Universidad Tecnológica de Panamá

Facultad de Ingeniería en Sistemas Computacionales

Asignatura: Desarrollo Software III

Ejercicio Práctico2

Profesor: Napoleón Ibarra Valor: 100 puntos

Nombre: Jorge Javier Jiménez Ruiz Cédula: 4-826-874

Procedimiento:

- ✓ De manera individual, desarrolle los problemas.
- ✓ Se debe entregar al profesor: Documento digital: entrega en la plataforma (TEAM) el y/o los códigos desarrollando los problemas. Sustente su trabajo en el aula de clases.

Criterios de Evaluación:

Criterios	Puntos (Mínimo=1, Máximo=5)	Porcentaje
Desarrollo	1-5	70 %
Sustentación	1-5	15 %
Puntualidad	1-5	15 %

I Parte. Desarrollo de problemas en JAVA. Valor 70 Puntos

El Colegio XZ de le contrata a Usted (es) por servicios profesionales para que le desarrolle un proyecto. Requiere que se confeccione un prototipo (ver Figura 1) de desarrollo con interfaz gráfica. Verificar que los botones sean funcionales, hacer las pruebas y ajustes correspondientes. Presente una factura de sus servicios a la empresa, quien está representada por el Profesor.

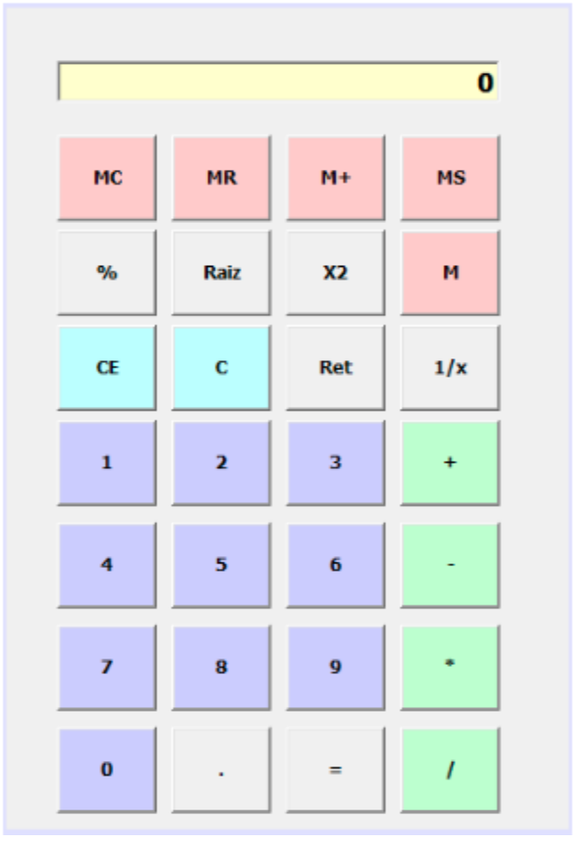


Figura 1. Prototipo de calculadora.

Desarrollo – Código de la calculadora

Fue hecho de manera manual:

Calculadora.java

Source | History

```
1  package calculadorapratica;
2
3  import javax.swing.*;
4  import java.awt.*;
5  import java.awt.event.*;
6
7  public class Calculadora extends JFrame implements ActionListener {
8
9      private JTextField display;
10     private double memory = 0;
11     private String lastCommand = "=";
12     private boolean start = true;
13     private double result = 0;
14     private StringBuilder expression = new StringBuilder();
15
16     public Calculadora() {
17         setTitle("Calculadora");
18         setSize(400, 500);
19         setDefaultCloseOperation(EXIT_ON_CLOSE);
20         setLocationRelativeTo(null);
21         setLayout(new BorderLayout());
22
23         display = new JTextField("0");
24         display.setEditable(false);
25         display.setHorizontalAlignment(JTextField.RIGHT);
26         display.setFont(new Font("Arial", Font.BOLD, 24));
27         add(display, BorderLayout.NORTH);
```

