FORTRAN

副程式（subroutine）

副程式SUBROUTINE

- 副程式：更多功能、更有彈性的自訂函數（FUNCTION），能夠指定輸入、回傳多個變數，也可以更動輸入的變數
- 副程式則要在主程式「呼叫」（CALL）之後才開始執行，

  CALL SUB_NAME(arg_list)

  （SUB_NAME: 副程式名稱，arg_list: 傳入與回傳的變數清單）

（主程式）

```
PROGRAM  main
…
CALL SUB_NAME(arg_list)
…
END PROGRAM main a
```

（副程式）

```
SUBROUTINE SUB_NAME(arg_list)
…
…
RETURN
END SUBROUTINE SUB_NAME
```

副程式本身的語法

```
SUBROUTINE SUB_NAME(arg_list)

(Declaration section)
TYPE, INTENT(in)::...
TYPE, INTENT(out)::...
TYPE, INTENT(inout)::...

(Execution section)


RETURN
END SUBROUTINE SUB_NAME
```

- 以SUBROUTINE開頭，以一行RETURN與一行END SUBROUTINE 結尾
- SUB_NAME：副程式的名字
- arg_list：由主程式傳入以及傳回主程式的變數清單
- INTENT(in) 變數只供傳入，在副程式中不能改變其數值
- INTENT(out) 變數會傳回主程式，在副程式中可以改變數值
- INTENT(inout) 變數可以從主程式傳入再傳回，在副程式中可以改變數值

主、副程式的變數傳遞 (1)

- FORTRAN在主、副程式之間傳遞變數時，是傳遞記憶體的位置，而不是變數名稱
- 主程式呼叫時給予的變數清單（arg_list），只要變數種類、個數相符，就會按照arg_list的順序傳入副程式，就算變數名稱不一致也沒關係。舉例：

（主程式）

```
PROGRAM  main
IMPLICIT NONE
INTEGER :: a=1, b=0
CALL sub1(a,b)
WRITE(*,*) 'in main',a,b
END PROGRAM main
```

執行結果 →

```
in sub 1 0
in main 1 99
```

（副程式）

```
SUBROUTINE sub1(x,y)
INPLICIT NONE
INTEGER,intent(in) :: x
INTEGER,intent(inout):: y
WRITE(*,*) 'in sub', x,y
y=99
RETURN
END SUBROUTINE sub1
```

主、副程式的變數傳遞 (1)

- 主程式按照清單順序，將a, b的記憶體位置依序傳給副程式，副程式將記憶體中儲存的數值按順序分配給副程式中的x, y。
- 副程式一開始時，x=1, y=0，副程式結束時x=1, y=99，將結果按照清單順序回傳給主程式，
- 主程式依序分配結果給a, b變數。呼叫副程式後，主程式的a=1, b=99

（主程式）

```
PROGRAM  main
IMPLICIT NONE
INTEGER :: a=1, b=0
CALL sub1(a,b)
WRITE(*,*) 'in main',a,b
END PROGRAM main
```

執行結果 →

```
in sub 1 0
in main 1 99
```

（副程式）

```
SUBROUTINE sub1(x,y)
INPLICIT NONE
INTEGER,intent(in) :: x
INTEGER,intent(inout):: y
WRITE(*,*) 'in sub', x,y
y=99
RETURN
END SUBROUTINE sub1
```

主程式呼叫時給予的變數清單，只要變數種類、個數相符，就會按照順序傳入副程式，就算變數名稱不一致也沒關係。

主、副程式的變數傳遞 (2)

- 副程式宣告的變數名稱，與主程式是完全獨立無關的。
- 就算主、副程式使用了相同的變數名稱，彼此也沒有關連，舉例：

（主程式）

```
PROGRAM  main2
IMPLICIT NONE
INTEGER :: a = 1
CALL sub2()
WRITE(*,*) 'in main', a
END PROGRAM main2
```

（副程式）

```
SUBROUTINE sub2()
INPLICIT NONE
INTEGER :: a=2
WRITE(*,*) 'in sub', a
RETURN
END SUBROUTINE sub2
```

執行結果 →

```
in sub 2
in main 1
```

注意：此例中主程式與副程式彼此沒有傳遞任何變數

錯誤的副程式變數傳遞—預習問題

（主程式）

```
Program bad_call
IMPLICIT NONE

REAL :: x = 1.0

CALL bad_arg(x)

END PROGRAM bad_call
```

（副程式）

```
SUBROUTINE bad_arg(i)
IMPLICIT NONE

INTEGER, INTENT(in) :: i
write(*,*) 'input=', i

RETURN
END SUBROUTINE bad_arg
```

執行結果 input=106535321.6

上面的例子，主程式雖然設定x=1.0，傳入副程式後，輸出的數值卻是錯誤的。
請問要如何修正副程式，才能進行正確的變數傳遞？

在副程式中宣告陣列的大小

- 在副程式中使用陣列變數，也要宣告陣列的大小
- 如果副程式中使用的陣列大小，必須由主程式決定，在撰寫副程式的時候，就要讓陣列的大小用一個整數變數傳入副程式，並且利用這個整數變數來宣告陣列大小。舉例：

