计算机应用基础
学习指导书

王利 主编

中央广播电视大学出版社

计算机应用基础学习指导书

王 利 主编

中央广播电视大学出版社

（京）新登字163号

计算机应用基础学习指导书

王 利 主编

中央广播电视大学出版社出版

新华书店总店科技发行所发行

一二〇二工厂印装

开本787×1092 1/16 印张10 251千字

1991年10月第1版 1991年10月第1次印刷

印数 1—90000

定价 4.00元

ISBN 7—304—00632—3/TP·30

前　言

本书是为电大经济类学员学习计算机语言及应用课程而编写的辅助教材。它与主教材《计算机应用基础》和实验教材《计算机应用基础实验》配套使用。

本书包括三个部分的内容：第一部分为BASIC语言辅导，第二部分为汉字dBASE Ⅲ辅导，第三部分为习题与解答。第一部分和第二部分分别对主教材中的重点、难点和疑点进行了详细地说明和辅导，第三部分对期末复习和日常做作业均有参考作用。

本书由主讲教师王利主编，中央广播电视大学主持教师徐孝凯和大连广播电视大学教师朱嵬参加了部分章节的编写。

由于水平有限，加之时间仓促，不当之处在所难免，希望广大读者给予批评指正。

编　者

1991年4月

目　录

第一部分　BASIC语言辅导

第二部分　汉字dBASE Ⅲ辅导

第一部分　BASIC语言辅导

第一讲　分　支

一、流程图的使用

流程图是分析或处理一个问题的图形表示，它由一些框和带箭头的线连接而成。这些框一般被分为起止框、叙述框、判断框和循环框四种。起止框用两侧为圆弧的矩形框表示，框中的内容为“开始”或“结束”字样，注明“开始”时，则为开始框，注明“结束”时，则为“结束”框。叙述框（又叫处理框）用矩形框表示，框中的内容一般是用文字或公式指明要进行的具体操作。判断框（又叫条件框）用菱形框表示，它有一个入口和两个出口，其中标有“Y”的那一个表示条件成立的出口，标有“N”的那一个表示条件不成立的出口，判断框中的内容为一个条件（即一个逻辑表达式）。循环框包括FOR循环框和WHILE循环框，它们分别在主教材的第114页和第127页给出。

流程图对于分析和编写BASIC程序都很重要。当分析一个现成的BASIC程序时，首先根据程序画出对应的流程图，然后通过分析和察看流程图就能够清楚而直观地看出程序的逻辑结构和所具有的功能。当根据问题编写一个BASIC程序时，首先分析出解决问题的算法（即求解方法或求解步骤），接着画出相应的流程图，然后就能够比较容易地按照流程图编写出程序。当然对于一些简单的问题，可直接由算法编写出程序，或直接写出程序。

解决一个问题，其算法可能有一种，也可能有许多种。对于编程者来说，首先就是要确定算法或从若干个算法中找出一个最合适的算法。这些对于初学者来说都是比较困难的，这需要多阅读一些别人的程序，并不断地积累经验。

例1，根据求两个正整数的最大公约数的算法，画出流程图并编写程序。

已知求两个正整数的最大公约数的算法如下：

（1）设置两变量M和N存放正整数，设置变量R存放M除以N后的余数；

（2）输入M和N；

（3）如果M＜N，则交换它们的值；

（4）用M除以N并把余数送给R；

（5）如果R = 0，则表明N的当前值为所求的最大公约数，应把它打印出来并结束运行；

（6）把N送给M，R送给N，然后转向第（4）步。

在算法的第（3）步，如果M＜N，则需要交换它们的值。在计算机中交换两变量的值需要设置一个临时变量，其交换的过程是：首先把第一个变量的值送给临时变量，接着把第二个变量的值送给第一个变量，然后把临时变量的值（即第一个变量的原有值）送给第二个变量。如果不采用临时变量，直接把第一个变量的值送给第二个变量，再把第二个变量的值送给第一个变量，这样不能使两个变量的值交换，其原因是当把第一个变量的值送给第二个变量时，第二个变量中的原有值就丢失了。在BASIC语言中，若采用临时变量T来交换M和

