**Lab7:**

* **Stack interface**
* **Stack implementation using array**
* **Stack implementation using Singly Linked List**
* **Reverse an array using Stack**

**Code:**

public interface Stack<E> {  
  
 boolean isEmpty ();  
 int Size();  
 E top();  
 void push (E element);  
 E pop();  
  
}

public class ArrayStack<E> implements Stack<E> {  
  
 static int *capacity*=1000;  
 int t=-1;  
 E data[];  
  
 public ArrayStack(int capacity) {  
 data=(E[]) new Object[capacity];  
 }  
  
 public ArrayStack() {  
 this(*capacity*);  
 }  
  
 @Override  
 public boolean isEmpty() {  
 return t==-1;  
 }  
  
 @Override  
 public int Size() {  
 return t+1;  
 }  
  
 @Override  
 public E top() {  
 if (isEmpty()) return null;  
 return data[t];  
 }  
  
 @Override  
 public void push(E element) {  
 if( Size()== data.length) throw new IllegalStateException("Stack is full!");  
  
  
 t++;  
 data[t]=element; // we can do it in one line like data[++t]=element;  
  
 }  
  
 @Override  
 public E pop() {  
 if(isEmpty()) return null;  
 E del =data[t];  
 data[t]=null;  
 t--;  
 return del;  
 }  
}

public class LinkedStack<E> implements Stack<E> {  
  
 SinglyLinkedList <E> list=new SinglyLinkedList<E>();  
  
 @Override  
 public boolean isEmpty() {  
 return list.isEmpty();  
 }  
  
 @Override  
 public int Size() {  
 return list.Size();  
 }  
  
 @Override  
 public E top() {  
 return list.first();  
 }  
  
 @Override  
 public void push(E element) {  
 list.addFirst(element);  
  
 }  
  
 @Override  
 public E pop() {  
 return list.removeFirst();  
 }  
}

public class lab7 {  
 public static void main(String[] args) {  
 ArrayStack<Integer> stack=new ArrayStack<>(3);  
 stack.push(11);  
 stack.push(12);  
 stack.push(13);  
 //stack.push(14);  
 while (!stack.isEmpty())  
 System.*out*.println(stack.pop());  
 }  
}

public class lab71 {  
 public static void main(String[] args) {  
 LinkedStack<Integer> stack=new LinkedStack<>();  
 stack.push(11);  
 stack.push(12);  
 stack.push(13);  
 //stack.push(14);  
 while (!stack.isEmpty())  
 System.*out*.println(stack.pop());  
 }  
}