APPLE-Ⅱ(讲义)

——图形及游戏

圖形、遊戲與其他

老師札記 15

一些捷徑

本課涵蓋：

?	用來PRINT
LET	可以省略
:	用在同一列的各個敍述之間
INPUT	與提示訊息一齊使用

前面的衝刺已經結束了，我們曾談過RND和如何將程式保存在磁碟上，以及其他的事情，所有的這些要素都是爲了讓學生能眞正寫作程式而設計的。

冒號在程式中是用來將幾個敍述放在同一列上使程式看起來短一些並且清晰一些，這種程式列應該包括一些具有共同特性的敍述。

冒號也可能將程式弄得一團糟，有一些敍述是用GOTO跳過去的。因此若您要這種程式列中間的一個敍述，您將在執行這程式時就會得到一個錯誤訊息。

有一種錯誤是有經驗的程式設計師也偶而會犯的，那就是將一個敍述移到含有 IF 的某一列的後面去，這麼做會改變程式的邏輯，因爲該敍述只在 IF 條件爲眞時才執行。

另一方面，冒號在Applesoft裏面允許人們放入一個 IF 和數個在它後面的敍述組成一個副程式（subroutine），這樣可以不需利用 GOTO 也能到達程式的擴充部分：一個較短而又不混亂的程式；所以，冒號在使程式的敍述簡潔方面是一個很有威力而又有必要的工具。

當使用 INPUT 而沒有隨同的提示訊息時，螢光幕上將顯示"？"符號後面跟著閃爍的輸入游標；當 INPUT 與提示訊息一齊使用時，"？"不會出現而只顯現閃爍的游標，所以，如果提示訊息是一個問句，它應該以一個問號作結尾。

問答題：

1. "？"提供什麼捷徑？
2. 您如何知道LET這個字從一個LET指令中不見了？
3. 一個 INPUT 指令可以使用有引號括起的提示訊息，什麼標點符號必須跟在那引號後面？
4. 爲什麼有時用冒號將兩個敍述放在同一列上是比較好的？
5. 下面這些列中有什麼錯誤？

```
10 REM BEGINNING:GOTO 1000
10 GOTO 50:S$="FAST"
```

第 15 課

一些捷徑

個印字的捷徑

我們不再需要打入 PRINT，而只打入一個問號即可。

輸入：

```
10 ? "HI"
LIST 10
```

電腦會用一個PRINT來取代問號。

一個LET捷徑

下面兩列處理同樣的事情：

```
10 LET A=41 和 10 A=41
```

下面兩列也做同樣的事情：

```
20 LET B$="HI" 和 20 B$="HI"
```

您可以在LET 敍述中省略LET 這個字！不論什麼時候，只要某一列以一個變數的名字開頭而跟著一個“＝”號，電腦便知道您指的是LET。

一個INPUT捷徑

以前是：

```
10 PRINT "ENTER YOUR NAME"(輸入您的名字)
20 INPUT N$
```

現在，您可以這樣：10 INPUT "ENTER YOUR NAME"; N$

在" ENTER YOUR NAME"(輸入您的名字）的提示訊息與變數之間插入一個分號。

例如：

```
10 INPUT "AGAIN ? <Y OR N>"; Y$ (再玩一次？)
20 INPUT "LOCATION"; X,Y (位置)
30 INPUT "MONTH, DAY, YEAR";M$,D,Y (月，日，
                                    年)
```

一個LIST的捷徑

下面是LIST（列出）指令的五個用法：

LIST	列出整個程式
LIST 48	列出編號 48 的那一列
LIST 50—75	列出編號 50 至 75 的每一列
LIST —27	列出開頭至編號 27 的每一列
LIST 90—	列出編號 90 至末尾的每一列

一個冒號的捷徑

幾個敍述可以用冒號“：”分開而放在同一列上。這樣可以節省空間。

以前是：

```
10 Q=17*3
20 R=Q+2
30 PRINT R
```

現在您可以寫成　10 Q=17*3:R=Q+2:? R

在記憶體中，這一列是相當於下面這樣子的：

```
10 Q=17*3:R=Q+2:PRINT R
```

什麼時候使用冒號

在下面時機可以使用：

1.使程式更清楚。

將類似的敍述放在同一列上。例如：

不再用：

```
10 X=0
12 Y=0
14 Z=0
```

而寫成：

```
10 X=0:Y=0:Z=0
```

2.使程式更簡短。

3.在程式列後頭加上REM的時候。

例如：

```
40 H=X+Y/66 : REM H IS THE HEIGHT
```

（H代表高度）

在IF指令之後的冒號

您可以利用冒號使 IF 敍述更簡潔。

不用冒號時：

```
50 IF A=0 THEN GOTO 80
60 B=Q
62 C=B*D
64 FLASH
66 PRINT "WRONG"
80 FOR . . .
```

使用冒號時：

```
50 IF A<>0 THEN B=Q:C=B*D:FLASH:PRINT
"WRONG"
80 FOR . . .
```

所有在“A = 0爲眞”路徑上的指令都放在THEN之後的同一列。

小心！

不要把不相干的東西放在 IF 列的後面。

例如：

```
35 IF A=B THEN PRINT "ALIKE"(相像)
40 Q=R
```

與下面一列是不同的：

```
37 IF A=B THEN PRINT "ALIKE":Q=R
```

因爲編號 40 那一列的Q＝R是一定會做到的，不管A＝B是眞或假。但是在編號 37 那一列裏的Q＝R則只在A＝B爲眞時才執行。

使用冒號常犯的錯誤

若一列中有 REM 或 GOTO，則它們必須要寫在該列最後，因爲跟在它們後面的任何東西都將被忽略掉。

正確：`35 P=3:REM P IS THE PRICE ( P代表價格 )`

錯誤：`35 REM P IS THE PRICE:P=3`

因爲電腦忽略掉REM 之後的每一樣東西。

正確：
```
40 R=P+1:GOTO 88
42 S=3
```

錯誤：`40 R=P+1:GOTO 88:S=3`

指令，敍述與程式列

指令是用來告訴電腦做什麼事的。截至目前爲止，我們用過的指令有:

HOME, PRINT, NEW, RUN, LIST, REM, INPUT, LET, GOTO, FLASH, INVERSE, NORMAL, IF, SPEED, SAVE, LOAD, CATALOG, DELETE

指令被用在編了號的列中之後便就叫做“敍述”，如果單獨使用。它們永遠稱爲“指令”。

輸入：　HOME　　我們說；我們已“輸入一個指令”了。

但是如果我們在一個程式中將它寫成：

20　HOME　我們說編號 20 的這一列有一個“敍述”，那就是HOME 指令。

有些列含有若干個敍述，各個敍述之間是以冒號分開的。

```
30 HOME:PRINT:LET Z=55
```

就是一個含有三個敍述的程式列。

習題15：

1. 寫一段至少利用一次上面介紹的各種捷徑的程式。
2. 寫一個“度假”的程式，它會詢問您要花多少錢，然後它會告訴，您應該去那裏或您應該做什麼。
3. 寫一個“亂整”的程式，它會詢問您的姓名，然後以一個可笑的口吻說您瘋了。這個程式從您預先存入的一些話中隨意選出一句，並把它印在您的名字後面。

老師札記 16

在螢光幕上隨意移動：VTAB與HTAB指令

HTAB和VTAB指令是用來把輸出游標移到螢光幕的任何地方，這些令可用來彈性整理程式和圖形。

前面幾課中討論過的TAB指令是用在PRINT指令中的，但是HTAB VTAB則正好相反，它們是用在PRINT指令之前的。

為了有效運用這些指令，螢光幕需要看成一個水平方向有40格而垂方向有24列的格子點；這個觀點在稍後討論低解析度螢光幕的圖形上會度提到。

答題：

1. 如果您要印出第12列上的下一個字，您要用什麼指令？
2. 如果您要把下一個字母印在第6列，向右移進去12格，您要用那兩個指令（用分號隔開）？
3. 在第2題的答案中，該列中的兩個指令的先後次序有關係嗎？
4. 說明如何將兩個字“FAT”和“CAT”印在同一列；“CAT”從第25格開始先印，然後“FAT”從第5格開始印。

第 16 課

在螢光幕上隨意移動：VTAB 與 HTAB指令

/TAB指令

在螢光幕的垂直方向上有 24 列的打字空間，VTAB（垂直定位）利用1到 24 的任意一個數目來選擇一列。

執行下面程式：

```
10 REM VTAB DEMO (VTAB示範)
15 HOME:SPEED=1
10 VTAB 10 :PRINT "LINE 10 FIRST"
20 VTAB 1  :PRINT "LINE 1 NEXT"
30 VTAB 24 :PRINT "LINE 24 LAST"
40 SPEED=255
```

與這個程式：

```
10 REM ::: WHICH LINE ::: (那一列)
20 HOME
30 INPUT "WHICH LINE";L
40 VTAB L
50 PRINT L;" HERE. ";(這裏)
60 GO TO 30
```

如果VTAB後面的數目是零或大於 24 的數目，電腦將印出"ILLEGAL QUANTITY ERROR IN LINE 40." （錯在編號40的列上有不含規定的數量）。

HTAB指令

HTAB（水平定位）指令告訴電腦要從某一列的什麼地方開始印字；這要使用 1 到 40 的數目。

它有點像您在前面學過的 TAB指令；TAB指令是用在 PRINT 指令裏面，但是HTAB 指令却是單獨使用的。

試試下面程式：

```
10 REM --- HTAB DEMO --- (HTAB示範)
12 HOME:SPEED= 1
20 VTAB 12
20 VTAB 12: HTAB 35:PRINT"HERE " (這裏)
30 VTAB 12: HTAB 5:PRINT"THEN HERE" (然後那兒)
40 SPEED=255
```

