**DHAMINI SAHU**

**12108871**

**BODY MASS INDEX (BMI) CALCULATOR:**

**TABLE OF CONTENTS:**

1. **TITLE**
2. **Overview of the Implementation of GUI using JFrame:**
3. **Why BMI calculator?**
4. **Java program to calculate BMI**
5. **Functions and methods Required for the Framework and GUI**
6. **Code**
7. **Expected output**
8. **Conclusion**

**INTRODUCTION**

**Overview of the Implementation of GUI using JFrame:**

The Goal is to develop a Graphical User Interface (GUI) using a Java Swing and JFrame implementation of a Body Mass Index (BMI) calculator application.

The code initially creates a JFrame object and defines its dimensions, background color, and layout. The user input fields and labels for entering their weight and height are then added as several JLabels and JTextFields to the frame.

The "Calculate" button, an instance of the JButton class with an ActionListener connected to it, is also added to the frame. When the button is pressed, the ActionListener analyzes the user-entered weight and height values before using the BMI calculation to determine the BMI value.

Another JLabel is then used to display the BMI value on the frame. In addition, the code adds a "Clear" button to the frame that, when pressed, clears the input fields and the result label.

In summary, the code uses JFrame to develop a simple BMI calculator application that accepts user input, conducts calculations, and shows the results on the screen.

**Why BMI calculator?**

The Body Mass Index (BMI) calculator is a valuable tool for keeping track of one's health and fitness. Body mass index (BMI) is a measurement of body fat based on height and weight. The higher the BMI, the greater the likelihood of having health issues such as heart disease, diabetes, and high blood pressure. As a result, it is critical to maintain a healthy BMI to limit the risk of various health conditions.

The importance of having a BMI calculator tool cannot be emphasized. First and foremost, it assists people in keeping track of their weight and staying healthy. Individuals can identify whether they are underweight, normal weight, overweight, or obese by assessing their BMI. If their BMI is over the acceptable limit, individuals can lose weight by healthy food and exercise.

Another practical benefit of the BMI calculator is that it is easy to use. The user simply needs to input their height and weight, and the calculator does the rest. This simplicity means that anyone can use the calculator, regardless of their level of technical knowledge.

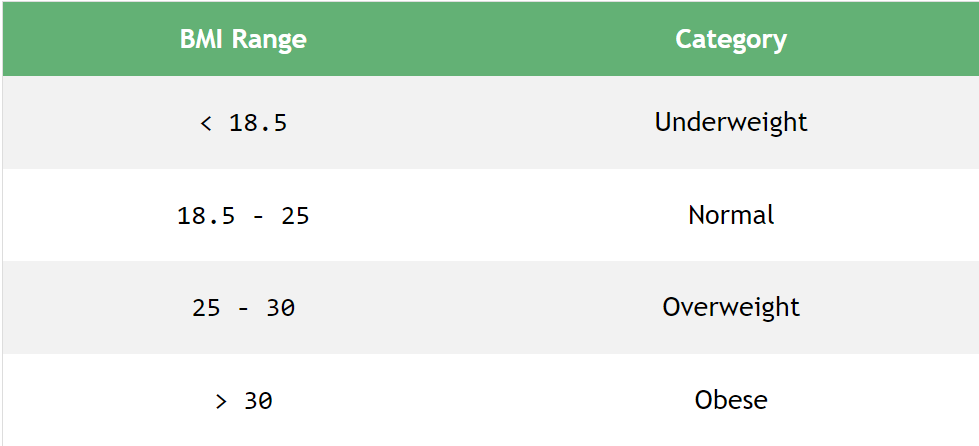
BMI stands for Body Mass Index. It's a value derived from an individual's height and weight.

With the help of BMI, we can find out whether an individual's weight is healthy or not.

Let's have a look at the formula for calculating A person's categorized as Underweight, Normal, Overweight, or Obese based on the BMI range:

BMI:

**BMI = Weight in Kilograms / (Height in Meters \* Height in Meters)**



For example, let's calculate the BMI of an individual with a weight equal to 100kg (Kilograms) and a height equal to 1.524m (Meters).

BMI = 100 / (1.524 \* 1.524)

BMI =  43.056

**Since BMI is greater than 30, the person is categorized as “Overweight”.**

**Functions and methods Required for the Framework and GUI**:

1. public class BMICalculator extends JFrame: This establishes a BMICalculator class that extends JFrame, a built-in Java class that offers a graphical user interface (GUI) framework for generating windows, menus, buttons, and other UI components.

2. private JPanel contentPane: This defines a private variable of type JPane named contentPane, which is a container for other UI components.

3. private JTextField textFieldHeight: This defines a private variable of type JTextField named textFieldHeight, which is a text field where users may enter their height in centimeters.

