第六章　BASIC語言介紹

6-4-3— 6-8

6-4-3　INPUT和PRINT 敍述

到目前爲止，都假設資料已進入程式處理，但有些資料必須由人提供（例如上一個例子裏的N%），所提供的資料到底要放在那個變數，什麼時候輸入，必須在程式裏言明；反過來資料輸出也必須在程式裏表達清楚（例如上一個例子裏的M%）。

輸入敍述的形式爲

INPUT　變數序列

變數序列包含一個或一個以上的變數，用逗點“，”分開，當執行到 INPUT 敍述時，螢光幕會顯示出一個問號“？”，表示正在等待資料的輸入。此時應把所需各項資料，依正確型態按順序以逗點隔開鍵入，然後按 RETURN 鍵，程式就繼續執行。

例。

```
150  INPUT X
200  INPUT CITY$,STATE$,COUNTRY$
```

程式執行到 150 出現問號後，鍵入一個數字，例如 56.8 或 3.75E−4，繼續執行 200，再出現問號等待輸入資料，這時可鍵入三個字串如“TAIPEI”，“TAIWAN”，“REPUBLIC OF CHINA”。

輸出敍述的形式爲

PRINT　式子序列

式子序列包含一個或一個以上的式子，用分號“；”分開，而式子可以是常數、變數或運算式子。當執行完PRINT 敍述，螢光幕會將式子序列裏每個式子的值顯示出來，而程式繼續往後執行。

例。

```
100  INPUT X%
200  PRINT X%;" SQUARE IS ";X%^2
```

執行到100 會出現問號，若鍵入 5 則X%的值變成 5，執行 200 後螢光幕顯示

```
 5  SQUARE IS 25
```

例。

```
10  PI=3.14
20  INPUT R
30  PRINT "THE RADIUS IS ";R
40  A=PI*R^2
50  PRINT "AREA OF THE CIRCLE IS **** ";A
```

執行到20出現問號，若鍵入7.4，則R的值爲7.4，執行30後螢光幕顯示

```
THE RADIUS IS 7.4
```

執行50後螢光幕顯示

```
AREA OF THE CIRCLE IS **** 171.946
```

40、50兩行可以合併成

```
50  PRINT "AREA OF THE CIRCLE IS **** ";PI*R^2
```

若要將所有的輸出一次顯示出來，可將30一齊合併寫成

```
50  PRINT "RADIUS IS ";R;"AREA OF CIRCLE IS ";PI*R^2
```

6-4-4 STOP和END敍述

程式執行完最後（行號最大的）一個敍述，若沒有跳回前面的敍述，就會自然終止而回到BASIC命令階段，但這不是很適當的停止方法，最好是用敍述讓程式停頓，停止敍述只是一個符號

STOP

停止敍述可以放在程式任何位置，一個程式也可以有好幾個停止指令。當程式執行到停止敍述時，會回到BASIC 命令階段，螢光幕即出現字樣表示停止在那一行。如果要繼續往下執行，可鍵入命令

CONT↙

會回到程式原來停頓的地方，繼續往下執行。

例.

```
10  INPUT A,B,C
20  K=A^2*5.3: L=B^B/.26
30  STOP
40  M=C*K+100: PRINT M

RUN ↙          (程式開始執行)
?1,2,3         (輸入資料)
BREAK IN 30    (停頓在30)
OK             (回到BASIC命令階段)
CONT ↙         (繼續執行)
115.9          (輸出資料)
OK             (程式執行完畢，回到BASIC命令階段)
```

STOP 敍述只是讓程式停頓，暫時回到BASIC 命令階段，若要眞正離開程式，可使用終止敍敍

END

終止敍述可放在程式任何位置，也可以有好幾個出現在同一程式裏，但一般好的程式設計只使用一個終止敍述，而且放在最後一行。

底下一個程式設計的例子，是用來複習這一節所提到的一些BASIC 基本敍述的使用方法。

例. 輸入任意個數字，求其平均值。困難之處在於不知道總共會有多少個數字輸入，解決的方法是在所有的數字輸入之後，多輸入一個其他的特別數字，譬如說以0當作資料的結尾。

