```java
package main;

import java.util.Arrays;
import java.util.Scanner;

import operator.*;

/**
 * This class represent the usage of the Calculator from a terminal/console. It
 * allows the user to enter values and operators with the keyboard, and the
 * state of the current value and stack is printed after each operations.
 *
 * @author Sacha Bron
 * @author Valentin Minder
 */
public class Calculator {

    public static void main(String[] args) {

        State state = State.getInstance();
        Scanner scan = new Scanner(System.in);
        System.out.println("Welcome to the REVERSE POLISH TERMINAL CALCULATOR");

        String line = "";
        boolean flag = true;

        while (flag) {
            System.out.print("> ");
            line = scan.nextLine().trim().toLowerCase();
            if (line.equals("exit")) {
                flag = false;
                break;
            } else if (line.equals("+")) {
                new PlusOperator().execute();
            } else if (line.equals("-")) {
                new MinusOperator().execute();
            } else if (line.equals("/")) {
                new DivOperator().execute();
            } else if (line.equals("*")) {
                new TimesOperator().execute();
            } else if (line.equals("sqrt")) {
                new SqrtOperator().execute();
            } else if (line.equals("1/x")) {
                new OneOverXOperator().execute();
            } else if (line.equals("x^2")) {
                new SquareOperator().execute();
            } else if (line.equals("mr")) {
                new MROperator().execute();
            } else if (line.equals("ms")) {
                new MSOperator().execute();
            } else if (line.equals("c")) {
                new COperator().execute();
            } else if (line.equals("ce")) {
                new CEOperator().execute();
```

```
55              } else if (line.equals("enter")) {
56                  new EnterOperator().execute();
57              } else {
58                  if (line.length() > 0) {
59                      boolean changeSign = false;
60                      if (line.charAt(0) == '-') {
61                          changeSign = true;
62                          line = line.substring(1, line.length());
63                      }
64                      if (line.charAt(0) == '+') {
65                          line = line.substring(1, line.length());
66                      }
67                      for (int i = 0; i < line.length(); i++) {
68                          char a = line.charAt(i);
69                          if (a == '.') {
70                              new DotOperator().execute();
71                          } else if (a != ' ') {
72                              try {
73                                  new DigitOperator(Integer.parseInt(a + ""))
74                                          .execute();
75                                  ;
76                              } catch (NumberFormatException e) {
77                                  System.err
78                                          .println("Not a valid number. Try again");
79                                  break;
80                              }
81                          }
82                      }
83                      if (changeSign) {
84                          new SignOperator().execute();
85                      }
86                  }
87                  // in order to make it immutable, so that the next
88                  // call doesnt modify the value but push it on the stack.
89                  new MSOperator().execute();
90                  new MROperator().execute();
91              }
92
93              System.out.print(state.getValueString() + " ");
94              System.out.println(Arrays.toString(state.getStackState()));
95          }
96          scan.close();
97      }
98  }
99
```