======Day-9-Assignment-Session-1======

1. Implementation of 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 reference.

\* Linked list has Node every Node has two parts

1. Data part-> it hold the data.

2. Next part-> it hold the address of the next Node.

if there is a last node the last node->next part hold the Null

Data Null

Data Next

Data Next

Head

1. Singly Linked List

A singly linked list is a concrete data structure consisting of a sequence of nodes.

Each node has two parts

1. Data part
2. Next part



Implementation:-

Class LinkedList {

    Node head;

    class Node {

        int data;

        Node next;

        Node(int d)

{

data = d;

}

     }

}

Advantages:

* Quick update operation like insertion, deletion
* No memory wastage

3. Doubly linked list

A Doubly Linked List contains an extra reference, typically called previous pointer together with next pointer and data which are there in singly linked list.

It has three parts:

\*Previous Node address part

\*Data part

\*Next Node address part



Implementation:

Class LinkedList {

    Node head;

    class Node {

        int data;

        Node next;

Node prev;

        Node(int d)

{

data = d;

}

     }

}

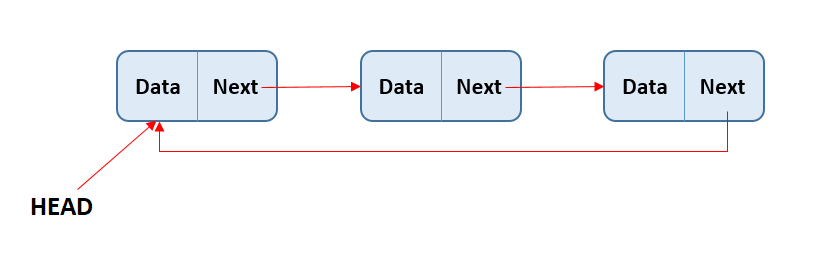
Advantages:

* Allows us to iterate in both direction
* We can delete a node easily as we have access to its previous node.
* Useful in implementing various other data structures.
* Reversing Easy

4.Circular Linked list

Circular linked list is a linked list where all nodes are connected to form a circle. There is no Null at the end.

A circular linked list can be singly circular linked list or doubly circular linked list.

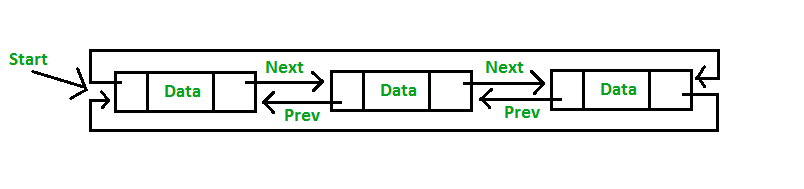


Advantages:

* Any node can be a starting point. We can traverse the whole list by starting from any point. We just need to stop when the first visited node is visited again.
* Never fall off the end of a list
* Continue to traverse even after passing the first or last node

ii) Doubly Circular linked list

Circular Doubly Linked List has properties of both doubly linked list and circular linked list in which two consecutive elements are linked or connected by previous and next reference and the last node points to first node by next reference and also the first node points to last node by the previous reference.



Advantages:

* In the circular linked list we will traverse the node only the one time. whereas in doubly linked list it's possible we will traverse the node more than one time.

6. 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).

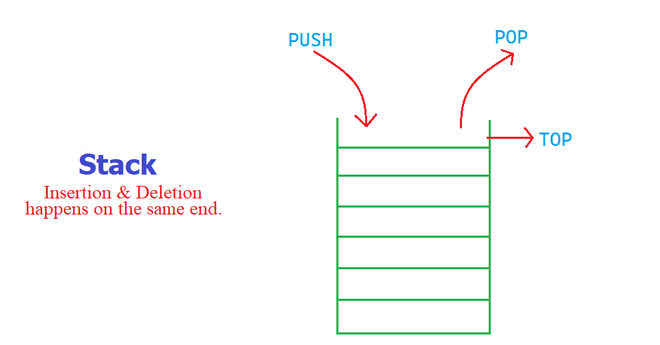
\*Methods

1. Push()-> Insert the element into stack

2. Pop()-> Delete the element from the stack

3. Peek()-> Returns the top element of the stack

4. IsEmpty()->Returns true if the stack is empty, else false



\*Applications:

recursion

editors

browser

parsing

6(i). 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).

Methods:

Enqueue->Insert the element into the Queue

Dequeue->delete the element form the Queue

peekFirst()->Returns the rare elements of the Queue

peekLast()->Returns the front elements of the Queue



Application:

* When a resource is shared among multiple consumers. Examples include[CPU scheduling](https://www.geeksforgeeks.org/cpu-scheduling-in-operating-systems/), [Disk Scheduling](https://www.geeksforgeeks.org/disk-scheduling-algorithms/).
* In Operating Systems

=======Session-2-Assignment======

1. Take array of list apply linear search

public class LinearSearch {

    public static void main(String[] args) {

        int arr[] = {10, 15, 17, 44, 56, 78, 98, 34, 12, 43};

        int key = 56;

        boolean flag = false;

        for(int i = 0; i < arr.length; i++)

           { if(arr[i] == key)

            {

                System.out.println("The element is found at index: "+i);

                flag = true;

                break;

            }

        }

            if(flag == false)

        {

            System.out.println("The element is not present in the list:");

        }

    }

}

2.Take array Sorted apply binary search

public class BinarySearch {

    public static void binSearch(int arr[], int l, int h, int key){

        int mid = (l + h)/2;

        while( l <= h )

        {

           if ( arr[mid] < key )

           {

             l = mid + 1;

           }else if ( arr[mid] == key )

           {

             System.out.println("Element is found at index: " + mid);

             break;

           }

           else

           {

            h = mid - 1;

        }

           mid = (l + h)/2;

        }

        if ( l > h )

        {

           System.out.println("Element is not found!");

        }

      }

    public static void main(String[] args) {

    int arr[] = {10, 15, 17, 23, 27, 44, 56, 78, 98};

        int key = 15;

        int l = 0;

        int h = arr.length - 1;

        binSearch(arr,l,h,key);

        }

}