N的值，则使用下面三条赋值语句可实现这个过程。

LET T=M:LET M=N:LET N=T

在算法的第（4）步，需要把M除以N后的余数送给R。那么在BASIC语言中如何来实现这一操作呢？我们可采用下面的赋值语句：

LET R=M-INT(M/N)*N

在这里INT(M/N)的值是不超过M/N值的最大整数，所以INT(M/N)*N为不超过M的N的最大整除数，因此，M-INT(M/N)*N为M除以N后的余数。如假定M=48，N=5，则：

M/N=9.6

INT(M/N)=9

INT(M/N)*N=45

M-INT(M/N)*N=3

从而得到3就是48除以5后的余数。

根据上面算法画出流程图如图1.1所示。

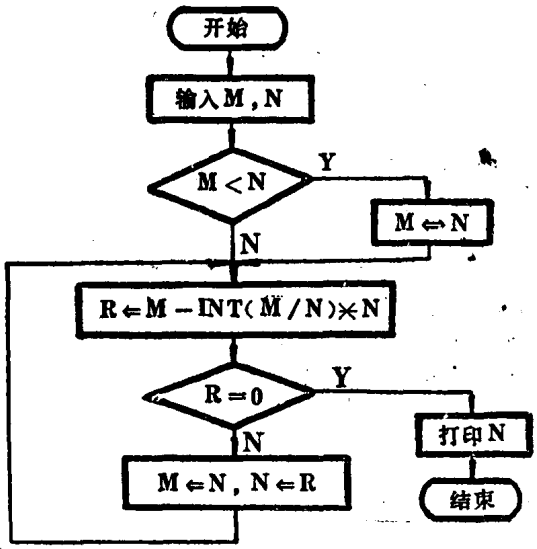

图 1.1

流程图中的“⟺”符号表示把两边变量值互换，“⟹”或“⟸”符号表示把等号一端表达式的值赋给箭头一端的变量。

按照流程图编写出程序如下：

```
10  INPUT M,N
20  IF M<N THEN T=M:M=
    N:N=T
30  R=M-INT(M/N)*N
40  IF R=0 THEN PRINT N:END
50  M=N:N=R
60  GOTO 30
```

假定从键盘上输入的两个正整数为120和84，则运行后打印出12，它就是120和84的最大公约数。在程序的运行过程中，当每次执行30语句后，M、N和R值的变化如表1.1所示。

表1.1

次　数	M	N	R
第一次	120	84	36
第二次	84	36	12
第三次	36	12	0

按标号写出该程序的执行过程为：10-20-30-40-50-60-30-40-50-60-30-40。

例2　根据分别求一批数据中正、负数之和的算法，画出流程图并编写程序。

已知分别求正、负数之和的算法如下：

（1）设置两个累加变量S1和S2，用S1累加正数之和，用S2累加负数之和，设置读数变量X（假定一批数据由DATA语句提供），用来从DATA语句中读数，设置结束数据处理的终止标志为0；

（2）分别给S1和S2赋初值0；

（3）读X；

（4）如果X＝0，则打印S1和S2并结束运行；

（5）如果X＞0，则把它累加到S1上，否则累加到S2上；

（6）转向第（3）步。

根据算法画出流程图，如图1-2所示：

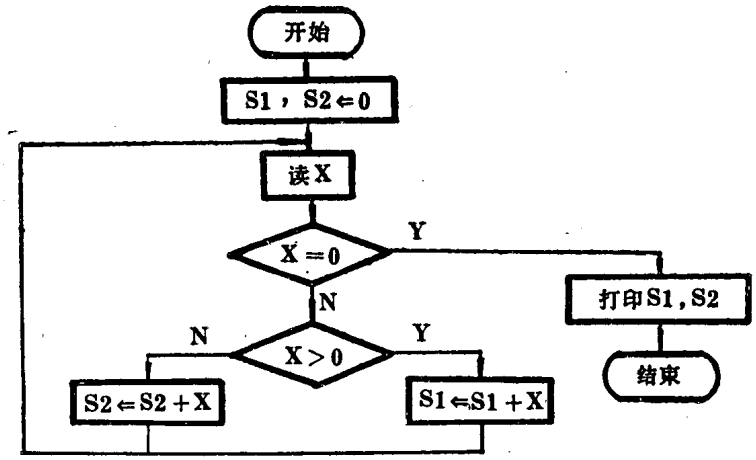

图 1.2

按照流程图编写出程序如下：

```
10  S1 = 0:S2 = 0
20  READ  X
30  IF  X = 0  THEN  70
40  IF  X>0  THEN  S1 = S1 + X  ELSE  S2 = S2 + X
50  GOTO  20
60  DATA  36, - 48,72,25, - 30, - 18, - 49,64,53,0
70  PRINT"S1 = "; S1,"S2 = ";S2
80  END
```

