More on Do Loops:
Cycle, Exit, Do while, Nested Loop

Do 迴圈—Cycle功能

- 直接跳回迴圈的開頭，執行下一次迴圈。舉例：

```
PROGRAM test_cycle
IMPLICIT NONE
INTEGER :: I

DO I=1,5
  IF (I==3) THEN
    CYCLE  ! Go back to top and continue
  ENDIF
  WRITE(*,*) I
ENDDO

WRITE(*,*) 'End Loop!'

END PROGRAM test_cycle
```

當迴圈進行到counter I＝3的時候，不執行之後的WRITE指令，直接跳回迴圈開頭，並且令I+1(=4)繼續迴圈，直到迴圈結束

Do 迴圈—Exit功能

- 在迴圈重複執行過程中，直接跳出迴圈。舉例

```
PROGRAM test_exit
IMPLICIT NONE
INTEGER :: I

DO I=1,5
  IF (I==3) THEN
    EXIT ! Finish loop
  ENDIF
  WRITE(*,*) I
ENDDO

WRITE(*,*) 'End Loop!'

END PROGRAM test_exit
```

當迴圈進行到counter I＝3的時候，直接跳出迴圈，執行END DO之後的指令。

Do 迴圈—DO While功能

- 之前學的迴圈語法，必須事先設定好重複遞迴執行的次數（如 Do i=1,10）
- 若無法事先決定迴圈要重複幾次，可以使用DO While 語法，舉例：

```
PROGRAM test_dowhile
IMPLICIT NONE
REAL :: k=1., S=0
DO WHILE (k>0)
  WRITE(*,*) 'Please input next number (>0):'
  READ(*,*) k
  IF (k>0) THEN
    S=S+k
    WRITE(*,*) 'Sum =', S
  ENDIF
END DO
WRITE(*,*) 'Do not input 0 or negative number!'
END PROGRAM test_dowhile
```

使用者每輸入一個正數，程式會輸出之前所有輸入數值累加的結果，直到使用者輸入0或負數，才跳出迴圈

巢狀迴圈（nested loop）

- 之前學過的Do迴圈只有一層，可以在迴圈中再寫入迴圈，形成「巢狀迴圈 」結構，舉例：

```
PROGRAM nest_loop
IMPLICIT NONE
INTEGER :: I,J
   DO J=1,5
      DO I=1,3
         WRITE(*,*) I, J
      END DO
   END DO
END PROGRAM nest_loop
```

迴圈運算順序：
外層迴圈的counter J 先保持在1，
讓內層迴圈counter I 由1到3繞完，
之後外層迴圈的J再增加1 (J=2)，
並且再進行一輪內層迴圈 (I=1,2,3)
直到外層迴圈也全部繞完 (J=5) 為止