以變數 INDATA 代表每次輸入的數字，以 SUM 來收集已經輸入數字的和，另外用變數 COUNT 記錄共輸入多少個數字，最後以 AVERAGE 輸出平均數，其演算法則如下：

設定 SUM 爲 0

輸入 INDATA

設定 COUNT 爲 1

若 INDATA 不等於 0 則重覆底下步驟：

SUM 更改爲 SUM + INDATA

輸入 INDATA

COUNT 更改爲 COUNT + 1

設定 AVERAGE 爲 SUM / (COUNT − 1)

輸出 AVERAGE

改寫成 BASIC 程式：

```
100  LET SUM=0
200  INPUT INDATA
300  LET COUNT=1
400  IF INDATA=0 THEN GOTO 500
410      LET SUM=SUM+INDATA
420      INPUT INDATA
430      LET COUNT=COUNT+1
440  GOTO 400
500  LET AVERAGE=SUM/(COUNT-1)
600  PRINT "THE AVERAGE IS ";AVERAGE
700  END
```

6-5 迴路敍述

在 6-4-2 節裏，曾利用 IF-THEN 和 GOTO 敍述組成迴路，若以 L_1，L_2 代表行號，其一般形式爲

L_1 IF 條件式子 THEN GOTO L_2

一些敍述

GOTO L_1

L_2 ……

條件式子如不成立時，就（再）進入迴路，一直到條件式子成立時，才離開迴路執行 L_2 那一行，所以該條件式子控制了整個迴路執行的次數。

如果迴路重覆執行的次數能夠預先確定，在 BASIC 裏有一種特別的迴路組合可以運用，一般把它叫做 FOR 迴路，是由一個 FOR 敍述、一個 NEXT 敍述和一些其他的敍述所構成的，其形式爲

FOR I = X TO Y

一些敍述

NEXT I

其中X和Y爲兩個數值式子，而變數I用來控制廻路重覆的次數，叫做控制變數（ control variable）。控制變數由原來X的值開始，廻路執行一次就加1，如果控制變數的值比原來Y的值大了，就結束廻路的執行，而到NEXT敍述的下一行繼續執行。因此原來X和Y的值是FOR 廻路的上下限，由它們可確定廻路重覆的次數。介於FOR和NEXT 兩個敍述間的敍述是主要的執行對象，叫做廻路本體（ loop body ）。

例。

```
10  INPUT N
20  FACT=1
30  FOR I=1 TO N
40      FACT=FACT*I
50      PRINT "FACTORIAL OF ";I;" IS ";FACT
60  NEXT I
70  END
RUN
?5
```

行號30到60是一個FOR廻路，I 是控制變數，由1逐次加1變到5（輸入資料N），每次經過廻路本體就把原來FACT的值乘上I的值，且把FACT 輸出。這個程式即是在計算從1到N的階乘。如果把50那一行去掉，而在FOR廻路之後才將FACT 輸出，則只顯示出N的階乘。

例。設計一個程式求 n 個數的最小值、最大值和平均值。

首先說明所使用變數的意義：

N　代表輸入資料的個數

X　輸入資料所存的地方

MIN　臨時最小值所存的地方，其最後的值是眞正最小值

MAX　臨時最大值所存的地方，其最後的值是眞正最大值

SUM　臨時的和，最後的值是所有輸入資料的和

AVE　平均值

開始時把第一個輸入的資料當成臨時的最小值和最大值，以後逐次輸入 $n-1$ 個資料，每次都拿剛輸入的資料和臨時的最小值最大值比較，以決定新的最小值和最大值，並且把和加以調整，結束重覆的步驟後再求平均值。演算法則如下：

