9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 使用子程序必须熟练掌握形参、实参及其数据传递关系。
- 定义子程序,一般要有若干形式参数(形参、哑元、虚元)。
- 调用子程序,一般要有若干实在参数(实参、实元)。
- 形实参结合规则：第I个实参与第I个形参结合。
- 形参在子程序调用前被看作形式符号，不占据存储单元，没有任何实际数据。形参在子程序调用执行过程中才占据存储单元，才有具体数据。
- 形实参之间有两种数据传递关系：引用传递、值传递。
  - 引用传递(换名传递、地址传递)：将实参地址传递给形参,形实参占相同存储空间，引用形参同引用实参。
  - 值传递：将实参的值传递给形,形实参占不同存储空间，改变形参值不影响实参。 （示例）

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 形参为普通变量时，实参可以是常数、常量、函数调用、表达式、变量、数组元素、字符子串。

实参　　常数 常量 函数 表达式 变量 数组元素 子串

形参　　普通变量

- 实参为常数、常量、函数调用、表达式，采用值传递方式。将实参值传递给对应形参。(示例1)
- 实参为变量、数组元素，采用引用传递方式。将实参地址传递给对应形参。(示例2)
- 实参为字符子串，采用引用传递方式。将子串地址传递给对应形参。(示例3)

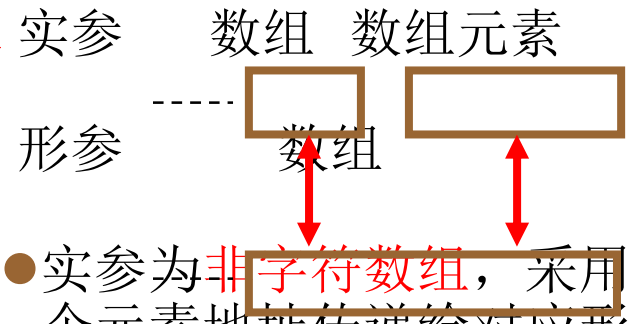
使用求最大公约数的函数Hcf，求两个正整数的最大公约数和最小公倍数。

```
read *,m,n
Print *,m, n
Print *,"2个数的最大公约数是:" ,Hcf(m, n)
Print *,"2个数的最小公倍数是:",m * n / Hcf(m, n)
CONTAINS
Function Hcf(m,n)
  INTEGER C,r
  If (m < n) Then
```

```
  c = m;   m = n;   n = c
  endif
  r =Mod(m,n)
Do While (r/=0)
          m = n  ;  n = r  ;   r =Mod(m,n)
 enddo
  Hcf = n
End Function
end
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名



- 形参为一般数组时，实参可以是数组、数组元素。形实参数组可以有不同维数，按存储结构顺序结合。
- 实参为非字符数组，采用引用传递方式。将实参数组第一个元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例1)
- 实参为非字符数组元素，采用引用传递方式。将实参数组元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例2)
- 实参为字符数组、字符数组元素，采用引用传递方式。将数组或数组元素地址传递给对应形参数组第一个元素，其后按所有元素组成长字符串对应传递。(示例3)

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

●可调数组：数组维说明由形参给定的形参数组，其大小可在实参数组范围内变化。

```
FUNCTION f(x,n)
  INTEGER  x(n)
  ……
END FUNCTION
```

●例如：统计30名学生的平均成绩。

分别用非可调数组和可调数组实现。

比较两种方法的异同点，可调数组有明显优点，实现时尽可能使用可调数需，要熟练掌握可调数组的使用规则。

可调数组大小不能超过实参数组大小。

```
!采用非可调数组实现
PROGRAM main
  INTEGER s(30)
  READ*,s
  av=f(s)
  PRINT*,av
CONTAINS
  FUNCTION f(x)
    INTEGER X(30)
    DO I=1,30
      sum=sum+x(I)
    ENDDO
    f=sum/30
  END FUNCTION
END
```

```
!采用可调数组实现
PROGRAM main
  INTEGER s(30)
  READ*,s
  av=f(s,30)
  PRINT*,av
CONTAINS
  FUNCTION f(x,n)
    INTEGER x(n)
    DO I=1,n
      sum=sum+x(I)
    ENDDO
    f=sum/n
  END FUNCTION
END
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 假定大小数组：最末维用“*”说明的形参数组，最末维由由实参数组确定。

```
SUBROUTINE sub(A,B,C,n)
  REAL A(n,n),B(10,*),C(2*n,*)
  ……
END SUBROUTINE
```

- 假定大小数组必须是形式数组。
- 假定大小数组不能出现在输入输出语句中。
- 假定大小数组可为常界数组，也可为可调数组。
- 例如：生成3×5数组数据。使用假定大小数组实现。

```
!采用假定大小数组实现
PROGRAM F914
   PARAMETER(m=3,n=5)
   DIMENSION A(m,n)
   CALL sub(A,m)
   WRITE(*,'(1X,5F5.0)') ((A(I,J),J=1,n),I=1,m)
CONTAINS
  SUBROUTINE sub(B,m)
  DIMENSION B(m,*)
  DO I=1,m
    DO J=1,m
      B(I,J)=I+J-1
    ENDDO
  ENDDO
  END SUBROUTINE
END
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 假定形状数组是不指定数组维说明上下界的形式数组。

```
SUBROUTINE sub(A,B,C)
  REAL A(:,:),B(:,:),C(:,:)
  ……
END SUBROUTINE
```

- 假定形状数组必须是形式数组。
- 假定形状数组可出现在输入输出语句中。
- 假定形状数组不能为常界数组和可调数组。
- 例如：生成3×5数组数据。使用假定形状数组实现。

```
!采用假定形状数组实现
PROGRAM F914
   PARAMETER(m=3,n=5)
   DIMENSION A(m,n)
   CALL sub(A)
   WRITE(*,'(1X,5F5.0)') ((A(I,J),J=1,n),I=1,m)
CONTAINS
  SUBROUTINE sub(B)
  DIMENSION B(:,:)
  DO I=LBOUND(B,1),UBOUND(B,1)
    DO J=LBOUND(B,2),UBOUND(B,2)
      B(I,J)=I+J-1
    ENDDO
  ENDDO
  END SUBROUTINE
END
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 形式子程序(函数、子例行程序)：用作形参的子程序。

```
SUBROUTINE proc(x,y,p)
  INTEGER x,y
  CALL p(x,Y)
  ……
END SUBROUTINE
```

