```java
import java.awt.*;
import java.lang.reflect.InvocationTargetException;
import java.util.Scanner;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Calculator {
    private JFrame frame;
    private static JTextField textField1;
    private static JTextField textField2;
    private static JTextField textFieldResult;
    private JButton buttonAdd, buttonSubtract, buttonMultiply, buttonDivide;
    private JPanel panel;
    private Scanner scanner;

    public Calculator() {
        frame = new JFrame("Calculator");

        textField1 = new JTextField(0);
        textField1.setBackground(Color.DARK_GRAY);
        textField1.setFont(new Font("Arial", Font.BOLD, 20));
        textField1.setForeground(Color.WHITE);

        textField2 = new JTextField(0);
        textField2.setBackground(Color.DARK_GRAY);
        textField2.setFont(new Font("Arial", Font.BOLD, 20));
        textField2.setForeground(Color.WHITE);

        buttonAdd = new JButton("+");
        buttonAdd.setBackground(Color.DARK_GRAY);
        buttonAdd.setFont(new Font("Arial", Font.BOLD, 20));
        buttonAdd.setForeground(Color.WHITE);

        buttonSubtract = new JButton("-");
        buttonSubtract.setBackground(Color.DARK_GRAY);
        buttonSubtract.setFont(new Font("Arial", Font.BOLD, 20));
        buttonSubtract.setForeground(Color.WHITE);

        buttonMultiply = new JButton("*");
        buttonMultiply.setBackground(Color.DARK_GRAY);
        buttonMultiply.setFont(new Font("Arial", Font.BOLD, 20));
        buttonMultiply.setForeground(Color.WHITE);

        buttonDivide = new JButton("/");
        buttonDivide.setBackground(Color.DARK_GRAY);
        buttonDivide.setFont(new Font("Arial", Font.BOLD, 20));
        buttonDivide.setForeground(Color.WHITE);

        textFieldResult = new JTextField(0);
        textFieldResult.setBackground(Color.DARK_GRAY);
        textFieldResult.setFont(new Font("Arial", Font.BOLD, 20));
        textFieldResult.setForeground(Color.WHITE);


        panel = new JPanel();
        panel.setLayout(new GridLayout(5, 4));
        panel.add(textField1);
        panel.add(textField2);
        panel.add(buttonAdd);
```

```
        panel.add(buttonSubtract);
        panel.add(buttonMultiply);
        panel.add(buttonDivide);
        panel.add(textFieldResult);
        frame.add(panel, BorderLayout.CENTER);

        frame.setSize(640, 480);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        scanner = new Scanner(System.in);

        buttonAdd.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                double number1 = Double.parseDouble(textField1.getText());
                double number2 = Double.parseDouble(textField2.getText());
                double result = calculate(number1, number2, "+");
                textFieldResult.setText("REZULTAT: " + String.valueOf(result));
            }
        });

        buttonSubtract.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                double number1 = Double.parseDouble(textField1.getText());
                double number2 = Double.parseDouble(textField2.getText());
                double result = calculate(number1, number2, "-");
                textFieldResult.setText("REZULTAT: " + String.valueOf(result));
            }
        });

        buttonMultiply.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                double number1 = Double.parseDouble(textField1.getText());
                double number2 = Double.parseDouble(textField2.getText());
                double result = calculate(number1, number2, "*");
                textFieldResult.setText("REZULTAT: " + String.valueOf(result));
            }
        });

        buttonDivide.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                double number1 = Double.parseDouble(textField1.getText());
                double number2 = Double.parseDouble(textField2.getText());
                double result = calculate(number1, number2, "/");
                textFieldResult.setText("REZULTAT: " + String.valueOf(result));
            }
        });
    }

    public double calculate(double num1, double num2, String operation) {
        // Check the value of the "operation" argument and perform the corresponding
    calculation
        if (operation.equals("+")) {
            return num1 + num2;
        } else if (operation.equals("-")) {
            return num1 - num2;
        } else if (operation.equals("*")) {
```

```
            return num1 * num2;
        } else if (operation.equals("/")) {
            return num1 / num2;
        } else {
            // If the operation is invalid, return 0
            return 0;
        }
    }

    public static void main(String[] args) {
        try {
            // Create a new Calculator object
            Calculator calculator = new Calculator();
            // Set the frame to be visible
            calculator.frame.setVisible(true);
            SwingUtilities.invokeAndWait(new Runnable() {
                @Override
                public void run() {
                    calculator.frame.setVisible(true);