在编写与第一个条件框相对应的30语句时，其标有"Y"出口处的两个框，可以紧接着THEN后写出来，也可以拿到程序最后写出。当拿到程序最后写出时，只要暂时将THEN后面空着，待写出所对应的打印语句和结束语句后，再返回来把打印语句的标号填在THEN的后面。

另外，由于这个例子中没有给出具体的数据，所以60语句中的数据是任意设定的。

二、给变量赋初值

我们知道，对于编写不同问题的程序来说，需要设置不同的变量。如对于编写统计问题的程序来说，需要设置统计变量；对于编写累加问题的程序来说，需要设置累加变量（有时还需要设置计数变量）；对于编写累乘问题的程序来说，需要设置累乘变量；对于编写求最大值、最小值之类的程序来说，需要设置比较用的变量（简称比较变量）等。当然对于编写综

合性问题的程序来说，需要设置更多的变量。在利用这些变量进行数值计算或数据处理时，必须在程序开始时给它们赋初值。只有初值正确，才能得到正确的运行结果，否则将导致错误的运行结果。下面通过两个例子来说明如何给这些变量赋以正确的初值。

例1　编一程序计算$1+3+3^2+3^3+\cdots+3^{10}$的值。

显而易见，这是一个和式，是一个累加计算问题，所采用的算法如下：

（1）设置计数变量I和累加变量S；

（2）给I和S赋初值；

（3）计数变量I增1；

（4）累加变量S增加3↑I；

（5）如果I＜10，则转向第（3）步；

（6）打印S并结束运行。

在这个算法中，如何确定计数变量I和累加变量S的初值呢？首先假设I的初值为0，则通过分析第(3)步至第(5)步的循环过程（即反复执行的过程），可知和式中的第一项1没有被累加到S上，因此S的初值应为1。按照这个算法编写出的程序如下：

```
10  I=0:S=1
20  I=I+1
30  S=S+3↑I
40  IF I<10  THEN 20
50  PRINT  S
60  END
```

上面的算法是先计数后累加，若我们改为先累加后计数，则得到如下另一个算法：

（1）设置计数变量I和累加变量S；

（2）给I和S赋初值；

（3）把3↑I累加到S上；

（4）I增1；

（5）如果I≤10，则转向第(3)步；

（6）打印S并结束运行。

在这个算法中，首先假定I的初值为0，则通过分析第（3）步至第（5）步的循环过程可知：和式中的第一项已经被累加到S上了，因此S的初值应为0。若假定I的初值为1，则通过重新分析第（3）步至第（5）步的循环过程，可知和式中的第一项1没有被累加到S上，因此S的初值应为1。按照这个算法编写出程序如下：

```
10  I=0:S=0
20  S=S+3↑I
30  I=I+1
40  IF I<=10  THEN 20
50  PRINT  S
60  END
```