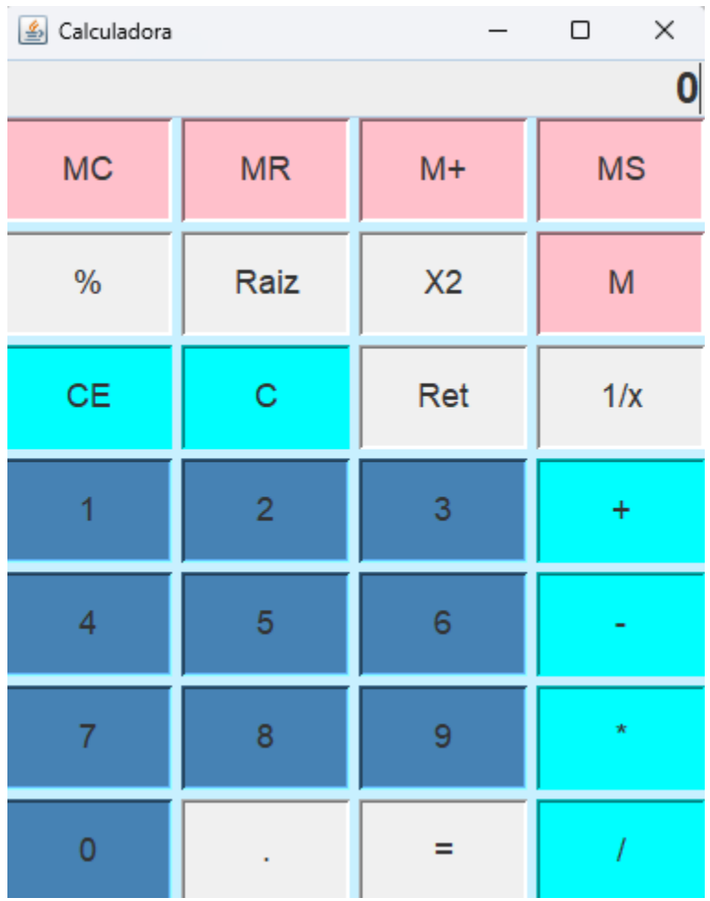
```
28
29          String[][] buttons = {
30              {"MC", "MR", "M+", "MS"},
31              {"%", "Raiz", "X2", "M"},
32              {"CE", "C", "Ret", "1/x"},
33              {"1", "2", "3", "+"},
34              {"4", "5", "6", "-"},
35              {"7", "8", "9", "*"},
36              {"0", ".", "=", "/"}
37          };
38
39          JPanel panel = new JPanel();
40          panel.setLayout(new GridLayout(7, 4, 5, 5));
41          panel.setBackground(new Color(200, 240, 255)); // fondo
42
            for (String[] row : buttons) {
                for (String text : row) {
45                  JButton btn = new JButton(text);
46                  btn.setFont(new Font("Arial", Font.PLAIN, 18));
                    btn.addActionListener(this);
48                  btn.setFocusPainted(false); // quita borde de enfoque
49
50                  // Colores personalizados según función
51                  if (text.matches("MC|MR|M\\+|MS|M")) {
52                      btn.setBackground(new Color(255, 192, 203)); // rosado
53                  } else if (text.matches("CE|C|\\+|\\-|\\*|/")) {
54                      btn.setBackground(Color.CYAN);
55                  } else if (text.matches("[0-9]")) {
56                      btn.setBackground(new Color(70, 130, 180)); // azul oscuro suave
57                  } else {
58                      btn.setBackground(new Color(240, 240, 240)); // gris claro
59                  }
60
61                  btn.setBorder(BorderFactory.createBevelBorder(1)); // borde 3D
62                  panel.add(btn);
63              }
64          }
65
66          add(panel, BorderLayout.CENTER);
67          setVisible(true);
68      }
69
        public void actionPerformed(ActionEvent e) {
71          String cmd = e.getActionCommand();
72          try {
                switch (cmd) {
74                  case "CE":
75                  case "C":
76                      display.setText("0");
77                      expression.setLength(0);
78                      start = true;
79                      break;
80                  case "Ret":
81                      if (expression.length() > 0) {
```

```
                    expression.setLength(expression.length() - 1);
                    display.setText(expression.length() > 0 ? expression.toString() : "0");
                }
                break;
            case "Raiz":
                double raiz = Math.sqrt(getCurrentNumber());
                displayResult(raiz);
                break;
            case "X2":
                double val = getCurrentNumber();
                displayResult(val * val);
                break;
            case "1/x":
                double d = getCurrentNumber();
                if (d == 0) throw new ArithmeticException("División entre cero");
                displayResult(1 / d);
                break;
            case "%":
                double p = getCurrentNumber() / 100;
                displayResult(p);
                break;
            case "MS":
                memory = getCurrentNumber();
                break;
            case "MR":
                expression = new StringBuilder(formatNumber(memory));
                display.setText(expression.toString());
                break;
            case "M+":
                memory += getCurrentNumber();
                break;
            case "MC":
                memory = 0;
                break;
            case "=":
                calculate(getCurrentNumber());
                lastCommand = "=";
                expression = new StringBuilder(formatNumber(result));
                break;
            case "+":
            case "-":
            case "*":
            case "/":
                calculate(getCurrentNumber());
                lastCommand = cmd;
                expression.append(" ").append(cmd).append(" ");
                display.setText(expression.toString());
                start = true;
                break;
            case ".":
                if (!getLastToken().contains(".")) {
                    expression.append(".");
                    display.setText(expression.toString());
                }
```

```
                    break;
                default: // Números
                    if (start && isLastCharOperator()) {
                        expression.append(cmd);
                    } else if (start || display.getText().equals("0")) {
                        expression.append(cmd);
                    } else {
                        expression.append(cmd);
                    }
                    display.setText(expression.toString());
                    start = false;
                    break;
            }
        } catch (Exception ex) {
            JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
        }
    }

    private void calculate(double x) {
        switch (lastCommand) {
            case "+": result += x; break;
            case "-": result -= x; break;
            case "*": result *= x; break;
            case "/": result /= x; break;
            case "=": result = x; break;
        }
        displayResult(result);
    }

    private void displayResult(double val) {
        String formatted = formatNumber(val);
        display.setText(formatted);
        expression = new StringBuilder(formatted);
    }

    private double getCurrentNumber() {
        String[] parts = expression.toString().trim().split(" ");
        String last = parts[parts.length - 1];
        return Double.parseDouble(last);
    }

    private String formatNumber(double val) {
        return (val == (int) val) ? Integer.toString((int) val) : Double.toString(val);
    }

    private String getLastToken() {
        String[] tokens = expression.toString().split(" ");
        return tokens[tokens.length - 1];
    }

    private boolean isLastCharOperator() {
        if (expression.length() == 0) return false;
        char c = expression.charAt(expression.length() - 1);
        return c == '+' || c == '-' || c == '*' || c == '/';
```

```
190        }
191
192        public static void main(String[] args) {
193            SwingUtilities.invokeLater(() -> new Calculadora());
194        }
195    }
```

Resultado:



(También grabe un video anteriormente mostrando que funciona)