Academic Year: 2020

Semester 3

Course Code: CS-212L

Course Title: Data structures and Algorithms Lab

CS212L-Data Structure and algorithms

Type of Lab: Open Ended **Weightage:**

CLO 1: CLO’s.

State the Rubric	**Cognitive/Understanding**	CLO1	Rubric A

Rubric A: Cognitive Domain

Evaluation Method: GA shall evaluate the students for Question according to following rubrics.

CLO	**0**	**1**	**2**	**3**	**4**
CLO1	Mention Milestones with respect to rubrics				

CS212L-Data Structure and algorithms

Lab 3

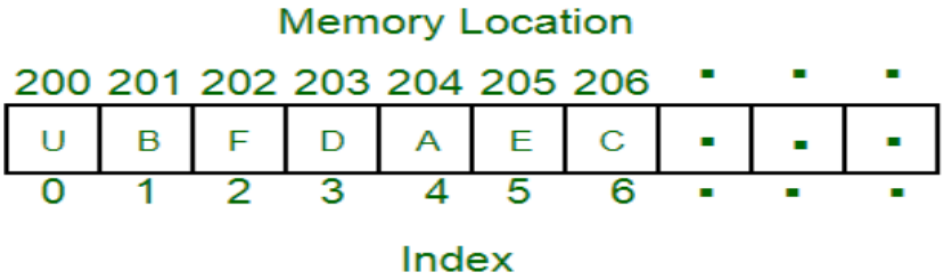
Analysis of Algorithms with respect to Time complexity

Objectives: To get familiar with array and its types

Processing steps:

Step 1: what is an array?

An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array). The base value is index 0 and the difference between the two indexes is the offset.



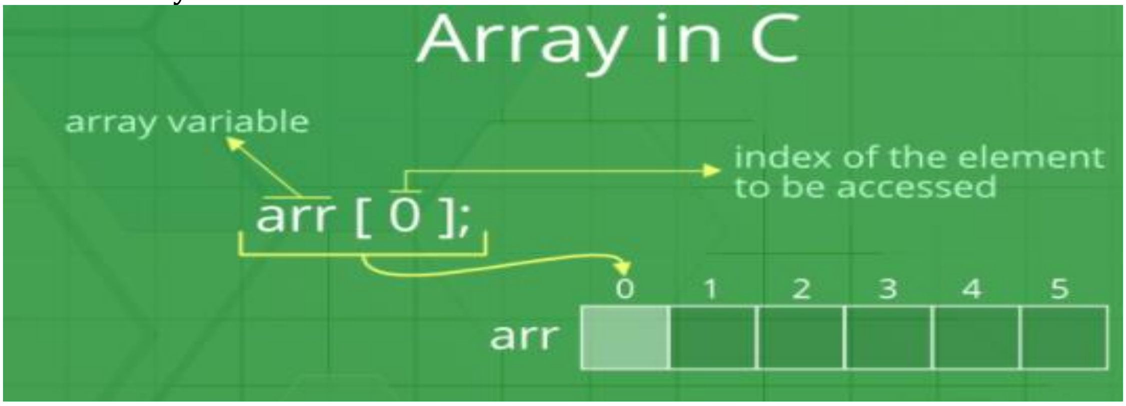
Array size:

In C language array has the fixed size meaning one's size is given to it. It can't change i.e. can't shrink it, can't expand it. The shrinking will not work because array, when declared, it gets memory statically, and thus compiler is the only one to destroy it.

Types of indexing in array:

- 0 (zero-based indexing): The first element of the array is indexed by subscript of 0
- 1 (one-based indexing): The second element of the array is indexed by subscript of 1
- n (n-based indexing): The base index of an array can be freely chosen. Usually programming languages allowing n-based indexing also allow negative index

values and other scalar data types like enumerations, or characters may be used as an array index.



Advantages of using arrays:

- Arrays allow random access of elements. This makes accessing elements by position faster.
- Arrays have better cache locality that can make a pretty big difference in performance.

Disadvantages of using arrays:

You can’t change the size i.e. once you have declared the array you can’t change its size because of static memory allocated to it.

Now if take an example of implementation of data structure Stack using array there are some obvious flaw.

Examples :

```
// A character array in C/C++/Java
char arr1[] = {'g', 'e', 'e', 'k', 's'};


// An Integer array in C/C++/Java
int arr2[] = {10, 20, 30, 40, 50};


// Item at i'th index in array is typically accessed
// as "arr[i]".  For example arr1[0] gives us 'g'
// and arr2[3] gives us 40.
```

Step 2: what is Dynamic Array?

Dynamic array in which size are dynamic .array can be of any size which is given by the user. Dynamic Array is created in Heap section.