当然把10程序行改为10 I=1:S=1也是正确的。

通过分析这个例子可以看出，一个变量的初值与算法有关，有时也与其它变量的初值有

关。如例1中所使用的累加变量S，其初值与算法有关，同时也与计数变量I的初值有关。当算法确定后，计数变量I的初值确定后，S的初值就能够确定下来了。

例2　编一程序，求出一批数据中的最大值、最小值和总和。

我们先看一下如何求出25，48，36，72，64这五个常数中的最大值。若凭肉眼去看，则一眼就看出72是它们之中的最大值，但计算机不能对它们同时比较，只能一一进行比较，其比较过程是：首先将前两个数比较，取其大者（即48）再同第三个数比较，又取其大者（仍是48）同第四个数比较，又取其大者（即72）同第五个数比较，因为所有五个常数都比较过一遍，所以最后一次比较的大者(即72)就是它们之中的最大值。若不止5个常数，假定为N个常数，则需要反复进行(N－1)次比较后才能得到最大值。若我们设置一个比较变量，假定用M表示，M的初值取一批数据中的第一个数，再设置一个读数变量X或输入变量X，每次给X读入(对于READ语句而言）或输入（对于INPUT语句而言）一个新值后，都同M进行比较并将其大者存入M中，这样待所有的数据都比较完毕后，M的值就是它们之中的最大值。求一批数据中的最小值的方法也同求最大值的方法相同。求最小值时，需要设置一个存放最小值用的比较变量，假定用N表示，也让它的初值为第一个数，每次给X读入或输入一个新值后，都同N进行比较并将其小者存入N中，这样待所有的数据都比较完毕后，N的值就是它们之中的最小值。

根据以上分析，可知此题的算法如下：

（1）设置用于存放最大值、最小值用的比较变量M和N，设置用于求总和的累加变量S，设置输入变量(假定一批数据由键盘输入)X和数据输入的终止标志－1(假定待处理的数据中不包括－1这个常数)；

（2）输入第一个常数并送给M、N和S，作为它们的初值；

（3）输入X；

（4）如果X＝－1，则转向第(9)步；

（5）如果X＞M，则将X值送给M；

（6）如果X＜N，则将X值送给N；

（7）把X累加到S上；

（8）转向第(3)步；

（9）打印最大值M、最小值N和总和S；

(10) 结束。

在这个算法中，把第一个常数作为初值赋给累加变量S是必要的；若按照一般的惯例，盲目地给S赋初值0，将使得第一个常数没有被累加到S上，从而导致错误结果。

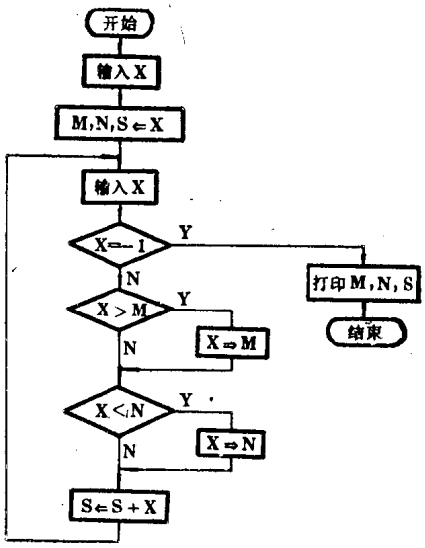

图 1.3

根据算法画出流程图如图1.3所示：

按照图1.3编写出程序如下：

```
10  INPUT  X
20  M=X:N=X:S=X
30  INPUT  X
40  IF X=－1  THEN 90
```

```
50  IF X>M  THEN M=X
60  IF X<N  THEN N=X
70  S=S+X
80  GOTO  30
90  PRINT  "M=";  M,"N=";  N,"S=";S
95  END
```

若将上述算法的第(2)步改写为：

(2) 给M、N和S赋初值；

也就是说，不输入第一个常数作为M、N和S的初值，那么它们的初值究竟如何确定呢？通过对改写后的算法分析可知，M、N将从第一个数开始，依次对一批数据中的每一个数进行比较，若M的初值大于一批数据中的所有数，则在比较过程中，M的值始终不变，最后打印出的M值是其初值而不是一批数据中的最大数；同理，若N的初值小于一批数据中的最小数，则在比较的过程中，N的值始终不变，最后打印出的N值也不是一批数据中的最小数。因此，M的初值最好取比一批数据中的所有数都小的数，N的初值最好取比一批数据中的所有数都大的数。如假定一批数据都为正数，则M的初值可取小于0的任一常数，N的初值应取很大很大的正数。因为对改写后的算法来说，S将从第一个数开始，依次累加所有的数，所以S的初值应为0。

通过这个例子分析也可以看出：变量的初值与算法有关，算法不同，其初值可能不同。

对于编程时经常使用的统计变量和累乘变量来说，其初值一般分别为0和1，这方面的例子就不列举了。

三、利用IF语句和GOTO语句构成循环

IF语句和GOTO语句配合使用，可以使某一程序段反复执行，构成循环。下面通过一些例子来说明如何使用它们构成循环，从而解决具体问题的。

例1，已知$y=\begin{cases} e^x\cos x & |x|\leqslant 1 \\ 3x^2+2 & |x|>1 \end{cases}$，x值由键盘输入，编一程序计算y值。