輸入N

輸入X

設定SUM，MIN，MAX均爲X

重覆底下步驟N－1次：

　輸入X

　設定SUM爲SUM+X

　若X比MIN小則設定MIN爲X

　若X比MAX大則設定MAX爲X

求AVE

輸出MIN，MAX，AVE

很自然的可以把演算法則改寫成下面程式：

```
100   INPUT N
200   INPUT X
300   SUM=X: MIN=X: MAX=X
400   FOR I=1 TO N-1
410       INPUT X
420       SUM=SUM+X
430       IF X<MIN THEN MIN=X: GOTO 500
440       IF X>MAX THEN MAX=X
500   NEXT I
600   AVE=SUM/N
700   PRINT "MINIMUM ****";MIN
800   PRINT "MAXIMUM ****";MAX
900   PRINT "AVERAGE ****";AVE
1000  END
```

廻路本體可含有任何數目的敍述，甚至於可能出現其他廻路，而構成多層廻路（nested loop）。FOR廻路使用起來很方便，但有些要點要特別注意，茲列舉如下：

1. 控制變數不要在FOR廻路外面使用。
2. 控制變數在廻路本體可以使用，但不可用指定敍述改變它的值。
3. 在FOR敍述裏若Y後面爲空白，則表示控制變數的值每次增加1，若想採取其他改變方式，可在Y後面加上STEP Z，式子Z表示控制變數每次改變的量，例如

   FOR I=X TO Y STEP −2

   表示I每次減少2。

   若Z爲正，則X≤Y的條件在開始時必須成立；若Z爲負，則X≥Y的條件在開始時必須成立，否則整個廻路將被省略而跳過去。
4. X和Y在廻路本體裏可以加以改變，但控制變數的上下限以X和Y原來的值爲根據。
5. 要執行FOR廻路本體裏的敍述，必須經由FOR敍述，在FOR廻路外面不可用GOTO敍述直接進入廻路本體。

6-6 陣列的使用

在上一個例子裏，只透過一個變數逐次將資料輸入，在輸入另一個資料時，原來的資料就被取代而不見了。如果在求最小值、最大值和平均值之後，想計算每項資料和平均值的差，就必須留下所有的輸入資料，因此就必須有足夠的變數來容納它們。因爲輸入的資料可能很多，變數也就很多，書寫起來很不方便，倒不如給一個總稱，然後以註標（subscript）來區分它們。BASIC提供了一種基本而有用的變數寫法，將一群型態相同的資料並列在一起，稱它們爲一個陣列（array）。陣列必須在使用之前加以宣告，以決定容納資料的多少，宣告的方法是用一個DIM敍述：

DIM 陣列名稱（正整數序列）

DIM是dimension的簡寫，陣列名稱可以是任何變數符號，爲該陣列的總稱。正整數序列可以是一個或一個以上的正整數，以逗點分開。陣列的宣告等於是預約了一定數量的記憶空間，一個DIM敍述可以同時宣告數個陣列，把它們排在DIM後面。以下用一些例子來說明陣列的空間結構和使用方法。

例。

```
10  DIM A(5)
20  FOR I=1 TO 5
30      INPUT A(I)
40  NEXT I
```

A是陣列的名稱，在其名下有6個空間A(0)，A(1)，A(2)，A(3)，A(4)，A(5) 可供使用，但習慣都從A(1)開始，而不使用A(0)。用圖畫出來就像是6個方格子並列在一起：

A(0)	A(1)	A(2)	A(3)	A(4)	A(5)

程式將5項資料利用一個FOR迴路輸入，INPUT 敍述執行了5次，每次I的值都不同，也就是陣列的註標不同，因此輸入資料就存入了不同的位置A(1)，A(2)，A(3)，A(4)，A(5)。

例。

```
10  DIM A(4,5)
20  FOR I=1 TO 4
30      FOR J=1 TO 5
40          INPUT A(I,J)
50      NEXT J
60  NEXT I
```

DIM(4,5)中有兩個正整數4和5，表示這個陣列的空間在兩個方向上延伸，各有5（從0到4）和6（從0到5）個位置，所以在A的名稱下就有了5×6＝30個空間可供使用：

