**1. Prime Numbers up to that Integer**

**SOURCE CODE:**

import java.util.Scanner;

public class PrimeNumbers {

public static void main(String[] args) {

try (Scanner scanner = new Scanner(System.in)) {

System.out.print("Enter an integer: ");

int number = scanner.nextInt();

System.out.println("Prime numbers up to " + number + ":");

for (int i = 2; i <= number; i++) {

if (isPrime(i)) {

System.out.print(i + " ");

}

}

}

}

public static boolean isPrime(int n) {

if (n <= 1) return false;

if (n == 2) return true;

if (n % 2 == 0) return false;

for (int i = 3; i <= Math.sqrt(n); i += 2) {

if (n % i == 0) return false;

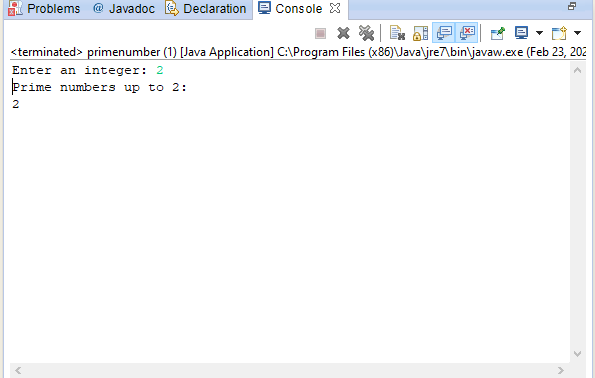
}

return true;

}

}

OUTPUT:



**2. To create a Multiply two given Matrices**

**SOURCE CODE:**

import java.util.Scanner;

public class MatrixMultiplication {

public static void main(String[] args) {

try (Scanner scanner = new Scanner(System.in)) {

System.out.println("Enter the number of rows of the first matrix:");

int rows1 = scanner.nextInt();

System.out.println("Enter the number of columns of the first matrix:");

int cols1 = scanner.nextInt();

System.out.println("Enter the number of rows of the second matrix:");

int rows2 = scanner.nextInt();

System.out.println("Enter the number of columns of the second matrix:");

int cols2 = scanner.nextInt();

if (cols1 != rows2) {

System.out.println("Matrices cannot be multiplied due to incompatible dimensions.");

return;

}

int[][] matrix1 = new int[rows1][cols1];

int[][] matrix2 = new int[rows2][cols2];

System.out.println("Enter the elements of the first matrix:");

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols1; j++) {

matrix1[i][j] = scanner.nextInt();

}

}

System.out.println("Enter the elements of the second matrix:");

for (int i = 0; i < rows2; i++) {

for (int j = 0; j < cols2; j++) {

matrix2[i][j] = scanner.nextInt();

}

}

int[][] result = new int[rows1][cols2];

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

for (int k = 0; k < cols1; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

System.out.println("Resultant matrix:");

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

System.out.print(result[i][j] + " ");

}

System.out.println();

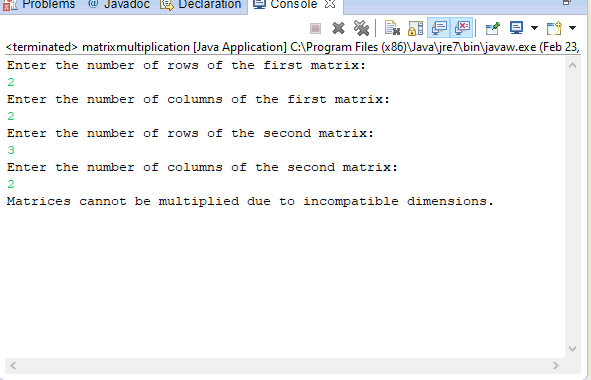
}

}

}

}

**OUTPUT:**

****

**3. To display the number of characters,lines and words in a text**

**SOURCE CODE:**

package javapractical;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class TextFileAnalyzer {

public static void main(String[] args) {

// Initialize counters

int characterCount = 0;

int wordCount = 0;

int lineCount = 0;

// Specify the path to the text file

String filePath = "D:/balaa/java practical program/textfile.txt";

try {

// Open the file for reading

BufferedReader reader = new BufferedReader(new FileReader (filePath));

