# DATA STRUCTURE

Khairul Islam

10 October,2018

## 1 INTRODUCTION TO DATA STRUCTURE :

Data Structure is a way of collecting and organizing data in such a way that we can perform operations on these data in an effective way. so that various operations can be performed on it easily. It represents the knowledge of data to be organized in memory. It should be designed and implemented in such a way that it reduces the complexity and increases the efficiency.

## 2 BASIC DATA STRUCTURES :

As we have discussed above, anything that can store data can be called as a data structure, hence Integer, Float, Boolean, Char etc, all are data structures. They are known as Primitive Data Structures or Built-in Data Structures. Also we have some complex Data Structures, which are used to store large and connected data. Some example of Abstract Data Structure are:

- Stack
- Queue

- Linked list

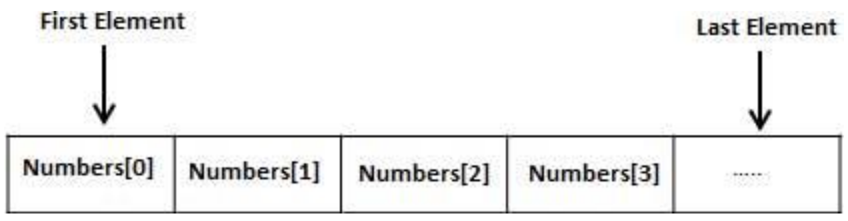
We will discuss them by two steps. first one is midterm, and secondly final.

## 3 Midterm

### 3.1 ARRAY

Arrays a kind of data structure that can store a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

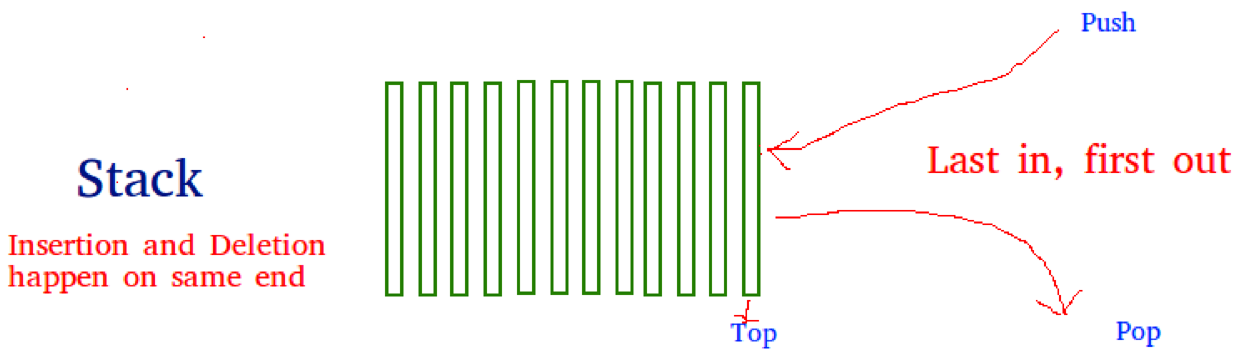
All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.



In our course, we have learnt what is array, how to store data of array, how to print element of array. There we have learnt about 1D array,2D array.

### 3.2 STACK

Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO(Last In First Out) or FILO(First In Last Out).



