Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,\*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class Calculator extends JFrame implements ActionListener {

private JTextField textField;

public Calculator() {

super("Calculator");

setSize(250, 200);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new BorderLayout());

JPanel buttonPanel = new JPanel();

buttonPanel.setLayout(new GridLayout(4, 4, 5, 5));

String[] buttonLabels = {"7", "8", "9", "/",

"4", "5", "6", "\*",

"1", "2", "3", "-",

"0", ".", "=", "+"};

for (String label : buttonLabels) {

JButton button = new JButton(label);

button.addActionListener(this);

buttonPanel.add(button);

}

textField = new JTextField();

add(textField, BorderLayout.NORTH);

add(buttonPanel, BorderLayout.CENTER);

}

public void actionPerformed(ActionEvent e) {

String command = e.getActionCommand();

if (command.equals("=")) {

try {

double result = evaluate(textField.getText());

textField.setText(Double.toString(result));

} catch (ArithmeticException ex) {

textField.setText("Divide by zero error");

} catch (Exception ex) {

textField.setText("Error");

}

} else {

textField.setText(textField.getText() + command);

}

}

private double evaluate(String expression) throws Exception {

char[] tokens = expression.toCharArray();

Stack<Double> values = new Stack<Double>();

Stack<Character> ops = new Stack<Character>();

for (int i = 0; i < tokens.length; i++) {

if (tokens[i] == ' ') {

continue;

}

if (tokens[i] >= '0' && tokens[i] <= '9') {

StringBuilder sb = new StringBuilder();

while (i < tokens.length && (tokens[i] >= '0' && tokens[i] <= '9' || tokens[i] == '.')) {

sb.append(tokens[i++]);

}

values.push(Double.parseDouble(sb.toString()));

} else if (tokens[i] == '+' || tokens[i] == '-' || tokens[i] == '\*' || tokens[i] == '/') {

while (!ops.empty() && hasPrecedence(tokens[i], ops.peek())) {

values.push(applyOperation(ops.pop(), values.pop(), values.pop()));

}

ops.push(tokens[i]);

}

}

while (!ops.empty()) {

values.push(applyOperation(ops.pop(), values.pop(), values.pop()));

}

return values.pop();

}

private boolean hasPrecedence(char op1, char op2) {

if (op2 == '(' || op2 == ')') {

return false;

}

if ((op1 == '\*' || op1 == '/') && (op2 == '+' || op2 == '-')) {

return false;

}

return true;

}

private double applyOperation(char op, double b, double a) throws ArithmeticException {

switch (op) {

case '+':

return a + b;

case '-':

return a - b;

case '\*':

return a \* b;

case '/':

if (b == 0) {

throw new ArithmeticException();

}

return a / b;

}

throw new IllegalArgumentException("Invalid operator: " + op);

}

public static void main(String[] args) {

Calculator calculator = new Calculator();

calculator.setVisible(true);

}

}

//This program creates a JFrame with a text field and a grid of buttons for

performing arithmetic operations. The actionPerformed method is called whenever a button is pressed. If the button is not the equals button, the corresponding character is appended to the text field. If the equals button is pressed, the evaluate method is called to calculate the result of the expression in the text field. If an ArithmeticException is thrown, the text field is set to "Divide by zero error". If any other exception is thrown, the text field is set to "Error".

The evaluate method evaluates the expression using a stack-based algorithm for evaluating postfix expressions. If a division by zero is attempted, an ArithmeticException is thrown.

To use this program, simply run the main method. You should see a window with a calculator interface. You can enter numbers and perform arithmetic operations using the buttons. The result will be displayed in the text field. If you attempt to divide by zero or enter an invalid expression, an error message will be displayed in the text field.

You can see that the interface has a grid of buttons for the digits and arithmetic operations, and a text field for displaying the result. You can enter expressions by clicking on the buttons or typing on the keyboard. When you press the equals button, the program evaluates the expression and displays the result in the text field. If an error occurs, an appropriate error message is displayed instead.

Note that this program is a very simple calculator and does not implement advanced features like parentheses, exponentiation, or trigonometric functions. However, it provides a good starting point for building more complex calculator programs.

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

implementation of a simple calculator program in Java using Swing and BorderLayout.

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class Calculator {

private JFrame frame;

private JPanel panel;

private JTextField display;

private String operator = "";

private double firstNumber = 0.0;

private boolean clearOnNextDigit = false;

public Calculator() {

frame = new JFrame("Calculator");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

panel = new JPanel(new BorderLayout());

display = new JTextField();

display.setEditable(false);

panel.add(display, BorderLayout.NORTH);

JPanel buttonPanel = new JPanel(new GridLayout(4, 4));

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

for (String label : buttonLabels) {

JButton button = new JButton(label);

button.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (label.matches("[0-9.]")) {

if (clearOnNextDigit) {

display.setText("");

clearOnNextDigit = false;

}

display.setText(display.getText() + label);

} else if (label.equals("+") || label.equals("-") || label.equals("\*") || label.equals("/")) {

firstNumber = Double.parseDouble(display.getText());

operator = label;

clearOnNextDigit = true;

} else if (label.equals("=")) {

double secondNumber = Double.parseDouble(display.getText());

double result = 0.0;

switch (operator) {

case "+":

result = firstNumber + secondNumber;

break;

case "-":

result = firstNumber - secondNumber;

break;

case "\*":

result = firstNumber \* secondNumber;

break;

case "/":

if (secondNumber == 0.0) {

display.setText("Divide by zero error");

return;

} else {

result = firstNumber / secondNumber;

break;

}

}

display.setText(Double.toString(result));

clearOnNextDigit = true;

}

}

});

buttonPanel.add(button);

}

panel.add(buttonPanel, BorderLayout.CENTER);

frame.add(panel);

frame.pack();

frame.setVisible(true);

}

public static void main(String[] args) {

new Calculator();

}

}

This implementation uses BorderLayout to arrange the display and buttons on the panel. The buttons are arranged in a grid using GridLayout. Each button has an ActionListener that handles the input and performs the necessary calculations. The calculator supports basic arithmetic operations and can handle division by zero errors. The program also checks for consecutive decimal points in the input and ignores them.