String line;

// Process each line

while ((line = reader.readLine()) != null) {

// Increment line counter

lineCount++;

// Increment character counter

characterCount += line.length();

// Split the line into words and increment word counter

String[] words = line.split("\\s+");

wordCount += words.length;

}

// Close the file

reader.close();

// Display the counts

System.out.println("Number of characters:" + characterCount);

System.out.println("Number of words: " + wordCount);

System.out.println("Number of lines:"+ lineCount);

} catch (IOException e) {

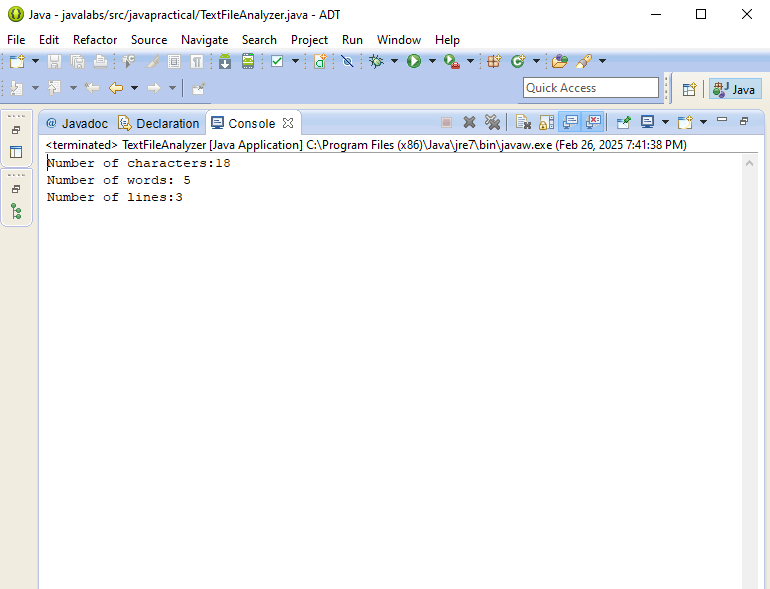
e.printStackTrace();

}

}

}

**OUTPUT:**

****

**4. Random numbers between two given limits using Random class and print messages according to the range of the value generated**

**SOURCE CODE:**

import java.util.Random;

public class RandomNumberGenerator {

public static void main(String[] args) {

int lowerLimit = 10;

int upperLimit = 50;

Random random = new Random();

int randomNumber = random.nextInt(upperLimit - lowerLimit) + lowerLimit;

System.out.println("Generated Random Number: " + randomNumber);

if (randomNumber < 20) {

System.out.println("The number is less than 20.");

} else if (randomNumber >= 20 && randomNumber < 40) {

System.out.println("The number is between 20 and 39.");

} else {

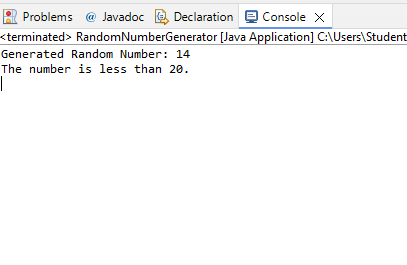
System.out.println("The number is 40 or more.");

}

}

}

**OUTPUT:**



**5.** **To do String Manipulation using Character Array and perform the following string operations: a) String length b) Finding a character at a particular position c)**

**Concatenating two strings**

**SOURCE CODE:**

public class StringManipulation {

public static void main(String[] args) {

String str1 = "ESSM";

String str2 = "College";

char[] charArray1 = str1.toCharArray();

char[] charArray2 = str2.toCharArray();

// a) String Length

System.out.println("Length of the first string: " + charArray1.length);

System.out.println("Length of the second string: " + charArray2.length);

// b) Finding a character at a particular position

int position = 2;

try {

System.out.println("Character at position " + position + " in the first string: " + charArray1[position]);

System.out.println("Character at position " + position + " in the second string: " + charArray2[position]);

} catch (IndexOutOfBoundsException e) {

System.out.println(e.getMessage());