- 对于形式子程序，可以使用标准子程序(如：SIN，COS，EXP等)或用户自定义子程序作为实参子程序调用。
- 在主程序中，一般用INTRINSIC语句声明标准子程序。
- 在主程序中，一般用EXTERNAL语句声明自定义子程序。
- 假定形状数组不能为常界数组和可调数组。
- 示例。

```
!形式子程序示例
PROGRAM ext
  INTRINSIC SIN
  EXTERNAL prt
   REAL f1
  f1=fun(3.1415926/6,SIN)
  PRINT*,'f1=',f1
  CALL proc(10,20,prt)
CONTAINS
  FUNCTION fun(x,f)
    REAL x
    fun=f(x)
  END FUNCTION
  SUBROUTINE
 proc(x,y,p)
    INTEGER x,y
    CALL p(x+y)
  END SUBROUTINE
END
SUBROUTINE prt(x)
  INTEGER x
  PRINT*,'x=',x
END SUBROUTINE
!输出结果:
!f1=  0.5000000
!x=       30
```

9.5 子程序形实参之间的数据传递关系

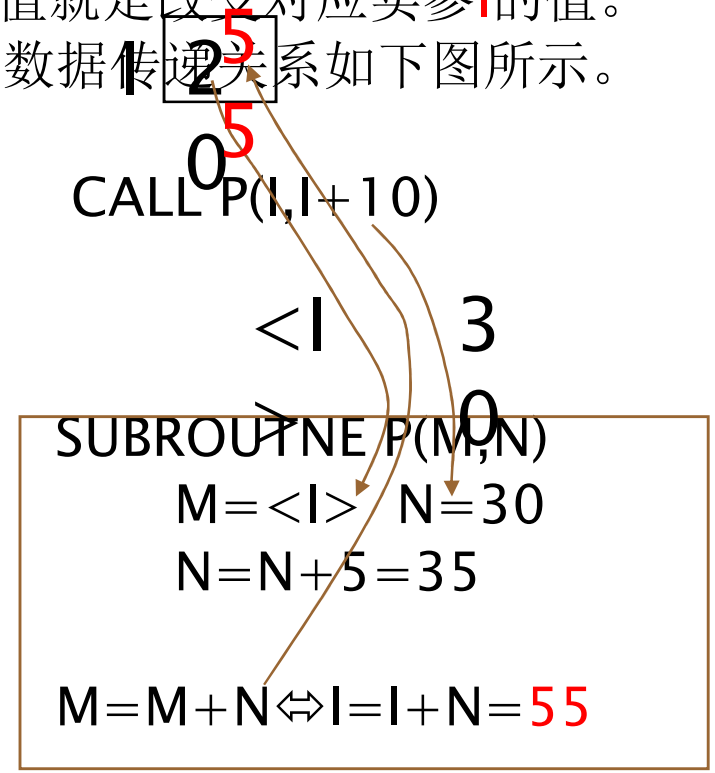
- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 使用子程序必须熟练掌握形参、实参及其数据传递关系。
- 定义子程序,一般要有若干形式参数(形参、哑元、虚元)。
- 调用子程序,一般要有若干实在参数(实参、实元)。
- 形实参结合规则：第I个实参与第I个形参结合。
- 形参在子程序调用前被看作形式符号，不占据存储单元，没有任何实际数据。形参在子程序调用执行过程中才占据存储单元，才有具体数据。
- 形实参之间有两种数据传递关系：引用传递、值传递。
  - 引用传递(换名传递、地址传递)：将实参地址传递给形参,形实参占相同存储空间，引用形参同引用实参。
  - 值传递：将实参的值传递给形,形实参占不同存储空间，改变形参值不影响实参。 （示例）

```
PROGRAM main
  INTEGER :: I=20
  PRINT*,I
  CALL P(I,I+10)
  PRINT*,I
CONTAINS
  SUBROUTNE P(M,N)
    INTEGER M,N
    N=N+5
    M=M+N
  END SUBROUTINE
END
```

输出结果：20
55

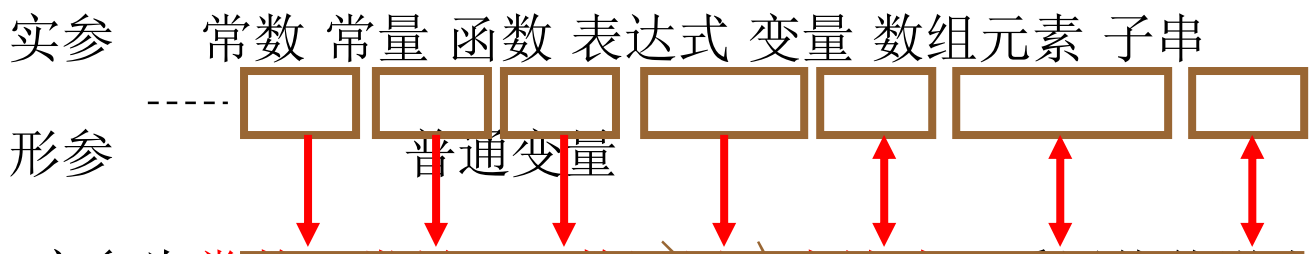
- 实参I与形参M采用引用传递方式,将地址传递给M
- 实参I+10与形参N采用值传递方式，将值传递给N
- 调用执行子程序P，形参N获得实参值30，改变N对对应实参无影响，形参M获得实参I的地址，改变M的值就是改变对应实参I的值。
- 形实参数据传递关系如下图所示。



9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

●形参为普通变量时，实参可以是常数、常量、函数调用、表达式、变量、数组元素、字符子串。



●实参为常数、常量、函数调用、表达式，采用值传递方式。将实参值传递给对应形参。(示例1)

●实参为变量、数组元素，采用引用传递方式。将实参地址传递给对应形参。(示例2)