此题的算法很简单，其步骤如下：

(1) 输入x；

(2) 根据公式计算y；

(3) 打印y；

(4) 结束。

程序如下：

```
10  INPUT  X
20  IF ABS(X)<=1  THEN  Y=EXP(X)*COS(X)  ELSE  Y=3*X↑2+2
30  PRINT  Y
40  END
```

例2，仍采用例1中计算y的公式，而x的一批值由键盘输入，编一程序分别计算出对应的y值。

根据题意，我们可以在例1算法的基础上得到例2的算法，也就是说，通过使用IF语

句和GOTO语句，使例1算法中的第（1）步至第（3）步循环执行，直到输入终止标志（设定为9999）为止。例2的算法如下：

（1）输入x；

（2）如果x=9999，则结束运行；

（3）根据公式计算y；

（4）打印y；

（5）转向第（1）步。

程序如下：

```
10  INPUT  X
20  IF  X=9999  THEN  END
30  IF  ABS(X)<=1  THEN  Y=EXP(X)*COS(X)
        ELSE  Y=3*X↑2+2
40  PRINT  X, Y
50  GOTO  10
```

在这个程序中，IF语句和GOTO语句配合使用，使10—50程序段反复执行，构成循环，从而达到了利用同一计算公式进行多次计算的目的。

例3，从键盘上输入一组数据，其中第一个数据为某个学生的姓名，第二、第三个数据分别为该学生第一门和第二门课的考试成绩，试编一程序，若两门课的考试成绩的总和不小于170，同时第一门课的考试成绩不小于90时，则在打印机上打印出该组数据。

此题的算法也很简单，其算法之一为：

（1）设置X$、A、B三个变量存放从键盘上输入的一组数据；

（2）输入X$、A、B；

（3）如果A+B≥170，同时A≥90，则在打印机上打印X$、A、B；

（4）结束。

该算法对应的程序如下：

```
10  INPUT  X$, A, B
20  IF  A+B>=170  AND  A>=90  THEN  LPRINT  X$, A, B
30  END
```

其算法之二为：

（1）设置X$、A、B三个变量存放从键盘上输入的一组数据；

（2）输入X$、A、B；

（3）如果A+B<170，或者A<90，则结束运行；

（4）在打印机上打印X$、A、B。

这个算法中第（3）步的条件同上一算法中第（3）步的条件正好相反，也就是说，当算法1的条件为真时，算法2的条件必为假；当算法1的条件为假时，算法2的条件必为真。因此，这两个算法具有完全相同的功能，都能够在打印机上打印出满足题目要求的一组数据。

算法2所对应的程序如下：

```
10  INPUT  X$, A, B
20  IF  A+B<170  OR  A<90  THEN  END
```

```
30  LPRINT  X$, A, B
```

当计算机执行30语句后，因没有其它语句可执行，将自动由执行状态返回到命令状态。

例4，要求与例3相同，但要从键盘上输入若干组数据而不只是一组数据。

根据题意，我们可以在例3算法的基础上得到例4的算法。也就是说，通过使用IF语句和GOTO语句使例3中的第一种算法的第(2)步至第(3)步循环执行，直到输入终止标志(假定终止标志为("-1", 0, 0))为止。例4的第一种算法如下：

(1) 设置X$、A、B三个变量存放从键盘上输入的一组数据；

(2) 输入X$、A、B；

(3) 如果X$="-1"，则结束运行；

(4) 如果A+B≥170，同时A≥90，则在打印机上打印X$、A、B；

(5) 转向第(2)步。

该算法对应的程序如下：

```
10  INPUT  X$, A, B
20  IF  X$="-1"  THEN  END
30  IF  A+B>=170  AND  A>=90  THEN  LPRINT  X$, A, B
40  GOTO  10
```

