Program No:	**3**
Roll No :	**1525**
Title of Program :	Stack with postfix operation
Objective :	Queue

Parbalance>>>>>>
source code:

```
public class ParBal {


  public static boolean isBalanced(String expr) {
    // Create a stack
    char[] stack = new char[expr.length()];
    int tos = -1; // top of stack index


    for (int i = 0; i < expr.length(); i++) {
      char ch = expr.charAt(i);

      if (ch == '(') {
        tos++;
        stack[tos] = ch;
      }

      else if (ch == ')') {

        if (tos == -1) {
          return false;
        }
        tos--;
      }
    }


    return tos == -1;
  }


  public static void main(String[] args) {
    String expr = "(a+b)+(c+d)";

    if (isBalanced(expr)) {
```

```
      System.out.println("Parentheses are balanced");
    } else {
      System.out.println("Parentheses are not balanced");
    }
  }
}
```

output:

```
PS C:\Users\mcamock\DSAlab\queue\newbhava\stack\bal> java  ParBal
Parentheses are balanced
```

POSTFIX CODE>>>>>
SOURCE CODE:

```
public class PostEval
{
  public static void main(String[] args)
  {
    //sameple postfix expression
    String expr = "23*5+62";

    int result = Evaluate(expr);
    System.out.println("Result of postfix evaluation: "+result);

  }
  public static int Evaluate(String ex)
  {
    //create a stack
    int[] stack = new int [ex.length()];
    int tos=-1;

    //loop through the postfix ex
    for(int i=0;i<ex.length();i++)
    {
      char ch = ex.charAt(i);
      if(Character.isDigit(ch))  //if ch is a digit - push
      {
        tos++;
        stack[tos] = ch='0'; //convert char to int

      }
```

```
      else if (ch=='+'||ch=='-' || ch== '*' || ch== '/') //if ch is operator
      {
        int op2 = stack[tos--]; //pop the 2nd operand
        int op1 = stack[tos--]; //pop the 2nd operand

        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;


        }
        //push the result back on the stack
        tos++;
        stack[tos] = res;

      }
    } // end of forl
    return stack[tos];
  }//end of evalutate
}//end of PostEval
```

OUTPUT:

```
PS C:\Users\mcamock\DSAlab\queue\newbhava\stack> java  PostEval
Result of postfix evaluation: 48
```