SATYAM TRIPATHI
202151141

LAB 6

1.) STACK USING AL

```
import java.util.*;
class ArrayLinearList{
    protected Object elements[];
    protected int size;
    public ArrayLinearList(int initialCapacity){
//Paratemetrized Constructor
        elements=new Object[initialCapacity];
        if(initialCapacity<1)
        {
            throw new IllegalArgumentException("Initial capacity
cannot be less than 1");
        }
        size=0;
    }
    public ArrayLinearList(){       //Non Parametrized constructor
        this(10);
    }
    public int size(){
        return size;
    }
    public void extendArray(){          //Function to double array
elements[] length
        int temp=elements.length;
        temp=temp*2;
        Object arr[]=new Object[temp];
        for(int i=0;i<size;i++)
        {
            arr[i]=elements[i];
        }
        elements=arr;
    }
    public void show(){
        for(int i=0;i<size;i++)
        {
            System.out.print(elements[i]+" ");
        }
```

SATYAM TRIPATHI
202151141

```
        System.out.println();
    }
    public void add(Object obj,int index){          //Add an object
at a particular index
        if(index>=elements.length)
        throw new IndexOutOfBoundsException("Arraylist capacity is
"+elements.length);
        if(size==elements.length)
        {
            extendArray();
        }
        Object temp=obj;

        for(int i=index;i<=size;i++){


                Object temp1=elements[i];
                elements[i]=temp;
                temp=temp1;
        }
        size++;
    }
    public Object deleteIndex(int index){       //Delete an object
at particular index
        Object temp=elements[index];
        for(int i=index;i<size-1;i++){

            elements[i]=elements[i+1];
        }
        elements[size-1]=null;
        size--;
        return temp;
    }
    public void addRear(Object a){      //Add an object to rear
        if(size==elements.length)
        {
            extendArray();
        }
        elements[size++]=a;
    }
```

SATYAM TRIPATHI
202151141

```
    public void addFront(Object a){      //Add an object to front
        if(size==elements.length)
        {
            extendArray();
        }
        Object temp=a;

        for(int i=0;i<=size;i++){


                Object temp1=elements[i];
                elements[i]=temp;
                temp=temp1;
        }
        size++;
    }
    public Object deleteRear(){          //Delete an object from Rear
        Object temp=elements[size-1];
        elements[--size]=null;
        return temp;
    }
    public Object deleteFront(){         //Delete an object from
front
        Object temp=elements[0];
        for(int i=0;i<size-1;i++){

            elements[i]=elements[i+1];
        }
        elements[size-1]=null;
        size--;
        return temp;
    }
    public Object get(int index){        //access an element at that
index
        return elements[index];
    }
}

class StackAL {
    ArrayLinearList list=new ArrayLinearList();
```

SATYAM TRIPATHI
202151141

```
    public void push(int v){
        list.addRear(v);
    }
    public int pop(){
        if(isEmpty()){
            throw new StackOverflowError("Overflow");
        }
        int a=(int)list.deleteRear();
        return a;
    }
    public int peek(){
        if(isEmpty()){
            throw new StackOverflowError("Overflow");
        }
        return (int)list.get(list.size-1);
    }
    public boolean isEmpty(){
        return list.size==0;
    }
}
public class ImplementationStackAL {
    public static void main(String[] args) {
        StackAL s=new StackAL();
        s.push(1);
        s.push(2);
        s.push(3);
        s.push(4);
        s.push(5);
        System.out.println("Peek = "+s.peek());
        System.out.println("Pop = "+s.pop());
        System.out.println("Pop = "+s.pop());
        System.out.println("Peek = "+s.peek());
    }
}
```

OUTPUT

SATYAM TRIPATHI
202151141

```
Peek = 5
Pop = 5
Pop = 4
Peek = 3
```

2.) Stack Using Linked List

```
import java.util.EmptyStackException;


class Node{
    int data;
    Node next;
    public Node(int v){
        data=v;
        next=null;
    }
    public Node(){}
}
class LinkedList {
    protected Node first;
    protected int size;
    public LinkedList(){
        first=null;
        size=0;
    }
    public boolean isEmpty(){
        if(size==0)
        return true;
        return false;
    }
    public int size(){
        return size;
    }
    public void checkIndex(int index){
        if(index<0 || index>size){
```

SATYAM TRIPATHI
202151141

```
            throw new IndexOutOfBoundsException("index = "+index+"
for size = "+size);
        }
    }
    public int get(int index){
        checkIndex(index);
        Node current=first;
        for(int i=0;i<index;i++)
        {
            current=current.next;
        }
        return current.data;

    }

    public void add(int element,int index){
        checkIndex(index);

        size++;
        Node temp=first;
        if(index==0)
        {
            first=new Node(element);

            return;
        }
       for(int i=1;i<index;i++)
       {
            temp=temp.next;
       }
       Node n1=new Node(element);
       n1.next=temp.next;
       temp.next=n1;

    }

    public int remove(int index){
        checkIndex(index);
        Node temp=first;
        Node prev=null;
```