}

// c) Concatenating two strings

StringBuilder sb = new StringBuilder();

sb.append(charArray1);

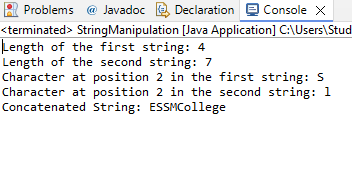
sb.append(charArray2);

System.out.println("Concatenated String: " + sb.toString());

}

}

OUTPUT:



**6. To perform the following string operations using String class: a) String Concatenation b) Search a substring c) To extract**

**substring from given string**

**SOURCE CODE:**

package StringOperations;

public class StringOperations {

public static void main(String[] args) {

// Initialize two strings for concatenation

String str1 = "Vijay";

String str2 = "Shankar E S";

// a) String Concatenation

String concatenatedString = concatenateStrings(str1, str2);

System.out.println("Concatenated String: " + concatenatedString);

// b) Search for a Substring

String searchString = "ay";

boolean isSubstringPresent = searchSubstring(str1, searchString);

System.out.println("Is \"" + searchString + "\" present in \"" + str1 + "\": " + isSubstringPresent);

// c) Extract a Substring

int startIndex = 1;

int endIndex = 4;

String extractedSubstring = extractSubstring(str1, startIndex, endIndex);

System.out.println("Substring from index " + startIndex + " to " + endIndex + " in \"" + str1 + "\": " + extractedSubstring);

}

// Method to concatenate two strings

public static String concatenateStrings(String str1, String str2) {

return str1 + " " + str2;

}

// Method to search for a substring

public static boolean searchSubstring(String str, String subStr) {

return str.contains(subStr);

}

// Method to extract a substring

public static String extractSubstring(String str, int startIndex, int endIndex) {

return str.substring(startIndex, endIndex);

}

}

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

**7. To perform string operations using**

**StringBuffer class: a) Length of a string b) Reverse a string c) Delete a substring from the given string**

**SOURCE CODE:**

package StringOperations;

public class StringBufferOperations {

public static void main(String[] args) {

// Initialize a StringBuffer with an initial string

StringBuffer stringBuffer = new StringBuffer("Vijay Shankar E S");

// a) Length of a string

System.out.println("Length of the string: " + getStringLength(stringBuffer));

// b) Reverse a string

StringBuffer reversedString = reverseString(stringBuffer);

System.out.println("Reversed String: " + reversedString);

// c) Delete a substring from the given string

int startIndex = 5;

int endIndex = 8;

StringBuffer modifiedString = deleteSubstring(new StringBuffer("Vijay Shankar E S"), startIndex, endIndex);

System.out.println("String after deletion: " + modifiedString);

}

// Method to get the length of the StringBuffer

public static int getStringLength(StringBuffer sb) {

return sb.length();

}

// Method to reverse a string

public static StringBuffer reverseString(StringBuffer sb) {

return sb.reverse();

}

// Method to delete a substring

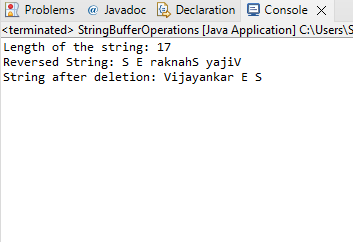
public static StringBuffer deleteSubstring(StringBuffer sb, int startIndex, int endIndex) {

return sb.delete(startIndex, endIndex);

}

}

OUTPUT:



**8.To implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number**

**SOURCE CODE:**

import java.util.Random;

class NumberGenerator extends Thread {

private final SharedResource sharedResource;

public NumberGenerator(SharedResource sharedResource) {

this.sharedResource = sharedResource;

}

@Override

public void run() {

Random random = new Random();

while (true) {

int number = random.nextInt(100); // Generates a random integer between 0 and 99

System.out.println("Generated Number: " + number);

sharedResource.setNumber(number);

try {

Thread.sleep(1000); // Sleep for 1 second

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

class SquareCalculator extends Thread {

private final SharedResource sharedResource;

public SquareCalculator(SharedResource sharedResource) {

this.sharedResource = sharedResource;

}

@Override

public void run() {

while (true) {

synchronized (sharedResource) {

if (sharedResource.isEven()) {

int number = sharedResource.getNumber();

System.out.println("Square of " + number + ": " + (number \* number));

}

}

}

}

}

class CubeCalculator extends Thread {

private final SharedResource sharedResource;

public CubeCalculator(SharedResource sharedResource) {

this.sharedResource = sharedResource;

}

@Override

public void run() {

while (true) {

synchronized (sharedResource) {

if (!sharedResource.isEven()) {

int number = sharedResource.getNumber();

System.out.println("Cube of " + number + ": " + (number \* number \* number));

}

}

}

}

}

class SharedResource {

private int number;

public synchronized void setNumber(int number) {

this.number = number;

}

public synchronized int getNumber() {

return number;

}

public synchronized boolean isEven() {

return number % 2 == 0;

}

}

public class MultiThreadApplication {

public static void main(String[] args) {

SharedResource sharedResource = new SharedResource();

NumberGenerator numberGenerator = new NumberGenerator(sharedResource);

SquareCalculator squareCalculator = new SquareCalculator(sharedResource);

CubeCalculator cubeCalculator = new CubeCalculator(sharedResource);

numberGenerator.start();

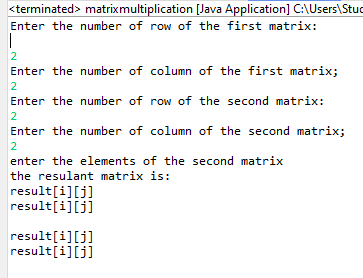
squareCalculator.start();

cubeCalculator.start();

}

}

**OUTPUT:**

****

**9.** **A threading program which uses the same method asynchronously to print the numbers 1 to 10 using Thread1 and to print 90 to 100 using Thread2**

**SOURCE CODE:**

package NumberPrinter;

class NumberPrinter implements Runnable {

private int start, end;

public NumberPrinter(int start, int end) {

this.start = start;

this.end = end;

}

@Override

public void run() {

for (int i = start; i <= end; i++) {

System.*out*.println(Thread.*currentThread*().getName() + ": " + i);

try {

Thread.*sleep*(100); // Simulate async execution

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

public static void main(String[] args) {

Thread thread1 = new Thread(new NumberPrinter(1, 10), "Thread1");

Thread thread2 = new Thread(new NumberPrinter(90, 100), "Thread2");

thread1.start();

thread2.start();

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**10. to demonstrate the use of following exceptions. a) Arithmetic Exception b) Number Format Exception c) Array Index Out of Bound Exception d) Negative Array Size Exception**

**SOURCE CODE:**

package ExceptionDemo;

public class ExceptionDemo {

public static void main(String[] args) {

// a) Arithmetic Exception

try {

int a = 10, b = 0;

int result = a / b; // Division by zero

} catch (ArithmeticException e) {

System.out.println("ArithmeticException: Division by zero is not allowed.");

}

// b) Number Format Exception

try {

String invalidNumber = "abc123";

int num = Integer.parseInt(invalidNumber);

} catch (NumberFormatException e) {

System.out.println("NumberFormatException: Invalid number format.");

}

// c) Array Index Out of Bound Exception

try {

int[] numbers = {1, 2, 3, 4, 5};

int invalidIndexValue = numbers[10];

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println("ArrayIndexOutOfBoundsException: Invalid index accessed.");

}

// d) Negative Array Size Exception

try {

int[] invalidArray = new int[-5];

} catch (NegativeArraySizeException e) {

System.out.println("NegativeArraySizeException: Array size cannot be negative.");

}

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**11.** **To reads on file name from the user, then displays information about whether the file exists, whether the file is readable, whether the file is writable, the type of file and the length of the file in bytes**

**SOURCE CODE:**

import java.io.File;

import java.util.Scanner;

public class FileInfo {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the file name: ");

String fileName = scanner.nextLine();

File file = new File(fileName);

if (file.exists()) {

System.out.println("File exists.");

System.out.println(file.canRead() ? "File is readable." : "File is not readable.");

System.out.println(file.canWrite() ? "File is writable." : "File is not writable.");

