import static java.lang.System.exit;

// Create Stack Using Linked list

class StackUsingLinkedlist {

// A linked list node

private class Node {

int data; // integer data

Node link; // reference variable Node type

}

// create global top reference variable global

Node top;

// Constructor

StackUsingLinkedlist()

{

this.top = null;

}

// Utility function to add an element x in the stack

public void push(int x) // insert at the beginning

{

// create new node temp and allocate memory

Node temp = new Node();

// check if stack (heap) is full. Then inserting an

// element would lead to stack overflow

if (temp == null) {

System.out.print("\nHeap Overflow");

return;

}

// initialize data into temp data field

temp.data = x;

// put top reference into temp link

temp.link = top;

// update top reference

top = temp;

}

// Utility function to check if the stack is empty or not

public boolean isEmpty()

{

return top == null;

}

// Utility function to return top element in a stack

public int peek()

{

// check for empty stack

if (!isEmpty()) {

return top.data;

}

else {

System.out.println("Stack is empty");

return -1;

}

}

// Utility function to pop top element from the stack

public void pop() // remove at the beginning

{

// check for stack underflow

if (top == null) {

System.out.print("\nStack Underflow");

return;

}

// update the top pointer to point to the next node

top = (top).link;

}

public void display()

{

// check for stack underflow

if (top == null) {

System.out.printf("\nStack Underflow");

exit(1);

}

else {

Node temp = top;

while (temp != null) {

// print node data

System.out.printf("%d->", temp.data);

// assign temp link to temp

temp = temp.link;

}

}

}

}

// main class

public class GFG {

public static void main(String[] args)

{

// create Object of Implementing class

StackUsingLinkedlist obj = new StackUsingLinkedlist();

// insert Stack value

obj.push(11);

obj.push(22);

obj.push(33);

obj.push(44);

// print Stack elements

obj.display();

// print Top element of Stack

System.out.printf("\nTop element is %d\n", obj.peek());

// Delete top element of Stack

obj.pop();

obj.pop();

// print Stack elements

obj.display();

// print Top element of Stack

System.out.printf("\nTop element is %d\n", obj.peek());

}

}