Métodos Computacionais em Física

Aula 05

Arranjos de dados: Matrizes, vetores e etc...

Suponha que vc deseja definir um vetor

```
REAL:: a(5)
```

O resultado será a criação de cinco espaços de memória denotados por

```
a(1)  a(2)  a(3)  a(4)  a(5)
```

Que poderão ser tratados como cinco variáveis independentes

Fazendo com que o índice não comece em 1

```
REAL:: a(-2:2)
```

```
a(-2)  a(-1)  a(0)  a(1)  a(2)
```

Como declarar

```
REAL,DIMENSION(5):: a,b,c

REAL:: d(5),e(5),f(5)

REAL,DIMENSION(5):: g,h(3),i

REAL,DIMENSION(-1:4):: j
```

Matrizes

```
REAL:: a(3,4)

a(1,1)  a(1,2)  a(1,3)  a(1,4)

a(2,1)  a(2,2)  a(2,3)  a(2,4)

a(3,1)  a(3,2)  a(3,3)  a(3,4)
```

Matrizes

```
REAL:: a(0:2,-1:2)

a(0,-1)  a(0,0)  a(0,1)  a(0,2)

a(1,-1)  a(1,0)  a(1,1)  a(1,2)

a(2,-1)  a(2,0)  a(2,1)  a(2,2)
```

Outros Arranjos

- Se um vetor é um arranjo de números A com apenas um índice: A_i
- Se uma matrix é um arranjo de números A com dois índice: A_{ij}
- Um arranjo de números com três índices é declarado como:

```
REAL:: a(2,3,5)
```

a(1,1,1)	a(1,2,1)	a(1,3,1)	a(2,1,1)	a(2,2,1)	a(2,3,1)
a(1,1,2)	a(1,2,2)	a(1,3,2)	a(2,1,2)	a(2,2,2)	a(2,3,2)
a(1,1,3)	a(1,2,3)	a(1,3,3)	a(2,1,3)	a(2,2,3)	a(2,3,3)
a(1,1,4)	a(1,2,4)	a(1,3,4)	a(2,1,4)	a(2,2,4)	a(2,3,4)
a(1,1,5)	a(1,2,5)	a(1,3,5)	a(2,1,5)	a(2,2,5)	a(2,3,5)

Escrevendo (ou lendo) dados de um arranjo

```
WRITE(1,*) (a(2,i),i=2,7)
```

Operações com arrays

```
REAL:: a(3),b(3),c(3)

c(1)=a(1)+b(1)
c(2)=a(2)+b(2)
c(3)=a(3)+b(3)
```

Operações com arrays

```
REAL:: a(3),b(3),c(3)

DO i=1,3
  c(i)=a(i)+b(i)
END DO
```

Operações com arrays

```
REAL:: a(3),b(3),c(3)

c=a+b
```

Operações com arrays

```
REAL:: a(3),b(3),c(3)


c=a+b
c=a-b
c=a*b
c=EXP(a)
c=SIN(b)
```

Operações com arrays

```
REAL:: a(3),b(3),c(5,3)

c(5,:)=a+b
c(5,:)=a-b
c(5,:)=a*b
```

Operações com arrays

```
REAL:: a(3,7),b(3),c(3)

c=a(:,7)+b
c=a(:,7)-b
c=a(:,7)*b
```

Operações com arrays

```
REAL:: a(7),b(3),c(3)

c=a(3:5)+b
c=a(3:5)-b
c=a(3:5)*b
```

Funções intrinsecas com arrays

```
MAXVAL(A)
MINVAL(A)
MINLOC(A)
MAXLOC(A)
SUM(A)
TRANSPOSE(A)
```

Matrizes alocatáveis

```
INTEGER,ALLOCATABLE:: A(:,:),B(:,:,:),v(:)
```

Como alocar?

```
ALLOCATE(A(10,20))

ALLOCATE(B(2,5,8),v(10))
```

Como desalocar?

```
DEALLOCATE(A)

DEALLOCATE(B,v)
```

Não alocar o que já está alocado
Não desalocar o que não está alocado

Como verificar antes

```
logical_var=ALLOCATED(A)
```

Mas é melhor saber de antemão se uma array está alocada ou não.

Como testar uma (des)alocação

```
REAL:: a(:)
INTEGER:: ios

ALLOCATE(a(5),IOSTAT=ios)
IF(ios.ne.0)THEN
  WRITE(*,*) 'Failed to allocate a.'
  STOP
END IF

DEALLOCATE(a,IOSTAT=ios)
IF(ios.ne.0)THEN
  WRITE(*,*) 'Failed to deallocate a.'
  STOP
END IF
```

Hands on

- 7.1
- 7.2
- 7.5
- 7.7
- 7.8
- 7.14