與這個程式：

```
10 REM === WHICH ROW, COLUMN === (那一列，行？)
20 HOME
30 VTAB 1:HTAB 1:INPUT "WHICH ROW, COLUMN?
";R,C
35 VTAB 1: HTAB 1:PRINT"
40 VTAB R: HTAB C:PRINT "*"
45 FOR T=1 TO 2000:NEXT T
60 GOTO 30
```

按下 RESET 鍵可以使它停止。

編號 30 的那一列要您打進兩個由逗點分開的數目，如果您只打進一個數目便按下 RETURN 鍵，電腦會印出：

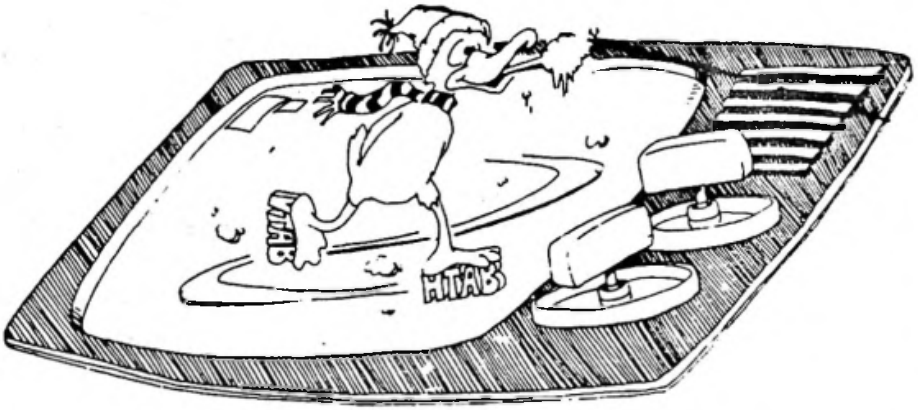
? ?

來告訴您要輸入另一個數目。

如果您打進太多個由逗點分開的數目，電腦會印出：

EXTRA IGNORED （多的，忽略掉了）

這表示多餘的數目根本不會用到。



螢光幕上的圖形

螢光幕就像一張畫圖紙，數目由左上角開始算起，垂直方向由 1 到24，以變數名字 Y 來代表它們，水平方向由 1 到 40 ，以變數名字 X 來代表

們。執行下面的程式：

```
10 REM --- GRAPHS --- (圖形)
20 HOME
25 PRINT"GIVE ME SOME X AND Y'S. "
27 PRINT" X FROM 1 TO 40, Y FROM 1 TO 24"(給我一些X和Y。)
28 PRINT
30 INPUT "ENTER 'C' TO CONTINUE ";X$ (輸入"C"以便繼續)
32 HOME
35 INPUT X,Y
40 HTAB X :VTAB Y
50 PRINT "*"
55 HTAB 1 :VTAB 1 :PRINT"
60 HTAB 1 :VTAB 1 :GOTO 35
```

題16：

1. 利用RND( )函數在螢光幕上任選一點寫下你的名字，利用它將您的名字在螢光幕上印很多次。
2. 寫一段可以在螢光幕的任意一點印出"HERE"（這裏）的程式。每當這個字被寫下之後便擦掉它，然後再寫一個。您可以把印字的部份放在一個延遲迴圈中，以免"　" HERT"跳得太快。

老師札記 17

FOR-NEXT 迴圈

本課要說明如何利用FOR,NEXT 和 STEP命令構成迴圈，這個迴圈是由兩個敍述組成，一個是以FOR開頭，而另外一個是以NEXT開頭。這兩個敍述之間可以有好幾列，但是它們仍然是強烈地相互依賴的。這一點對您的學生可能有點困擾，前面教過的延遲迴圈將有助於了解FOR和NEXT的結合方式，我們將留待以後才說明在迴圈中間重覆一系列命令的用法。

套疊式(nested)迴圈是使用一個包含層層相疊的延遲迴圈的例子來說明的。

有一些比較難懂的問題沒有包括在本課中，不過，我們遲早是會碰到它們的。迴圈永遠是至少要通過一次的，因爲出口(exit)測驗是在NEXT敍述中完成，而這個敍述是一定要通過迴圈之後才能碰到。

FOR敍述只有在開始進入迴圈的時候才會計算一次數值；它將迴圈變數的開始值存入一個變數儲存區；在那裏，它就好像任何其他的數字變數一樣。 STEP（改變的級距），最後數值，和 FOR 之後的第一個敍述的地址(address)都被寄放在一個堆疊(stack)中，從這個時候開始，所有的迴圈動作都發生在 NEXT 指令裏。一旦到達 NEXT ，迴圈變數便以 STEP 的數值增加，且與最終數值比較；如果迴圈變數大於最後數值（或當負STEP 時小於它），NEXT 便將控制權交給它本身之後的敍述；否則它便將控制權交給FOR指令之後的敍述

因爲BASIC 把迴圈變數與其他任何變數一樣看待，因此它可以在迴圈本身或在迴圈中間改變；要改變它的數值必須非常小心，因爲它也會被NEXT 改變，而NEXT 又用它去決定迴圈是否結束了。

在程式執行時跳進一個迴路中間的結果通常都是很悲慘的；但在NEXT產生出口以前跳出一個迴圈却是件平常的事，但是有些情形下（特別是牽到副程式的地方）會很難發現錯在那裏。

問答題：

1. 寫一個可以從 0 印到 20 的 迴圈。
2. 寫一個程式迴圈讓它從 30 開始每次以 2 的間隔倒數至 20。
3. 以一對套疊式迴路的方式寫一個程式，讓它印出 100、200、300 並在它們之間各用單獨的一列分別印出 1 ， 2 、 3 ， 4 ， 5 。

第 17 課

FOR-NEXT 迴圈

記得延遲迴圈嗎？電腦從 1 數到 2000 ，然後再重頭繼續。

```
30 FOR I=1 TO 5:NEXT I
```

電腦不僅僅只會這樣而已，它可以在計數時做其他的事情。

執行這個程式：

```
10 REM COUNTING （計數）
20 HOME
30 FOR I=5 TO 20
40 PRINT I
50 NEXT I
```

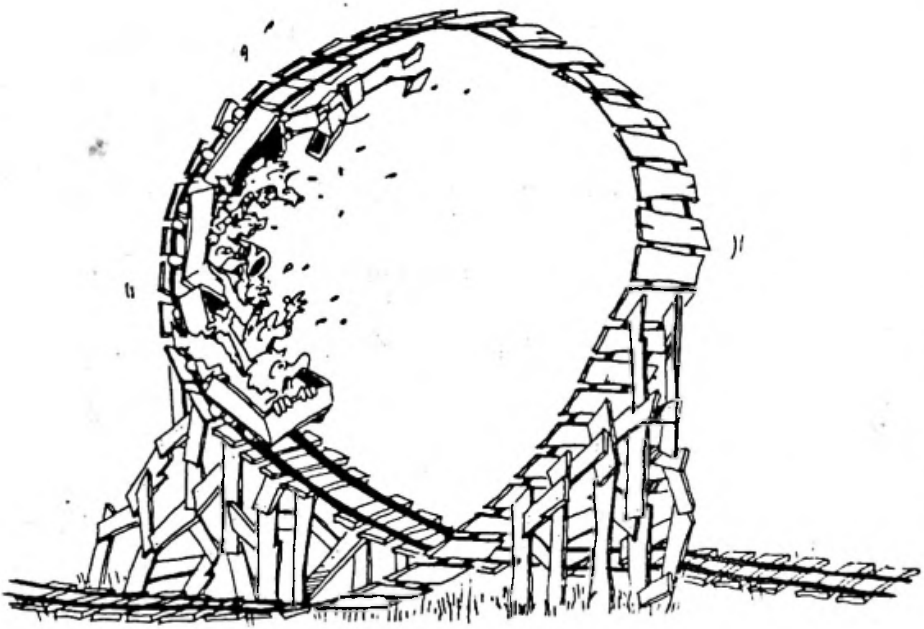
迴圈可以從任何數目開始，並在任何較高的數目上停下來；您不妨試著以下面的方法將編號 30 的那一列改成：

```
30 FOR I=100 TO 101
30 FOR I=-7 TO 13
30 FOR I=1.3 TO 5.7
```

在您的程式中作記號

用箭頭指出迴圈在那裏：

```
10 REM ON PAPER （在紙上作記號）
20 HOME
30 FOR I=0 TO 7
40 PRINT I
50 NEXT I
```



STEP指令

電腦在前面的程式中是以 1 增加的，爲了使它以 2 增加，可以把編號30的那一列改爲：

```
30 FOR I=10 TO 30 STEP 2
```

STEP指示每一次所要改變的量。

習題17A：

1. 令電腦從0開始每次間隔5計數到100。

倒數迴圈

您可以利用負的STEP讓電腦倒數。

試試這個程式：

```
10 REM *** APOLLO 11 *** (阿波羅11號)
20 HOME:SPEED=100
30 PRINT"T MINUS 12 SECONDS AND COUNTING"
40 FOR I=11 TO 0 STEP -1          (T-12秒開始計數)
50 PRINT I:PRINT CHR$(7)
60 FOR J=1 TO 740:NEXT J:REM TIMING LOOP (計時迴圈)
70 NEXT I
75 INVERSE (所有引擎運轉中，準備發射)
80 PRINT "ALL ENGINES RUNNING, LIFT OFF."
82 PRINT "WE HAVE A LIFT OFF."
84 PRINT "32 MINUTES PAST THE HOUR."(整時過32分鐘)
86 PRINT "LIFT OFF ON APOLLO 11." (阿波羅11號昇空)
99 SPEED=255: NORMAL
```