SATYAM TRIPATHI
202151141

```
        size--;
        while(temp.next!=null){
            prev=temp;
            temp=temp.next;
        }
        if(temp==first)
        {
            int v=temp.data;
            first=null;
            return v;
        }
        prev.next=null;
        int v=temp.data;
        temp=null;
        return v;
    }
    public void show(){
        Node temp=first;
        while(temp!=null){
            System.out.print(temp.data+" ");
            temp=temp.next;
        }
        System.out.println();
    }



}

public class StackLL  {
    LinkedList list=new LinkedList();
    public void push(int v){
        list.add(v,list.size());

    }
    public int pop(){
        if(list.isEmpty()){
            throw new EmptyStackException();
        }
```

SATYAM TRIPATHI
202151141

```
        return list.remove(list.size()-1);

    }
    public int peek(){
        return list.get(list.size()-1);
    }
    public int size(){
        return list.size();
    }
    public void show(){
        list.show();
    }

}
class ImplementationStack{
    public static void main(String[] args) {
        StackLL s=new StackLL();
        s.push(1);
        s.push(2);
        s.push(3);
        s.push(4);
        s.push(5);
        s.show();
        s.pop();
        s.pop();
        s.show();
        s.peek();
    }
}
```

OUTPUT

```
1 2 3 4 5
1 2 3
```

SATYAM TRIPATHI
202151141

3.) Driver Class

```
import java.util.*;
class ArrayLinearList{
    protected Object elements[];
    protected int size;
    public ArrayLinearList(int initialCapacity){
//Paratemetrized Constructor
        elements=new Object[initialCapacity];
        if(initialCapacity<1)
        {
            throw new IllegalArgumentException("Initial capacity
cannot be less than 1");
        }
        size=0;
    }
    public ArrayLinearList(){       //Non Parametrized constructor
        this(10);
    }
    public int size(){
        return size;
    }
    public void extendArray(){          //Function to double array
elements[] length
        int temp=elements.length;
        temp=temp*2;
        Object arr[]=new Object[temp];
        for(int i=0;i<size;i++)
        {
            arr[i]=elements[i];
        }
        elements=arr;
    }
    public void show(){
        for(int i=0;i<size;i++)
        {
            System.out.print(elements[i]+" ");
        }
        System.out.println();
    }
```

SATYAM TRIPATHI
202151141

```
    public void add(Object obj,int index){          //Add an object
at a particular index
        if(index>=elements.length)
        throw new IndexOutOfBoundsException("Arraylist capacity is
"+elements.length);
        if(size==elements.length)
        {
            extendArray();
        }
        Object temp=obj;

        for(int i=index;i<=size;i++){


                Object temp1=elements[i];
                elements[i]=temp;
                temp=temp1;
        }
        size++;
    }
    public Object deleteIndex(int index){       //Delete an object
at particular index
        Object temp=elements[index];
        for(int i=index;i<size-1;i++){

            elements[i]=elements[i+1];
        }
        elements[size-1]=null;
        size--;
        return temp;
    }
    public void addRear(Object a){      //Add an object to rear
        if(size==elements.length)
        {
            extendArray();
        }
        elements[size++]=a;
    }
    public void addFront(Object a){     //Add an object to front
        if(size==elements.length)
```

```
        {
            extendArray();
        }
        Object temp=a;

        for(int i=0;i<=size;i++){


                Object temp1=elements[i];
                elements[i]=temp;
                temp=temp1;
        }
        size++;
    }
    public Object deleteRear(){          //Delete an object from Rear
        Object temp=elements[size-1];
        elements[--size]=null;
        return temp;
    }
    public Object deleteFront(){         //Delete an object from
front
        Object temp=elements[0];
        for(int i=0;i<size-1;i++){

            elements[i]=elements[i+1];
        }
        elements[size-1]=null;
        size--;
        return temp;
    }
    public Object get(int index){        //access an element at that
index
        return elements[index];
    }
}

class StackAL {
    ArrayLinearList list=new ArrayLinearList();
    public void push(int v){
        list.addRear(v);
```

SATYAM TRIPATHI
202151141

```
    }
    public int pop(){
        if(isEmpty()){
            throw new StackOverflowError("Overflow");
        }
        int a=(int)list.deleteRear();
        return a;
    }
    public int peek(){
        if(isEmpty()){
            throw new StackOverflowError("Overflow");
        }
        return (int)list.get(list.size-1);
    }
    public boolean isEmpty(){
        return list.size==0;
    }
}
public class ImplementationStackAL {
    public static void main(String[] args) {
        StackAL s=new StackAL();
        s.push(2);
        s.push(0);
        s.push(2);
        s.push(1);
        s.push(5);
        s.push(1);
        s.push(1);
        s.push(4);
        s.push(1);
        System.out.println("Peek = "+s.peek());
        System.out.println("Pop = "+s.pop());
        System.out.println("Pop = "+s.pop());
        System.out.println("Peek = "+s.peek());
    }
}
```

SATYAM TRIPATHI
202151141

OUTPUT