*HARSH KASHYAP  
CSE 4*

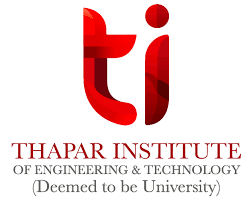
*101917088*

[*hkashyap\_be19@thapar.edu*](mailto:hkashyap_be19@thapar.edu)

A Practical activity Report submitted

for Data Structures (UCS301)

**DATA STRUCTURES**

****

Computer Science and Engineering

Patiala Campus

**2020**

Submitted to

Maninder Kaur

**Assignment 4**

**Question 1**

**Develop a menu driven program demonstrating the following operations on a Stack:**

* **push(),**
* **pop(),**
* **isEmpty(),**
* **isFull(),**
* **display(), and**
* **peek().**

**SOLUTION CODE**

**/\***

**#1) Develop a Menu driven program to demonstrate the following operations on Arrays**

**#——MENU——-**

**# 1.PUSH**

**# 2.POP**

**# 3.PEEK**

**# 4.EMPTY**

**# 5.FULL**

**# 6.DISPLAY**

**# 7.EXIT**

**\*/**

**import java.util.\*;**

**import java.io.\*;**

**class a4ques1**

**{**

**static Scanner scr = new Scanner(System.in);**

**static int a[]= new int[100];**

**static int top = -1;**

**public static void main(final String[] args)**

**{**

**int ch=0;**

**do {**

**ch = menu();**

**switch (ch) {**

**case 1:**

**push();**

**break;**

**case 2:**

**pop();**

**break;**

**case 3:**

**peek();**

**break;**

**case 4:**

**if (isEmpty())**

**System.out.println("Stack is empty .Underflow\n");**

**else**

**System.out.println("Stack is not empty \n");**

**break;**

**case 5:**

**if (isFull())**

**System.out.println("Stack is full. Overflow\n");**

**else**