編號 60 的那一列是計時迴圈。

套疊式迴圈(NESTED LOOPS)

在上面的程式中，我們用一個迴圈包含了另一個迴圈；在外圍的迴圈由編號 40 的那一列開始到編號 70 的一列結束，裏面的迴圈則在編號60的那一列中，這就是“套疊式”迴圈”，它就好像是一種一層套在另一層裡面的結構。

迴圈變數

爲了確保每一個FOR指令都知道那一個NEXT命令是屬於它的，NE
指令是以迴圈變數名字作爲結束的。請看編號60的那一列：

```
60 FOR J=1 TO 740:NEXT J
```

此地J是迴圈變數。至於從編號 40 的那一列開始的迴圈：

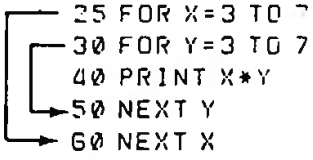
```
40 FOR I=12 TO 0 STEP -1
   . . .
70 NEXT I
```

I就是迴圈變數。

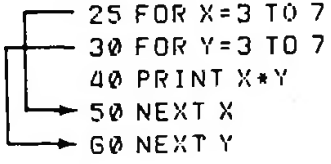
不良的套疊式迴圈

內圈的迴圈要一直保持在內圈

正確的：



錯誤的：



習題17B：

1. 寫一段可以印 15 次您的名字的程式。
2. 現在，讓它在螢光幕上每次隔開兩格並向下降一列呈斜向鋸齒狀印出您的名字。在迴圈中利用 TAB 。
3. 現在，讓它由左下角開始慢慢向上印 24 次您的名字。在迴圈中利用 VTAB 指令。
4. 現在，讓它在一列上寫下您的名字，在另一列上寫下您的朋友的名字這樣地一共每一個名字都印 5 次爲止。

老師札記 18

編修與執行模式,與計算機

本課要說明電腦的編修模式（EDIT MODE）和執行模式（ RUN MODE）。我們將這些最基本的材料放在這麼後面才討論是因爲它們是相當抽象，而且我們希望一口氣教完BASIC指令的精髓，不要因爲講解這些材料而緩慢下來。

不過，我們仍然顧慮到您可能會想要把本課插在前面的課程中，因此本課只使用了下面兩個指令：

PRINT 和 RUN

這些模式還有一些其他的名稱：

編輯模式：　指令模式　立即模式
　　計算機模式　直接模式

執行模式：　延緩的（deferred）執行模式

編修模式是電腦使用者的“本壘板”。在編修模式下，您可以輸入一列，所有的字元都進入輸入緩衝區（input buffer）（$02記憶頁），它是一個多達256字元的記憶區。

當 RETURN 鍵被按下後，電腦就會檢查這一列是否以數目字開頭，如果是，它就把這列存放在程式的記憶空間裏，找到適當的位置使各列按

號碼順序排列；如果這列並不是用數目字開頭的，電腦就立刻執行剛剛被輸入緩衝區的這一列，這種情形最普通的就是這列包含像HOME或RUN之類的單一指令。立即模式是一個非常有威力的方式，主要在於即使是一列相當長的程式也可以執行；這個特點在程式設計時期和除錯時期都是很方便的，在程式設計期間，有關設計的一些運算在輸入程式各列之前就可以完成了。

問答題：

1. 電腦在“執行模式”期間做些什麼事？
2. 您如何分辨電腦是否處於“編修模式”？
3. 在編修模式下，您可以做那些事？
4. 如果您輸入的一列是以數目開頭的，電腦將做些什麼事？
5. 如果您輸入的一列沒有包含該列的號碼，電腦將做些什麼事？

第 18 課

編修與執行模式,與計算機

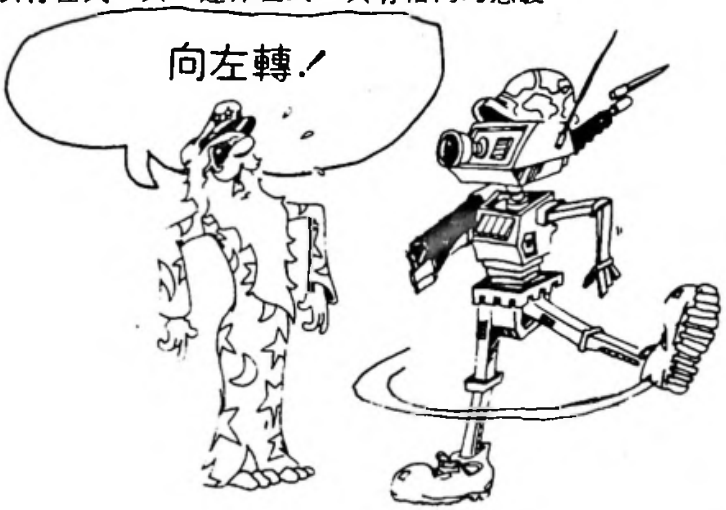
請輸入：　NEW

HOME　現在，您已準備好可以開始上課了。

執行與運作

我們所謂的“執行”（execution）就好像軍人在做“向左轉”的指一樣，而不是行刑班在“處死”死刑犯。（註：在英文中，execution作執行與處死解釋。）

“執行程式”與“運作程式”具有相同的意義。



延緩的執行(DEFERRED EXECUTION)

輸入並執行下面程式：

```
10 PRINT "HI"
```

這是令程式運作的最普通方法，我們稱之爲“延緩的執行”。

在這種情況下，電腦會一直等到您輸入“RUN”（執行）指令之後會開始執行的。

延緩執行的規則：

如果某一列是以數目開頭的，它就被放進記憶體內，該列便成爲記憶體內程式的一部分，這程式可用“RUN”指令去執行它。

立即執行

這是一條捷徑。照下面方式輸入（不要在前面加上列號碼）；

```
PRINT  “HI”
```

這一次電腦不會等待您輸入RUN便立刻印出“HI”了。這就是所謂“立即執行”。

立即執行的規則：

如果某一行並不是用數目開頭，電腦便立刻執行這一道指令（只要按下 [RETURN] 鍵）。

試試下面這個較長的例子：

```
FOR I=1 TO 20:PRINT I:NEXT I:PRINT:PRINT"DONE"
```

規則：

在立即執行模式下，您可以執行一列包含有好幾個分開的敍述的程式

睡着了或醒著？

人們在醒著時是一個樣子而睡着時又是另外一個樣子，他們有兩種“運作模式”。

您可以因爲他們會打鼾而分辨他們是否睡著。（雖然不是每一個人都打鼾，但是爲了解釋電腦與人有多麼相像，我們假設人們睡著時都會打鼾。）

電腦也有兩種“運作模式”，它們就是“編修模式”和“執行模式”。

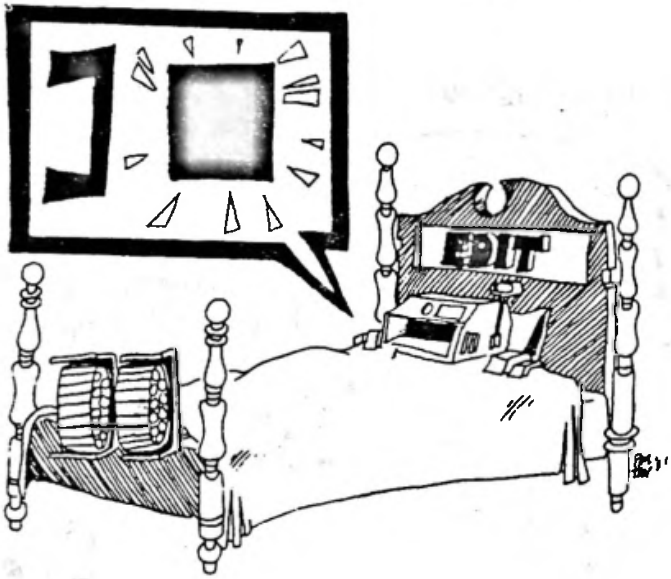
編輯模式

請按下 RESET 鍵。

您將看到一個“] ”符號和一個閃爍的小方塊；“] ”符號稱爲一個“系統標示”（ prompt ），表示電腦正處於Applesoft BASIC的“編修模式”下。當電腦處於編修模式時，“] ”符號就是它的鼾聲。

會閃爍的小方塊稱爲“游標”，它告訴我們電腦正等著您打進一些東西；當電腦處於編修模式時，我們要打入的下一個字母就是在游標的位置。

您可以將程式以號碼開頭的形式打入，您也可以把電腦當作一個口袋型計算機一樣使用，那可是一個很大的口袋哦！



您可以改正程式中的錯誤，這就是“編修”（edit）程式，也正是“編修模式”名稱的由來。我們將在本書稍後學習如何編修程式。

執行模式

輸入 RUN 就可以離開編修模式而進入執行模式。

當電腦處於執行模式時：

存在記憶體中的程式在運作

當程式執行完畢後，電腦便自動回到編修模式。

習題18：

1. 解釋什麼是“立即執行”。將電腦當作計算機去做一些算術問題。
2. 解釋什麼是“延緩執行”，寫一段有幾列的程式；其中一列可以印出“22 plus 67 is（22加67是）”，然後在另一行中執行加法並印出答案。
3. 您如何分辨電腦是否處於編修模式中？
4. 電腦在“執行模式”期間做些什麼事？
5. 當電腦執行程式完畢後，它將進入什麼模式？
6. 您如何知道您要打入的下一個字母會出現在螢光幕上那個地方？

