 变量类型的定义

每一个 BASIC 版本工作时，都有一个常规变量类型的定义。例如有些 BASIC 规定，凡不作特殊声明的变量一律定义为长浮点型变量。程序员可以利用变量名后缀的标志来规定变量的类型。例如，A% 表示变量A为整型变量，B# 表示变量B为短浮点型变量，而不加任何后缀则为长浮点型变量。有些 BASIC 版本则允许程序员用语句来规定变量的类型。其格式如下：

IMODE,	LFMODE,	SFMODE
INTEGER,	LONG,	SHORT

其中前三个用来改变整体变量类型的定义，后三个则用来定义某些具体变量的类型。例如：

```
10  IMODE: x = 2.5: IF x = 2.5 THEN RUN
20  SHORT X, Y, Z
30  LONG R, S, T
```

这些语句执行后，整个程序中，除 X，Y，Z 为短浮点型变量，R，S，T 为长浮点型变量外，其他所有变量均定义为整形变量。其中语句行 10 是为了保证常规定义类型为整型变量而设置的。

在 BASIC 中，一些内部函数的自变量类型是规定的，调用时若不符则自动转换为相应的类型。例如，计算 cos(5)，在执行时把 5 自动转换为长浮点常量来处理。而执行到 TAB(5.6) 语法项时，将把5.6换成整型数 6 来处理。与常量计算一样，当一个算术表达式里含有不同类型变量时，由于计算次序的不同而产生不同的结果。例如：

```
    10 SHORT S,L :INTEGER I
    20 S=7 : I=1
    30 L=I/3*S : PRINT L
    40 L=I*S/3 : PRINT L
>>RUN
0
2.33333
```

在一个算术表达式中，两个不同类型的变量(或常量)运算后，其结果将采用较长的类型，然后自动转换成赋值号左边变量所定义的类型．由于整型数之间的运算速度比较快，所以在不影响运算结果的前提下，应尽可能地把整型变量之间的运算放在前面．例如：设 I 为整型变量，L 为长浮点型变量，虽然 L＝I＊I＊I＊I＊L 与 L＝L＊I＊I＊I＊I 的结果是一样的，但前者的速度要比后者快．

1.2 输入及输出语句

1.2.1 输出语句

格式 1 〈行号〉 PRINT

2 〈行号〉 PRINT A_1 [分隔符] A_2 [分隔符]$\cdots$ A_n [分隔符]

这里〈行号〉的意思为行号可有可无，若有，则为程序的一个语句行，若无，则将为由 BASIC 立即执行的指令．以后各种语句格式中均为此意，不再一一解释．

其中 $A_i(i=1,2,\cdots,n)$ 可以是数值常量、数值变量及算术表达式(将输出它们的当前值)．也可以是字符串及字符串变量．格式中的 PRINT 可以用@符号来代替．例如 PRINT A 与 $@A$ 等价．格式中[分隔符]可用如下三种：

；(分号)表示前面的内容输出结束后，继续输出后面的内容，中间不留任何空隙．

，(逗号)表示前面的内容输出结束后，将在下一个区段的开始输出后面的内容．微型机采用的显示器一般每行能显示 80 个字符，为了输出格式的整齐，BASIC 把每行分成若干个区段．有分四个区段(每区段占 20 个字符)，也有分八个区段(每区段占 10 个字符)．更详细的请查阅使用说明．

[CR] (回车换行键)表示前面的内容输出结束后，将产生一个回车换行符，后面的内容将在下一行的开始输出．若我们把 PRINT 后面的内容称为输出表，显然回车换行这种分隔符只能用在输出表的最后面．

格式 1 是格式 2 的特例，即无输出表，而以回车换行作分隔符，因此其结果是输出一个回车换行符．

下面均是正确的输出语句：

PRINT A, B; C,

@ A＊A＋B/C; A＋C, C

@ R＋S; A＋B; C＋D;

实际上 BASIC 解释系统中有一个打印指针，每输出一个字符后，指针右移一位．遇

到分隔符为分号时，指针不移动，而遇到分隔符为逗号时，指针立即移至下一区段的开始．满一行后自动产生一个回车换行信息．在输出语句执行时，如果打印机是打开的，则显示器上输出的所有内容都将打印出来．因此打印机每行只能打印 80 个字符．这对于具有每行输出 132 个字符能力的打印机来说，是个很大的浪费．所以一般 BASIC 往往配有改变区段宽度及打印机输出宽度的指令．

