

Array



- **INTRODUCTION**
- **ONE-DIMENSIONAL ARRAY**
- **MULTIDIMENSIONAL ARRAY**

Introduction



- It holds multiple values of same type.
- Each block of array is stored consecutively in memory.

SYNTAX:

```
data-type name[size1][size2].....[sizen];
```

Example:

```
int a[6];
```

Advantage of Array



- Huge amount of data can be stored under single variable name.
- Searching of data item is faster.
- 2 dimension arrays are used to represent the matrices.
- It is helpful in implementing other data structure like linked list, queue, stack.

One dimensional Array

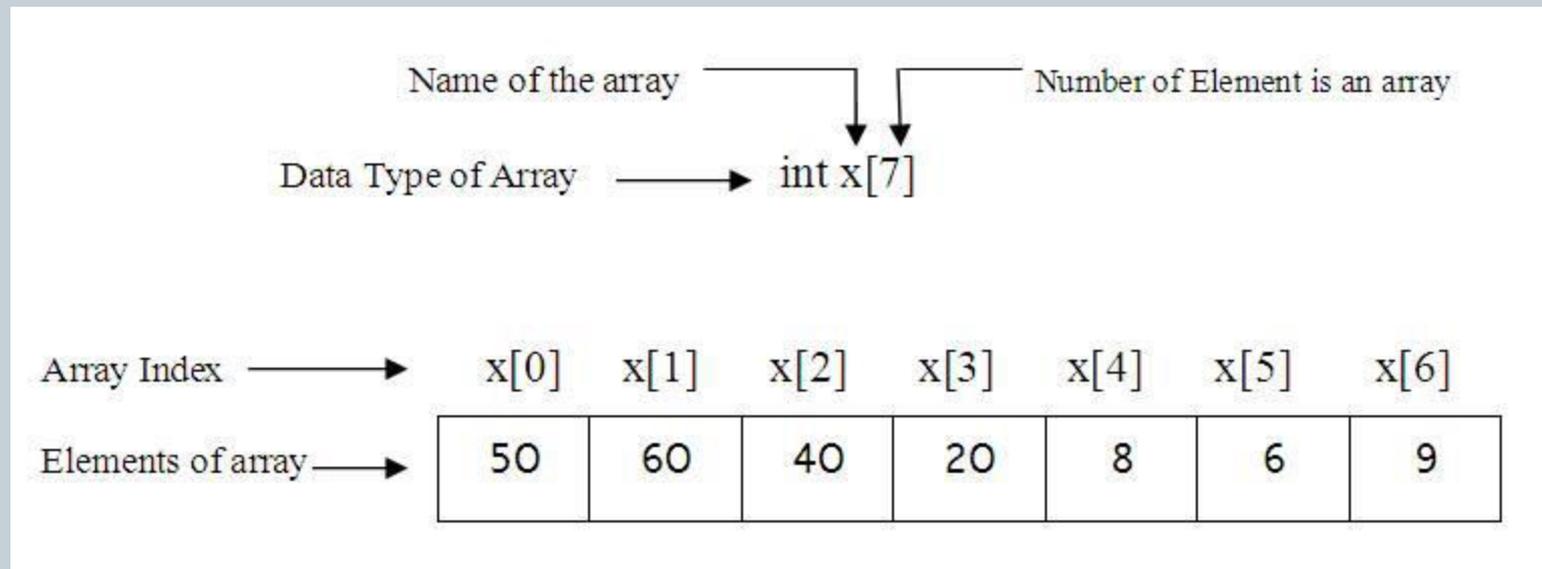


SYNTAX:

data-type name[index];

EXAMPLE:

```
int num[10];
```



Initialization



- `int num[6]={2,4,6,7,8,12};`
- Individual elements can also be initialize as:
 - `num[0]=2;`
 - `num[1]=4;`
 - `num[2]=6;`
 - `num[3]=7;`
 - `num[4]=8;`
 - `num[5]=12;`

Reading Data from User



- for loop is used to read data from the user.

```
for (i=0; i<10; i++)  
{  
    scanf ("%d", &num[i]);  
}
```

Class work



- WAP to read 10 numbers from the user and display them.
- WAP to read 20 numbers from the user and find out the highest number.

Multi-dimensional Array



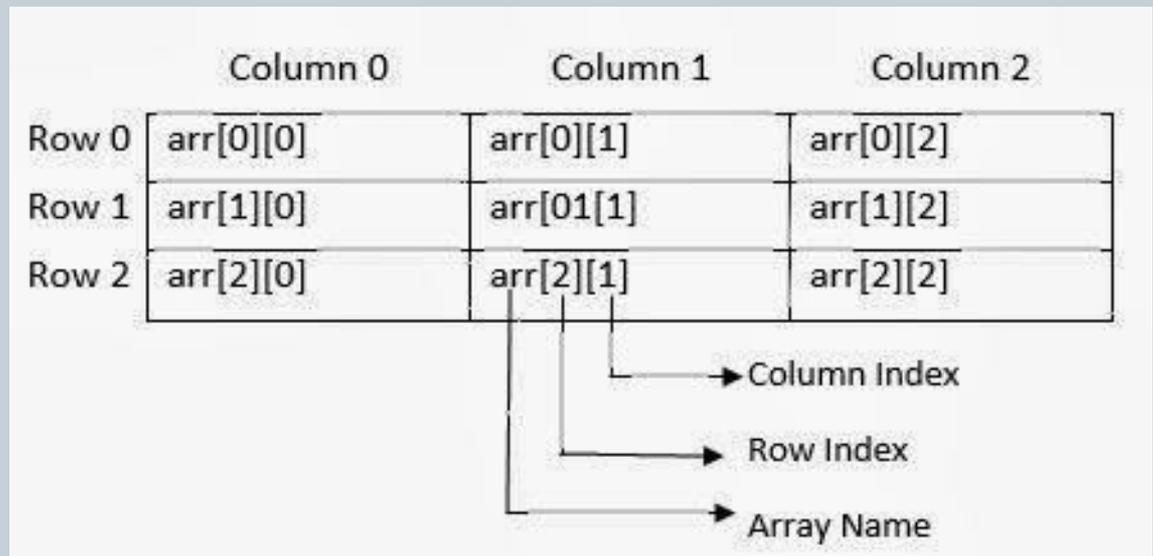
- In multi-dimensional we focus on the two dimensional array.

SYNTAX:

data-type name[row-size][column-size];

EXAMPLE:

```
int a[3][4];
```



Initialization



- `int odd[3][2]={1,3,5,7,9,11};`
- Individual element can also be assigned as:
 - `Odd[0][0]=1;`
 - `Odd[0][1]=3;`
 - `Odd[1][0]=5;`
 - `Odd[1][1]=7;`
 - `Odd[2][0]=9;`
 - `Odd[2][1]=11;`

Reading Data from the user



- Nested for Loop is used.

```
for (i=0; i<3; i++)  
{  
    for (j=0; j<2; j++)  
        scanf ("%d", &odd[i][j]);  
}
```