老師札記 19

移動由字串組成的圖形

本課將顯示一個有趣的小程式，它可以“射”一支箭橫越螢光幕。

本課也要示範一些字串變數的威力；例如：加入一些空格來移動那支箭，同時還要擦掉前面的影子，也用到了字串的連接，並且用了一個清晰而且直接了當的迴圈來加深迴圈觀念和盒子觀念的印象。

問答題：

1. 對於會移動的圖形，您必須在繪入新圖形之前擦去舊圖，在“射”箭程式中，這個擦去的動作是如何完成的？
2. 字串盒子可以是空的；您要如何告訴電腦您想要字串變數 D$ 有一個空盒子？
3. 有一個迴圈以 NEXT I 指令作爲結尾。NEXT 對變數 I 有什麼作用？NEXT 做了什麼測驗？NEXT 有什麼樣的選擇並且下一個步驟要執行那一列呢？

第 19 課

移動由字串組成的圖形

有一種畫圖的方法就是利用字串。執行下面的程式：

```
10 REM >>>---ARROW----> (箭)
20 HOME
30 S$="  ":REM        S$ IS A SPACE ( S$是一個空白)
40 A$=">>>------>"
42 VTAB 10
44 FOR I=1 TO 29
46 HTAB I
48 PRINT S$+A$;:REM      GLUING TWO STRINGS(黏合兩個
50 NEXT I                                    字串)
```

將它保存在磁碟內。

這個程式射一支箭橫越螢光幕：

列 30 在引號之間有一個空白，所以在"S$"的盒子中存了一個空白。

列 40 這個字串看起來像一支箭。

列 42 VTAB 選擇編號 10 的那一列開始讓這支箭向前移動。

列 44　這是迴圈的開始。它在編號 50 的那一列結束。

列 46　從這列的左邊開始印字。

列 48　首先 S$ 盒中的空白連接在箭的後面，然後印出空白和箭。

列 50　NEXT 檢查 I 是否已大於 29，如果不是，程式便回到列 46，因爲這是在 FOR I ……之後的第一列。

列 46　現在，從左邊算起一格開始印出這支箭。

列 48　這支箭的羽狀尾巴後面印了一個空白。這個動作將原來的箭擦掉了。

列 50　NEXT 再度檢查。

現在，您繼續用這種方法追踪迴圈兩次。

列 46_____

列 48_____

列 50_____

列 46_____

列 48_____

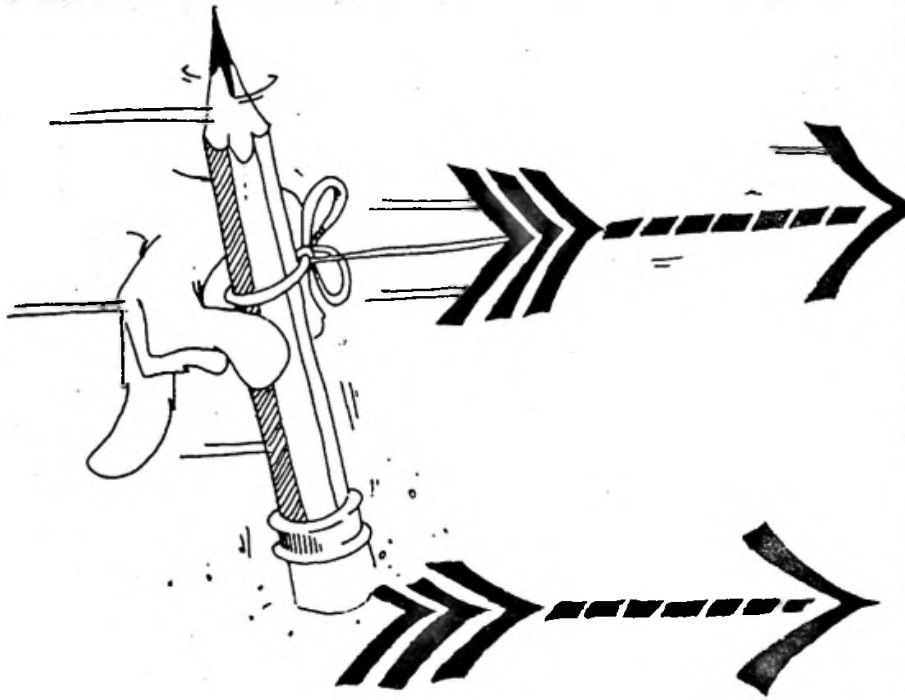
列 50_____

擦掉動作的眞正工作情形

我們把那支箭的前三個步驟寫出來；利用“ b ”代表那支箭後面的空白以便您可以看到它，您將看到：

```
b>>>------->
 b>>>------->
  b>>>------->
```

每一次，除了羽狀尾巴的最後一個“ > ”，新的箭身都寫在舊的箭身上，最後一個“ > ”是由下一支箭後面跟著的空白擦掉。



空字串和空白字串

執行下面程式：

```
20 A$="A":B$="B"
30 S$=" "
40 PRINT A$;S$;B$
```

電腦將在A與B之間印一個空白，空白是由編號 30 的那一列來的，它就是引號之間的那一個，所以 S$ 的盒中有一個空白在裏面。

現在，將編號 30 的那一列改成：

```
30 S=""
```

現在，引號之間空無一物，所以在 S$ 的盒中沒有東西。

規則：

一個空白字串和一個空字串是不同的。

習題19：

1. 令那支箭飛越得慢一些，令那支箭的後面噴出一些烟。
2. 寫一段可以讓您的名字從左邊滑到右邊的程式。
3. 現在，另外寫一段可以讓您的朋友的名字由上向下滑的程式。提示：利用 VTAB，並且不要忘記在印出“新”名字之前擦掉“舊”名字。
4. 現在，寫一段可以讓您的名字橫越螢光幕，而讓您的朋友的名字滑下來且在螢光幕中心交會的程式。

老師札記 20

變數的名稱

本課討論給變數名字的規則。

有意義的變數名字能使程式更清楚；另一方面，它們需要打進去的字自然也比較多，而且還有兩個隱藏著的“機關”。一個“機關”是 BASIC　只記錄一個名字的前面兩個字母，這表示那些長長的，美麗的，有意義的名字並不是唯一的，即使有經驗的設計師也偶而會陷在那裏，很難找出錯誤所在。另一個“機關”是“保留字”是不可以包含在名字的任何地方的，不論開始，中間或結尾，這些保留字是 BASIC 指令和函數。

如果一個名字中包含有保留字，則電腦在列出程式時將把變數名字分開並顯示出保留字本身；這種事第一次發生時程式設計師可能會以爲是打字錯誤，可是，這種錯誤在改正打字後還是會出現的。唯一的解決方法是選一個新名字。

APPLESOFT BASIC 裏面的保留字列在附錄C中。

問答題：

1. 使用兩個字元以上的變數名字可能會帶給您麻煩，爲什麼？
2. 包含數目在裏面的名字可能是好的名字，也可能是錯誤的名字。您怎麼來區別它們是否是好的？

3. 名字是不能有標點與符號的，除了一個例外，您知道什麼樣的標點或符號可以使用，應該放在那裏，和它會告訴您什麼事嗎？

4. 長的名字可能包含了“保留字”在裏面，那些字是“保留的”？在那裏可以找到這張保留字的表？

第 20 課

變數的名稱

舊規則：

1. 字串變數的名字需以金錢符號“ $ ”作結尾。
2. 數字變數的名字就不需要。

更完整的規則：

3. 名字是由字母和數目組成的。
4. 名字必須以字母開頭。
5. 名字中不能有任何其他符號或標點符號，（除了字串名字以一個“ $ ”作爲結尾的是個例外。）
6. 名字的長度是不受限制的。但是……
7. 只有最前面兩個字元才有重要性，其餘的都被忽略掉（除了在字串名字最後的“ $ ”之外）。
8. 保留字不能出現在名字的任何地方，這些字是像 REM, LIST, FOR, TO, LET, IF, PRINT 等之類的字，您可以可在附錄 C 中找到它們。

正確的與錯誤的名字

以下是一些正確的名字：

```
A77
LEFTSIDE
X3AB
APE
NAME$
D5$
```

以下是一些錯誤的名字：

```
2X
X*
S[$
$NAME
3RDNAME$
LIST
COLOR
STATE$
TOM
```

將 COLOR, STATE$, 和 TOM 像下面一樣各放在一列裏，

做做看： 10 TOM = 1

然後　　LIST

如果有保留字在裏面，您將看到名字被分開來。

實際相同的不同名字

如果您不遵守“只有最前面兩個字元才有重要性”這條規則可能使您的程式執行起來變得非常奇怪！

電腦是無法區分出下面各對的名字有什麼不同的：

它將認爲	HOSE	與	HOP	相同
	X1	與	X10	相同
	NAME1$	與	NAME2$	相同

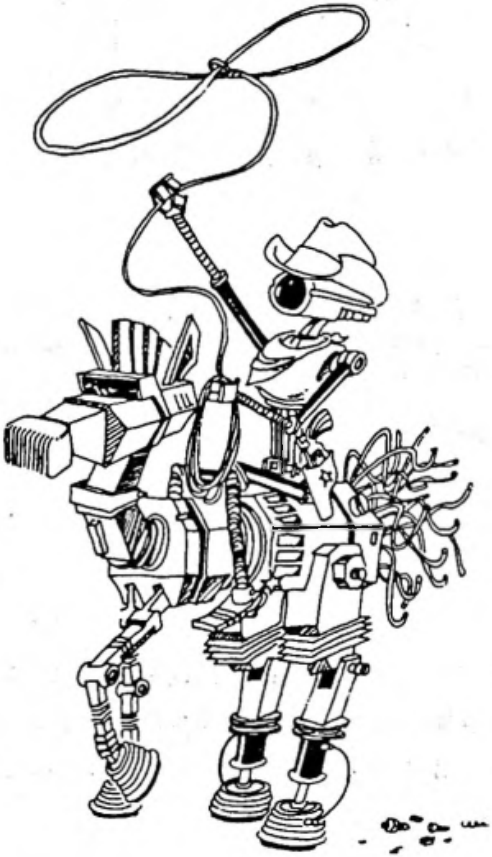
試試下面程式：

```
10 HOP$  = "BUNNY"(兔寶)
20 HOSE$ = "LONG AND NARROW'(長又窄)
30 PRINT HOP$
```