1.2.2 输入语句

格式 1 〈行号〉 INPUT X_1, X_2, ···, X_n

2 〈行号〉 INPUT "一串字符", X_1, X_2, ···, X_n

这里 $X_i(i = 1, 2, \cdots, n)$ 可为数值变量或字符串变量．一串字符(它们两边必须用引号括起来)是用来对输入变量作一些必要的提示．如果没有，则在用户键盘输入数据前，显示器上将出现一个问号．当用户输入数据的个数小于输入表中规定的个数时，则将再出现两个问号．

例 1

```
    60 INPUT A,B,C,D
    70 PRINT A+B+C+D
>>RUN
?1,5
??9,22
37
```

例 2

```
    90 INPUT"X=",X
    95 INPUT"Y",Y
   100 PRINT"X*X+2*X*Y+Y*Y=";X*X+2*X*Y+Y*Y
>>RUN
X=4
Y=7
X*X+2*X*Y+Y*Y=121
```

在 INPUT 语句输入表的最后，还可以用分号来结尾，以使输出格式更能满足一些特殊要求．

例 3

```
   115 INPUT"EXAMPL: ",A;
   120 B=A+5
   125 PRINT"+5=";B
>>RUN
EXAMPL: 7+5=12
```

INPUT 是键盘输入语句，亦即程序执行时将停下来，等待用户从键盘上输入必要的数据．由于输入表中各变量可以是数值变量，也可以是字符串变量．当然也可能是混合的，所以用户在输入时必须保持其一一对应的关系．

1.2.3 数据存取语句

格式 1 行号 DATA A_1, A_2, ···, A_n

2 〈行号〉 READ X_1, X_2, ···, X_n

这里 DATA 语句中行号不带尖括号，用以说明此语句中行号是必要的，也就是说不能作为指令来立即执行．其中 $A_i(i=1,2,\cdots,n)$ 可为数值常量、变量及算术表达式，也可为字符串及字符串变量．若是字符串则其两边必须用引号括起来．其中 $X_i(i=1,2,\cdots,n)$ 可为数值变量或字符串变量．因此 DATA 语句和 READ 语句使用时必须注意它们之间变量类型的一致性．

在 BASIC 解释系统中，有一个数据指针，当一个程序用 RUN 指令开始执行后，此指针指向程序中第一个 DATA 语句的第一个数据．每当用 READ 语句读出一个数据后，指令顺序向后移一个数据．当 DATA 语句中的数据全部读出后，再执行 READ 语句就要出错．因此，DATA 语句可以出现在程序的任意位置，而不影响正常运行，下面的例子说明了此问题：

例 4

```
 190 DATA 5
 195  FOR I=1 TO 5
 200  DATA 3**2
 205  READ A
 210  DATA 8/3
 215  PRINT A
 220  DATA 400.0
 225  NEXT I
 227 DATA 1/2
>>RUN
5
9
2
400
0
```

如果一些数据需要重复使用，或者程序员要按规定的地方去取数据，则可用 RESTORE 语句，使数据指针向前或退后若干个数据位置．

例 5

```
 138 DATA 1,2,3
 142 DATA 4,5,6
 146 READ A,B,C
 150 DATA 7,8,9
 158 RESTORE
 162 READ D,E,F
 166 RESTORE 146
 170 READ G,H,I
 174 PRINT A,B,C : PRINT D,E,F
 178 PRINT G,H,I
>>RUN
1             2             3
1             2             3
7             8             9
```

1.2.4 输出格式编辑语句

格式 1 〈行号〉 PRINT SPC (表达式)[分隔符]

2 〈行号〉 PRINT TAB (表达式)[分隔符]

3 〈行号〉 PRINT USING "编辑格式", X_1, X_2, ···, X_n

这里 PRINT 可以用符号@来代替，即 PRINT SPC(5) 和@SPC(5) 是等价的.

为了使输出格式更加丰富多彩,更适应程序设计的要求，一般 BASIC 解释系统都有为打印格式服务的一些函数和编辑格式语句. 格式 1 称为输出空格函数,此语句执行时，将按表达式的值输出空格字符. 它将给出前后两个输出内容的相对距离(空格数).

例 6

```
235 A=2 : B=1234
240 PRINT A;SPC(10);B
245 A=123456.0
250 PRINT A;SPC(10);B
>>RUN
2           1234
123456            1234
```

上例说明 SPC(10); 的作用在于: A 的全部内容输出结束后，将连续输出 10 个空格字符,然后再输出 B 的值.

格式 2 称为输出制表函数. 它将规定其后面输出的内容在显示器(或打印机)上的绝对位置.

例 7

```
260 A=2 : B=1234
265 PRINT TAB(5);A;TAB(15);B
270 A=123456.0
275 PRINT TAB(5);A;TAB(15);B
>>RUN
    2           1234
    123456      1234
```

从例 7 可以看出,当执行 PRINT TAB(15); B 语句时,不管前面输出内容如何, B 的值将从此行的第15个位置开始输出. 如果前面内容全部输出结束后，BASIC 打印指针所指位置超过 TAB 函数中表达式的值，则将在下一行的指定位置开始输出后面的内容.

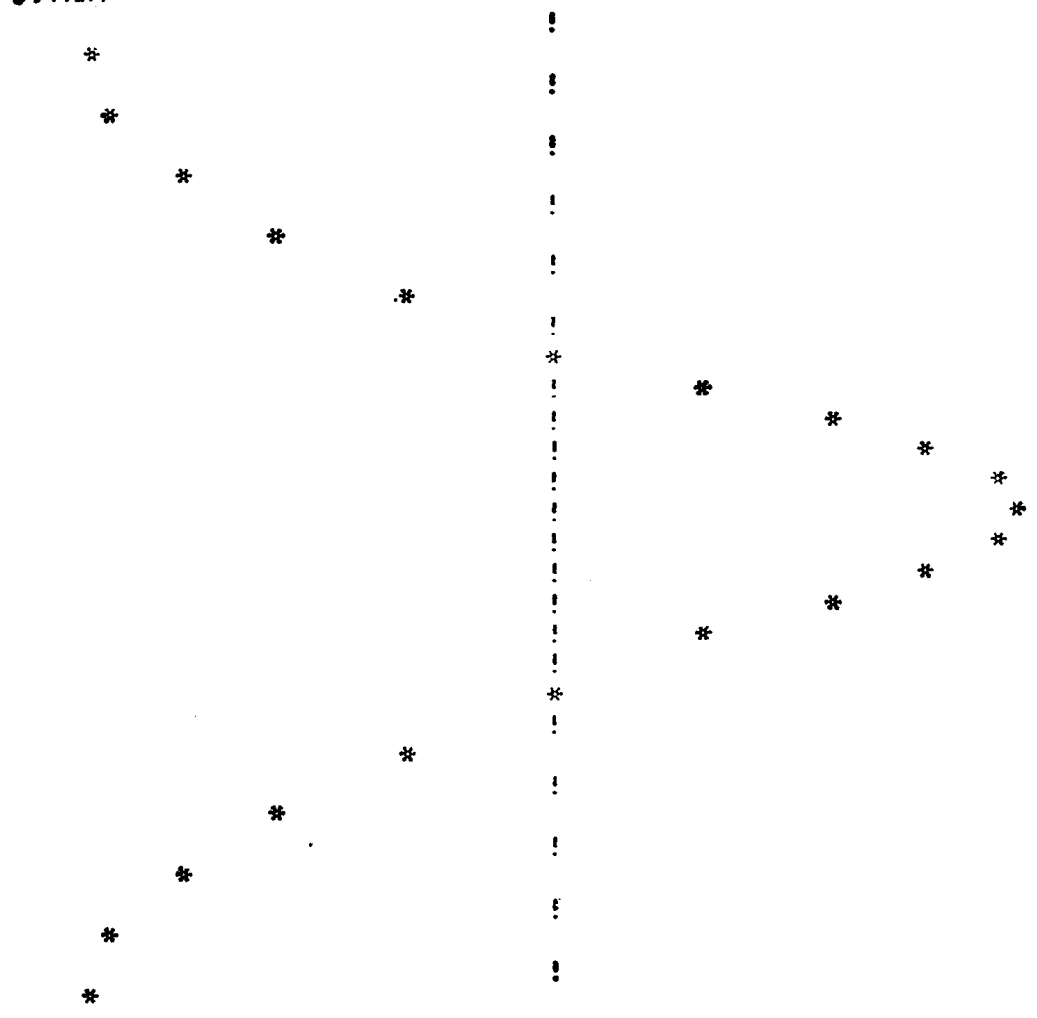
格式 1 和格式 2 中的分隔符可以是分号也可以是逗号. 它们的作用同 1.2.1 中所叙述的一样. 对初学者来说，TAB, SPC 以及一些相应的[分隔符]，再加上下面将介绍的编辑格式混合起来使用会引起一些麻烦. 所以使用时必须小心谨慎.例 8、例 9 给出了输出一条 X = SIN(Y) 曲线. 其中常数 30 是为了把 Y 轴画在第 30 列位置，而常数 25 则是为了适当地加大 SIN(Y) 的值，以使能画出曲线. 例 8 的程序有点毛病，而例 9 是正确的. 读者如果手头有机器可以试一试. 如果想在显示器屏幕上出现一条"运动"着的曲线,最简单的方法是增添下面两个语句:

```
313  FOR J = 0 TO 100
350  NEXT J
```

有兴趣的读者可以一试.