++++++++++++++++++++++++

implementation of a simple calculator program in Java using GridBagLayout.

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class Calculator {

private JFrame frame;

private JPanel panel;

private JTextField display;

private double num1 = 0;

private double num2 = 0;

private String operation = "";

public Calculator() {

frame = new JFrame("Calculator");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

panel = new JPanel(new GridBagLayout());

GridBagConstraints gbc = new GridBagConstraints();

gbc.gridx = 0;

gbc.gridy = 0;

gbc.gridwidth = 4;

gbc.fill = GridBagConstraints.HORIZONTAL;

display = new JTextField();

display.setEditable(false);

panel.add(display, gbc);

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

for (int i = 0; i < buttons.length; i++) {

JButton button = new JButton(buttons[i]);

gbc = new GridBagConstraints();

gbc.gridx = i % 4;

gbc.gridy = i / 4 + 1;

gbc.fill = GridBagConstraints.BOTH;

gbc.insets = new Insets(5, 5, 5, 5);

button.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String buttonText = ((JButton) e.getSource()).getText();

if (buttonText.matches("\\d")) {

display.setText(display.getText() + buttonText);

} else if (buttonText.equals(".")) {

if (!display.getText().contains(".")) {

display.setText(display.getText() + ".");

}

} else if (buttonText.equals("+")) {

num1 = Double.parseDouble(display.getText());

operation = "+";

display.setText("");

} else if (buttonText.equals("-")) {

num1 = Double.parseDouble(display.getText());

operation = "-";

display.setText("");

} else if (buttonText.equals("\*")) {

num1 = Double.parseDouble(display.getText());

operation = "\*";

display.setText("");

} else if (buttonText.equals("/")) {

num1 = Double.parseDouble(display.getText());

operation = "/";

display.setText("");

} else if (buttonText.equals("=")) {

num2 = Double.parseDouble(display.getText());

double result = 0;

switch (operation) {

case "+":

result = num1 + num2;

break;

case "-":

result = num1 - num2;

break;

case "\*":

result = num1 \* num2;

break;

case "/":

if (num2 == 0) {

display.setText("Divide by zero error");

return;

} else {

result = num1 / num2;

break;

}

}

display.setText(Double.toString(result));

}

}

});

button.setFont(new Font("SansSerif", Font.PLAIN, 16));

panel.add(button, gbc);

}

frame.add(panel);

frame.pack();

frame.setLocationRelativeTo(null);

frame.setVisible(true);

}

public static void main(String[] args) {

new Calculator();

}

}

This implementation uses GridBagLayout to arrange the display and buttons on the panel. The buttons are created using a loop, and each button has an ActionListener that handles the input and performs the necessary calculations. The calculator supports basic arithmetic operations and can handle division by zero errors. The program also checks for consecutive decimal points in the input and ignores them.

+++++++++++++++++++++++++

implementation of a simple calculator in Java using a different layout and design approach:

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Calculator {

private JFrame frame;

private JPanel panel;

private JTextField display;

private JButton[] buttons;

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

private String operation = "";

private double num1 = 0;

private double num2 = 0;

public Calculator() {

frame = new JFrame("Calculator");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

panel = new JPanel(new GridLayout(5, 4, 5, 5));

display = new JTextField();

display.setEditable(false);

buttons = new JButton[buttonLabels.length];

for (int i = 0; i < buttonLabels.length; i++) {

buttons[i] = new JButton(buttonLabels[i]);

buttons[i].addActionListener(new ButtonHandler());

panel.add(buttons[i]);

}

frame.add(display, BorderLayout.NORTH);

frame.add(panel, BorderLayout.CENTER);

frame.pack();

frame.setVisible(true);

}

private class ButtonHandler implements ActionListener {

public void actionPerformed(ActionEvent e) {

String buttonLabel = e.getActionCommand();

switch (buttonLabel) {

case "C":

display.setText("");

num1 = 0;

num2 = 0;

operation = "";

break;

case "+":

case "-":

case "\*":

case "/":

if (operation.equals("")) {

num1 = Double.parseDouble(display.getText());

operation = buttonLabel;

display.setText("");

} else {

num2 = Double.parseDouble(display.getText());

calculate();

num1 = Double.parseDouble(display.getText());

operation = buttonLabel;

display.setText("");

}

break;

case "=":

num2 = Double.parseDouble(display.getText());

calculate();

operation = "";

break;

default:

display.setText(display.getText() + buttonLabel);

break;

}

}

private void calculate() {

switch (operation) {

case "+":

display.setText(Double.toString(num1 + num2));

break;

case "-":

display.setText(Double.toString(num1 - num2));

break;

case "\*":

display.setText(Double.toString(num1 \* num2));

break;

case "/":

if (num2 == 0) {

JOptionPane.showMessageDialog(frame, "Cannot divide by zero", "Error", JOptionPane.ERROR\_MESSAGE);

display.setText("");

} else {

display.setText(Double.toString(num1 / num2));

}

break;

}

}

}

public static void main(String[] args) {

new Calculator();

}

}

This implementation uses a **GridLayout** to create a 4x5 grid of buttons. The **ButtonHandler** class implements the **ActionListener** interface and handles button clicks in a switch statement similar to the previous implementations.

This implementation also includes a "C" button to clear the display and reset the calculator state.

+++++++++++++++++++++++++++++++