GUI CALCULATOR:→By Janak Sapkota

#PROGRAM SOURCE CODE

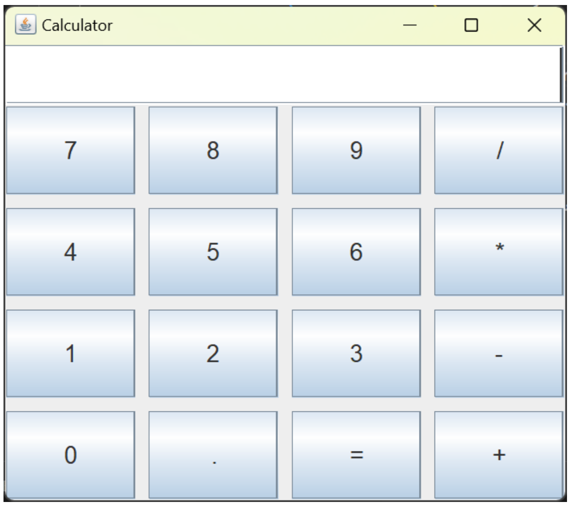
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class CalculatorGUI extends JFrame {

    private JTextField textField;

    public CalculatorGUI() {
        setTitle("Calculator");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setPreferredSize(new Dimension(300, 400));

        textField = new JTextField();
        textField.setFont(new Font("Arial", Font.PLAIN, 35));
        textField.setHorizontalAlignment(JTextField.RIGHT);
        add(textField, BorderLayout.NORTH);

        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new GridLayout(4, 4, 10, 10));

        String[] buttonLabels = {
                "7", "8", "9", "/",
                "4", "5", "6", "*",
            "1", "2", "3", "-",
            "0", ".", "=", "+"
        };

        for (String label : buttonLabels) {
            JButton button = new JButton(label);
            button.setFont(new Font("Arial", Font.PLAIN, 18));
            button.addActionListener(new ButtonClickListener());
            buttonPanel.add(button);
        }

        add(buttonPanel, BorderLayout.CENTER);

        pack();
        setLocationRelativeTo(null);
    }

    private class ButtonClickListener implements ActionListener {
        public void actionPerformed(ActionEvent event) {
            JButton source = (JButton) event.getSource();
            String buttonText = source.getText();

            if (buttonText.equals("=")) {
                evaluateExpression();
            } else {
                textField.setText(textField.getText() + buttonText);
            }
        }
```

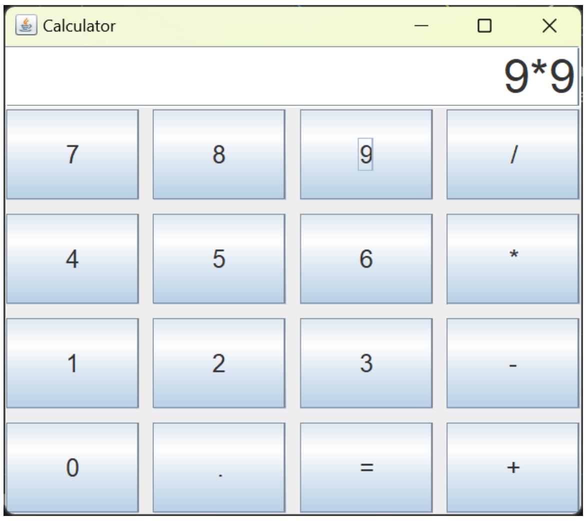
```
private void evaluateExpression() {
    try {
        String expression = textField.getText();
        double result = evaluate(expression);
        textField.setText(String.valueOf(result));
    } catch (Exception e) {
        textField.setText("Error");
    }
}

private double evaluate(String expression) {
    return (double) new Object() {
        int pos = -1, ch;

        void nextChar() {
            ch = (++pos < expression.length()) ? expression.charAt(pos) : -1;
        }

        boolean eat(int charToEat) {
            while (Character.isWhitespace(ch))
                nextChar();
            if (ch == charToEat) {
                nextChar();
                return true;
            }
            return false;
        }

        double parse() {
```

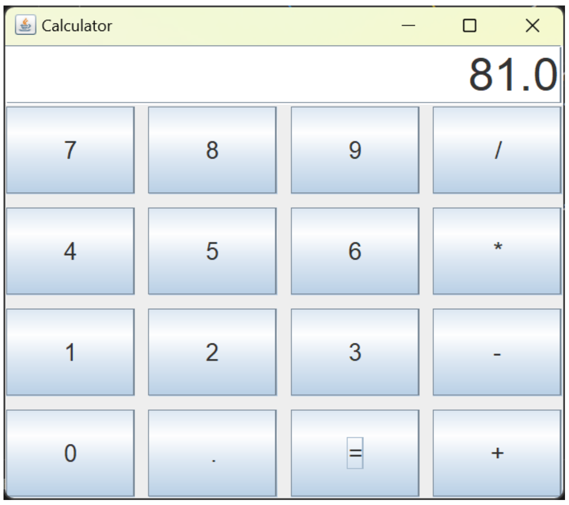
```
    nextChar();
    double x = parseExpression();
    if (pos < expression.length())
        throw new RuntimeException("Unexpected: " + (char) ch);
    return x;
}

// Grammar:
// expression = term | expression `+` term | expression `-` term
// term = factor | term `*` factor | term `/` factor
// factor = `+` factor | `-` factor | `(` expression `)` | number

double parseExpression() {
    double x = parseTerm();
    for (;;) {
        if (eat('+'))
            x += parseTerm(); // addition
        else if (eat('-'))
            x -= parseTerm(); // subtraction
        else
            return x;
    }
}

double parseTerm() {
    double x = parseFactor();
    for (;;) {
        if (eat('*'))
            x *= parseFactor(); // multiplication
            else if (eat('/'))
                x /= parseFactor(); // division
            else
                return x;
        }
    }

    double parseFactor() {
        if (eat('+'))
            return parseFactor(); // unary plus
        if (eat('-'))
            return -parseFactor(); // unary minus

        double x;
        int startPos = pos;
        if (eat('(')) { // parentheses
            x = parseExpression();
            eat(')');
        } else if ((ch >= '0' && ch <= '9') || ch == '.') { // numbers
            while ((ch >= '0' && ch <= '9') || ch == '.')
                nextChar();
            x = Double.parseDouble(expression.substring(startPos, pos));
        } else {
            throw new RuntimeException("Unexpected: " + (char) ch);
        }

        return x;
    }
}.parse();
```

```
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            CalculatorGUI calculator = new CalculatorGUI();
            calculator.setVisible(true);
        });
    }
}
```

#PROGRAM OUTPUT




9*9


81.0