讓主程式傳入代表陣列大小的變數

```
SUBROUTINE sub1(data, nx, ny, ...)
IMPLICIT NONE

INTEGER, INTENT(in) :: nx, ny
REAL, INTENT(in), DIMENSION(nx, ny) :: data

...

RETURN
END SUBROUTINE
```

用傳入的變數宣告副程式中陣列的大小 (explicit-shape array)

使用副程式或自訂函數，如何編譯程式碼？

- 方法一：把SUBROUTINE或FUNCTION貼在主程式碼的下方（END PROGRAM之後），compile主程式即可。

```
>f95 main.f95 -o main.exe
```

- 方法二：把SUBROUTINE、FUNCTION存成個別的.f95檔案，一起compile

```
>f95 main.f95 sub1.f95 func2.f95 -o main.exe
```

- 方法三：把SUBROUTINE、FUNCTION存成個別的.f95檔案，先個別編譯產生.o檔，再編輯主程式，把.o檔連結進去：

```
>f95 -c sub1.f95
>f95 -c func2.f95
>f95 main.f95 sub1.o func2.o -o main.exe
```

副程式SUBROUTINE vs. 自訂函數 FUNCTION

- FUNCTION回傳給主程式的只有單一變數，而且主程式傳入的變數（用intent(in)宣告），在FUNCTION中不能更動
- 副程式能夠輸入、回傳多個變數，也可以更動輸入的變數

- 主程式使用FUNCTION時，FUNCTION用自己的名字當作變數回傳給主程式（例如 y=cot(x)）
- 副程式則要在主程式呼叫執行（例如CALL sort(x,y)），傳遞的變數與副程式的名稱無關

- 在主程式與自訂函數的宣告區，都要宣告自訂函數本身
- 在主程式與副程式的宣告區，都不可以宣告與副程式名稱相同的變數。

副程式的範例 sort_sub.f95

- 程式範例檔：
  /home/teachers/weitingc/lectures_ex/sort_sub.f95

- 將sort.f95的程式中，排序的指令另外寫成副程式sortdata
- 功能：將輸入檔案rain.txt內的10個數值由小到大排序，輸出到rain_sorted.txt中
- 流程：
  - 讀取檔案內的數值，存在陣列rain中
  - 排序（CALL sortdata(...)）
  - 輸出：原來的十個數值，與按照大小排序好的結果

Main Program
sort.f95

```
PROGRAM sort
IMPLICIT NONE
…
! Open input data file and read data
…
READ (100,*) rain(i)
…
!Sort the data
sorted=rain
DO i=1,n-1
…
  DO j=i+1,n
  …
  ENDDO
…
ENDDO
! Output results
…
WRITE(101,999) sorted(i)
…
END PROGRAM sort
```

CALL sortdata(rain, sorted, n)

```
SUBROUTINE sortdata(inputdata, sorted, ndata)
IMPLICIT NONE
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Receive array "inputdata" (size=ndata)
! from the main Program, returnt the
! sorted results in array "sorted"
!
INTEGER, Intent(in) :: ndata
REAL, Intent(in), DIMENSION(1:ndata) :: inputdata
REAL, Intent(out),DIMENSION(1:ndata) :: sorted
REAL :: minimum   ! temporary variable for swapping
INTEGER :: i,j,k  ! counter for do loop

…

…



RETURN
END SUBROUTINE sortdata
```

```
SUBROUTINE sortdata(inputdata, sorted, ndata)
…
…
sorted=inputdata
DO i=1,ndata-1
  k=i
  minimum=sorted(i)
  ! find the minimum value in sorted(i) to sorted(n)
  ! and store it temporarily in minimum
  DO j=i+1,ndata
    IF (sorted(j) < minimum) THEN
      k=j
      minimum=sorted(k)
    ENDIF
  ENDDO
 ! swap the minimum value with sorted(i)
  sorted(k)=sorted(i)
  sorted(i)=minimum
ENDDO
RETURN
END SUBROUTINE sortdata
```

```
PROGRAM sort_sub
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Read in 10 numbers from the input file, sort it
! into ascending order.
! The sorting is done by call sortdata(...)
! Write the original and sorted values into the
! the output file
! (Wei-Ting, 2013/10/07
IMPLICIT NONE
INTEGER , PARAMETER :: n=10      ! size of data array
REAL, DIMENSION(1:n) :: rain     ! data array to sort
REAL , DIMENSION(1:n) :: sorted  ! sorted array
INTEGER :: i                     ! counter for do loop
```

有幾個變數完全移入副程式

```
! Open input data file and read data
OPEN(unit=100,file='rain.txt')
READ(100,*) ! skip header
DO i=1,n
  READ (100,*) rain(i)
ENDDO
```

```
! Sort the data
CALL sortdata(rain,sorted,n)
```

將rain, n 傳進副程式sortdata
回傳排序後的結果sorted

```
! Output results
OPEN(unit=101,file='rain_sorted.txt')
! Write the original data
WRITE(101,*) ' original data is '
DO i=1,n
  WRITE(101,999) rain(i)
999 format(1x, f7.1)
ENDDO
! Write the sorted data
WRITE(101,*) ' Sorted data is '
DO i=1,n
  WRITE(101,999) sorted(i)
ENDDO

END PROGRAM sort_sub
```