A(0,0)	A(0,1)	A(0,2)	A(0,3)	A(0,4)	A(0,5)
A(1,0)	A(1,1)	A(1,2)	A(1,3)	A(1,4)	A(1,5)
A(2,0)	A(2,1)	A(2,2)	A(2,3)	A(2,4)	A(2,5)
A(3,0)	A(3,1)	A(3,2)	A(3,3)	A(3,4)	A(3,5)
A(4,0)	A(4,1)	A(4,2)	A(4,3)	A(4,4)	A(4,5)

程式裏20到60是一個FOR迴路，而它的迴路本體30到50又是一個迴路，形成了兩層迴路，其執行的情況是這樣子的：首先I爲1，J由1變到5，因此經過一次外層迴路就輸入了5個資料，分別存入A(1,1)，A(1,2)，A(1,3)，A(1,4)，A(1,5)的位置，也就是陣列的第一列。然後I變成2，J再度由1變到5，再輸入5個資料到陣列的第二列A(2,1)，A(2,2)，A(2,3)，A(2,4)，A(2,5)，如此重覆到I爲4而輸入最後一列爲止。

例。全班有50個學生，考試科目爲國文、英文及數學三科，寫一個程式輸入每一個學生的三科成績，計算每個學生的平均分數，和全班各科的平均分數。

以陣列SCORE(50,3)存全班學生各科成績：SCORE(I,1)，SCORE(I,2)，SCORE(I,3)分別代表第I個學生的國文、英文及數學成績。陣列TOTAL(3)存各科全班總分，陣列STUDENT(50)存每個學生的三科平均分數，CHINESE，ENGLISH，MATH 各代表國文、英文、數學的全班總平均分數。

```
  5  DIM SCORE(50,3),TOTAL(3),STUDENT(50)
 10  FOR I=1 TO 3: TOTAL(I)=0: NEXT I
 20  FOR I=1 TO 50
 30      INPUT SCORE(I,1),SCORE(I,2),SCORE(I,3)
 40  NEXT I
 50  FOR J=1 TO 3
 60      FOR I=1 TO 50
 70          TOTAL(J)=TOTAL(J)+SCORE(I,J)
 80      NEXT I
 90  NEXT J
100  FOR I=1 TO 50
110      STUDENT(I)=(SCORE(I,1)+SCORE(I,2)+SCORE(I,3))/3
120  NEXT I
130  CHINESE=TOTAL(1)/50
140  ENGLISH=TOTAL(2)/50
150  MATH=TOTAL(3)/50
     - - - -
```

行號5的DIM敘述宣告程式中所使用的陣列容量，10把TOTAL各位置均設定爲0，20到40輸入所有的成績，50到90計算全班各科總分，100到200計算每個學生的平均分數，130到150計算全班各科平均分數。

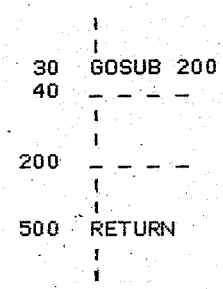
DIM敘述中的正整數表示陣列的容量，而在使用陣列時，( )中可放任何有意義的常數、變數或運算式子，其值表示註標，設計程式時要特別注意註標是在陣列容量範圍內。陣列爲一簡單而常用的資料結構(data structure)，廣被高階語言所採用，陣列如能配合迴路使用，更可發揮它的功能，值得用心練習。

6-7 副程式

一個程式的執行過程中，可能在某一個段落裏完成一項很具體的計算，例如求出餘數、交換兩個變數的內容、求出平方根等。這些段落本身可視爲功能獨立的程式片斷，相同的片斷或許會在同一程式裏重覆出現，或許會在不同的程式裏出現，如果一一的加以書寫，就顯得累贅，而且太佔程式的記憶空間。功能獨立的程式片斷，可以像寫一個完整的程式一般，分別給予構思，以寫成副程式(subprogram)，使用時再呼叫(call)調用。

副程式有兩種存在方式，一種是程式設計者在自己的程式裏寫下來的，只能在該程式使用，這是使用者自定的副程式(userdefined subprogram)；另一種是預先建立在語言系統裏的副程式，可供任何人在任何程式裏使用，是爲內建副程式(built-insubprogram)。