**System.out.println("Stack is not full \n");**

**break;**

**case 6:**

**display();**

**break;**

**case 7:**

**System.out.println("We are done ");**

**break;**

**default:**

**System.out.println("Not an option \n");**

**}**

**} while (ch != 7);**

**}**

**static int menu()**

**{**

**System.out.println( "\n——MENU——- \n1.PUSH\n2.POP\n3.PEEK\n4.EMPTY\n5.FULL\n6.DISPLAY\n7.EXIT\nEnter your choice");**

**int ch = scr.nextInt();**

**return ch;**

**}**

**static void push()**

**{**

**if (isFull())**

**{**

**System.out.println("Overflow ");**

**return;**

**}**

**System.out.print("Enter elements to be pushed : ");**

**int pos = scr.nextInt();**

**a[++top]=pos;**

**}**

**static void pop()**

**{**

**if (isEmpty())**

**{**

**System.out.println("Underflow ");**

**return;**

**}**

**System.out.println("Popping out element... : "+a[top]);**

**a[top--]=0;**

**}**

**static void peek()**

**{**

**if (isEmpty())**

**{**

**System.out.println("Underflow ");**

**return;**

**}**

**System.out.println("Viewing the top element ... : "+a[top]);**

**}**

**static boolean isFull()**

**{**

**return top==100;**

**}**

**static boolean isEmpty()**

**{**

**return top==-1;**

**}**

**static void display()**

**{**

**if (isEmpty())**

**{**

**System.out.println("Underflow ");**

**return;**

**}**

**System.out.println("Viewing the stack ");**

**for(int i=top;i>=0;i--)**

**{**

**System.out.println(" | "+a[i]+" |");**

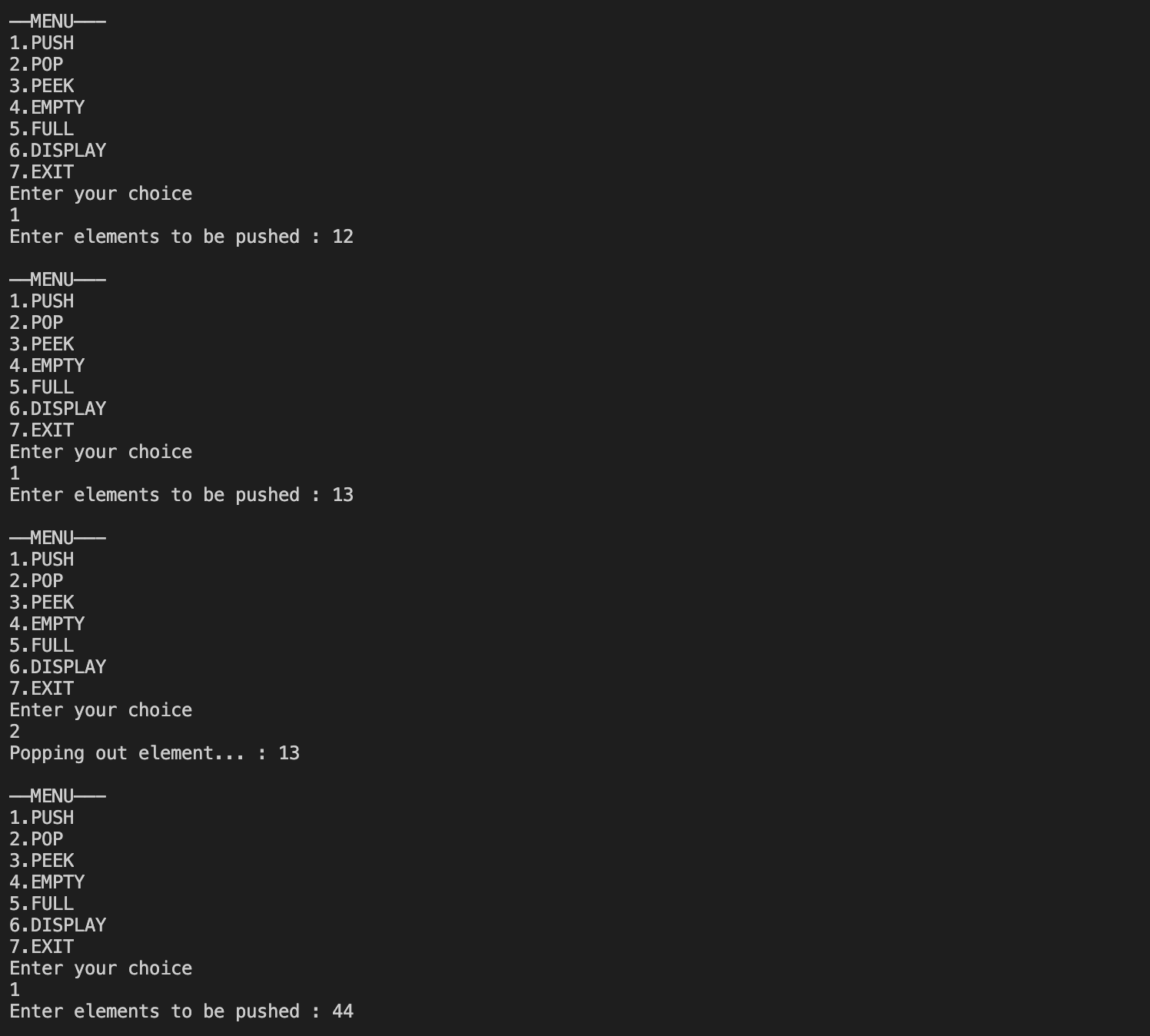
**System.out.println(" -----");**

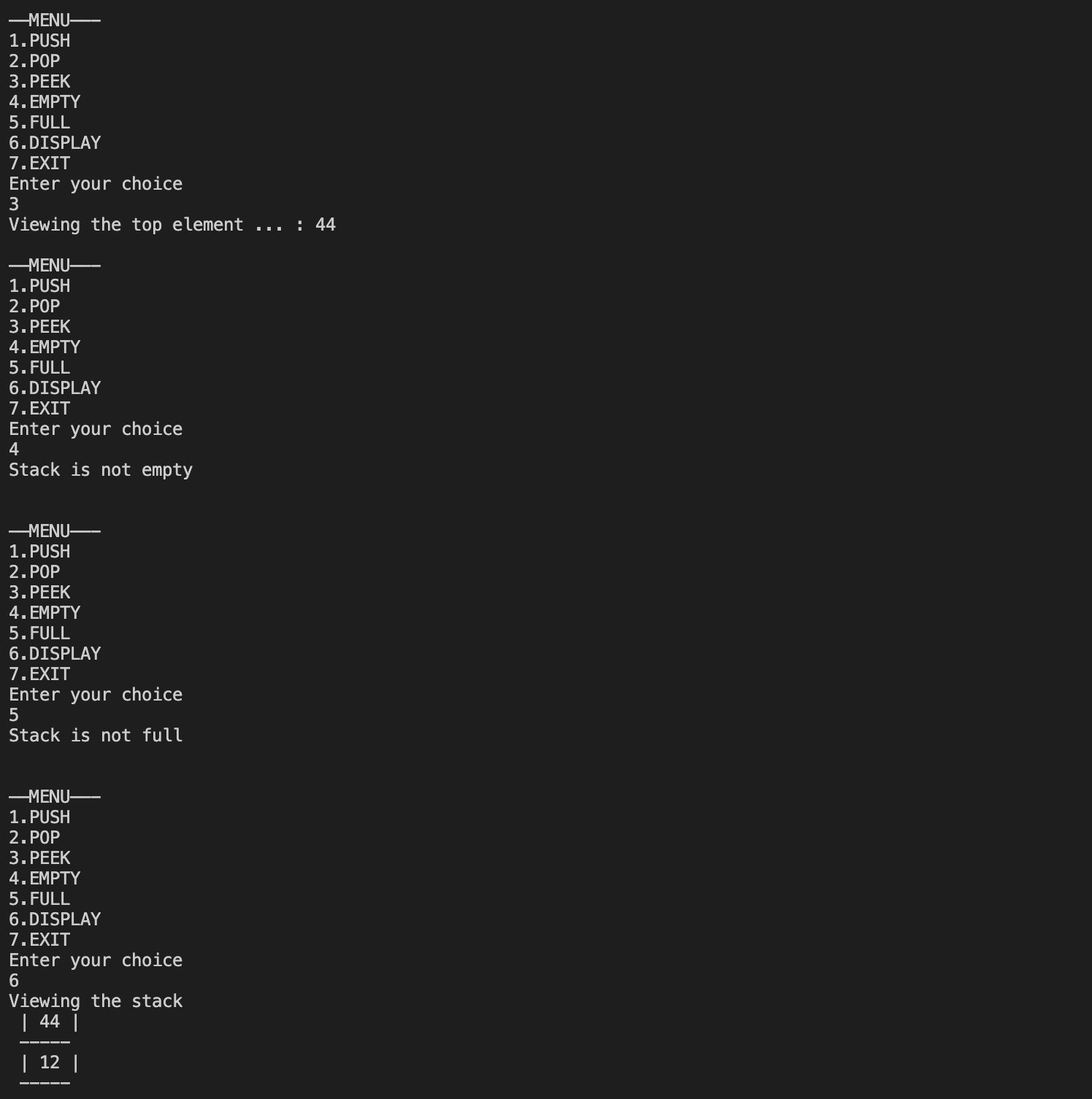
**}**

**}**

**}**

**OUTPUT**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 2**

**Given a String, Reverse it using STACK. For example “data structure” should be output as “erutcurtsatad.”**

**/\***

**Given a String, Reverse it using STACK. For example “data structure” should be**

**output as “erutcurtsatad.”**

**\*/**

**import java.util.\*;**

**import java.io.\*;**

**class a4ques2**

**{**

**static Scanner scr = new Scanner(System.in);**

**static int top=-1; //counter**

**public static void main(final String[] args)**

**{**

**System.out.println( "Enter string to be reversed. ");**

**String st = scr.nextLine();**

**char ch[] = new char[st.length()];**

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

**{**

**push(ch, st.charAt(i));**

**}**

**System.out.println( "The reversed string is : ");**

**while(!isEmpty())**

**{**

**pop(ch);**

**}**

**System.out.println();**

**}**

**static void push(char c[], char element)**

**{**

**if(element!=' ')**

**c[++top]=element;**

**}**

**static void pop(char ch[])**

**{**

**System.out.print(ch[top]);**

**ch[top--]=' ';**

**}**

**static boolean isEmpty()**

**{**

**return top==-1;**

**}**

**}**

**OUTPUT**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 3**

**Write a program to check for balanced parentheses in an expression.**

**SOLUTION CODE**

**/\***

**Write a program that checks if an expression has balanced parentheses.**

**\*/**

**import java.util.\*;**

**import java.io.\*;**

**class a4ques3**

**{**

**static Scanner scr = new Scanner(System.in);**

**static int top=-1; //counter**

**static char ch[]= new char [100];**

**public static void main(final String[] args)**

**{**

**System.out.println( "Enter paranthesis");**

**String s=scr.next();**

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

**{**

**char p=s.charAt(i);**

**if(p=='{' || p=='[' || p=='(')**

**{**

**push(p);**

**}**

**else if (p=='}' || p==']' || p==')')**

**{**

**if (isEmpty())**

**{**

**System.out.println("Not Balanced ");**

**return;**

**}**

**char para=pop();**

**if (para=='{' && p!='}')**

**{**

**System.out.println("Not Balanced ");**

**return;**

**}**

**else if (para=='[' && p!=']')**

**{**

**System.out.println("Not Balanced ");**

**return;**

**}**

**else if (para=='(' && p!=')')**

**{**

**System.out.println("Not Balanced ");**

**return;**

**}**

**}**

**}**

**if(!isEmpty())**

**System.out.println("Not Balanced ");**

**else**

**System.out.println("Balanced ");**

**}**

**static void push(char element)**

**{**

**ch[++top]=element;**

**}**

**static char pop()**

**{**

**char n=ch[top];**

**ch[top--]=0;**

**return n;**

**}**

**static boolean isEmpty()**

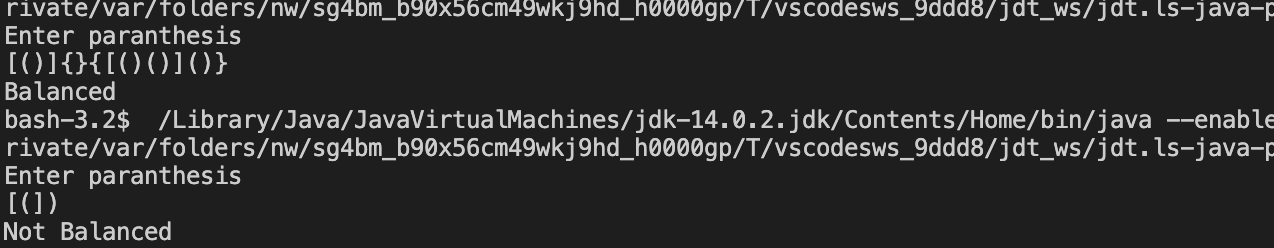
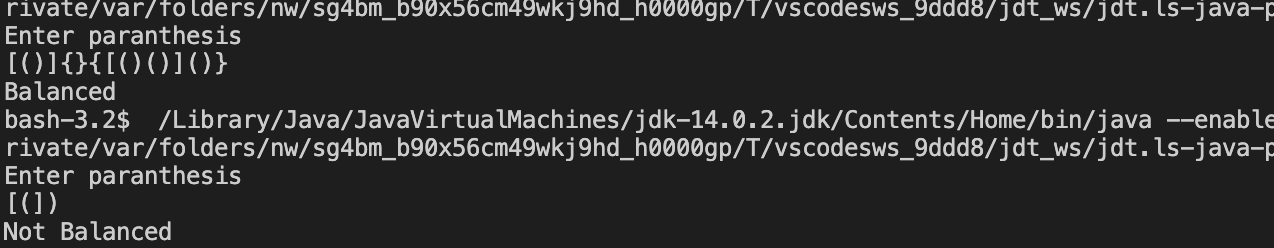
**{**

**return top==-1;**

**}**

**}**

***OUTPUT***

******

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 4**

**Write a program to convert an Infix Expression into a Postfix expression.**

**SOLUTION CODE**

**//Write a program to convert an Infix expression into a Postfix expression.**

**import java.util.\*;**

**import java.io.\*;**

**class a4ques4**

**{**

**static Scanner scr = new Scanner(System.in);**

**static char st[]= new char [100];**

**static int top = -1;**

**public static void main(final String[] args)**

**{**

**System.out.println("Enter the infix expression");**

**String exp=scr.next();**

**System.out.println("Infix expression : "+exp);**

**exp="("+exp+")";**

**String post="";**

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

**{**

**char ch=exp.charAt(i);**

**switch (ch)**

**{**

**case '(':**

**push(ch);**

**break;**

**case ')':**

**while(true)**

**{**

**char putout=pop();**

**if(putout=='(')**

**break;**

**post=post+putout;**

**}**

**break;**

**case '^':**

**push(ch);**

**break;**

**case '\*':**

**case '/':**

**while(true)**

**{**

**char putout=peek();**

**if (peek()=='^' || peek()=='\*' || peek()=='/')**

**{**

**putout=pop();**

**post=post+putout;**

**}**

**else**

**{**

**push(ch);**

**break;**

**}**

**}**

**break;**

**case '+':**

**case '-':**

**while(true)**

**{**

**char putout=peek();**

**if (peek()=='^' || peek()=='\*' || peek()=='/' || peek()=='+' || peek()=='-')**

**{**

**putout=pop();**

**post=post+putout;**

**}**

**else**

**{**

**push(ch);**

**break;**

**}**

**}**

**break;**

**default:**

**post=post+ch;**

**}**

**}**

**System.out.println("Postfix expression : "+post);**

**}**

**static void push(char pos)**

**{**

**st[++top]=pos;**

**}**

**static char pop()**

**{**

**char ch= st[top];**

**st[top--]=' ';**

**return ch;**

**}**

**static char peek()**

**{**

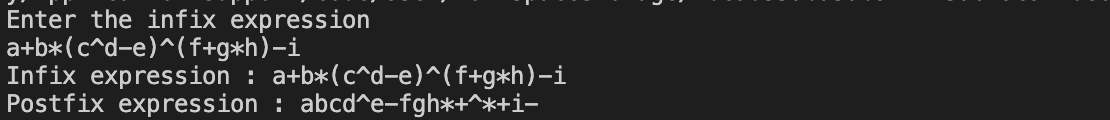
**if(top==-1)**

**return '.';**

**return st[top];**

**}**

**}**

***OUTPUT***

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 5**

**Write a program for the evaluation of a Postfix expression.**

**SOLUTION CODE**

**//Write a program for the evaluation of a Postfix expression.**

**import java.util.\*;**

**import java.io.\*;**

**class a4ques5**

**{**

**static Scanner scr = new Scanner(System.in);**

**static int st[]= new int [100];**

**static int top = -1;**

**public static void main(final String[] args)**

**{**

**System.out.println("Enter the postfix expression");**

**String exp=scr.nextLine();**

**exp+=" ";**

**int num=0;**

**System.out.println("Evaluating the postfix expression : "+exp);**

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

**{**

**char ch=exp.charAt(i);**

**if (Character.isDigit(ch))**

**{**

**num=num\*10+(ch-'0');**

**}**

**else if (Character.isWhitespace(ch))**

**{**

**if(num!=0)**

**push(num);**

**num=0;**

**}**

**else**

**{**

**int a = pop();**

**int b = pop();**

**switch(ch)**

**{**

**case '+': push(b+a);**

**break;**

**case '-': push(b-a);**

**break;**

**case '\*': push(b\*a);**

**break;**

**case '/': push(b/a);**

**break;**

**case '^': push((int)Math.pow(b,a));**

**break;**

**}**

**}**

**}**

**System.out.println("Postfix expression : "+pop());**

**}**

**static void push(int pos)**

**{**

**st[++top]=pos;**

**}**

**static int pop()**

**{**

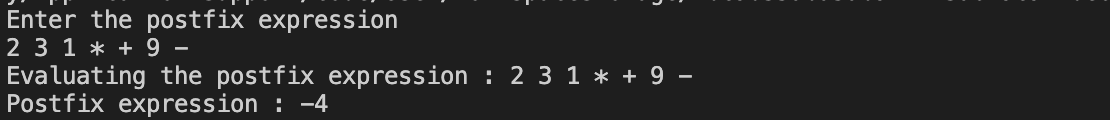
**int ch= st[top];**

**st[top--]=' ';**

**return ch;**

**}**

**}**

***OUTPUT***

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**EXTRA**

**Question**

**Given an array, find the nearest smaller element G[i] for every element A[i] in the array such that the element has an index smaller than i.**

**SOLUTION CODE**

**//Given an array, find the nearest smaller element G[i] for every element A[i] in the array such that the element has an index smaller than i.**

**//This one has been tried using the Library class of JAVA to understand that.**

**// Java code for stack implementation**

**import java.io.\*;**

**import java.util.\*;**

**class NearElement**

**{**

**static Scanner scr= new Scanner(System.in);**

**public static void main (String[] args)**

**{**

**System.out.println("Enter no. of elements ");**

**int n=scr.nextInt();**

**int arr[]=new int[n];**

**for(int i=0;i<n;i++)**

**{**

**System.out.print("Enter element : ");**

**arr[i]=scr.nextInt();**

**}**

**System.out.println("Near elements are ");**

**displayNear(arr);**

**}**

**static void displayNear(int a[])**

**{**

**System.out.println("For "+a[0]+" near element is : "+(-1));**

**Stack<Integer> st = new Stack<Integer>();**

**st.push(a[0]);**

**int k=0;**

**for(int i=1;i<a.length;i++)**

**{**

**if(st.peek() < a[i])**

**k=st.peek();**

**else if(st.peek() >= a[i])**

**{**

**while (!st.empty() && st.peek() >= a[i] )**

**st.pop();**

**k= st.empty()?-1:st.peek();**

**}**

**System.out.println("For "+a[i]+" near element is : "+k);**

**st.push(a[i]);**

**}**

**}**

**}**

***OUTPUT***

******

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question**

**Design a stack that supports getMin() in O(1) time and O(1) extra space**

**SOLUTION CODE**

**/\***

**Design a stack that supports getMin() in O(1) time and O(1) extra space**

**\*/**

**import java.util.\*;**

**import java.io.\*;**

**class min**

**{**

**static Scanner scr = new Scanner(System.in);**

**static int a[]= new int[100];**

**static int top = -1;**

**static int min;**

**public static void main(final String[] args)**

**{**

**int ch=0;**

**do**

**{**

**ch = menu();**

**switch (ch) {**

**case 1:**

**System.out.print("Enter elements to be pushed : ");**

**int pos = scr.nextInt();**

**push(pos);**

**break;**

**case 2:**

**pop();**

**break;**

**case 3:**

**getMin();**

**break;**

**case 4:**

**if (isEmpty())**

**System.out.println("Stack is empty .Underflow\n");**

**else**

**System.out.println("Stack is not empty \n");**

**break;**

**case 5:**

**if (isFull())**

**System.out.println("Stack is full. Overflow\n");**

**else**

**System.out.println("Stack is not full \n");**

**break;**

**case 6:**

**System.out.println("We are done ");**

**break;**

**default:**

**System.out.println("Not an option \n");**

**}**

**} while (ch != 6);**

**}**

**static int menu()**

**{**

**System.out.println("\n——MENU——- \n1.PUSH\n2.POP\n3.GET MIN\n4.EMPTY\n5.FULL\n6.EXIT\nEnter your choice");**

**int ch = scr.nextInt();**

**return ch;**

**}**

**static boolean isFull()**

**{**

**return top == 100;**

**}**

**static boolean isEmpty() {**

**return top == -1;**

**}**

**static void getMin()**

**{**

**if (isEmpty())**

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

**else**

**System.out.println("Minimum Element in the " + " stack is: " + min);**

**}**

**// prints top element of MyStack**

**static void peek()**

**{**

**if (isEmpty())**

**{**

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

**return;**

**}**

**int t = a[top]; // Top element.**

**System.out.print("Top Most Element is: ");**

**if (t < min)**

**System.out.println(min);**

**else**

**System.out.println(t);**

**}**

**// Removes the top element from MyStack**

**static void pop()**

**{**

**if (isEmpty())**

**{**

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

**return;**

**}**

**System.out.print("Top Most Element Removed: ");**

**int t = a[top];**

**a[top--] = 0;**

**if (t < min) {**

**System.out.println(min);**

**min = 2 \* min - t;**

**}**

**else**

**System.out.println(t);**

**}**

**// Insert new number into MyStack**

**static void push(int x)**

**{**

**if (isEmpty())**

**{**

**min = x;**

**a[++top]=x;**

**System.out.println("Number Inserted: " + x);**

**return;**

**}**

**if (x < min)**

**{**

**a[++top]=2\*x - min;**

**min = x;**

**}**

**else**

**a[++top]=x;**

**System.out.println("Number Inserted: " + x);**

**}**

**}**