System.out.println(file.isDirectory() ? "The file is a directory." : "The file is a regular file.");

System.out.println("File size: " + file.length() + " bytes.");

} else {

System.out.println("The file does not exist.");

}

scanner.close();

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**12.** **To accept a text and change its size and font. Include bold italic options. Use frames and controls**

**SOURCE CODE:**

package FontChanger;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class FontChanger extends JFrame implements ActionListener {

private JTextArea textArea;

private JComboBox<String> fontSizeComboBox, fontFamilyComboBox;

private JCheckBox boldCheckBox, italicCheckBox;

public FontChanger() {

// Set up the main window

setTitle("Font Changer");

setSize(500, 400);

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

setLocationRelativeTo(null);

setLayout(new BorderLayout());

// Create a text area

textArea = new JTextArea("Enter your text here...");

textArea.setFont(new Font("Arial", Font.*PLAIN*, 14));

add(new JScrollPane(textArea), BorderLayout.*CENTER*);

// Create a control panel

JPanel controlPanel = new JPanel();

controlPanel.setLayout(new FlowLayout());

// Font Family Selection

String[] fonts = GraphicsEnvironment.*getLocalGraphicsEnvironment*().getAvailableFontFamilyNames();

fontFamilyComboBox = new JComboBox<>(fonts);

controlPanel.add(new JLabel("Font:"));

controlPanel.add(fontFamilyComboBox);

// Font Size Selection

fontSizeComboBox = new JComboBox<>(new String[]{"12", "14", "16", "18", "20", "24", "28", "32", "36"});

controlPanel.add(new JLabel("Size:"));

controlPanel.add(fontSizeComboBox);

// Bold & Italic Checkboxes

boldCheckBox = new JCheckBox("Bold");

italicCheckBox = new JCheckBox("Italic");

controlPanel.add(boldCheckBox);

controlPanel.add(italicCheckBox);

add(controlPanel, BorderLayout.*NORTH*);

// Add action listeners

fontFamilyComboBox.addActionListener(this);

fontSizeComboBox.addActionListener(this);

boldCheckBox.addActionListener(this);

italicCheckBox.addActionListener(this);

}

@Override

public void actionPerformed(ActionEvent e) {

String fontFamily = (String) fontFamilyComboBox.getSelectedItem();

int fontSize = Integer.*parseInt*((String) fontSizeComboBox.getSelectedItem());

int fontStyle = (boldCheckBox.isSelected() ? Font.*BOLD* : Font.*PLAIN*) |

(italicCheckBox.isSelected() ? Font.*ITALIC* : Font.*PLAIN*);

textArea.setFont(new Font(fontFamily, fontStyle, fontSize));

}

public static void main(String[] args) {

SwingUtilities.*invokeLater*(() -> new FontChanger().setVisible(true));

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**13. To handles all mouse events and shows the event name at the center of the window when a mouse event is fired. (Use adapter classes).**

**SOURCE CODE:**

package MouseEventHandler;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

public class MouseEventHandler extends JFrame {

private String eventName = ""; // To store the name of the current event

public MouseEventHandler() {

// Set up the frame

setTitle("Mouse Event Handler");

setSize(400, 400);

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

setLocationRelativeTo(null);

// Add mouse listener using adapter

addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

eventName = "Mouse Clicked";

repaint();

}

@Override

public void mousePressed(MouseEvent e) {

eventName = "Mouse Pressed";

repaint();

}

@Override

public void mouseReleased(MouseEvent e) {

eventName = "Mouse Released";

repaint();

}

@Override

public void mouseEntered(MouseEvent e) {

eventName = "Mouse Entered";

repaint();

}

@Override

public void mouseExited(MouseEvent e) {

eventName = "Mouse Exited";

repaint();

}

});

addMouseMotionListener(new MouseAdapter() {

@Override

public void mouseDragged(MouseEvent e) {

eventName = "Mouse Dragged";

repaint();

}

@Override

public void mouseMoved(MouseEvent e) {

eventName = "Mouse Moved";

repaint();

}

});

}

@Override

public void paint(Graphics g) {

super.paint(g);

g.setFont(new Font("Arial", Font.*BOLD*, 24));

g.setColor(Color.*BLACK*);

int windowWidth = getWidth();

int windowHeight = getHeight();