可以看到HOSE$ 和 HOP$指的是相同的變數，實際上只有"HO$"寫在它的盒子前面。

習題20：

1. 從規則1 讀到規則 8 ，針對每一條規則找兩個名字，一個正確的，而另一個是違反該規則的。（規則 6 沒有錯誤的名字。）把每個名字以下面的形式 10 NAME=1 或 10 NAME$="A" 試一試，看看它是否是一個合法的名字。



老師札記 21

低解析度圖形

本課要介紹GR,TEXT,COLOR 和 PLOT 指令，下一課將探討 VLIN 和 HLIN指令。

如果您有一部黑白電視機或螢光幕，那麼低解析度（LO-RES）圖形也是很有趣；事實上，圖形會更清楚一些，除了黑色以外，您必須選擇另一個顏色。（例如，第 15 號的顏色，白色。）

彩色與黑白的繪圖方式是不同的，當電腦以一個顏色畫一大片區域時，低解析度的圖畫看起來還不錯。但是，有些顏色的線條却並不很清晰。

如果您有一台彩色螢光幕或彩色電視機，那麼您的學生應該調整一下它的控制以得到令人喜悅的顏色。SYSTEM MASTER 磁碟上的 COLOR DEMOSOFT 程式對這項工作相當有用，這就是爲什麼在說明如何爲您的學生準備一片磁碟的附錄中，要您將這個程式拷貝一份在您學生專用的磁碟上的緣故。

LO-RES代表低解析度（low resolution），表示影像的點（也有人叫做掃描點）並不很細緻。它們是長方形的，而且每一個點就好像顯示文字時一樣寬，但是只有一半高。您的學生使用的這個模式將有 40 個方塊寬， 40 個方塊高，而且在下面還有 4 列寫文字的空間，叫做文字幕。我們所謂的方塊其實是長方形的。

一點一點來畫圖是相當沉悶的，利用下一課的VLIN 和HLIN 指令將會省點事。不管用那種方法，先用方格紙以方塊描出圖形往往是很有幫助

的，我建議利用一個變數先指定圖畫的一個角落（或中央），從這角落往外推算定出圖畫中其他點或線的偏置量（offset），然後如果有必要製作成卡通或者只是做構圖修正，就很容易將整個圖形移位了。

問答題：

1. 如果您給了 GR 指令，將會發生什麼事？
2. 如果你利用 GR 和 PLOT 寫了一段程式，但是當您執行它時您並未看到任何圖畫，想想看您在 GR 指令之後忘了放什麼指令？
3. TEXT指令的作用是什麼？
4. 有多少種顏色供我們選擇？
5. 在 PLOT X,Y 的指令中，什麼範圍的數目是可以使用的？這些數與HTAB 和 VTAB指令可以使用的範圍是有什麼不同？

第 21 課

低解析度圖形

利用很少的彩色方塊繪成的圖就叫做“低解析度圖形”。

調整電視機的顏色

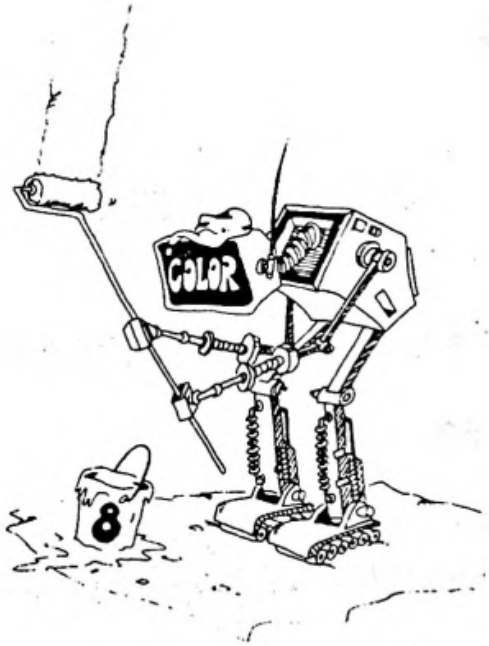
如果您的電視或螢光幕是黑白的，請跳到繪圖指令那一節去，否則從您的磁碟上把 COLOR　DEMOSOFT 這個程式抄入記憶體並執行它。

```
LOAD COLOR DEMOSOFT
RUN
```

從下面的“顏色選擇表”中選擇顏色的號碼或是顏色的名稱並打入它，旋轉您彩色電視機或彩色螢光幕上面的色彩和色調旋鈕使畫面的條紋合乎我們所了解的顏色。

各種顏色爲：

0 BLACK（黑）	8 BROWN（棕）
1 MAGENTA（紫紅）	9 ORANGE（橘）
2 DARK BLUE（深藍）	10 GREY 2（第二種灰色）
3 PURPLE（紫）	11 PINK（粉紅）
4 DARK GREEN（深綠）	12 LIGHT GREEN（淺綠）
5 GREY 1（第一種灰色）	13 YELLOW（黃）
6 MEDIUM BLUE（中藍）	14 AQUAMARINE（水色）
7 LIGHT BLUE（淺藍）	15 WHITE（白）



繪圖指令

執行下面程式：

```
10 REM (((( SMILE ))))(微笑)
20 GR
30 INPUT "WHAT COLOR? <1-15> ";C(什麼顏色?)
35 COLOR=C
40 PLOT 20,20
45 PLOT 21,21
50 PLOT 22,22
55 PLOT 23,22
60 PLOT 24,22
65 PLOT 25,21
70 PLOT 26,20
85 PRINT CHR$(7):REM PEEP(嗶一聲)
```

(接下頁)

```
88 HOME
90 INPUT "AGAIN? <Y/N> ";A$ (再做一次)
95 IF A$="Y" THEN GOTO 30
99 TEXT:HOME

SAVE SMILE (將SMILE保存起來)
```

在試過一些其他顏色後，對 "WHAT COLOR?" (什麼顏色？)這個問題回答 " 0 "，電腦便以 " 黑色 " 畫出那張笑臉，但您在黑色的螢光幕上是看不到它的。

在這個程式中使用的新指令有：

GR	TEXT
COLOR	PLOT

編號 20 的那一列使用了 GR 指令，GR 代表 " 圖形 " (graphics)。

規則：

在開始畫圖之前要先利用 GR 告訴電腦它必須畫一張圖。

編號 30 的那一列使用了 COLOR (顏色) 指令，它選擇下一個要用的顏色號碼。

規則：

在 GR 指令之後，您必須使用附有 1 至 15 之間的一個號碼的 COLOR 指令。

爲什麼？在 GR 指令執行之後，顏色永遠是黑色的（它的號碼爲 0），如果您忘記改爲另一個顏色，您將看不到圖畫；因爲您是無法從一個黑色螢光幕上看到一幅黑色圖畫的。

編號 99 的那一列使用了TEXT 指令。

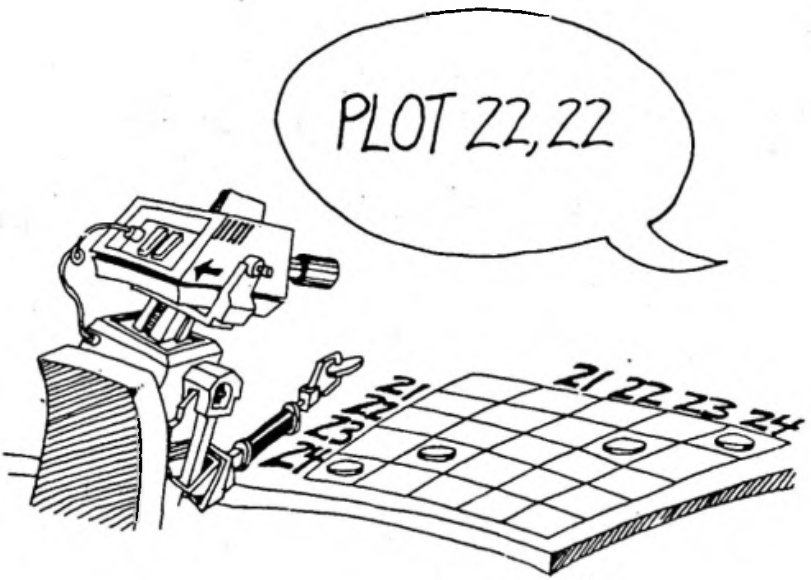
規則：

利用 TEXT 指令來結束畫圖使電腦重新準備好顯示文字。

在這個程式中有好幾列都使用了 PLOT 指令。

規則：

PLOT X,Y 指令表示要把一個小方格放在螢光幕上橫座標 X 與縱座標 Y 的地方，也就是第 X 行，第 Y 列的所在。此地 X 是一個由 0 到 39 的數目或變數；Y 也是一樣。