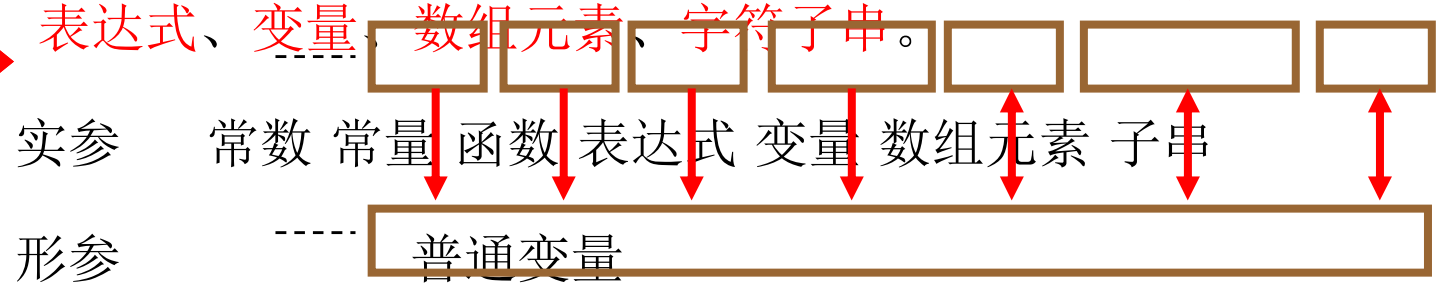
●实参为字符子串，采用引用传递方式。将子串地址传递给对应形参。(示例3)

```
PROGRAM AAA
  I=10
  CALL sub(I+I,20,I)
  WRITE(*,*) I
CONTAINS
SUBROUTINE sub(J,N,K)
  WRITE(*,*) J,N,K
  K=J*2+N-K+10
END SUBROUTINE
END
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

●形参为普通变量时，实参可以是常数、常量、函数调用、表达式、变量、数组元素、字符子串。



●实参为常数、常量、函数调用、表达式，采用值传递方式。将实参值传递给对应形参。(示例1)

●实参为变量、数组元素，采用引用传递方式。将实参地址传递给对应形参。(示例2)

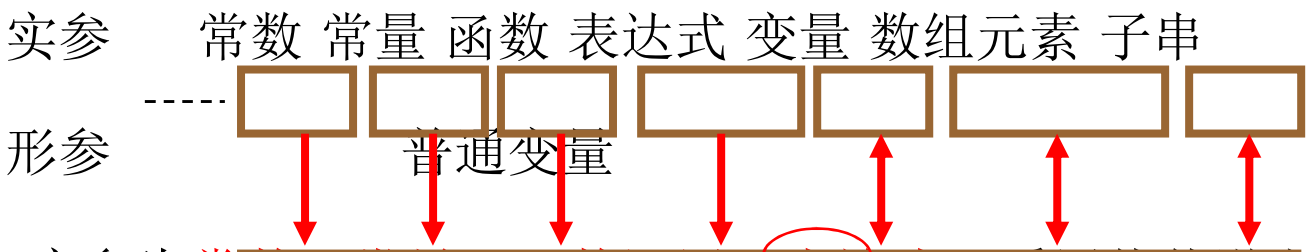
●实参为字符子串，采用引用传递方式。将子串地址传递给对应形参。(示例3)

```
PROGRAM BBB
   INTEGER A(10)
  DO J=1,10
    A(J)=J+10
  ENDDO
  I=4
  WRITE(*,"(1X,2I4)") A(I),I
  CALL sub(A(I),I)
  WRITE(*,"(1X,2I4)") A(I),I
CONTAINS
SUBROUTINE sub(B,J)
  INTEGER      B,J
  WRITE(*,"(1X,2I4)") B,J
  B=2*B;J=J+1;
  WRITE(*,"(1X,2I4)") B,J
END SUBROUTINE
END
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 形参为普通变量时，实参可以是常数、常量、函数调用、表达式、变量、数组元素、字符子串。



- 实参为常数、常量、函数调用、表达式，采用值传递方式。将实参值传递给对应形参。(示例1)
- 实参为变量、数组元素，采用引用传递方式。将实参地址传递给对应形参。(示例2)
- 实参为字符子串，采用引用传递方式。将子串地址传递给对应形参。(示例3)

```
PROGRAM CCC
  CHARACTER *10 :: ch='My name is baiyun.‘
  I=3
  CALL  csub(ch(I+1:7))
CONTAINS
SUBROUTINE csub(charl)
  CHARACTER*4 charl
  WRITE(*,*) charl
END SUBROUTINE
END
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 形参为一般数组时，实参可以是数组、数组元素。形实参数组可以有不同维数，按存储结构顺序结合。

实参　　数组　数组元素

形参　　数组

- 实参为非字符数组，采用引用传递方式。将实参数组第一个元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例1)
- 实参为非字符数组元素，采用引用传递方式。将实参数组元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例2)
- 实参为字符数组、字符数组元素，采用引用传递方式。将数组或数组元素地址传递给对应形参数组第一个元素，其后按所有元素组成长字符串对应传递。(示例3)

```
PROGRAM main
  PARAMETER(N=3)
  INTEGER D(N,N),C(N*N)
  DO I=1,N
    READ(*,*) (D(I,J),J=1,N)
  ENDDO
  CALL dim(D,C)
  DO I=1,N
    PRINT *, (D(I,J),J=1,N)
  ENDDO
  PRINT*,(C(I),I=1,N*N)
CONTAINS
  SUBROUTINE dim(A,B)

    ……
  END SUBROUTINE
END
```

输入数据：

```
1  2  3
4  5  6
7  8  9
```

输出结果为：

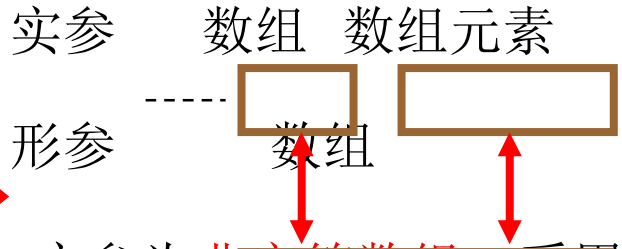
```
1  4  7
2  5  8
3  6  9
1  2  3  4  5  6  7  8  9
```

	1	2	3	4	5	6	7	8	9
数组D	D(1,1)	D(2,1)	D(3,1)	D(1,2)	D(2,2)	D(3,2)	D(1,3)	D(2,3)	D(3,3)
数组A	A(1,1)	A(2,1)	A(3,1)	A(1,2)	A(2,2)	A(3,2)	A(1,3)	A(2,3)	A(3,3)
	1	2	3	4	5	6	7	8	9
数组C	C(1)	C(2)	C(3)	C(4)	C(5)	C(6)	C(7)	C(8)	C(9)
数组B	B(1,1)	B(2,1)	B(3,1)	B(1,2)	B(2,2)	B(3,2)	B(1,3)	B(2,3)	B(3,3)

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

●形参为一般数组时，实参可以是数组、数组元素。形实参数组可以有不同维数，按存储结构顺序结合。



●实参为非字符数组，采用引用传递方式。将实参数组第一个元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例1)

●实参为非字符数组元素，采用引用传递方式。将实参数组元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例2)

●实参为字符数组、字符数组元素，采用引用传递方式。将数组或数组元素地址传递给对应形参数组第一个元素，其后按所有元素组成长字符串对应传递。(示例3)

```
PROGRAM main
  PARAMETER(N=3)
  INTEGER D(N,N),C(N*N)
  DO I=1,N
    READ(*,*) (D(I,J),J=1,N)
  ENDDO
  CALL dim(D,C)
  DO I=1,N
    PRINT *, (D(I,J),J=1,N)
  ENDDO
  PRINT*,(C(I),I=1,N*N)
CONTAINS
  SUBROUTINE dim(A,B)
    ......
  END SUBROUTINE
END
```

输入数据：