                    textField1.setText("");
                    textField2.setText("");
                }
            });
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (InvocationTargetException e) {
            e.printStackTrace();
        }
    }
}
```

```java
import java.awt.*;
import java.lang.reflect.InvocationTargetException;
import java.util.Scanner;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.concurrent.ExecutionException;
import javax.swing.*;

public class Calculator {
    private JFrame frame;
    private static JTextField textField1;
    private static JTextField textField2;
    private static JTextField textFieldResult;
    private JButton buttonAdd, buttonSubtract, buttonMultiply, buttonDivide;
    private JPanel panel;
    private Scanner scanner;

    public Calculator() {
        frame = new JFrame("Calculator");

        textField1 = new JTextField(1);
        textField1.setBackground(Color.DARK_GRAY);
        textField1.setFont(new Font("Arial", Font.BOLD, 20));
        textField1.setForeground(Color.WHITE);

        textField2 = new JTextField(1);
        textField2.setBackground(Color.DARK_GRAY);
        textField2.setFont(new Font("Arial", Font.BOLD, 20));
        textField2.setForeground(Color.WHITE);

        buttonAdd = new JButton("+");
        buttonAdd.setBackground(Color.DARK_GRAY);
        buttonAdd.setFont(new Font("Arial", Font.BOLD, 20));
        buttonAdd.setForeground(Color.WHITE);

        buttonSubtract = new JButton("-");
        buttonSubtract.setBackground(Color.DARK_GRAY);
        buttonSubtract.setFont(new Font("Arial", Font.BOLD, 20));
        buttonSubtract.setForeground(Color.WHITE);

        buttonMultiply = new JButton("*");
        buttonMultiply.setBackground(Color.DARK_GRAY);
        buttonMultiply.setFont(new Font("Arial", Font.BOLD, 20));
        buttonMultiply.setForeground(Color.WHITE);

        buttonDivide = new JButton("/");
        buttonDivide.setBackground(Color.DARK_GRAY);
        buttonDivide.setFont(new Font("Arial", Font.BOLD, 20));
        buttonDivide.setForeground(Color.WHITE);

        textFieldResult = new JTextField("Nakon odabira operatora ovdje će se pojaviti rezultat.",2);
        textFieldResult.setBackground(Color.DARK_GRAY);
        textFieldResult.setFont(new Font("Arial", Font.ITALIC, 12));
        textFieldResult.setForeground(Color.WHITE);
        textFieldResult.setEditable(false); // make the result field non-editable

        panel = new JPanel();
        panel.setLayout(new GridLayout(4, 2));
        panel.add(textField1);
```

```
        panel.add(textField2);
        panel.add(buttonAdd);
        panel.add(buttonSubtract);
        panel.add(buttonMultiply);
        panel.add(buttonDivide);
        panel.add(textFieldResult);
        frame.add(panel, BorderLayout.CENTER);

        frame.setSize(640, 480);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        scanner = new Scanner(System.in);

        buttonAdd.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                // get the numbers from the text fields
                double number1 = Double.parseDouble(textField1.getText());
                double number2 = Double.parseDouble(textField2.getText());

                // perform the addition in a separate thread using a SwingWorker
                SwingWorker<Double, Void> worker = new SwingWorker<Double, Void>() {
                    @Override
                    protected Double doInBackground() throws Exception {
                        return number1 + number2;
                    }

                    @Override
                    protected void done() {
                        try {
                            // get the result of the calculation and update the result
   field
                            // in the event dispatch thread using the invokeLater() method
                            final double result = get();
                            SwingUtilities.invokeLater(new Runnable() {
                                public void run() {
                                    textFieldResult.setText(String.valueOf(result));
                                }
                            });
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        } catch (ExecutionException e) {
                            throw new RuntimeException(e);
                        }
                    }
                };
                worker.execute();
            }
        });

        buttonSubtract.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                // get the numbers from the text fields
                double number1 = Double.parseDouble(textField1.getText());
                double number2 = Double.parseDouble(textField2.getText());

                // perform the subtraction in a separate thread using a SwingWorker
                SwingWorker<Double, Void> worker = new SwingWorker<Double, Void>() {
                    @Override
                    protected Double doInBackground() throws Exception {
                        return number1 - number2;