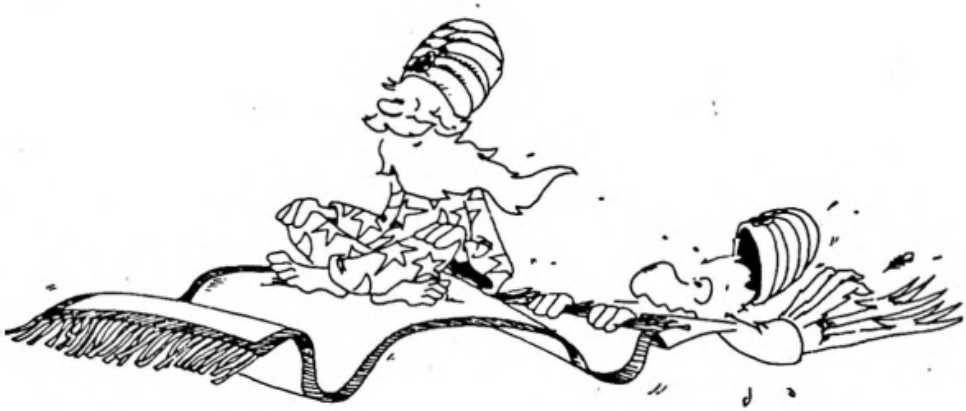
可惜！

非常抱歉，記得 VTAB 和 HTAB 指令在螢光幕上是分別由 1 做到 24 以及 1 到 40 的。如果 PLOT 指令也可以從 1 做到 40 ，那不是很好嗎！可惜它不是這樣，它是由 0 做到 39 的。

習題21：

1. 在那“笑臉”程式中加上眼睛和鼻子。
2. 在第 13 課的數字猜謎遊戲中加入一段程式，在猜對時顯示一個彩色的星星，並利用一個計時迴圈在遊戲重新開始之前讓那顆星星顯示幾秒鐘。
3. 寫一個程式來畫“辛巴達的魔氈”，讓使用者選一些顏色，然後在螢光幕上畫一張樣本。您很可能願意看看在 COLOR DEMOSOFT 程式中是如何畫的。只要：

```
LOAD COLOR DEMOSOFT
LIST 700-799
```



老師札記 22

利用HLIN與VLIN繪圖

本課要解釋HLIN和VLIN繪圖指令。

我們將利用水平和垂直線來畫低解析度的圖，實心區域也可以利用線來填滿。

問答題：

1. 在 HLIN　A,B　AT　C 指令中，會畫出那一種線來？字母A和B告訴指令什麼事？字母C又說些什麼事？
2. 對 VLIN　A,B　AT　C 指令，回答上面相同的問題。
3. 如果您在一條線的上面畫下一條線，或在一條線的旁邊畫下一條線會怎麼樣？
4. 您要如何畫一個四方形，邊線是藍色而裏面是實心紅色呢？

第 22 課

利用HLIN與VLIN繪圖

現在，我們將利用線來畫圖而不用點。

首先，利用COLOR DEMOSOFT 程式和您的電視或彩色螢光幕上的旋鈕來調整顏色。

水平線條

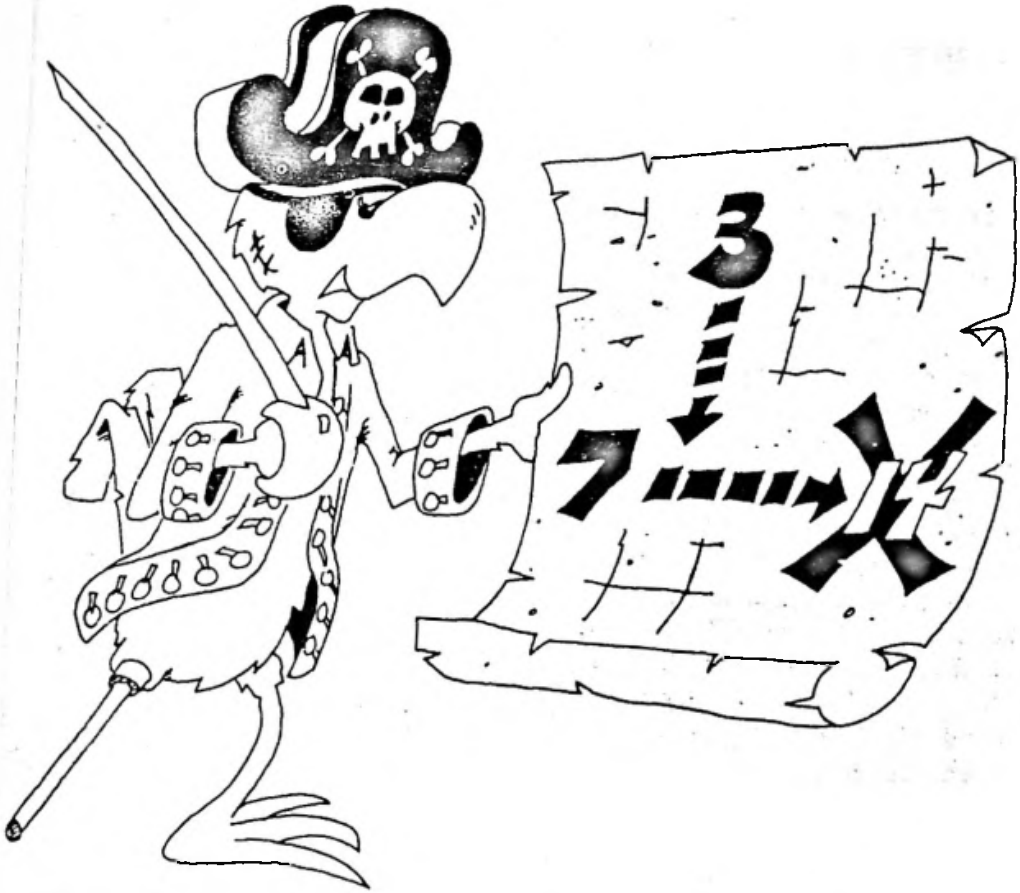
讓我們畫一條彩色的水平線，您需要下達附加三個號碼的HLIN指令，每一個號碼必須在0與39之間。

執行下面程式：

```
10 GR:COLOR=12: REM COLOR IS GREEN (顏色是綠色)
30 HLIN 7,14 AT 3
```

電腦從螢光幕上端算起的第四列畫下一條綠線。（這就是“AT 3”所代表的意義。）這條綠線由螢光幕左邊算起的第8格開始，畫到第15格結束；請不要忘記，列的編號是從0算起的，AT 0表示第一列，AT 1表示第二列，……等等。

換句話說： 30 HLIN 7,14 AT 3 表示：
這條線畫在從上端算起第四列，從第8格開始，一直畫到第15格為止。



垂直線條

將編號 30 的那一列改成：　30 VLIN 8,15 AT 4

現在，水平線改成垂直線了。這個指令代表：垂直的綫是畫在自左邊算起第 5 行（請注意，AT 0 在第一行，AT 1 在第二行，…）上的第 8 格（從上往下算的第 9 格）到第 15 格爲止。

練習程式

執行這程式：

```
10 GR:REM -- MAKE LINES -- (畫線)
12 HOME
15 INPUT"HORIZONTAL OR VERTICAL? <H / V> ";D$
20 INPUT"COLOR? <1-15> ";C                (水平或垂直的)
25 COLOR=C
40 IF D$="V" THEN GOTO 70
42 PRINT"HORIZONTAL LINE" (水平線)
44 INPUT"HOW FAR FROM THE TOP? <0-39> ";Y (從上算起有多遠
46 INPUT"START WHERE? <0-39> ";X1 (從那裏開始？)
48 INPUT"END   WHERE? <0-39> ";X2 (到那裏結束？)
50 HLIN X1,X2  AT Y
55 GOTO 12
70 PRINT"VERTICAL LINE " (垂直線)
74 INPUT"HOW FAR FROM THE LEFT? <0-39> ";X
76 INPUT"START WHERE? <0-39> ";Y1
78 INPUT"END   WHERE? <0-39> ";Y2
80 VLIN Y1,Y2 AT X
85 GOTO 12
```

將這程式以“MAKE LINES”(畫線)的檔案名字保存起來。

一個較長的例子

```
1  REM  L22A
100  REM === PICK UP STICKS ===(揀棍子)
105  HOME
110  GR
120  FOR I = 1 TO 50
125  FOR T = 1 TO 300: NEXT T
130  REM  VERTICAL LINES (垂直線)
135  COLOR=  RND  (8) * 16
140 X =  RND (8) * 39
142 Y1 =  RND (8) * 39
144 Y2 =  RND (8) * 39
146  IF Y1 > Y2 THEN YT = Y1:Y1 = Y2:Y2 = YT
148  VLIN Y1,Y2 AT X
155  COLOR=  RND (8) * 16
157  FOR T = 1 TO 300: NEXT T
160  REM  HORIZONTAL LINES (水平線)
162 Y =  RND (8) * 39
164 X1 =  RND (8) * 39
166 X2 =  RND (8) * 39
168  IF X1 > X2 THEN YT = X1:X1 = X2:X2 = YT
170  HLIN X1,X2 AT Y
190  NEXT I
192  PRINT " D O N E"(完成了)
195  FOR T = 1 TO 5000: NEXT T
199  TEXT : HOME
```

在這個程式中找出下列事項：

1. 三個計時迴圈。每一個各有什麼作用？
2. 一個主要迴圈。
3. 迴圈中的兩項主要活動。

4. 當程式執行完畢時把螢光幕恢復正常的那一列。
5. 告訴電腦準備好畫圖的指令。
6. 在這張圖畫中用了幾個顏色？有黑色的棍子嗎？選擇顏色的是那一列？

輸入這個程式並執行它。然後以"PICK UP STICKS"（撿棍子）的檔案名字將它保存起來。

移動您的圖畫

爲了使您的圖畫移動，您必須：

畫它
擦掉它
移開一點再畫它
再擦掉它
如此繼續下去

要完成這件事最好的方法是利用一個副程式來畫這個圖，我們將在第24課告訴您如何做這件事。

習題22：

1. 寫一段可以畫一個方框的程式，讓使用者自由選擇顏色，將它保存在磁碟上。
2. 在第一題的程式中加入一段程式使它可以產生1到6個點之中的一個，使它看起來像個骰子。
3. 寫一段可以畫出您名字的第一個字的程式，令它從紅色（1號顏色