```
1  2  3
4  5  6
7  8  9
```

输出结果为：

```
1  4  7
2  5  8
3  6  9
1  2  3  4  5  6  7  8  9
```

	1	2	3	4	5	6	7	8	9
数组D	D(1,1)	D(2,1)	D(3,1)	D(1,2)	D(2,2)	D(3,2)	D(1,3)	D(2,3)	D(3,3)
数组A	A(1,1)	A(2,1)	A(3,1)	A(1,2)	A(2,2)	A(3,2)	A(1,3)	A(2,3)	A(3,3)

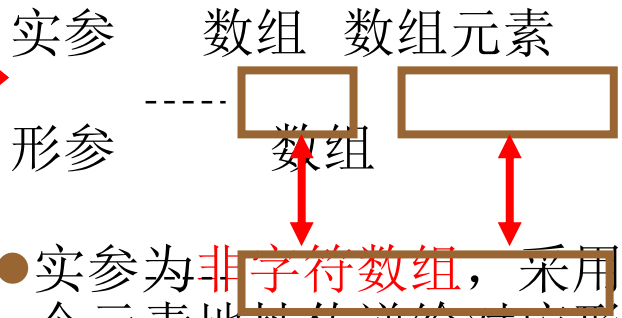
	1	2	3	4	5	6	7	8	9
数组C	C(1)	C(2)	C(3)	C(4)	C(5)	C(6)	C(7)	C(8)	C(9)
数组B	B(1,1)	B(2,1)	B(3,1)	B(1,2)	B(2,2)	B(3,2)	B(1,3)	B(2,3)	B(3,3)

```
SUBROUTINE dim(A,B)
  INTEGER A(N,N),B(N,N)
  DO I=1,N
    DO J=I+1,N
      K=A(I,J); A(I,J)=A(J,I); A(J,I)=K
    ENDDO
  ENDDO
  DO I=1,N
    DO J=1,N
      B(I,J)=A(I,J)
    ENDDO
  ENDDO
END SUBROUTINE
```

9.5 子程序形实参之间的数据传递关系

- 概述
- 形式参数
  - 普通变量
  - 一般数组
  - 可调数组
  - 假定大小
  - 假定形状
  - 子程序名

- 形参为一般数组时，实参可以是数组、数组元素。形实参数组可以有不同维数，按存储结构顺序结合。



- 实参为非字符数组，采用引用传递方式。将实参数组第一个元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例1)
- 实参为非字符数组元素，采用引用传递方式。将实参数组元素地址传递给对应形参数组第一个元素，其后元素按存储结构顺序对应传递。(示例2)
- 实参为字符数组、字符数组元素，采用引用传递方式。将数组或数组元素地址传递给对应形参数组第一个元素，其后按所有元素组成长字符串对应传递。(示例3)

```
PROGRAM main
   REAL A(9)
   CALL sub(A(4))
CONTAINS
   SUBROUTINE sub(B)
     REAL B(2,2)
      …
   END SUBROUTINE
END
```

	1	2	3	4	5	6	7	8	9
数组 A	A(1)	A(2)	A(3)	A(4)	A(5)	A(6)	A(7)	A(8)	A(9)
数组 B				B(1, 1)	B(2, 1)	B(1, 2)	B(2, 2)		

9.5 子程序形实参之间的数据传递关系

```
PROGRAM main
  CHARACTER *5 ch(5)
  DATA ch / '12345','BBBBB','CCCCC','DDDDD','EEEEE' /
  CALL sub(ch)
CONTAINS
    SUBROUTINE sub(B)
    CHARACTER *4  B(5)
    WRITE(*,100)(B(I),I=1,5)
100 FORMAT(1X,' ',A)
    END SUBROUTINE
END
```

	ch(1)	ch(2)	ch(3)	ch(4)	ch(5)
数组ch	12345	BBBBB	CCCCC	DDDDD	EEEEE
数组B	1234	5 BBB	BBCC	CCCD	DDDD
	B(1)	B(2)	B(3)	B(4)	B(5)

9.6 重入口语句(ENTRY语句)

- 通常调用子程序从子程序第1条可执行语句开始执行。
- 使用ENTRY语句可实现从子程序中任意语句处开始执行。
- 含有ENTRY语句的子程序结构为：
- 含有ENTRY语句的子程序示例：

```
PROGRAM 918
READ *,num
IF (num .GE. 0) THEN
  CALL sign
ELSE
  CALL negative
END IF
END
```

```
SUBROUTINE sign
 PRINT *, "It's
  positive.“
 RETURN
ENTRY negative
 PRINT*,"It's
  negative."
 RETURN
END SUBROUTINE
```

9.6 重入口语句(ENTRY语句)

- ENTRY语句为非执行语句，可出现在程序的任何位置。
- 调用执行子程序可从一ENTRY语句开始执行至程序结束。
- ENTRY 语句的一般形式：ENTRY <name>([形式参数表])
- 调用多重入口子程序的一般形式：<name>([实在参数表]) 或 CALL <name>([实在参数表])
- ENTRY语句声明和调用与函数和子例行程序相同。

9.6 重入口语句(ENTRY语句)

- 概述
- 语句形式
- 语句限制
- 应用示例

- 在子程序内,重入口名不能用作形参。
- 在子程序体内可出现多个ENTRY语句。
- 重入口名不能当作外部过程出现在EXTERNAL语句中。
- ENTRY语句可出现在FUNCTION语句或SUBROUTINE语句之后的任何地方,但不能出现在块IF、DO或DO WHILE语句之内。
- ENTRY语句中形参的名字、个数、次序和类型与所在子程序的FUNCTION或SUBROUTINE或其他的ENTRY语句中形参的名字、个数、次序和类型可以不一致。
- ENTRY语句之后不能出现显式的任何说明语句。
- 若ENTRY语句在外部函数体中出现,则其入口函数的类型可在ENTRY语句前的外部函数说明语句部分给出说明。
- 在外部函数中,如果name是字符类型,那么在函数中的所有name都必须为字符型,且所有name的长度必须相同。

9.6 重入口语句(ENTRY语句)


```
integer add,sub,mul,div
print*,add(20,10),sub(20,10),mul(20,
  10),div(20,10)
end
function arithmetic(x,y)
integer arithmetic,x,y,add,sub,mul,div
entry add(x,y)
  add=x+y; return
entry sub(x,y)
  sub=x+y; retu
entry mul(x,y)
  mul=x*y; return
entry div(x,y)
  div=x/y; return
end function
```

输出结果:

```
  30     10     200     2
```

Back

9.6 重入口语句(ENTRY语句)

- 概述
- 语句形式
- 语句限制
- 应用示例

- 通常调用子程序从子程序第1条可执行语句开始执行。
- 使用ENTRY语句可实现从子程序中任意语句处开始执行。
- 含有ENTRY语句的子程序结构为:
- 含有ENTRY语句的子程序示例:

```
SUBROUTINE <子程序名>(<形参表>)
ENTRY  <入口名1>[([<形参表>])]
ENTRY  <入口名2>[([<形参表>])]
… …
ENTRY  <入口名n>[([<形参表>])]
… …
END SUBROUTINE
FUNCTION  <外部函数名>([<形参表>])
ENTRY  <入口名1>[([<形参表>])]
ENTRY  <入口名2>[([<形参表>])]
… …
ENTRY  <入口名n>[([<形参表>])]
… …
END FUNCTION
```

9.7 递归子程序

- 递归定义
- 递归函数
- 递归子例行程序
- 示例

●递归子程序:子程序体内直接或间接地调用子程序自身的子程序。称这种子程序调用为递归调用。

●递归子程序有两种:递归函数和递归子例行程序。

●对于递推公式或递归问题,可使用递归子程序实现。

如n！表示为下面递推公式,可使用递归函数求解。

```
RECURSIVE FUNCTION factorial(n) RESULT (fact)
INTEGER n,fact
  IF (n= =1) THEN
     fact=1
  ELSE
     fact=n*factorial(n-1)
  ENDIF
END FUNCTION
```

$$n!=\begin{cases}1 & n=1\\ n*(n-1)! & n>1\end{cases}$$

9.7 递归子程序

- 递归定义
- 递归函数
- 递归子例行程序
- 示例

●一般格式:

RECURSIVE FUNCTION <name>([形参表])[RESULT(结果名)]

{说明语句部分}

{执行语句部分}

[内部子程序部分]

END FUNCTION [<name>]

例：使用递归函数计算阶乘n!,1≤n≤10。

解：将阶乘表示为递推公式。

$$n! = \begin{cases} 1 & n=1 \\ n*(n-1)! & n>1 \end{cases}$$

编写程序。

9.7 递归子程序

- 递归定义
- 递归函数
- 递归子例行程序
- 示例

●一般格式:

RECURSIVE SUNROUTINE <name>([形参表])
    {说明语句部分}
    {执行语句部分}
    [内部子程序部分]
END SUBROUTINE [<name>]

例：使用递归子例行程序计算阶乘n!,1≤n≤10。

解：将阶乘表示为递推公式。

编写程序。

$$n! = \begin{cases} 1 & n=1 \\ n*(n-1)! & n>1 \end{cases}$$

例：使用递归函数计算下面勒让德多项式的值。
解：根据递推公式编写递归函数。

```
RECURSIVE  FUNCTION  pn(n,x)  RESULT (pnx)
    REAL x,pnx
    IF (n= =0) THEN
      pnx=1
    ELSE IF (n= =1) THEN
      pnx=x
    ESLE
      pnx=((2*n-1))*pn(n-1,x)-(n-1)*pn(n-2,x))/n
    ENDIF
  END FUNCTION
```

Back

$$p_n(x) = \begin{cases} 1 & n=0 \\ x & n=1 \\ ((2n-1)p_{n-1}(x)-(n-1)p_{n-2}(x))/n & n>1 \end{cases}$$

9.7 递归子程序

- 递归定义
- 递归函数
- 递归子例行程序
- 示例

●一般格式:

RECURSIVE FUNCTION <name>([形参表])[RESULT(结果名)]

{说明语句部分}

{执行语句部分}

[内部子程序部分]

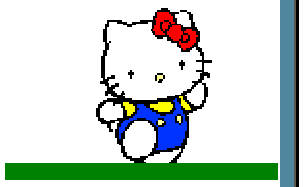
END FUNCTION [<name>]

例：使用递归函数计算阶乘n!,1≤n≤10。

解：将阶乘表示为递推公式。

$$n! = \begin{cases} 1 & n=1 \\ n*(n-1)! & n>1 \end{cases}$$

编写程序。



```
PROGRAM F919
INTEGER  n
PRINT *,'请输入阶乘的阶数(小于10):'
READ *,n
WRITE（*,100） n,factorial(n)
100 FORMAT(I3,'!=',I12)
CONTAINS
RECURSIVE FUNCTION factorial(m) RESULT (fact)
  INTEGER m,fact
  IF (m= =1) THEN
    fact=1
  ELSE
    fact
  ENDI
END FU
END
```

输入数据：5
输出结果：5!= 120

执行过程分析

9.7 递归子程序

- 递归定义
- 递归函数
- 递归子例行程序
- 示例

●一般格式:

RECURSIVE FUNCTION <name>([形参表])[RESULT(结果名)]

{说明语句部分}

{执行语句部分}

[内部子程序部分]

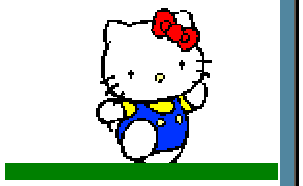
END FUNCTION [<name>]

例：使用递归函数计算阶乘n!,1≤n≤10。

解：将阶乘表示为递推公式。

$$n! = \begin{cases} 1 & n=1 \\ n*(n-1)! & n>1 \end{cases}$$

编写程序。



```
PROGRAM F919
INTEGER  n
PRINT *,'请输入阶乘的阶数(小于10):'
READ *,n
WRITE（*,100） n,factorial(n)
100 FORMAT(I3,'!=',I12)
CONTAINS
RECURSIVE FUNCTION factorial(m) RESULT (fact)
  INTEGER m,fact
  IF (m= =1) THEN
    fact=1
  ELSE
    fact=m*factorial(m-1)
  ENDIF
END FUNCTION
END
```

执行过程分析

表 9-2　factorial(5)递归函数调用执行结果的分析过程

第一层调用	第二层调用	第三层调用	第四层调用	第五层调用
factorial(5) m=5≠1 m*factorial(m-1) 5*factorial(4) 返回 120	调用 factorial(4) m=4≠1 m*factorial(m-1) 4*factorial(3) 返回 24	调用 factorial(3) m=3≠1 m*factorial(m-1) 3*factorial(2) 返回 6	调用 factorial(2) m=2≠1 m*factorial(m-1) 2*factorial(1) 返回 2	调用 factorial(1) m=1 fact=1 返回 1

9.7 递归子程序

- 递归定义
- 递归函数
- 递归子例行程序
- 示例

●一般格式:

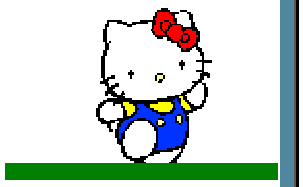
RECURSIVE SUNROUTINE <name>([形参表])
    {说明语句部分}
    {执行语句部分}
    [内部子程序部分]
END SUBROUTINE [<name>]

例：使用递归子例行程序计算阶乘n!,1≤n≤10。

解：将阶乘表示为递推公式。

编写程序。

$$n! = \begin{cases} 1 & n=1 \\ n*(n-1)! & n>1 \end{cases}$$



