विद्याधनं सर्वधन प्रधानम् ।

MUMBAI EDUCATIONAL TRUST

MET Institute of Computer Science

THE MET LEAGUE OF COLLEGES
MET
AS SHARP AS YOU CAN GET
Bhujbal Knowledge City

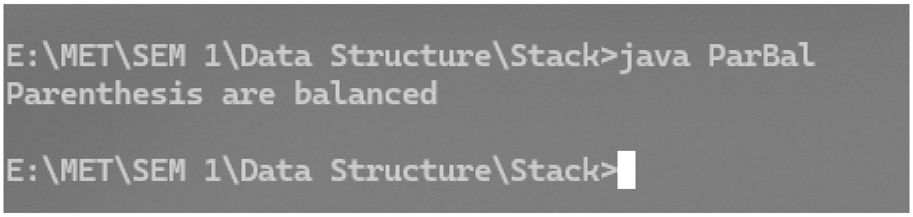
Name:	Advait Dhakad
Roll No:	1510
Title of Program:	Stacks & Queues: Stack Application
Objective:	1. Parenthesis Balancing 2. Postfix Evaluation

1) Parenthesis Balancing - Stack Application

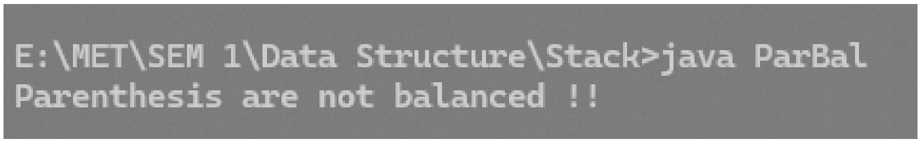
CODE:

```
public class ParBal {
    public static void main(String[] args) {
        String expression = "((a+b))+(c+d)";
        if (isBalanced(expression)) {
            System.out.println("Parenthesis are balanced");
        } else {
            System.out.println("Parenthesis are not balanced !!");
        }
    }

    public static boolean isBalanced(String ex) {
        char[] stack = new char[ex.length()];
        int tos = -1;
        for (int i = 0; i < ex.length(); i++) {
            char ch = ex.charAt(i);
            if (ch == '(') {
                tos++;
                stack[tos] = ch;
            } else if (ch == ')') // if closing parenthesis then pop
            {
                if (tos == -1) // No matching parenthesis
                {
                    return false;
                }
                tos--; // pop from the stack
            }
        } // end of for
        if (tos == -1) {
            return true;
        } else {
            return false;
        }
        // or above can be written as
        // return tos==-1;
    }// end of isBalanced
}// end of ParBal
```

OUTPUT:



```
E:\MET\SEM 1\Data Structure\Stack>java ParBal
Parenthesis are balanced

E:\MET\SEM 1\Data Structure\Stack>
```

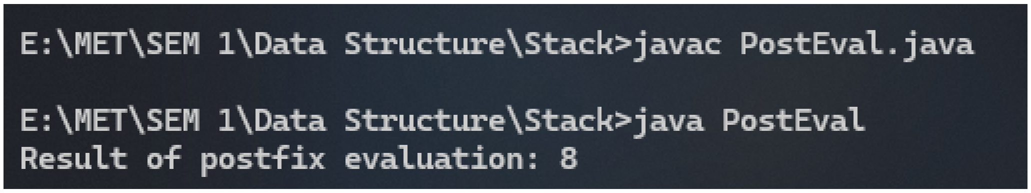


```
E:\MET\SEM 1\Data Structure\Stack>java ParBal
Parenthesis are not balanced !!
```

2) Postfix Evaluation

```
import java.util.*;

class PostEval {
    public static void main(String[] args) {
        String expr = "23*5+62/-";
        int result = evaluate(expr);
        System.out.println("Result of postfix evaluation: " +
                result);
    }

    public static int evaluate(String ex) {
        int[] stack = new int[ex.length()];
        int tos = -1;
        for (int i = 0; i < ex.length(); i++) {
            char ch = ex.charAt(i);
            if (Character.isDigit(ch)) // if ch is a digit -
            {
                tos++;
                stack[tos] = ch - '0';
            } else if (ch == '+' || ch == '-' || ch == '*' || ch == '/') // if ch is operator
            {
                int op2 = stack[tos--]; // pop the 2nd

                int op1 = stack[tos--]; // pop the 1st

                int res = 0;
                switch (ch) {
                    case '+':
                        res = op1 + op2;
                        break;
                    case '-':
                        res = op1 - op2;
                        break;
                    case '*':
                        res = op1 * op2;
                        break;
                    case '/':
                        res = op1 / op2;
                        break;
                } // -end of switch-case
                  // Push the result back on the stack
                tos++;
                stack[tos] = res;
            }
        } // -end of for loop
          // Final result at tos
        return stack[tos];
    } // -end of evaluate()
} // -end of PostEval class
```

OUTPUT



```
E:\MET\SEM 1\Data Structure\Stack>javac PostEval.java

E:\MET\SEM 1\Data Structure\Stack>java PostEval
Result of postfix evaluation: 8
```