4. private JTextField textFieldWeight: This defines a private variable of type JTextField named textFieldWeight, which is a text field where users may enter their weight in kilos.

5. private JLabel lblResult: This defines a private variable of type JLabel named lblResult, which is a label that shows the BMI calculation result.

6. public BMICalculator(): This is a constructor function that initializes the BMI calculator UI by specifying the frame title, size, layout, and other UI components such as labels, text fields, and buttons.

7. private void calculateBMI(): This is a helper function that calculates the BMI based on the user's height and weight input and shows the result in the lblResult label.

8. private void clearFields(): This is a helper function that clears the input fields and result label.

10. private class ClearButtonHandler implements ActionListener: This is an inner class that implements the ActionListener interface and handles the clear button's click event by calling the clearFields() function.

11. CalculateButtonHandler is a private class that implements ActionListener: This is an inner class that implements the ActionListener interface and handles the calculate button's click event by calling the calculateBMI() function.

In summary, the code uses JFrame to develop a simple BMI calculator application that accepts user input, conducts calculations, and shows the results on the screen.

**CODE:**

// import necessary packages

import java.awt.Color;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.\*;

public class BMIcalculator {

public static void main(String[] args) {

// create JFrame and set its properties

JFrame frame = new JFrame("BMI Calculator");

frame.setSize(525, 525); // set size of the frame

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); // set default close operation

frame.setLayout(null); // set layout to null for absolute positioning

frame.getContentPane().setBackground(Color.BLACK); // set background color of the content pane

    // create and add heading label to the frame

    JLabel heading = new JLabel("BMI Calculator");

    heading.setBounds(125, 15, 300, 45); // set size and position of the label

    heading.setFont(new Font("Times New Roman", Font.BOLD, 24)); // set font of the label

    heading.setForeground(Color.WHITE); // set foreground color of the label

    frame.add(heading);

    // create and add height label and text field to the frame

    JLabel heightLabel = new JLabel("Height (in cm):");

    heightLabel.setBounds(60, 75, 150, 45); // set size and position of the label

    heightLabel.setForeground(Color.WHITE); // set foreground color of the label

    frame.add(heightLabel);

    JTextField heightTextField = new JTextField();

    heightTextField.setBounds(225, 75, 225, 45); // set size and position of the text field

    frame.add(heightTextField);

    // create and add weight label and text field to the frame

    JLabel weightLabel = new JLabel("Weight (in kg):");

    weightLabel.setBounds(60, 135, 150, 45); // set size and position of the label

    weightLabel.setForeground(Color.WHITE); // set foreground color of the label

    frame.add(weightLabel);

    JTextField weightTextField = new JTextField();

    weightTextField.setBounds(225, 135, 225, 45); // set size and position of the text field

    frame.add(weightTextField);

    // create and add result label to the frame

    JLabel resultLabel = new JLabel();

    resultLabel.setBounds(60, 300, 300, 45); // set size and position of the label

    resultLabel.setForeground(Color.WHITE); // set foreground color of the label

    resultLabel.setFont(new Font("Times New Roman", Font.BOLD, 21)); // set font of the label

    frame.add(resultLabel);

    // create and add calculate button to the frame

    JButton calculateButton = new JButton("Calculate");

    calculateButton.setBounds(225, 210, 150, 45); // set size and position of the button

    calculateButton.addActionListener(new ActionListener() {

        public void actionPerformed(ActionEvent e) {

            try {

                // get height and weight values from text fields and calculate BMI

                double height = Double.parseDouble(heightTextField.getText());

                double weight = Double.parseDouble(weightTextField.getText());

                double bmi = weight / ((height / 100) \* (height / 100));

                // format and display the result in the result label

                String result = String.format("Your Body Mass Index: %.2f", bmi);

                resultLabel.setText(result);

                resultLabel.setForeground(Color.WHITE); // set foreground color of the label

            } catch (NumberFormatException ex) {

                // display error message if invalid values are entered

                resultLabel.setText("Please enter valid values");

            }

        }

    });

    frame.add(calculateButton);

    // create and add clear button to the frame

    JButton clearButton = new JButton("Clear");

    clearButton.setBounds(60, 210, 150, 45);

    clearButton.addActionListener(new ActionListener() {

        public void actionPerformed(ActionEvent e) {

            heightTextField.setText("");

            weightTextField.setText("");

            resultLabel.setText("");

        }

    });

    frame.add(clearButton);

    frame.setVisible(true);

}

}

**EXPECTED OUTPUT:**

**Graphical user interface, application

Description automatically generated**

## Conclusion

In this article, we learned to create a BMI Calculator in Java. We also tested the implementation by writing a JUnit test.

**THANKYOU**

**DR. RANJITH KUMAR**