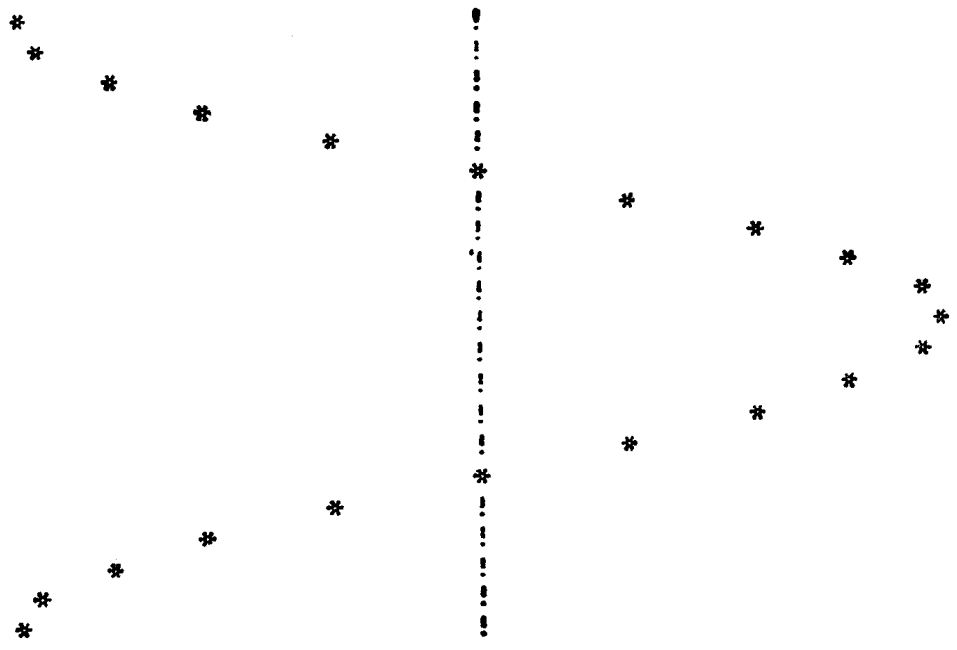
例 8

```
  290  FOR I=-3.1416 TO 3.1416 STEP 0.31416
  295  @TAB(30);"!";TAB(30+25*COS(I));"*"
  300  NEXT I
>>RUN
```



例 9

```
  310 INTEGER D
  315  FOR I=-3.1416 TO 3.1416 STEP 0.31416
  320  D=25*COS(I)
  325  IF D=0 THEN@TAB(30);"*" : GOTO 345
  330  IF D<0 THEN 340
  335  @TAB(30);"!";TAB(30+D);"*" : GOTO 345
  340  @TAB(30+D);"*";TAB(30);"!"
  345  NEXT I
>>RUN
```



格式 3 称为输出格式编辑语句. 其中编辑格式是由#、*、&、·、,、+、−、$、！及字符空格等专用符号组成.它们主要用于使数字报表输出较为整齐.前面叙述的输出方法全部属于左对齐,也就是说指定了输出内容的起始位置. 如果要求输出报表中,上下两行数字的小数点要对齐,若无小数点则个位数要对齐, 不足的部分用 0 (小数点后)或空格(小数点前)来补齐,则用前面的方法来实现是比较困难的. 因而一般 BASIC 都配有输出格式编辑语句,来满足右对齐的需要. 各种版本 BASIC 的格式也很不一样, 使用前必须仔细查阅有关说明. 下面给出一个例子供参考, 其中符号 ">"是为了指示打印起点而添加的.

数	编辑格式	输出结果
56.78	+##.##	>+56.78
3.4	+##.##	>+ 3.40
-6.4321	+##.##	>- 6.43
123	+##.##	>***.**
56.78	+++##.##	> +56.78
3.4	+++##.##	> + 3.4
-6.4321	+++##.##	> - 6.43
123	+++##.##	>+123.00
34.2	-&&&.&&	> 034.20
-1.78	-&&&.&&	>-001.78
123	-&&&.&&	> 123.00
1234	-&&&.&&	>****.**
34.2	****.**	>**34.20
-1.78	****.**	>***1.78
123	****.**	>*123.00
1234	****.**	>1234.00