）開始，並“嗶”一聲，然後改成深藍色（2號顏色），如此一直做到 15 號顏色。

老師札記23

秘密寫入與GET指令

本課要討論 GET （取得）指令。

GET是從鍵盤索取一個字元的方法。執行它時，電腦會停下來一直到鍵被敲下爲止，才再度有動作。這段期間內螢光幕上沒有任何顯示，甚至提示信號(prompt)或游標也沒有顯示出來；當敲鍵時，字元也不會反映(echo)到螢光幕上。

GET 指令的用途也就是基於以上的需要而設計的；例如，我們可以用一系列的 GET 指令來索取一個字而不讓別人看到。

與 INPUT 比較，GET 有另外一項優點就是不需要按 RETURN 鍵，這使得 GET 在使用者與電腦交談式的程式設計中變得非常有用。

Apple公司指出，最好是將輸入的字元放入一個字串變數內而不是直接使用數字變數。如果您要GET（取得）數目字，那末先把它當作字串取得之後，然後利用後面將談到的VAL( )函數轉換成數目。

問答題：

1. 比較 INPUT 和 GET 。一個是一次取得一個字母，而另一個是一次取得一個完整的字或句子；一個有游標而另一個沒有；一個是將訊息印在螢光幕上而另一個却不這麼做；一個需要按下 RETURN 鍵而另一個則不需要。究竟那一個指令會造成那一個現象呢？

第 23 課

秘密寫入與GET指令

INPUT指令

INPUT有兩種用法。

不包含提示訊息的用法：

```
10 INPUT A$
10 INPUT N
10 INPUT NAME$,AGE,DAY,MONTH$,YEAR
                                    (年齡，日，月，年)
```

包含提示訊息的用法：

```
10 INPUT "NAME,AGE";NAME$,AGE
10 INPUT "HOW ARE YOU? ";FEEL$ (您好嗎？)
```

不管用那一種方法都可以打進一個字，句子或數目。

GET指令與秘密寫入

GET 指令與 INPUT 不同，它可以從鍵盤取得字元。在程式執行到GET 指令之後，電腦會停止動作，一直等到有鍵被按下去了，才會恢復動作。

螢光幕上不會顯示任何東西：

沒有提示訊息
沒有問號
沒有游標
也不會顯示您所打入的字

電腦一直等到有鍵按下爲止，之後，您也不需要按 RETURN 鍵，電腦就立刻繼續執行程式。

您可以在猜謎遊戲中利用 GET 來輸入謎底的字或數目，其他人是無法看到的。

執行下面程式：

```
10 REM -----GET-----
20 HOME
30 PRINT "PRESS ANY KEY"(按下任何一個鍵)
40 GET K$
45 PRINT CHR$(7)
47 FOR T=1 TO 1000:NEXT T
50 PRINT "THE KEY YOU PRESSED WAS ";K$
                (您按下的鍵是…)
```

再執行下面程式：

```
10 REM *** BACKWARDS *** (反方向)
20 HOME
30 PRINT"TYPE IN A 5 LETTER WORD"(打入一個5個
35 PRINT                                子母的字)
40 GET A$:GET B$:GET C$:GET D$:GET E$
50 PRINT"NOW HERE IT IS BACKWARDS"(現在，它被
55 PRINT:INVERSE                        反過來了)
60 PRINT E$;D$;C$;B$;A$
70 NORMAL
```

由字母組成字

GET 指令一次取得一個字母。若要組成字，要先連接字串。

```
10 REM GET A WORD (取得一個字)
20 HOME                          (打進一個字，並在後面加一個句點。)
30 PRINT"TYPE A WORD, END IT WITH A PERIOD."
35 W$="": REM WORD STRING IS EMPTY(字串是空的)
40 GET L$:REM GET A LETTER(取得一個字母)
50 IF L$="." THEN GOTO 80:REM TO TEST FOR END(測驗是否結束了)
60 W$=W$+L$:REM ADD LETTER TO END OF WORD (把字母加到字的後面)
65 GOTO 40: REM TO GET ANOTHER LETTER (取得另一個字母)
80 REM WORD IS FINISHED (字已組成)
85 PRINT W$
```

當一個完整的字被打入後，電腦怎麼知道呢？它會看到字尾有個句點。編號 50 的那一列就是測驗是否有句點存在，當它發現句點時就結束這個字。

利用GET指令取得數目

GET 指令也可以用來輸入數目，但是會有一些麻煩，我們將在"轉換數目與字串"一課中詳細解釋。

習題23：

1. 設計一份可以供使用者選擇顏色"選擇表"的程式；使用者利用打進一個單一字母來選擇。請利用 GET 來取得字母。例如：

```
PRINT "WHICH COLOR? =R=RED, B=BLUE,
G=GREEN"
```

2. 寫一段造句程式，每--個句子須包含一個名詞主詞，一個動詞，和一個受詞；第一個人打一個名詞〔如 "The donkey"（驢子）〕，第二個人打一個動詞〔如 "sings"（唱）〕，第三個人打另一個名詞〔如 "the toothpick."（牙籤歌）〕。利用 GET，讓每個人無法看到別人打入的字，你可以將這些程式加以擴充，在名詞之前加入形容詞。

老師札記 24

漂亮的程式，GOSUB，RETURN 與 END 指令

本課要討論副程式。END 指令也將包括在本課中，因爲副程式通常都被放在程式中較高列號碼的地方，以致於在中間部分需要一列 END 。

副程式不僅在長程式中有用，在短程式中也可以將程式分成幾段而使程式更清楚。

對於某些學生（甚至一些專家），最難養成的習慣之一是在程式中採用結構（ structure ）的觀念。用結構的觀念來設計程式還有許多名稱，諸如“結構化程式設計”和“直敍式程式設計”（ top down programming ），並且有各種不同的方法來訓練程式設計師。

要提醒學生注意到有多種方式可以達到結構化的目的，而且它有思路清晰和程式設計簡單等優點。在本書中，利用 REM 敍述做詳細的註解並在程式中使用模組化（modular）的結構都是主要的方法。

GOSUB 在 BASIC 中是用來組成模組（ module ）的，本課將在“ cootie ”（虱子）程式的摘要中以例子說明模組化的結構。

問答題：

1. 執行 END 指令時會怎麼樣？
2. GOSUB 與 GOTO 有什麼不同？

3. 當執行RETURN 時會怎麼樣？
4. 如果在GOSUB之前執行 RETURN ，會怎麼樣？
5. “呼叫副程式”代表什麼意義？
6. 在一個程式中可以有幾個 END 指令？
7. 在程式中爲什麼需要副程式？

第 24 課

漂亮的程式, GOSUB, RETURN, 與 END 指令

輸入下面程式，在執行過確定沒有錯誤之後將它保存到磁碟中：

```
100   REM   MAIN PROGRAM (主程式)
101  :
105   HOME
110   PRINT "READY TO GO TO THE SUBROUTINE"
120   GOSUB 200                          (準備好到副程式去)
130   PRINT "BACK FROM THE SUBROUTINE" (從副程式回返)
133   PRINT
135   PRINT "GO TO THE SUBROUTINE AGAIN" (再到副程式去)
140   GOSUB 200
150   PRINT "BACK AGAIN" (再回返)
190   END
199  :
200   REM   SUBROUTINE (副程式)
201  :
210   PRINT "IN THE SUBROUTINE" (在副程式裏面)
215   FOR T = 1 TO 2000:NEXT T
217   PRINT CHR$(7)
290   RETURN
```

這是一個大程式的骨架，主程式由編號100的列開始而在編號190的那一列結束。在各個PRINT指令的地方，你可以放入其他程式。

列190內的END指令告訴電腦程式在此結束，於是電腦回到編修模

式去。在列120與140“呼叫副程式”，這表示電腦跳到副程式去執行那兒的指令，然後再回去。

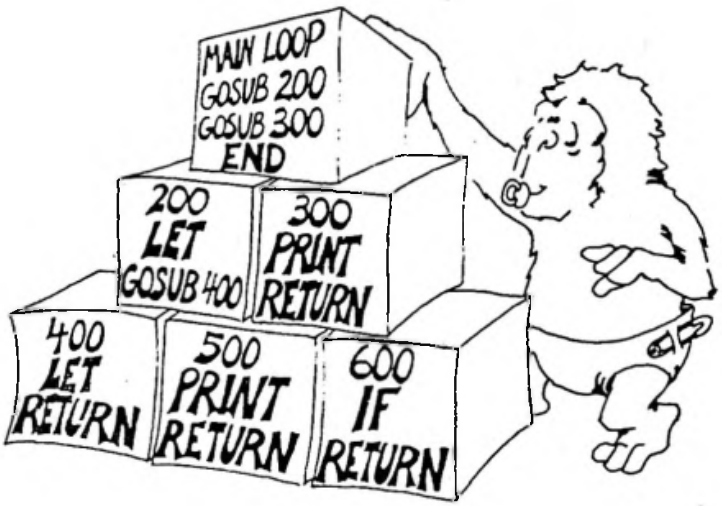
GOSUB 200 指令與 GOTO 200 指令是相似的，電腦會記住它們從那裡來，而 GOSUB 不同的是，電腦還會記住要回到那裡去。

