Arrays

1. introduction

From the five basic data types of Fortran (integer, real, character, logical, parameter) we can form a sequences of data called in programming **array**. An array is a sequence elements of the same type stored contiguously in memory. Array can have one or more dimensions. A dimension describes how to position an element according to a given reference: a dimension along a straight line; two dimensions according to a plan, etc. Particularly, arrays with only one dimension are called **vector** and arrays with two dimensions are called **matrix**.

Arrays can store a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

Numbers(1)	Numbers(2)	Numbers(3)	Numbers(4)	...

Arrays can be one- dimensional (like vectors), two-dimensional (like matrices) and Fortran allows you to create up to 7-dimensional arrays.

من أنواع البيانات (انواع المتغيرات) الخمسة الأساسية في فورتران (عدد صحيح، حقيقي، حرف، منطقي، ثابت) يمكننا تشكيل تسلسل من البيانات تسمى في لغة البرمجة جدول. **الجدول** عبارة عن عدد من المتغيرات من نفس النوع مخزنة بشكل متجاور في الذاكرة معرفة باسم واحد (اسم الجدول) وترتيب العنصر في الجدول. يمكن أن يحتوي الجدول على بُعد واحد أو أكثر. يصف البعد كيفية البلوغ لعنصر وفقًا لترتيبه : البعد على طول خط مستقيم؛ بعدان وفقًا للسطح، وما إلى ذلك. على وجه الخصوص، تسمى الجداول ذات البعد الواحد فقط **بالشعاع** و ذات البعدين تسمى **المصفوفة**.

يمكن للجداول تخزين مجموعة متسلسلة ذات حجم ثابت من العناصر من نفس النوع. يتم استخدام الجدول لتخزين مجموعة من البيانات، ولكن غالبًا ما يكون من الأفضل للمتغيرات من نفس النوع أن تكون مشتركة في العمليات و الاجراءات مثال قيم لـ درجات الحرارة, نقاط الطالب... إلخ.

2. Declaring Arrays

Arrays are declared with the **dimension** attribute.

For example, to declare a one-dimensional array named number, of real numbers containing 5 elements, you write,

```
real, dimension(5) :: numbers
```

The individual elements of arrays are referenced by specifying their subscripts. The first element of an array has a subscript of one. The array numbers contains five real variables –numbers(1), numbers(2), numbers(3), numbers(4), and numbers(5).

To create a **5 x 5** two-dimensional array of integers named matrix, you write −

```
integer, dimension (5,5) :: matrix
```

You can also declare an array with some explicit lower bound, for example −

```
real, dimension(2:6) :: numbers
integer, dimension (-3:2,0:4) :: matrix
```

3. Some Array Related Terms

The following table gives some array related terms

Term	Meaning
Rank	It is the number of dimensions an array has. For example, for the array named matrix, rank is 2, and for the array named numbers, rank is 1.
Extent	It is the number of elements along a dimension. For example, the array numbers has extent 5 and the array named matrix has extent 3 in both dimensions.
Size	It is the number of elements an array contains. For the array matrix, it is 9, and for the array numbers, it is 5.

خصائص الجداول

Rank (الرتبة) : هو عدد أبعاد الجدول . على سبيل المثال، بالنسبة للشعاع يمثل في جدول تكون الرتبة 1، وبالنسبة للمصفوفة ، تكون الرتبة 2.

Extent المدى : هو عدد العناصر على طول كل بعد. على سبيل المثال، الجدول numbers لها مدى 5 و مصفوفة (3*3) لها مدى 3 في كلا البعدين.

الحجم هو عدد العناصر التي يحتويها الجدول مثال (3*3) بالنسبة لمصفوفة هي 9

4. Operations of Arrays (vectors) العمليات على الجداول

4. 1. **Filling** ملء جدول

```
program fillArray
implicit none
  integer, dimension (5):: a
  integer :: i
   ! initialization array with one value example 0
 a=0
! initialization array with value from 1 to 5
do i = 1, 5
   a(i) = i
end do
! initialization array with user(keyboard)
do i = 1, 5
   read*,a(i)
end do
end program fillArray
```

4. **2. Calculate (sum, product, adds) :**

```
! example of  loop for calculate sum  array values

  s=0    ! initialization  value
do i = 1, 5
   s= s+ a(i)     ! add array element rank I to sum s
end do
print*,s      ! display result sum
```

4. **3. Research (max, min , nb occurrence value ,...) :**

```
! example of  loop for research  max  array values (greatest)

  m=0    ! initialization  value  max
do i = 1, 5
   if ( m < a(i) ) then    ! Conditional construct for reseach
      m = a(i)             ! Update value research max m
   end if
end do
print*,m      ! display result max
```

5. Matrix operations

```
program filling
implicit none
   integer, dimension (5,5):: a
   integer :: i ,j

! initialization array with one value example 0
  a=0
! initialization array with value from 1 to 25
k=1
do i = 1, 5
 do j = 1, 5
    a(i, j) = k
    k=k+1
  end do
 end do

! initialization array with user(keyboard)
do i = 1, 5
 do j = 1, 5
    read*,a(i, j)
  end do
 end do
```

```
end program filling
```

```
! example of  loop for calculate sum  matrix  values
```

```
  s=0    ! initialization  value
do i = 1, 5
 do j = 1,5
   s= s+ a(i, j)     ! add matrix element rank I to sum s
end do
end do
print*,s      ! display result sum
```

```
! example of  loop for research  max  matrix values (greatest)

  m=0    ! initialization  value  max
do i = 1, 5
do j = 1,5
   if ( m < a(i, j) ) then    ! Conditional construct for reseach
      m = a(i, j)             ! Update value research max m
   end if
end do
end do
print*,m      ! display result max
```