

ARRAYS AND MATRICES

In an array, multiple values of the same data type can be stored with one variable name. In computer, array elements are stored in a sequence of adjacent memory locations. Arrays are of two types:

1. *One dimensional array.*
2. *Multi-dimensional array.*

1. ONE DIMENSIONAL ARRAY

The position of an element in array is called **array index** or **subscript**. In the case of an array of five elements $A[4]=\{6, 7, 8, 9,\}$, their index or subscript values are 0, 1, 2, and 3. Note that count for array elements or subscripts starts from 0 as shown below.



The declaration of **one dimensional array** is done as illustrated below.

type name [number of elements in the array] ;

for example:

```
int A[10]; // Array 'A' has 10 elements of type integer.
float B[20]; // Array 'B' has 20 elements of type float.
double D[15]; // Array 'D' has 15 elements of type double.
char name[20]; // Array 'name' has 20 elements of type char.
```

INPUT/OUTPUT OF ONE DIMENSIONAL ARRAY

The input/output of an array is carried out element by element either a *for* loop or *while* loop may be used. For example, an array $Bill[5]$ having n elements are to be read as follow:

```
for (int i = 0; i<5; i++)
    cin>> Bill[i] ;
```

An array can be read by another way called “**static initialization**” as shown:

```
int Bill[5]={10, 20, 30,40, 50}; إعطاء قيم مباشرة عند تعريف المصفوفة
```

and the output (printing) is as follows:

```
for (int i = 0; i<5; i++)
    cout<< Bill[i]<<" "; لطباعتها على شكل صف
```

OR

```
for (int i = 0; i<5; i++)
    cout<< Bill[i]<<endl; لطباعتها على شكل عمود
```

2. TWO DIMENSIONAL ARRAYS (MATRIX)

The two dimensional array is represented by i^{th} rows and j^{th} columns. The figure below shows an array of two rows and five columns.

A[0][0] = 5		0	1	2	3	4
A[0][1] = 2	0	5	2	3	2	4
A[1][0] = 6	1	6	7	8	9	8
A[1][3] = 9						

A two dimensional array can be declared as below.

```
type name [number of rows][number of columns];
```

For example:

```
int A[2][5];
float B[10][20];
```

INPUT/OUTPUT OF TWO DIMENSIONAL ARRAY

The two dimensional array $A[m][n]$ can be read as follow:

```
for(i=0; i<m; i++)
    for(j=0; j<n; j++)
        cin>>A[i][j];
```

We can use the static initialization with the two dimensional array as follow:

```
float M[2][5]= {5.1, 2.2, 3.8, 2.5, 4.7, 6.1, 7.2, 8.8, 9.0, 8.4};
float M[2][5]= {{5.1, 2.2},{ 3.8, 2.5}, {4.7, 6.1}, {7.2, 8.8}, {9.0, 8.4}};
```

To print a two dimensional array we can use the following form:

```
for(i=0; i<m; i++)
{
    for(j=0; j<n; j++)
        cout<<A[i][j]<<" ";
    cout<<endl;
}
```

Ex: Write a program to read an array of 50 real numbers, the program calculates the sum and the average and the maximum element of the array.

```
#include<iostream.h>
void main( )
{
    float A[50];
    int i;
    for (i=0; i<50; i++)
        cin>>A[i];
    float max=A[0];
    for (i=0; i<50; i++)
    {
        float sum+=a[i];
        if (A[i]>max)
            max=A[i];
    }
    float av=sum/50;
    cout<<sum<<av<<max;
}
```

Ex: Write a program to sort an array of 10 integers in an ascending order.

```
#include<iostream.h>
void main( )
{
    int A[10]= {10,7,0,20,3,125,15,75,5,34};
    int i, j, t;
    for(i=0; i<9; i++)
    {
        for(j=i+1; j<10; j++)
        {
            if(A[i]>A[j])
            {
                t=A[i];
                A[i]=A[j];
            }
        }
    }
}
```

```
                A[j]=t;
            }
        }
    }
    for(i=0; i<10; i++)
        cout<<A[i]<<" ";
}
```

Ex: Write a program to find the average of even and odd numbers in an array of 10 numbers.

```
#include<iostream.h>
void main( )
{
    int A[10], se=0, so=0, i, ke=0, kn=0;
    float ave, avo;
    for(i=0; i<10; i++)
        cin>>A[i];
    for(i=0; i<10; i++)
    {
        if(A[i]%2==0)
        {
            se+=A[i];
            ++ke;
        }
        else
        {
            so+=A[i];
            ++ko;
        }
    }
    ave=float(se)/ke;
    avo=float(so)/ko;
    cout<<"Average of even= "<<ave<<endl;
    cout<<"Average of odd= "<<avo;
}
```