若從使用的觀點來看，副程式則可分成常式(subroutine)和函數(function)兩種。BASIC沒有內建常式，使用者自定的常式和主程式的寫法一樣，只是在結尾的地方多加一個RETURN敘述而已。且看一個程式形式：(如下頁)



行號30的敍述爲呼叫敍述，表示要調用以200爲開端的常式。200到500爲一常式，被呼叫時就開始執行，就像暫時把200到500整個片斷調去取代30那一行。常式執行到RETURN敍述那一行，等於30執行完畢，繼續執行40。

程式可以在不同的地方呼叫同一個常式，每次執行完常式後，都要回到原來呼叫的地方，繼續往下執行。一個程式依需要可有許多不同的常式，原則上都把它們放在END敍述後面，程式會顯得段落分明，容易閱讀、偵錯和更改。

例。 利用輾轉相除法求兩個正整數的最大公約數。

兩個正整數以變數A和B輸入，爲了保留原來的值，將A和B中大的值指定給X，小的值指定給Y，而以X和Y去做輾轉相除法。X除以Y，若整除則除數Y爲最大公約數，否則以除數爲新的被除數，以餘數爲新的除數，繼續以上的步驟，其演算法則如下：

輸入A，B

若A≥B則指定A給X且指定B給Y

否則指定B給X且指定A給Y

求X除以Y的商和餘數

若餘數不爲0則重覆底下步驟：

　指定Y給X

　指定餘數給Y

　求X除以Y的商和餘數

輸出A，B，Y

以Q代表商，R代表餘數，且上面演算法則中兩個求商和餘數的部分可以抽出來，當作一個常式處理，所得程式如下：

```
10  INPUT A%,B%
20  IF A%>=B% THEN X%=A%: Y%=B% ELSE X%=B%: Y%=A%
30  GOSUB 70
40  IF R%=0 THEN GOTO 50
41      LET X%=Y%
42      LET Y%=R%
43      GOSUB 70
44      GOTO 40
50  PRINT "G.C.D. OF ";A%;" AND ";B%;" IS ";Y%
60  END
70  LET Q%=INT(X%/Y%)
80  LET R%=X%-Q%*Y%
90  RETURN
```

70到90爲一常式，在程式裏有兩個地方（30和43）呼叫該常式。

BASIC 提供一些內建函數，使用時必須給適當的參數（parameter），常數、變數和運算式子都可當參數使用，它們的值就是求函數值時的參數值。基本算術函數，如三角函數、絕對值函數、平方根函數、亂數產生函數等，都有內建函數，讀者可參照第83頁，這裏只介紹其中幾個。

絕對值函數的符號是ABS，假如在程式中寫下ABS(A)，參數值是當時變數A的值，執行內建函數會把它的絕對值求回來，所以ABS(A) 就代表了那個求回來的值。平方根函數的符號是SQR，SQR(X+Y)代表X+Y值的平方根，當然X+Y的值不能小於0，否則會產生錯誤。

例。

```
10  LET X=ABS(Y-Z)
20  LET X=SQR(X)
```

10將Y－Z的絕對值指定給X，而20再把X的值開根號。如果將兩個敍述合併，就成爲

```
10  LET X=SQR(ABS(Y-Z))
```

亂數產生函數RND可以不給任何參數，呼叫它會產生一個介於0和1之間的數值，亂數在程式設計的應用上常用來模擬隨機發生的事件。

例。 寫一個程式模擬投擲銅板若干次，以統計出現正面（head）和反面（tail）的次數。如果亂數小於0.5代表正面，大於等於0.5代表反面，程式如下：

```
10  INPUT N
20  LET HEAD=0
30  FOR I=1 TO N
40      X=RND
50      IF X<0.5 THEN HEAD=HEAD+1
60  NEXT I
70  PRINT HEAD;" TOSSES ARE HEADS"
80  PRINT N-HEAD;" TOSSES ARE TAILS"
90  END
```