```
PROGRAM F920
INTEGER  n,f
READ *,n
CALL factorial(n,f)
WRITE（*,100） n,f
100 FORMAT(I3,'!=',I12)
CONTAINS
RECURSIVE SUBROUTINE  factorial(m,fact)
  INTEGER m,fact
  IF (m= =1) THEN
    fact=1
  ELSE
    CALL factorial(m-1,fact)
    fact=m*fact
  ENDIF
END SUBROUTINE
END
```

输入数据：5
输出结果：5!= 120

9.8 外部子程序

- 内部子程序：在程序单元CONTAINS结构内定义的子程序。内部子程序与主程序一起保存在一个源程序文件中，并与主程序一起输入、修改、编译。（示例1）
- 外部子程序：在主程序之外定义的子程序。外部子程序可与主程序一起保存在一个源程序文件中(放置在主程序PROGRAM语句之前或END语句之后)，也可与主程序分别保存在不同的源程序文件中，并与主程序分别输入、修改、编译，分别生成OBJ文件。（示例2）
- 使用外部子程序可实现程序的并行设计和编写工作，提高程序设计的效率。
- 主程序或其它程序单元中调用外部子程序，则需使用EXTERNAL语句声明所调用的外部子程序。如果外部子程序与主程序放在一个源程序文件中，则EXTERNAL语句不写。（示例3）

9.8 外部子程序

- <外部函数>→[<类型>] FUNCTION <name>([<形参表>])
  {<说明语句>}
  {<执行语句>}
  [CONTAINS
  {<内部子程序>}]
  END [FUNCTION [<函数名>]]
- <外部子例行程序>→SUBROUTINE <name>[([<形参表>])]
  {<说明语句>}
  {<执行语句>}
  [CONTAINS
  {<内部子程序>}]
  END [SUBROUTINE [<子例行程序名>]]
- 外部子程序内可含CONTAINS结构,允许定义内部子程序。

9.8 外部子程序

- <外部函数调用>→<name>([<实参表>])
- <内部子例行程序调用>→CALL <name>[([<实参表>])]
- 外部子程序调用方法同内部子程序。

9.8 外部子程序

- 概述
- 定义
- 调用
- EXTERNAL
- 示例1
- 示例2

- <EXTERNAL语句>→[<类型>]EXTERNAL[::]<外部子程序名表>
- EXTERNAL语句必须放置在USE和IMPLICIT语句之后。
- EXTERNAL语句有两个作用：
  - 用于解决外部子程序与系统预定义子程序的冲突问题。用EXTERNAL语句定义的外部子程序优先同名系统预定义子程序。如果不冲突，则可取消EXTERNAL语句。
  - 用于提高程序的可读性,程序中是否使用、使用哪些、使用几个外部子程序能一目了然,建议使用该语句。

[9.22]设计和编写计算标准偏差外部函数。

解：

计算平均值：

$$X_a = \frac{(X_1+X_2+X_3+X_4+X_5+X_6+X_7+X_8+X_9+X_{10})}{10}$$

计算均方差：

$$X_s = \sqrt{\frac{\sum_{i=1}^{10}(X_i - X_a)^2}{9}}$$

主程序单元为：

不使用内部函数的计算标准偏差的外部函数为：

使用内部函数的计算标准偏差的外部函数为：

输入10个实数：

5.2、4.3、3.5、8.8、9.5、7.9、2.4、4.4、5.8、9.4

输出结果为：

标准偏差为：　2.59178

[9.23]设计和编写矩阵相加的外部子例行程序。

解：已知矩阵$A_{m\times n}$、$B_{m\times n}$,求$C_{m\times n}=A_{m\times n}+B_{m\times n}$。

主程序单元为：

输入矩阵数据的外部子例行程序为：

输出矩阵数据的外部子例行程序为：

计算矩阵之和的外部子例行程序为：

输入两个3×4矩阵数据(整数)：

```
   12 34 41 55        10 10 10 10
A: 10 21 33 18     B: 20 20 20 20
   50 42 31 14        30 30 30 30
```

输出结果矩阵为：

```
22  44  51  65
30  41  53  38
80  72  61  44
```

Back

9.8 外部子程序

- 概述
- 定义
- 调用
-

EXTERNAL

- 示例1
- 示例2

- 内部子程序：在程序单元CONTAINS结构内定义的子程序。内部子程序与主程序一起保存在一个源程序文件中，并与主程序一起输入、修改、编译。（示例1）
- 外部子程序：在主程序之外定义的子程序。外部子程序可与主程序一起保存在一个源程序文件中(放置在主程序PROGRAM语句之前或END语句之后)，也可与主程序分别保存在不同的源程序文件中，并与主程序分别输入、修改、编译，分别生成OBJ文件。（示例2）
- 使用外部子程序可实现程序的并行设计和编写工作，提高程序设计的效率。
- 主程序或其它程序单元中调用外部子程序，则需使用EXTERNAL语句声明所调用的外部子程序。如果外部子程序与主程序放在一个源程序文件中，则EXTERNAL语句不写。（示例3）

```
!使用内部函数计算n个学生的平均成绩，如：n=5
PROGRAM main
INTEGER,PARAMETER :: max=100
INTEGER :: grade(max),n=5,i
REAL av
WRITE(*,"(1X,'输入',I3,'个学生的成绩数据(整数)：')") n
READ*,(grade(i),i=1,n)
!调用内部函数av,求n个数的平均值。将平均值从显示器输出
WRITE(*,"(1X,I3,'个学生平均成绩为：',F5.2)") n,av(grade,n)
CONTAINS
!定义一个内部函数子程序单元,m个数据保存于数组arr中。
REAL FUNCTION  av(arr,m)
INTEGER arr(m),m,sum
sum=0
DO i=1,m
  sum=sum+arr(i)
ENDDO
av=sum/m
END FUNCTION
END
```

9.8 外部子程序

- 概述
- 定义
- 调用
-

EXTERNAL

- 示例1
- 示例2

- 内部子程序：在程序单元CONTAINS结构内定义的子程序。内部子程序与主程序一起保存在一个源程序文件中，并与主程序一起输入、修改、编译。（示例1）
- 外部子程序：在主程序之外定义的子程序。外部子程序可与主程序一起保存在一个源程序文件中(放置在主程序PROGRAM语句之前或END语句之后)，也可与主程序分别保存在不同的源程序文件中，并与主程序分别输入、修改、编译，分别生成OBJ文件。（示例2）
- 使用外部子程序可实现程序的并行设计和编写工作，提高程序设计的效率。
- 主程序或其它程序单元中调用外部子程序，则需使用EXTERNAL语句声明所调用的外部子程序。如果外部子程序与主程序放在一个源程序文件中，则EXTERNAL语句不写。（示例3）