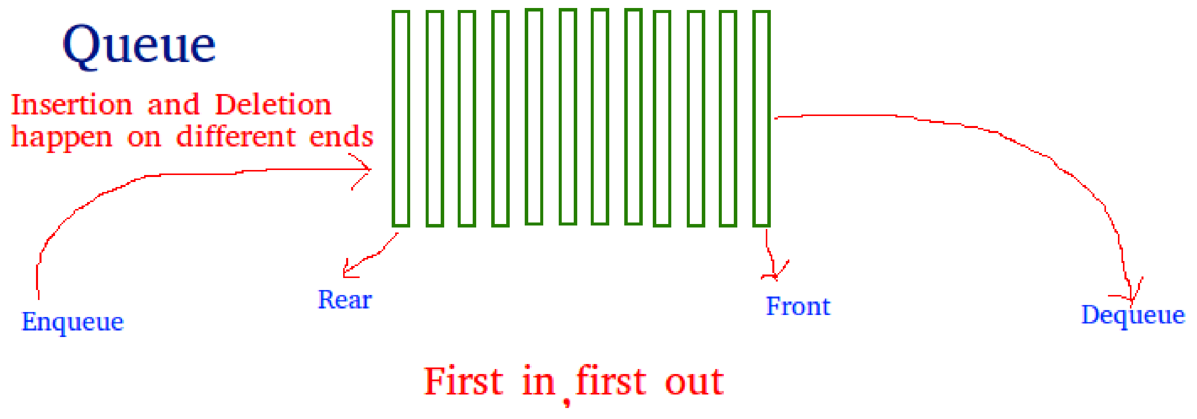
There are many real-life examples of a stack. Consider an example of plates stacked over one another in the canteen. The plate which is at the top is the first one to be removed, i.e. the plate which has been placed at the bottommost position remains in the stack for the longest period of time. So, it can be simply seen to follow LIFO(Last In First Out)/FILO(First In Last Out) order.

We know several operation of stack. The use of stack is done by some functions.

- Push
- Pop
- Display_stack
- Is_full
- Is_empty and etc.

## 3.3 QUEUE

A Queue is a linear structure which follows a particular order in which the operations are performed. The order is First In First Out (FIFO). A good example of a queue is any queue of consumers for a resource where the consumer that came first is served first. The difference between stacks and queues is in removing. In a stack we remove the item the most recently added; in a queue, we remove the item the least recently added.

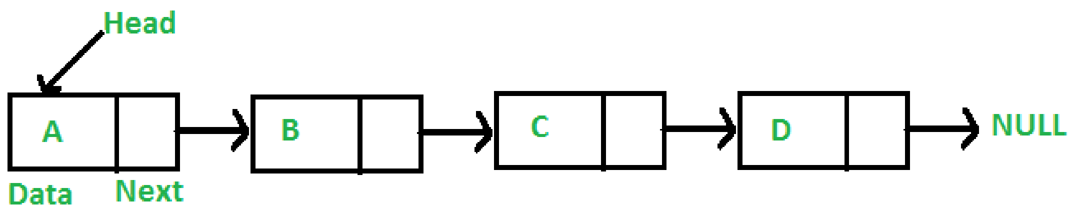


In queue also, there have several function by which we can do the operation of queue. They are:

- enQueue
- deQueue
- Display_queue
- Is_full
- Is_empty and etc.

## 3.4 LINKED LIST

A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers as shown in the below image:



In simple words, a linked list consists of nodes where each node contains a data field and a reference(link) to the next node in the list.

**Why Linked List?**
Arrays can be used to store linear data of similar types, but arrays have following limitations.
**1)** The size of the arrays is fixed: So we must know the upper limit on the number of elements in advance. Also, generally, the allocated memory is equal to the upper limit irrespective of the usage.
**2)** Inserting a new element in an array of elements is expensive, because room has to be created for the new elements and to create room existing elements have to shifted.

For example, in a system if we maintain a sorted list of IDs in an array id[ ].

id[ ] = [1000, 1010, 1050, 2000, 2040].

And if we want to insert a new ID 1005, then to maintain the sorted order, we have to move all the elements after 1000 (excluding 1000).
Deletion is also expensive with arrays until unless some special techniques are used. For example, to delete 1010 in id[ ], everything after 1010 has to be moved.

**Advantages over arrays**
**1)** Dynamic size
**2)** Ease of insertion/deletion

## REFERENCE :

1.Array. https://www.tutorialspoint.com/cprogramming/c_arrays.htm

2.Stack. https://www.geeksforgeeks.org/stack-data-structure/

3.Queue. https://www.geeksforgeeks.org/queue-data-structure/

4.Linked list. https://www.geeksforgeeks.org/data-structures/linked-list/