FontMetrics fm = g.getFontMetrics();

int textWidth = fm.stringWidth(eventName);

int textHeight = fm.getHeight();

int x = (windowWidth - textWidth) / 2;

int y = (windowHeight + textHeight) / 2;

g.drawString(eventName, x, y);

}

public static void main(String[] args) {

SwingUtilities.*invokeLater*(() -> {

MouseEventHandler frame = new MouseEventHandler();

frame.setVisible(true);

});

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**A screenshot of a computer

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**14. To 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 divide by zero**

**SOURCE CODE:**

package SimpleCalculator;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class SimpleCalculator extends JFrame implements ActionListener {

private JTextField display;

private double num1, num2, result;

private char operator;

public SimpleCalculator() {

setTitle("Simple Calculator");

setSize(400, 500);

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

setLocationRelativeTo(null);

display = new JTextField();

display.setEditable(false);

display.setHorizontalAlignment(SwingConstants.*RIGHT*);

display.setFont(new Font("Arial", Font.*BOLD*, 24));

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(5, 4, 10, 10));

String[] buttons = {

"7", "8", "9", "/",

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

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

"0", ".", "=", "+",

"%", "C"

};

for (String text : buttons) {

JButton button = new JButton(text);

button.setFont(new Font("Arial", Font.*BOLD*, 24));

button.addActionListener(this);

panel.add(button);

}

add(display, BorderLayout.*NORTH*);

add(panel, BorderLayout.*CENTER*);

}

@Override

public void actionPerformed(ActionEvent e) {

String command = e.getActionCommand();

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

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

} else if ("+-\*/%".contains(command)) {

try {

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

operator = command.charAt(0);

display.setText("");

} catch (NumberFormatException ex) {

display.setText("Error");

}

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

try {

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

switch (operator) {

case '+': result = num1 + num2; break;

case '-': result = num1 - num2; break;

case '\*': result = num1 \* num2; break;

case '/':

if (num2 == 0) {

display.setText("Cannot divide by zero");

return;

}

result = num1 / num2; break;

case '%': result = num1 % num2; break;

}

display.setText(String.*valueOf*(result));

} catch (NumberFormatException ex) {

display.setText("Error");

}

} else if (command.equals("C")) {

display.setText("");

}

}

public static void main(String[] args) {

SwingUtilities.*invokeLater*(() -> {

SimpleCalculator calculator = new SimpleCalculator();

calculator.setVisible(true);

});

}

}

**OUTPUT:**

A screenshot of a calculator

AI-generated content may be incorrect.

**15.** **To simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with “stop” or “ready” or “go” should appear above the buttons in a selected color. Initially there is no message shown**

**SOURCE CODE:**

package TrafficLightSimulator;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class TrafficLightSimulator extends JFrame implements ActionListener {

private JRadioButton redButton, yellowButton, greenButton;

private JLabel messageLabel;

public TrafficLightSimulator() {

setTitle("Traffic Light Simulator");

setSize(300, 200);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new FlowLayout());

messageLabel = new JLabel("");

messageLabel.setFont(new Font("Arial", Font.BOLD, 20));

add(messageLabel);

redButton = new JRadioButton("Red");

yellowButton = new JRadioButton("Yellow");

greenButton = new JRadioButton("Green");

redButton.addActionListener(this);

yellowButton.addActionListener(this);

greenButton.addActionListener(this);

ButtonGroup group = new ButtonGroup();

group.add(redButton);

group.add(yellowButton);

group.add(greenButton);

add(redButton);

add(yellowButton);

add(greenButton);

setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e) {

if (redButton.isSelected()) {

messageLabel.setText("Stop");

messageLabel.setForeground(Color.RED);

} else if (yellowButton.isSelected()) {

messageLabel.setText("Ready");

messageLabel.setForeground(Color.ORANGE);

} else if (greenButton.isSelected()) {

messageLabel.setText("Go");

messageLabel.setForeground(Color.GREEN);

}

}

public static void main(String[] args) {

new TrafficLightSimulator();

}

}

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.