```
!使用外部函数计算n个学生的平均成绩，如：n=5
!主程序
PROGRAM main
EXTERNAL av  !EXTERNAL 语句声明一个外部函数子程序av
INTEGER,PARAMETER :: max=100
INTEGER :: grade(max),n=5,i
REAL av
WRITE(*,"(1X,'输入',I3,'个学生的成绩数据(整数)：')") n
READ*,(grade(i),i=1,n)
!调用外部函数av,求n个数的平均值。将平均值从显示器输出
WRITE(*,"(1X,I3,'个学生平均成绩为：',F5.2)") n,av(grade,n)
END
!定义一个外部函数子程序单元,m个数据保存于数组arr中。
REAL FUNCTION  av(arr,m)
INTEGER arr(m),m,sum
sum=0
DO i=1,m
  sum=sum+arr(i)
ENDDO
av=sum/m
END FUNCTION
```

9.8 外部子程序

- 概述
- 定义
- 调用
-

EXTERNAL

- 示例1
- 示例2

- 内部子程序：在程序单元CONTAINS结构内定义的子程序。内部子程序与主程序一起保存在一个源程序文件中，并与主程序一起输入、修改、编译。（示例1）
- 外部子程序：在主程序之外定义的子程序。外部子程序可与主程序一起保存在一个源程序文件中(放置在主程序PROGRAM语句之前或END语句之后)，也可与主程序分别保存在不同的源程序文件中，并与主程序分别输入、修改、编译，分别生成OBJ文件。（示例2）
- 使用外部子程序可实现程序的并行设计和编写工作，提高程序设计的效率。
- 主程序或其它程序单元中调用外部子程序，则需使用EXTERNAL语句声明所调用的外部子程序。如果外部子程序与主程序放在一个源程序文件中，则EXTERNAL语句不写。（示例3）

```
!使用外部函数计算n×n方阵对角线元素之和，如：n=5
PROGRAM main      !主程序
EXTERNAL fsum     !EXTERNAL 语句声明一个外部子例行程序fsum
INTEGER,PARAMETER :: n=5
INTEGER :: matrix(n,n),i,j,s1,s2
WRITE(*,"(1X,'以行为主输入',I2,'×',I2,'方阵数据(整数)。')") n,n
READ*,((matrix(i,j),j=1,n),i=1,n)
!调用外部子例行程序fsum,求两对角线元素之和,分别放在s1和s2中。
CALL fsum(matrix,n,s1,s2)
WRITE(*,"(1X,'左上右下对角线元素之和为：',I4)") s1
WRITE(*,"(1X,'左下右上对角线元素之和为：',I4)") s2
END
!定义一个外部子例行程序单元,对角线元素分别放在s1和s2中。
SUBROUTINE fsum(arr,m,sum1,sum2)
INTEGER :: arr(m,m),m,sum1=0,sum2=0
DO i=1,m
  sum1=sum1+arr(i,i)
ENDDO
DO i=1,m
  sum2=sum2+arr(m-i+1,i)
ENDDO
END SUBROUTINE
```

9.8 外部子程序

- 概述
- 定义
- 调用
- EXTERNAL
- 示例1
- 示例2

[9.22]设计和编写计算标准偏差外部函数。

解：

计算平均值：

$$X_a = \frac{(X_1+X_2+X_3+X_4+X_5+X_6+X_7+X_8+X_9+X_{10})}{10}$$

计算均方差：

$$X_s = \sqrt{\frac{\sum_{i=1}^{10}(X_i - X_a)^2}{9}}$$

主程序单元为：

不使用内部函数的计算标准偏差的外部函数为：

使用内部函数的计算标准偏差的外部函数为：

输入10个实数：

5.2、4.3、3.5、8.8、9.5、7.9、2.4、4.4、5.8、9.4

输出结果为：

标准偏差为：　2.59178

```
!主程序
PROGRAM  s_deviation
IMPLICIT  NONE
EXTERNAL deviation     !声明一个外部函数deviation
INTEGER  :: n=10,i
REAL X(10),deviation
WRITE(*,"(1X,'请输入',I2,'个数：')") n
READ *,(X(i),i=1,n)
!调用外部函数deviation计算标准偏差,并从显示器输出
 WRITE(*,“(1X,‘标准偏差为:',F10.5)") deviation(X,n)
END
```

[9.22]设计和编写计算标准偏差外部函数。

解：

计算平均值：

$$X_a = \frac{(X_1+X_2+X_3+X_4+X_5+X_6+X_7+X_8+X_9+X_{10})}{10}$$

计算均方差：

$$X_s = \sqrt{\frac{\sum_{i=1}^{10}(X_i - X_a)^2}{9}}$$

主程序单元为:

不使用内部函数的计算标准偏差的外部函数为:

使用内部函数的计算标准偏差的外部函数为:

输入10个实数:

5.2、4.3、3.5、8.8、9.5、7.9、2.4、4.4、5.8、9.4

输出结果为:

标准偏差为:　2.59178

```
!外部函数
REAL FUNCTION deviation(X,n)
   REAL :: X(n),sum,Xa
   sum=0
   DO i=1,n
     sum=sum+X(i)
   ENDDO
   Xa=sum/n
   sum=0
   DO i=1,n
     sum=sum+(X(i)-Xa)**2
   ENDDO
   deviation=SQRT(sum/(n-1))
END FUNCTION deviation
```

[9.22]设计和编写计算标准偏差外部函数。

解：

计算平均值：$$X_a = \frac{(X_1+X_2+X_3+X_4+X_5+X_6+X_7+X_8+X_9+X_{10})}{10}$$

计算均方差：

$$X_s = \sqrt{\frac{\sum_{i=1}^{10}(X_i - X_a)^2}{9}}$$

主程序单元为：

不使用内部函数的计算标准偏差的外部函数为：

使用内部函数的计算标准偏差的外部函数为：

输入10个实数：

5.2、4.3、3.5、8.8、9.5、7.9、2.4、4.4、5.8、9.4

输出结果为：

标准偏差为：　2.59178