若对例3中的第二种算法进行修改和扩充，则得到此题的第二种算法如下：

(1) 设置X$、A、B三个变量存放从键盘上输入的一组数据；

(2) 输入X$、A、B；

(3) 如果X$="-1"，则结束运行；

(4) 如果A+B<170，或者A<90，则转向第(2)步；

(5) 在打印机上打印X$、A、B；

(6) 转向第(2)步。

该算法对应的程序如下：

```
10  INPUT  X$, A, B
20  IF  X$="-1"  THEN  END
30  IF  A+B<170  OR  A<90  THEN  10
40  LPRINT  X$, A, B
50  GOTO  10
```

例5，已知有10个常数：82, 75, 54, 62, 93, 58, 74, 83, 49, 62，编一程序分别统计出小于60和大于等于60的个数。

设读数变量为X，统计小于60和大于等于60的变量分别为C1和C2，很显然C1、C2的初值应为0，若我们只考虑处理一个常数，则程序如下：

```
10  DATA  82, 75, 54, 62, 93, 58, 74, 83, 49, 62, -1
20  C1=0:C2=0
30  READ  X
40  IF  X<60  THEN  C1=C1+1  ELSE  C2=C2+1
50  PRINT  "C1=", C1, "C2=", C2
60  END
```

若考虑处理所有的常数，直到X读入－1为止，则只要在上面程序的基础上，在合适的位置加入IF语句和GOTO语句，使程序的处理部分构成循环即可。下面就是此题的一个完整程序：

```
10  DATA  82, 75, 54, 62, 93, 58, 74, 83, 49, 62, -1
20  C1=0:C2=0
30  READ  X
40  IF  X=-1  THEN  70
50  IF  X<60  THEN  C1=C1+1  ELSE  C2=C2+1
60  GOTO  30
70  PRINT  "C1=";C1, "C2=";C2
80  END
```

这个程序明显地由三个部分组成：20语句为赋初值部分，30—60语句为处理部分，70—80语句为打印结果和结束部分。

第二讲　循　　环

一、对FOR循环的进一步说明

1．循环变量的初值、终值和步长可以是任意算术表达式

在多数情况下，循环变量的初值、终值和步长均为常数，但在有些情况下，循环变量的初值、终值和步长可能为变量、函数或通过运算符连接的算术表达式。

例1，编一程序计算2至100之间所有偶数的乘积。

这是一个累乘计算问题，设循环变量为I，累乘变量为P。很显然，循环变量的初值、终值和步长应分别为2、100和2，累乘变量的初值应为1。为了在程序中实现累乘计算，只要在循环体中使用LET　P=P*I语句即可。当循环结束后，P的值就是2至100之间所有偶数的乘积。根据分析编写出程序如下：

```
10  P = 1
20  FOR  I = 2  TO 100 STEP  2
30  P = P*I
40  NEXT  I
50  PRINT  P
60  END
```

可以看出：在这个程序中，循环变量的初值、终值和步长均为常数。

例2，编一程序，从键盘上输入任意两个正整数，计算出它们之间所有自然数的乘积。

这也是一个累乘计算问题，为此设置循环变量为I，累乘变量为P。因为要从键盘上接收两个正整数，所以要再设置两个变量（假定为M和N）来存放它们。按照题目要求，要从M累乘到N，所以I的初值和终值应分别为M和N，但I的步长应如何确定呢？当M<N时，步长应取1；当M>N时，步长应取－1；当M=N时，无须进行累乘计算，直接打印出M或N值即可。因此我们用SGN(N－M)来表示I的步长，正好可以满足不同的情况。根据分析编写出程序如下：

```
10  INPUT  M, N
20  IF M=N THEN PRINT M:END
30  P = 1
40  FOR I=M TO N STEP SGN(N-M)
50  P = P*I
60  NEXT I
70  PRINT P
80  END
```

可以看出：在这个程序中，循环变量的初值和终值为变量，步长为函数。

例3，编一程序，从键盘上输入两个正整数M和N，并确保M>N，计算出（M－N）至（M＋N）之间所有自然数平方的和。

很显然，这是一个累加计算问题，为此设循环变量为I，累加变量为S。循环变量的

初值、终值和步长应分别为（M－N）、（M＋N）和1，累加变量的初值应为0。为了进行累加计算，只要在循环体中使用LET S＝S＋I↑2语句即可。当循环结束后，S的值就是（M－N）至（M＋N）之间所有自然数平方的和。还要注意：按照题目要求，要在程序开始判断M是否大于N，若不大于，应重新输入M和N的值。根据分析编写出程序如下：

```
10  INPUT M, N
20  IF M<=N  THEN 10
30  S = 0
40  FOR I=M-N TO M+N
50  S=S+I↑2
60  NEXT I
70  PRINT S
80  END
```