除了內建函數之外，BASIC 使用者可以利用DEF 敍述，在程式裏自己定義所需要的函數：

DEF 函數名稱(變數序列)=函數計算式

函數名稱的前兩個字母必須是FN，後面可接任何變數符號，如FNA，FNPOLYN 等都可作爲函數名稱。變數序列是一個或一個以上的變數，而以逗點分開。

程式中用DEF 敍述定義了函數後，即可像內建函數一般去使用它，代入參數的個數必須和DEF敍述中參數變數的個數一致，且與它們的順序相對應。

例。

```
100  DEF FNS(X)=(3.14159*X)/180
200  DEF FNSQUARE(X)=X^2
300  DEF FNDIST(X,Y)=SQR(FNSQUARE(X)+FNSQUARE(Y))
```

100定義了一個函數將度度量化爲弳度量，200定義平方函數，300定義平面上一個點到原點的距離。

例。 寫一個求二次多項式在不同點上值的程式。

設二次多項式爲AX^2+BX+C，先輸入A，B，C的值，然後逐次輸入不同的X，利用所定義的函數FNPOLYN求多項式的值。

```
10  DEF FNPOLYN(X)=A*X^2+B*X+C
20  INPUT A,B,C
30  INPUT N
40  FOR I=1 TO N
50      INPUT X
60      PRINT "X= ";X;" P(X)= ";FNPOLYN(X)
70  NEXT I
80  END
```

BASIC內建算術函數（均含一個參數）

ABS	取絕對值
ATN	取ARCTANGENT值，參數爲弳度
CDBL	轉換爲雙精確度值
CINT	四捨五入小數部分成整數
COS	求COSINE值，參數爲弳度
CSNG	轉換爲單精確度值
FIX	捨棄小數部分成整數
INT	求小於等於參數的最大整數
LOG	取自然對數值
RND	產生一亂數
SGN	參數＞0爲1，參數＝0爲0，參數＜0爲－1
SIN	求SINE值，參數爲弳度
SQR	求平方根
TAN	求TANGENT值，參數爲弳度

6-8 輸入及輸出格式

利用INPUT敍述輸入資料，必須在執行到INPUT敍述時，即時從終端機將資料鍵入。另外還有一種輸入資料的方法，就是將需要輸入的資料預先在程式中列出，方式是以READ和DATA敍述配合使用，其格式分別爲：

READ 變數序列

DATA 常數序列

其中變數序列是一個或一個以上的變數，常數序列是一個或一個以上的常數，都是以逗點分開。READ敍述表示要輸入資料到變數裏，而輸入的資料就是由DATA敍述所列出的常數。

DATA敍述所列常數的個數並沒有限制，各別的DATA敍述可以在程式的任何位置，然而不同的DATA敍述裏所有的常數，却可以把它們想像成一連串有待輸入的資料，由READ敍述逐一去讀進，對應的變數和常數的資料型態必須一致。

例。

```
10  READ A,B
20  READ X,Y,Z$
30  PRINT A,B,X,Y,Z$
40  DATA 1,2
50  DATA 3,4,"FIVE"
60  END
```

上面程式是將在行號40的常數1，2輸入到A，B兩個變數裏，而在行號50的常數3，4，“FIVE”輸入到X，Y，Z$三個變數中，然後輸出5項資料1，2，3，4，“FIVE”。如果把40，50兩行合併成

```
40  DATA 1,2,3,4,"FIVE"
```

結果完全一樣。如果把40，50兩行改成

```
40  DATA 1,2,3
50  DATA 4,"FIVE"
```

結果還是一樣，主要是DATA敍述（不管在什麼位置）必須按照順序供給足夠的資料。

例。

```
80  FOR I=1 TO 10
90      READ A(I)
100 NEXT I
110 DATA 3.08,5.19,3.12,3.98,4.24
120 DATA 5.08,5.55,4.00,3.16,3.37
```