```
!外部函数
REAL FUNCTION deviation(X,n)
 REAL :: X(n),Xa
 Xa=av(X,n)
 deviation=SQRT(ss(X,n,Xa)/(n-1))
CONTAINS
 !计算平均值内部函数
 FUNCTION av(Y,m)
   REAL  Y(m),sum
   sum=0
   DO i=1,m
     sum=sum+X(i)
   ENDDO
   av=sum/m
 END FUNCTION av
 !计算平方和内部函数
 FUNCTION ss(Y,m,aver)
  REAL  Y(m),sum,ss
  sum=0
  DO i=1,n
   sum=sum+(Y(i)-
aver)**2
   ENDDO
   ss=sum
  END FUNCTION ss
END FUNCTION deviation
```

9.8 外部子程序

- 概述
- 定义
- 调用
- EXTERNAL
- 示例1
- 示例2

```
!主程序
PROGRAM  m_add
!声明3个外部子例行程序
EXTERNAL input,matrix_add,output
INTEGER,PARAMETER  :: m=3,n=4
INTEGER,DIMENSION(m,n) :: A,B,C
!调用执行外部子例行程序input输入A矩阵数据
CALL input(A,m,n)
!调用执行外部子例行程序input输入B矩阵数据
CALL input(B,m,n)
!调用外部子例行程序matrix_add求A、B矩阵和,得C矩阵
CALL matrix_add(A,B,C,m,n)
!调用执行外部子例行程序output输出C矩阵数据
CALL output(C,m,n)
END
```

9.8 外部子程序

- 概述
- 定义
- 调用
- EXTERNAL
- 示例1
- 示例2

[9.23]设计和编写矩阵相加的外部子例行程序。

解：已知矩阵$A_{m\times n}$、$B_{m\times n}$,求$C_{m\times n}=A_{m\times n}+B_{m\times n}$。

主程序单元为：

输入矩阵数据的外部子例行程序为：

输出矩阵数据的外部子例行程序为：

计算矩阵之和的外部子例行程序为：

输入两个3×4矩阵数据(整数)：

```
   12 34 41 55        10 10 10 10
A: 10 21 33 18     B: 20 20 20 20
   50 42 31 14        30 30 30 30
```

输出结果矩阵为：

```
22  44  51  65
30  41  53  38
80  72  61  44
```

```
!外部子例行程序
!输入矩阵数据,mat为数组,n1为行数,n2为列数
SUBROUTINE  input(mat,n1,n2)
 INTEGER mat(n1,n2),n1,n2,i,j
 WRITE(*,100) n1,n2
 WRITE(*,200) n1*n2
 READ*,((mat(i,j),j=1,n2),i=1,n1)
 100 FORMAT('以行为主输入',I2,'×',I2,'矩阵。'\)
 200 FORMAT(1X,'共',I4,'个数据:')
END SUBROUTINE
```

[9.23]设计和编写矩阵相加的外部子例行程序。

解：已知矩阵$A_{m\times n}$、$B_{m\times n}$,求$C_{m\times n}=A_{m\times n}+B_{m\times n}$。

主程序单元为：

输入矩阵数据的外部子例行程序为：

输出矩阵数据的外部子例行程序为：

计算矩阵之和的外部子例行程序为：

输入两个3×4矩阵数据(整数)：

```
   12 34 41 55        10 10 10 10
A:  10 21 33 18    B:  20 20 20 20
   50 42 31 14        30 30 30 30
```

输出结果矩阵为：

```
 22  44  51  65
 30  41  53  38
 80  72  61  44
```

```
!外部子例行程序
!输出矩阵数据,mat为数组,n1为行数,n2为列数
SUBROUTINE  output(mat,n1,n2)
 INTEGER mat(n1,n2),n1,n2,i,j
 WRITE(*,100) n1,n2
 WRITE(*,"(<n2>I4)") ((mat(i,j),j=1,n2),i=1,n1)
 100 FORMAT(1X,I2,'×',I2,'矩阵数据(整数)为:')
END SUBROUTINE
```

9.8 外部子程序

- 概述
- 定义
- 调用
- EXTERNAL
- 示例1
- 示例2

```
!外部子例行程序
!计算矩阵之和,A和B为已知矩阵，C为结果矩阵
SUBROUTINE  matrix_add(A,B,C,n1,n2)
 INTEGER n1,n2,i,j
 !声明3个n1×n2数组
 INTEGER,DIMENSION(n1,n2) :: A,B,C
 DO i=1,n1
   DO j=1,n2
     C(i,j)=A(i,j)+B(i,j)
   ENDDO
 ENDDO
END SUBROUTINE
```

习题九

1．用语句函数语句定义下列函数。

- $F(t)=1-2t+t^2$
- $Y=\dfrac{\ln(1+\sqrt{x})}{1+x^2}$
- $Y=\mathrm{arctg}\dfrac{x}{\sqrt{1-x^2}}$
- $Y=\sqrt{|ab|}$

2. 编写一个内部函数子程序求n个数的平方和 。

3．编写一个内部函数子程序ifac用来求n!,要求所得函数值为实型。函数要对变元进行检查,对不合理的变元给出错误信息,然后返回,调用此函数求：

$$C=\frac{R!}{(R-K)!K!}$$

习题九

4. 有A,B两个正数数列,从A数列中删除在B数列中出现的数,编写内部子例行程序实现之。
5. 编写内部函数子程序把十进制数转换成二进制数。二进制数用一个字符串来表示。
6. 编写一个打印信息的内部子例行程序,要求打印三种信息:
   ①PARAMETER ERROR
   ②NO SOLUTION
   ③END OF JOB
7. 输入n个字符串,请将字符串按实际长度的大小排列,编写内部子例行程序实现之。
8. 编写函数,用于检查字符是否是字母、数字或特殊字符。
9. 有m只猴子要选猴王,选举方法:所有猴子排成一列,从头到尾报数,能被n除尽者留下,其余退出。留下者再从头到尾报数,能被n除尽者留下,其余退出。反复报数,直到留下不足n只猴子,此时报1者为猴王。编写内部子例行程序实现之。

习题九

10．古代有一个塔,塔内有3个座A、B、C,开始时A座上有15个盘子,盘子大小不等,大的在下,小的在上,有一个小和尚想把这15个盘子从A座移到C座,但每次只允许移动一个盘子,而且在任何时候都必须保证3个座上的盘子是大在下小在上。编写递归子例行程序实现之。要求在移动过程中只能借助B座。

The end

文件名格式：班级 学号 姓名 简略实验名称

邮件标题同文件名

Any questions please 发送至

xingzhengwu@163.com