RETURN 指令告訴電腦回到GOSUB之後的敍述去。

副程式有什麼好處呢？

在短程式中，好處並不多，但是在長程式中，它有兩項好處：

1. 它可以節省工作與記憶空間，您不需要在程式的不同的部份重複相同的程式指令。
2. 它使程式更容易了解，寫作加快而且節省除錯的時間。



END 指令

程式可以有零個，一個或許多個 END 指令。

規則：

END 指令告訴電腦停止執行並且回到編修模式。

它眞的是這樣做的；您可以將 END 指令放在程式的任何地方，例如，放在 IF－THEN 敍述的後面。

移動畫面

```
10   REM ??? JUMPING J ??? (跳動的J)
20   HOME : GR
22  X = 15:Y = 15:D = 1
25   FOR J = 1  TO  10
26   FOR I = 1  TO  10
30   COLOR= 8: GOSUB 100: REM DRAW  (畫線)
35   COLOR= 0: GOSUB 100: REM ERASE(擦去)
45  Y  =  Y  -  D
50   NEXT  I
55  D  =  -  D
60   NEXT J
90   TEXT : HOME : END
100   REM
101   REM DRAW THE J
102   REM
110   HLIN X,X + 5 AT Y
120   VLIN Y + 1,Y + 7 AT X + 3
130   HLIN X,X + 2 AT Y+ 7
140   PLOT X,Y + 6
190   RETURN
```

畫面上是個“ J ”字。副程式是由編號100 的那一列開始畫“ J ”；在您 GOSUB100 （跳到副程式）之前，要利用一個COLOR 命令選定“ J ”是什麼顏色。請注意編號 30 和 50 的兩列。如果您選擇“ 0 ”號顏色，那麼副程式將從那一點擦去“ J ”。

副程式將“ J ”的左上角畫在螢光幕上的X，Y 點。當您改變X 或Y（或兩者）時，“ J ”就會畫在不同的地方了；編號 22 的那一列說明第一個“ J ”將畫在接近螢光幕的中央部分。

字母“D”告訴電腦下一個“J”距離這個“J”有多遠，編號22的那一列令“D”等於1，但55那一列在電腦畫好10張圖片後將D改爲－1；編號45的那一列則告訴電腦每一張圖要畫在比原來的Y增加D的Y地方。

習題24：

1. 輸入上面的“JUMPING J”（跳動的J）程式並執行它，然後做以下修改：

   將副程式改爲印您名字的第一個字母。

   將您名字的第一個字改爲藍色。

   將“jumping”改爲“sliding”（滑行的）（使J水平移動而不是垂直跳動）。

   將起點改到螢光幕的右下角而不是中央。

   將滑行的距離改爲20步而不是10步。

   將每一步的大小從1改爲2。

   將“滑行”改爲爬坡，利用

   ```
   X = X + D : Y = Y - D
   ```

   修改程式使名字的第一個字從紅色開始，每當它跳躍時改變爲下一個顏色；依次使用所有顏色一直到白色（15號顏色）爲止。

   將編號35那一列的COLOR = 0改爲COLOR = 1會怎麼樣？

如何寫一個長的程式

讓我們寫一個“劊子手”（hangman）遊戲。這是一個猜字遊戲，每當您猜錯一個字母時，您就在絞刑犯的圖上畫上一部份。

首先要做一個綱要，您可以在紙上或直接到螢光幕上做這件事．如果您不知道怎麼做，那麼只要依照遊戲的玩法在紙上玩一次，並追踪所發生的事，然後讓程式也照那樣去做。

下面是一個可能的綱要：

```
10 REM *** HANGMAN GAME ***  (劊子手遊戲)
200 REM INSTRUCTIONS         (玩法說明)
300 REM GET THE WORD TO GUESS(取得要猜的字)
400 REM MAKE A GUESS  (猜猜看)
500 REM TEST IF RIGHT(測驗是否猜對了)
600 REM ADD TO THE DRAWING (在圖畫上加上一部份)
700 REM TEST IF GAME IS OVER(測驗遊戲是否結束)
800 REM END GAME MESSAGE(遊戲結束的訊息)
```

做好這個綱要以後，在程式中加上一些細節：

```
10   REM *** HANGMAN GAME *** (劊子手遊戲)
99 :
100 REM MAIN LOOP(主迴圈)
101:
120 INPUT" NEED INSTRUCTIONS? <Y/N> "; Y$(需要指示嗎？)
122 IF Y$="Y" THEN GOSUB 200
130 GOSUB 300:REM GET WORD
132 STOP
135 GOSUB 400:REM MAKE GUESS
140 GOSUB 500:REM TEST GUESS
145 GOSUB 700:REM TEST IF GAME IS OVER
190 GOTO 135:REM MAKE ANOTHER GUESS(再猜一次)
199:
```

```
200 REM INSTRUCTIONS
……遊戲的玩法說明寫在此處
290 RETURN
299:
300 REM GET THE WORD TO GUESS
……利用 INPUT 向第一個人索取一個字。
……在被猜的各個字母上畫一條短線（ dash ）。
390 RETURN
399:
400 REM MAKE A GUESS
……第二個人猜一個字母。
490 RETURN
499:
500 REM TEST IF GUESS IS RIGHT
……如果猜錯了，GOSUB 600:REM  在劊子手的圖上加上一部分。
……如果猜對了，GOSUB 700:REM  看看遊戲是否結束了。
590 RETURN
599:
600 REM ADD TO THE DRAWING
……在劊子手的圖上加上一部分。
……測驗這張圖是否完成了。
……如果是，那麼 GOSUB 800
690 RETURN
699:
700 REM TEST IF GAME IS OVER
……看看是否所有字母都猜過了。
……如果是，那麼 GOSUB 900
```

```
790 RETURN
799:
800 REM END GAME MESSAGE
```

……若猜字者輸了，便印出這段訊息。

```
890 RETURN
899:
900 REM END OF GAME MESSAGE
```

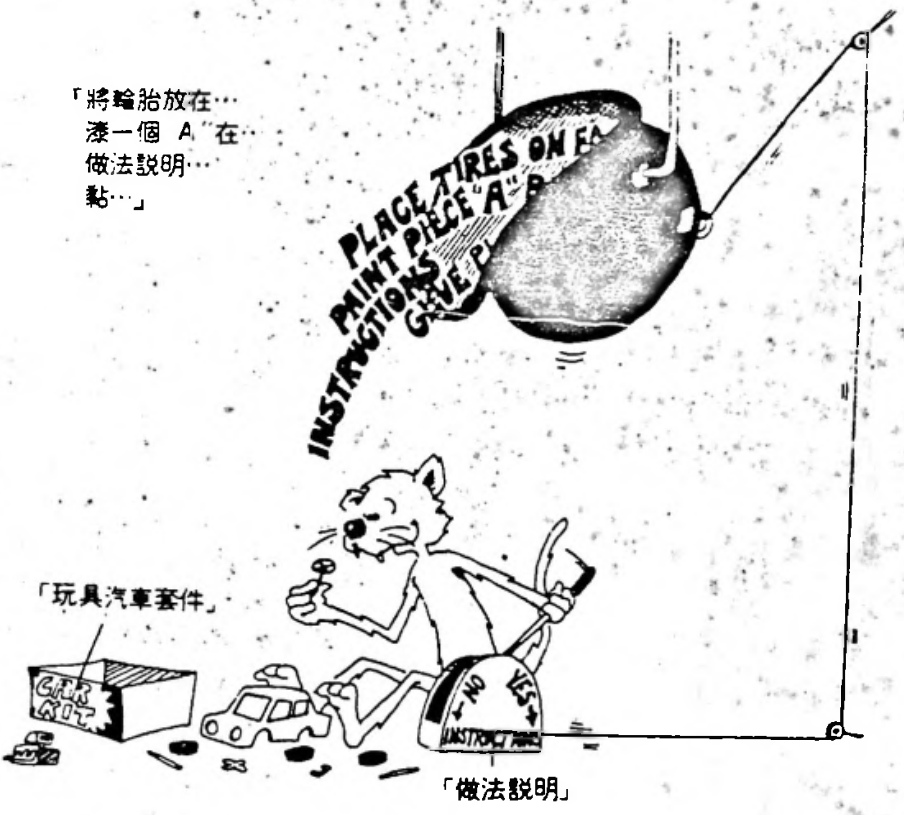
……若猜字者贏了，便印出這段訊息。

```
990 RETURN
```

這麼多事情在腦子裏面會有點混亂而不知如何結束這遊戲，因此，我將後半部留在後面，先寫程式的前半部並測試這一部分；我在編號132的那一列放了一個STOP（停止），因此只有第一個副程式會被執行。我再從列132的副程式，“GET A WORD”（索取一個字）開始寫。

習題24A：

1. 寫一段利用副程式的簡短程式，它不一定要做什麼有用的事，只要印一些無意義的話。將它們分成三個副程式：
   從主程式呼叫兩次它們之中的一個。
   從另一個副程式呼叫它們之中的一個。
   從一個 IF 敘述呼叫它們之中的一個。
2. 寫一個程式在螢光幕上寫下三個您的名字的第一個字，每一個名字各有不同的顏色，然後令它們同時跳上，跳下。
3. 將文中的“劊子手”遊戲完成。這是一個長的計畫，您可以先寫一部分，將它SAVE（保存）在磁碟上，稍後再完成其他部分。


「將輪胎放在…
漆一個 A 在…
做法說明…
黏…」
PLACE TIRES ON FA
PAINT PIECE "A" B
INSTRUCTIONS
GIVE P
「玩具汽車套件」
CAR KIT
NO
YES
「做法說明」