上面的程式片斷將10個資料輸入到陣列A中，例如A(1)會有3.08，A(7)會有5.55。如果將兩個DATA敍述移到80之前，結果仍然一樣。

如果用心觀察本章中程式例子的輸出情形，不難發現螢光幕上輸出資料之間都用一個空白位置分隔（注意正號雖不顯示出來也佔一位置），這是因爲PRINT敍述中的式子序列，以分號隔開的原故。這樣子顯示出來的資料，常顯得上下不對齊而很雜亂，以下要介紹一些以格式（format）輸出資料的方法。

最簡單的格式就是將一行分成幾個等長的區段，每項資料都在一區段的最左端出現。只要把PRINT敍述中的式子序列用逗點分開，即可達到這樣的效果。

例。

```
10  X=5
20  PRINT "OUTPUT1","OUTPUT2","OUTPUT3","OUTPUT4"
30  PRINT X+5,X-5,X*(-5),X^5
40  END
RUN
OUTPUT1         OUTPUT2         OUTPUT3         OUTPUT4
 10              0              -25              3125
```

輸出的資料中，10，0，3125前面的正號雖然不顯現，仍然佔有一個位置，如果仔細檢查，可以發現每項資料都在14個位置的區段裏。

如果要把輸出資料以某種特定格式顯示出來（例如定點數的位數和小數點位置、數值之前附加特殊符號、以浮點數方式輸出等），可以使用PRINT USING敍述：

PRINT USING “格式字串”；式子序列

式子序列以逗點分開，格式字串則是一些特別符號，以雙括號括起來，表示輸出的特定格式，常用的格式字串介紹如下：

(1) “＃”表示顯示數值之位數，小數點以後超出的位數以四捨五入方式捨棄。

(2) “．”指定小數點所在位置。

例.

```
PRINT USING "##.##";.78
 0.78
PRINT USING "###.##";987.654
987.65
PRINT USING "##.##    ";10.2,5.3,66.789,.234
10.20     5.30    66.79     0.23
```

(3) “＋”規定輸出數值包含正負號。

例.

```
PRINT USING "+##.##    ";-68.95,2.4,55.6,-.9
-68.95    +2.40    +55.60    -0.90
```

(4) “＊＊”可以使數值較格式多出兩個位置，不足的位數補以＊號。

例.

```
PRINT USING "**#.#    ";12.39,-0.9,765.1
*12.4    *-0.9    765.1
```

(5) “＄＄”使數值前加印一個＄號，在＄號之前則多空出一位。

例.

```
PRINT USING "$$###.##";456.78
 $456.78
```

(6) “＊＊＄”聯合“＊＊”及“＄＄”之功效。

例.

```
PRINT USING "**$##.##";2.34
***$2.34
```

(7) “∧∧∧∧”表示以指數格式輸出，其指數部分爲E＋xx，若不另規定正負號格式，正值之前爲空格。

例.

```
PRINT USING "##.##^^^^";234.56
 2.35E+02

PRINT USING "+.##^^^^";123
+.12E+03

PRINT USING ".####^^^^";888888
 .0889E+07
```

以上所討論的輸出敍述，PRINT敍述成PRINT USING敍述，都只能將資料顯示在螢光幕上；若把這兩種敍述中的PRINT改成LPRINT，則輸出資料就可從列表機印出，以便有更長的時間去研判分析程式執行所獲得的結果。

第六章　問　　題

1. 將求三個數目最大值的程式輸入執行，發現無誤後存檔。把存檔的程式再輸入，而改成可以求四個數目最大值的程式，再加以執行，發現無誤後用列表機把程式印出。
2. 將計算階乘函數的程式輸入。
   (1) 輸入大於50的整數給N，觀察程式執行的情形，並加以解釋。
   (2) 變數FACT改爲FACT％重做一遍(1)部分。
3. 寫一個解二次方程式的程式，輸入不同的係數加以執行。
4. 寫一個程式，以便在一個 $n \times n$ 矩陣上做如下的運作：
   (1) 計算每行的平均數，並將之輸出。
   (2) 計算每列的平均數，並將之輸出。
   (3) 將第一行和第一列交換，第二行和第二列交換，……，第 n 行和第 n 列交換。