Array

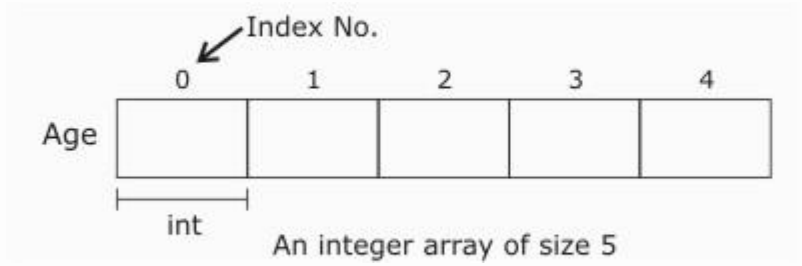
An array is a collection of data elements of same data type. It is described by a single name and each element of an array is referenced by using array name and its subscript no.

Declaration of Array

Type arrayName[numberOfElements];

For example,

```
int Age[5] ;
float cost[30];
```



Initialization of One Dimensional Array

An array can be initialized along with declaration. For array initialization it is required to place the elements separated by commas enclosed within braces.

```
int A[5] = {11,2,23,4,15};
```

It is possible to leave the array size open. The compiler will count the array size.

```
int B[] = {6,7,8,9,15,12};
```

Referring to Array Elements

In any point of a program in which an array is visible, we can access the value of any of its elements individually as if it was a normal variable, thus being able to both read and modify its value.

The format is as simple as:
name[index]

Examples:

```
cout << age[4];       //print an array element
age[4] = 55;          // assign value to an array element
cin >> age[4];        //input element 4
```

Using Loop to input an Array from user

```
int age [10], i ;
for (i = 0 ; i < 10; i++)
{
  cin >> age[i];
}
```

Arrays as Parameters

At some moment we may need to pass an array to a function as a parameter. In C++ it is not possible to pass a complete block of memory by value as a parameter to a function, but we are allowed to pass its address.
For example, the following function:
`void print(int A[])`
accepts a parameter of type "array of int" called A.
In order to pass to this function an array declared as:
`int arr[20];`
we need to write a call like this:
`print(arr);`
Here is a complete example:

```
#include <iostream>
using namespace std;

void print(int A[], int length)
{
  for (int n = 0; n < length; n++)
    cout << A[n] << " ";
  cout << "\n";
}
```

```
int main ()
{
  int arr[] = {5, 10, 15};
  print(arr,3);
  return 0;
}
```

Basic Operation On One Dimensional Array

Function to traverse the array A

```
void display(int A[], int n)
{
       cout << "The elements of the array are:\n";
       for(int i = 0; i < n; i++)
              cout << A[i];
}
```

Function to Read elements of the array A

```
void Input(int A[], int n)
{
      cout << "Enter the elements:";
      for(int i = 0; i < n; i++)
            cin >> A[i];
}
```

Function to Search for an element from A by Linear Search

```
void lsearch(int A[], int n, int data)
{
      for(int i = 0; i < n; i++)
      {
            if(A[i] == data)
            {
                  cout << "Data Found at : " << i;
                  return;
            }
      }
      cout << "Data Not Found in the array" << endl;
}
```

Function to Search for an element from Array A by Binary Search

```
int BsearchAsc(int A[], int n, int data)
{
       int Mid, Lbound = 0, Ubound = n-1, Found=0;
       while((Lbound <= Ubound) && !(Found))
       {
              Mid =(Lbound+Ubound)/2;        //Searching The Item
              if(data > A[Mid])
                     Lbound = Mid+1;
              else if(data < A[Mid])
                     Ubound = Mid-1;
              else
                     Found++;
       }
       if(Found)
              return(Mid+1);      //returning 1ocation, if present
       else
              return(-1);        //returning -1,if not present
}
```

Function to Sort the array A by Bubble Sort

```
void BSort(int A[], int n)
{
    int I, J, Temp;
    for(I = 0; I < n-1; I++) //sorting
    {
       for(J = 0; J < (n-1-I); J++)
            if(A[J] > A[J+1])
            {
               Temp = A[J]; //swapping
               A[J] = A[J+1];
               A[J+1] = Temp;
            }
    }
}
```

Function to Sort the array ARR by Insertion Sort

```
void ISort(int A[], int n)
{
       int I, J, Temp;
       for(I = 1; I < n; I++) //sorting
       {
           Temp = A[I];
           J = I-1;
           while((Temp < A[J]) && (J >= 0))
           {
               A[J+1] = A[J];
               J--;
           }
           A[J+1]=Temp;
       }
}
```

Function to Sort the array by Selection Sort

```
void SSort(int A[], int n)
{
    int I, J, Temp, Small;
    for(I = 0; I < n-1; I++)
    {
         Small = I;
         for(J = I+1; J < n; J++) //finding the smallest element
         if(A[J] < A[Small])
              Small = J;
         if(Small != I)
         {
            Temp = A[I]; //Swapping
            A[I] = A[Small];
            A[Small] = Temp;
         }
     }
}
```

Function to merge A and B arrays of lenghts N and M

```
void Merge(int A[], int B[], int C[], int N, int M, int &K)
{
      int I = 0, J = 0;
      K = 0;
      while (I < N && J < M)
      {
            if (A[I] < B[J])
                  C[K++] = A[I++];
            else if (A[I] > B[J])
                  C[K++] = B[J++];
            else
            {
                  C[K++] = A[I++];
                  J++;
            }
      }

      int T;
      for (T = I; T < N; T++)
            C[K++] = A[T];
      for (T = J; T < M; T++)
            C[K++] = B[T];
}
```