```

```
                    }

                    @Override
                    protected void done() {
                        try {
                            // get the result of the calculation and update the result field
                            // in the event dispatch thread using the invokeLater() method
                            final double result = get();
                            SwingUtilities.invokeLater(new Runnable() {
                                public void run() {
                                    textFieldResult.setText(String.valueOf(result));
                                }
                            });
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        } catch (ExecutionException e) {
                            throw new RuntimeException(e);
                        }
                    }
                };
                worker.execute();
            }
        });

        buttonMultiply.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                // get the numbers from the text fields
                double number1 = Double.parseDouble(textField1.getText());
                double number2 = Double.parseDouble(textField2.getText());

                // perform the multiplication in a separate thread using a SwingWorker
                SwingWorker<Double, Void> worker = new SwingWorker<Double, Void>() {
                    @Override
                    protected Double doInBackground() throws Exception {
                        return number1 * number2;
                    }

                    @Override
                    protected void done() {
                        try {
                            // get the result of the calculation and update the result field
                            // in the event dispatch thread using the invokeLater() method
                            final double result = get();
                            SwingUtilities.invokeLater(new Runnable() {
                                public void run() {
                                    textFieldResult.setText(String.valueOf(result));
                                }
                            });
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        } catch (ExecutionException e) {
                            throw new RuntimeException(e);
                        }
                    }
                };
                worker.execute();
            }
        });
```

```
177
178         buttonDivide.addActionListener(new ActionListener() {
179             public void actionPerformed(ActionEvent event) {
180                 // get the numbers from the text fields
181                 double number1 = Double.parseDouble(textField1.getText());
182                 double number2 = Double.parseDouble(textField2.getText());
183 
184                 // perform the division in a separate thread using a SwingWorker
185                 SwingWorker<Double, Void> worker = new SwingWorker<Double, Void>() {
186                     @Override
187                     protected Double doInBackground() throws Exception {
188                         return number1 / number2;
189                     }
190 
191                     @Override
192                     protected void done() {
193                         try {
194                             // get the result of the calculation and update the result
    field
195                             // in the event dispatch thread using the invokeLater() method
196                             final double result = get();
197                             SwingUtilities.invokeLater(new Runnable() {
198                                 public void run() {
199                                     textFieldResult.setText(String.valueOf(result));
200                                 }
201                             });
202                         } catch (InterruptedException e) {
203                             e.printStackTrace();
204                         } catch (ExecutionException e) {
205                             throw new RuntimeException(e);
206                         }
207                     }
208                 };
209                 worker.execute();
210             }
211         });
212     }
213 
214     public static void main(String[] args) {
215         try {
216             // Create a new Calculator object
217             Calculator calculator = new Calculator();
218             // Set the frame to be visible
219             calculator.frame.setVisible(true);
220             SwingUtilities.invokeAndWait(new Runnable() {
221                 @Override
222                 public void run() {
223                     calculator.frame.setVisible(true);
224                     textField1.setText("");
225                     textField2.setText("");
226                 }
227             });
228         } catch (InterruptedException e) {
229             e.printStackTrace();
230         } catch (InvocationTargetException e) {
231             e.printStackTrace();
232         }
233     }
234 }
235 
```