Lab 1: Array Based List

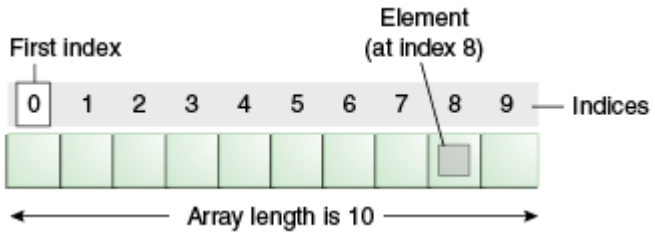
CLO:02

Objectives:

In this lab, we will implement a simple version of data structure, namely Int Array. We shall use MS Visual Studio for developing the programs. Learn how to implement Array List data structure.

Array:

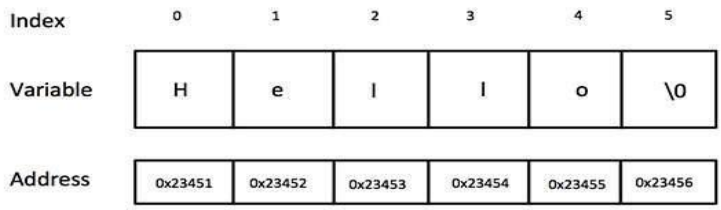
An array is a series of elements of the same type placed in contiguous memory locations that can be individually referenced by adding an index to a unique identifier. For example, we can store 5 values of type int in an array without having to declare 5 different variables, each one with a different identifier. Instead of that, using an array we can store 5 different values of the same type, with a unique identifier.



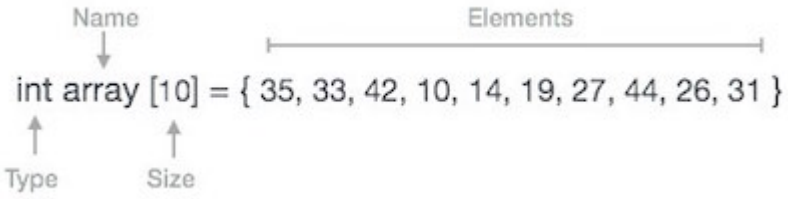
Following are the important terms to understand the concept of Array.

Element − Each item stored in an array is called an element.

Index − Each location of an element in an array has a numerical index, which is used to identify the element.



Array Representation:



Lab Task

1. In this Lab, you will have to implement the basic functions needed to build the List based on array as given below. You have to provide the implementation of given functions.

   Suppose we have an array like this:

   Array1

5	3	8	15	12

a. Function called **Insert** to add any value at the beginning of list.
   Now array1 will look like this: (If we insert 1 at start of the array)

1	5	3	8	15	12

b. Function called **Insert** to add value at the last of list.
   Now Array1 will look like this: (If we insert 17 after the last location of array)

5	3	8	15	12	17

c. Function called **remove** to remove new value after specific value in list.

5	3	8	15	12

   If we remove 3 from the array, the array will look like this:

5	8	15	12

d. Function called **display** to show all data of the array.

5	3	8	15	12

e. Function called **update** to update a value at certain index of array with the new value.

f. Function called **search** to find a specific value in array.

g. Function called **back** which take care that current location should not be less than the first location of array

h. Function called **front** which take care that current location should not go beyond the size of array.

i. Function called **Index_at** to show proper message according to the current index of the array**.**