Advantages of dynamic array over array:

- Dynamic array can be of any size, it given usually during run Time.
- Dynamic array are no static its array reference is in heap it is remain in the heap after the execution of the program
- Dynamic array can be extended at the run time whereas static array cannot be extended.

Step 5: How Dynamic array is implemented?

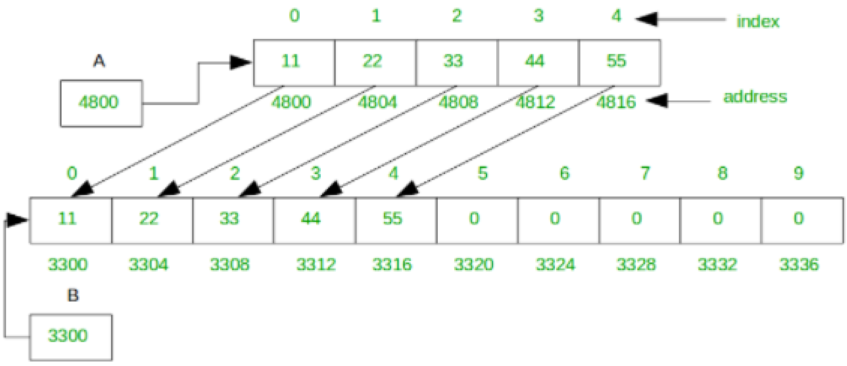
Syntax of dynamic Array:

```
int *a=new int[size];
```

size is taken during the run Time by the user. There are the operations which are implemented in the dynamic Array.

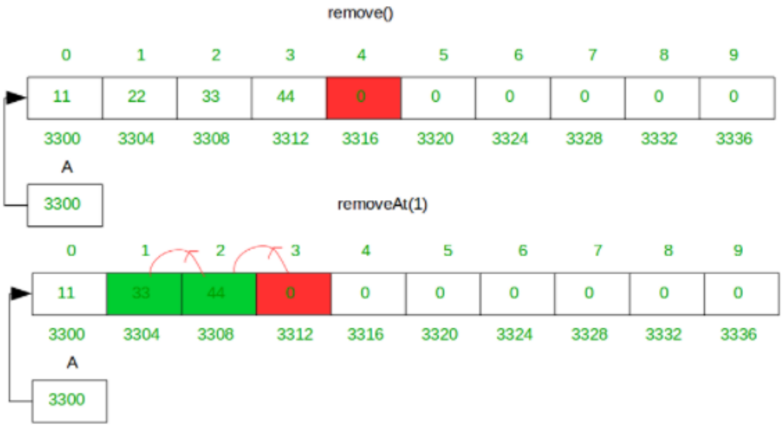
Insertion in the dynamic Array:

Add element at the end if the array size is not enough then extend the size of the array and add an element at the end of the original array as well as given index. Doing all that copying takes O(n) time, where n is the number of elements in our array. That's an expensive cost for an append. In a fixed-length array, appends only take O(1) time.



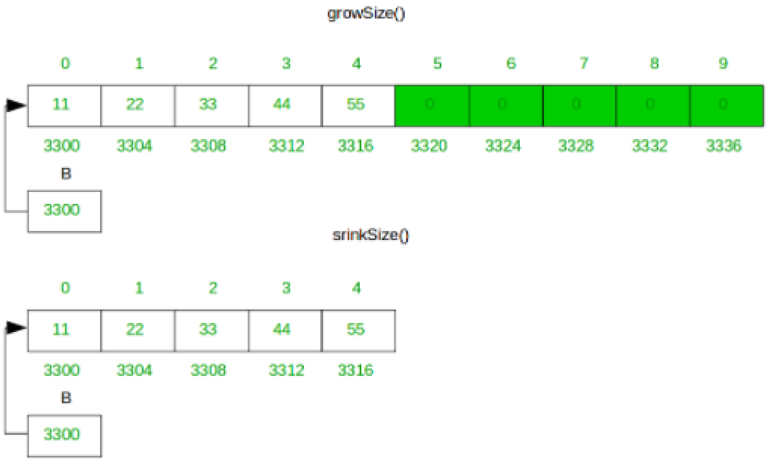
Deletion in dynamic array:

Delete an element from array, default remove() method delete an element from end, simply store zero at last index and you also delete element at specific index by calling removeAt(i) method where i is index. removeAt(i) method shift all right element in the left side from the given index.



Resize of Array Size:

When the array has null/zero data(exclude add by you) at the right side of the array whose take unrequited memory, the method srinkSize() free extra memory. When all space is consumed, and an additional element is to be added, then the underlying fixed-size array needs to increase size. Typically resizing is expensive because you have to allocate a bigger array and copy over all of the elements from the array you have overgrow before we can finally append our item.



Class activity

- Add element in dynamic Array at any position at the end of the array if array is full.
- Growth of the array.

Assignment

- Remove element at any position
- Shrink the array

REFRENCES: