| Program No: | 19 |
| --- | --- |
| Roll No: | 1510 |
| Title of Program: | Infix to Postfix - Stack Application |
| Objective: | Submit a PDF containing code for the Stack Applications  (1) Infix to Postfix Conversion the screenshots of its output |

**CODE:**

/\* Name: Advait Dhakad

Roll No: 1510

Unit 3: Stack & Queue

Program: Infix to Postfix - Stack Application \*/

import java.util.Scanner;

public class infixpostfix{

static boolean isOperator(char c){

return c=='+' || c=='-' || c =='\*' || c =='/';

} // end isOperator

static int precedence(char op){

switch(op){

case '+':

case '-':

return 1;

case '\*':

case '/':

return 2;

default:

return -1;

}

} // end precedence

static String infixtopostfix(String infix){

char[] stack = new char[infix.length()];

int tos = -1;

StringBuilder postfix = new StringBuilder();

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

char ch = infix.charAt(i);

if(Character.isLetterOrDigit(ch)){

postfix.append(ch);

}

else if (ch== '(' ){

tos++;

stack[tos] = ch;

}

else if (ch == ')' ){

while(stack[tos]!='('){

postfix.append(stack[tos--]);

}

tos --; // to discard the '('

}

else if(isOperator(ch)){

while(tos>=0 && precedence(ch) <= precedence(stack[tos])){

postfix.append(stack[tos--]);

}

tos++;

stack[tos] = ch;

}

}// end of for loop

while(tos>=0){

postfix.append(stack[tos]);

tos--;

}

return postfix.toString();

} // end infixtopostfix

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

System.out.println("\t\t\*\*\*\*\*PREFIX TO POSTFIX\*\*\*\*\*\*");

System.out.print("Enter the text you want to get converted: ");

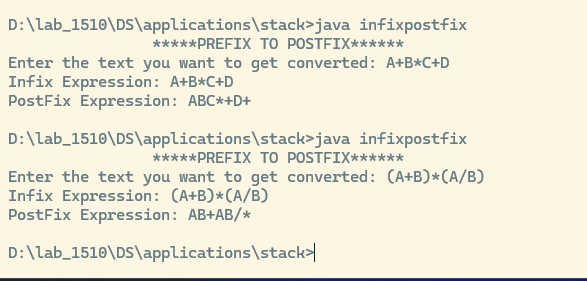
String text = sc.nextLine();

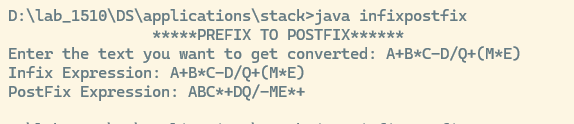
System.out.println("Infix Expression: " + text);

System.out.println("PostFix Expression: "+ infixtopostfix(text));

}}

**OUTPUT:**

****

****