可以看出：在这个程序中，循环变量的初值和终值为带有运算符的算术表达式。

2. 循环变量的初值、终值和步长值不受循环体的影响

下面先看一个程序：

```
10  M = 5 : N = 3
20  FOR I=M+N TO M*N STEP M-N
30  M=M + I
40  N = N*I
50  NEXT I
60  PRINT M, N
70  END
```

运行这个程序，当执行10语句时，分别把5和3赋值给M和N；执行20语句时，首先计算出循环变量I的初值、终值和步长值，初值为M＋N即5＋3＝8，终值为M*N即5*3＝15，步长值为M－N即5－3＝2，然后把初值8赋给I，把终值15和步长值2保存起来（由此可知：循环体将被执行 $\mathrm{INT}\left(\frac{15-8}{2}\right)+1$ 次，即4次）；第一次执行循环体的30和40语句时，分别把M＋I的值（即5＋8＝13）和M*I的值（3*8＝24）赋给M和N，此时M的值为13，N的值为24；第一次执行50语句时，把I增加一个步长值2（注意：M和N的值虽然由循环体改变了，但步长值始终不会变，因为步长值已经通过执行FOR语句时计算并保存起来了。再者说，NEXT语句没有计算功能，它无法根据求步长的公式计算出新的步长值），此时I值为10，因不超过终值15（同理，M和N值虽然由循环体改变了，但终值也始终不会变），所以转去再次执行循环体，以此类推，直到I超过终值为止；循环结束后，执行60语句时，打印出M和N的值，由上面分析可知M的值应为5＋8＋10＋12＋14＝49，N的值应为3*8*10*12*14＝40320；执行70语句时，结束运行返回到BASIC命令状态。

通过对这个程序的分析，可以清楚地认识到：循环变量的初值、终值和步长值是根据相应的算术表达式，执行FOR语句时计算并保存起来的，不管在循环体中如何改变这三个算术表达式中的变量值，都不会影响它们。

3. 使用循环变量的限制

（1）只能使用简单数值变量作为循环变量，不能使用其它任何变量作为循环变量。如：

① FOR A(1)=A(2) TO A(3) STEP A(4)

② FOR B(I, J)=M TO N

这两条语句都是错误的，因为它们分别使用了一维下标变量和二维下标变量作为循环变量。

（2）在循环体内不允许给循环变量赋值。如：

```
      :      :
50  FOR  I=1  TO  10
      :      :
80  I=I+2
      :      :
100  NEXT  I
      :      :
```

在这个程序中，使用80语句是错误的，因为它在循环体内给循环变量I赋了值。在循环体外，循环变量可以作为其它变量来使用，不受此限制。

4. 使用循环变量的应用场合

（1）循环变量不出现在循环体内，只起到控制循环体执行次数的作用。

程序1：

```
10  S=0
20  FOR I=1 TO 10
30  INPUT A, B, H
40  C=(A+B)*H/2
50  PRINT A, B, H, C
60  S=S+C
70  NEXT I
80  PRINT S
90  END
```

程序2：

```
10  FOR I=1 TO 8
20  READ X
30  IF X<3 OR X>5 THEN 50
40  PRINT X;
50  NEXT I
60  DATA 2.4, 3.8, 5.2, 4.5, 2.1, 4.6, 3.7, 6.8
70  END
```

在这两个程序中，循环变量I都没有出现在循环体内，只起到控制循环体执行次数的作用，其中第一个程序的循环体将执行10次，第二个程序的循环体将执行8次。顺便指出，第一个程序的作用是：若每次输入的三个数据分别表示一个梯形的上底、下底和高的长度，则计算并打印出10个梯形的面积和它们的总面积；第二个程序的作用是：以紧凑格式打印出3和5之间的所有数。