

****ASSINGMENT-4****

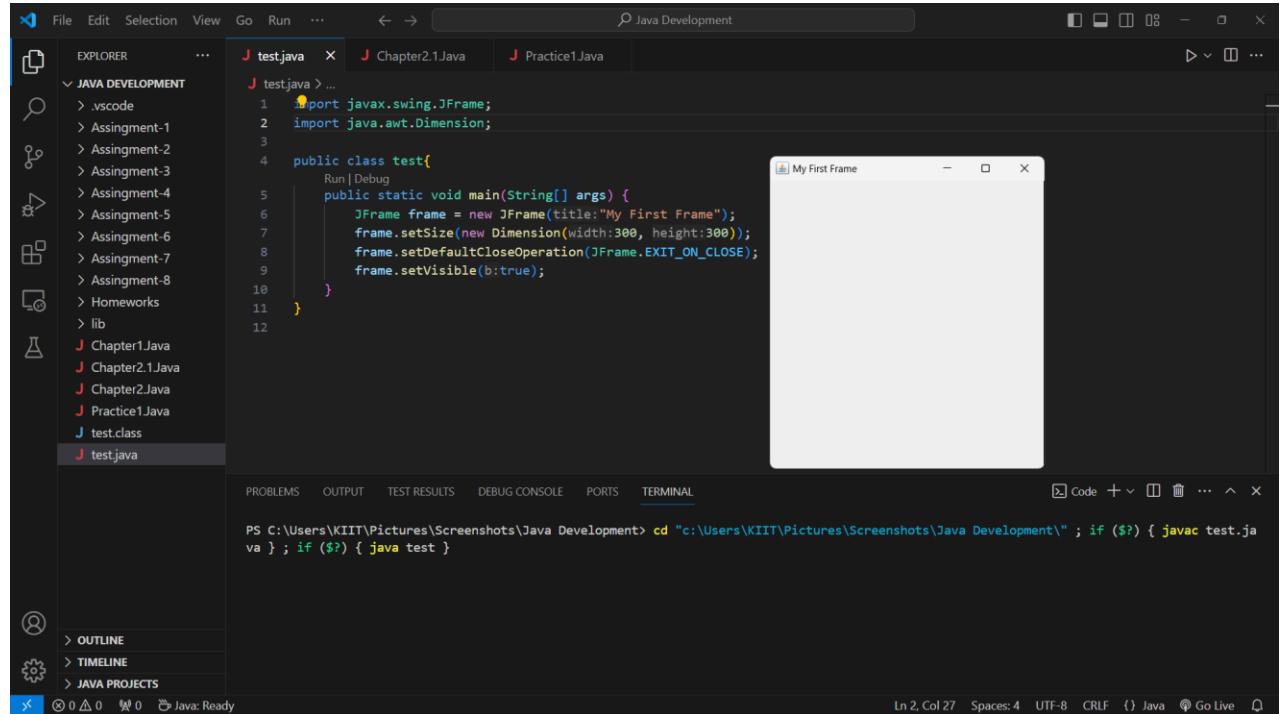
SUBMITTED BY: -

KANISHK

2205130

CSE-37

Q1- INPUT/OUTPUT



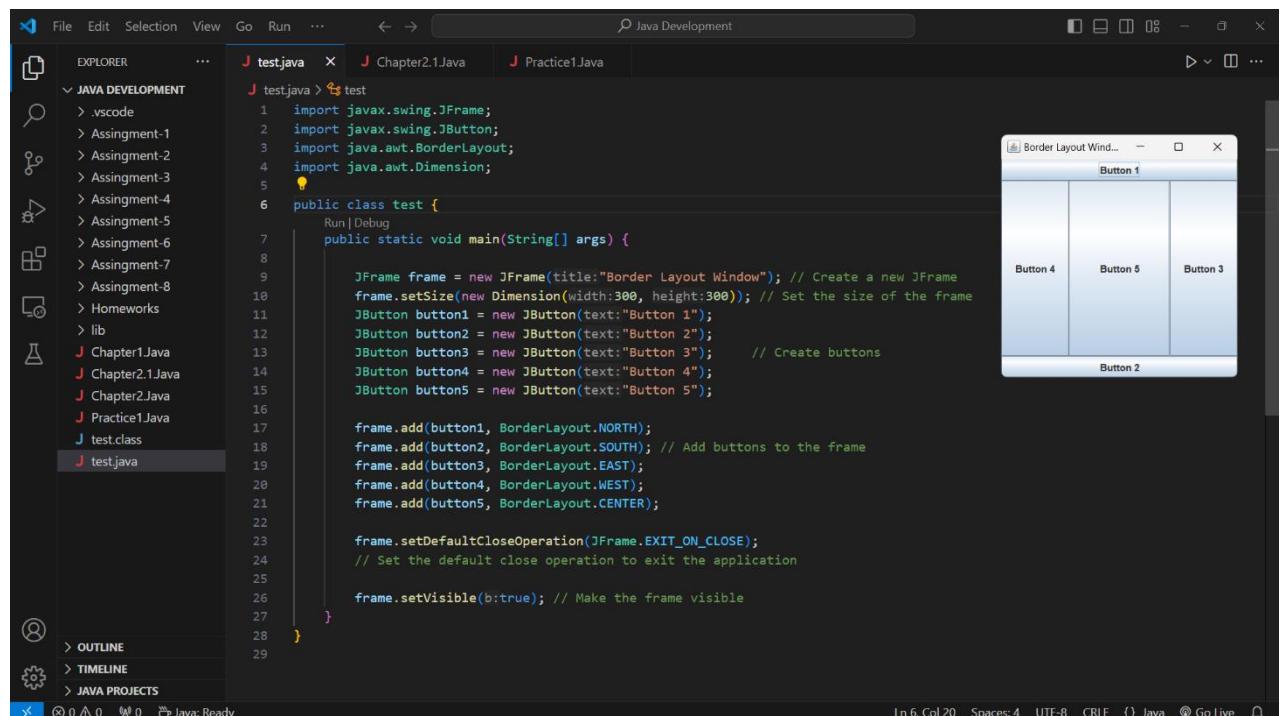
A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a file tree under 'JAVA DEVELOPMENT' containing '.vscode', 'Asssignment-1', 'Asssignment-2', 'Asssignment-3', 'Asssignment-4', 'Asssignment-5', 'Asssignment-6', 'Asssignment-7', 'Asssignment-8', 'Homeworks', 'lib', 'Chapter1Java', 'Chapter2.1Java', 'Chapter2Java', 'Practice1Java', 'test.class', and 'test.java'. The 'test.java' tab is active, displaying the following Java code:

```
import javax.swing.JFrame;
import java.awt.Dimension;

public class test{
    public static void main(String[] args) {
        JFrame frame = new JFrame("My First Frame");
        frame.setSize(new Dimension(width:300, height:300));
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(b:true);
    }
}
```

The right side of the interface shows a preview window titled 'My First Frame' which displays a blank white frame. Below the editor is a terminal window showing the command line output of running the Java code.

Q2: - INPUT/OUTPUT



A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a file tree under 'JAVA DEVELOPMENT' containing '.vscode', 'Asssignment-1', 'Asssignment-2', 'Asssignment-3', 'Asssignment-4', 'Asssignment-5', 'Asssignment-6', 'Asssignment-7', 'Asssignment-8', 'Homeworks', 'lib', 'Chapter1Java', 'Chapter2.1Java', 'Chapter2Java', 'Practice1Java', 'test.class', and 'test.java'. The 'test.java' tab is active, displaying the following Java code:

```
import javax.swing.JFrame;
import javax.swing.JButton;
import java.awt.BorderLayout;
import java.awt.Dimension;

public class test {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Border Layout Wind..."); // Create a new JFrame
        frame.setSize(new Dimension(width:300, height:300)); // Set the size of the frame
        JButton button1 = new JButton(text:"Button 1");
        JButton button2 = new JButton(text:"Button 2");
        JButton button3 = new JButton(text:"Button 3"); // Create buttons
        JButton button4 = new JButton(text:"Button 4");
        JButton button5 = new JButton(text:"Button 5");

        frame.add(button1, BorderLayout.NORTH);
        frame.add(button2, BorderLayout.SOUTH); // Add buttons to the frame
        frame.add(button3, BorderLayout.EAST);
        frame.add(button4, BorderLayout.WEST);
        frame.add(button5, BorderLayout.CENTER);

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        // Set the default close operation to exit the application

        frame.setVisible(b:true); // Make the frame visible
    }
}
```

The right side of the interface shows a preview window titled 'Border Layout Wind...' which displays a window with five buttons arranged in a BorderLayout: 'Button 1' at the top center, 'Button 2' at the bottom center, 'Button 3' on the right, 'Button 4' on the left, and 'Button 5' at the bottom-left corner. Below the editor is a terminal window showing the command line output of running the Java code.

Q3: - INPUT/OUTPUT

The screenshot shows the VS Code interface with the following details:

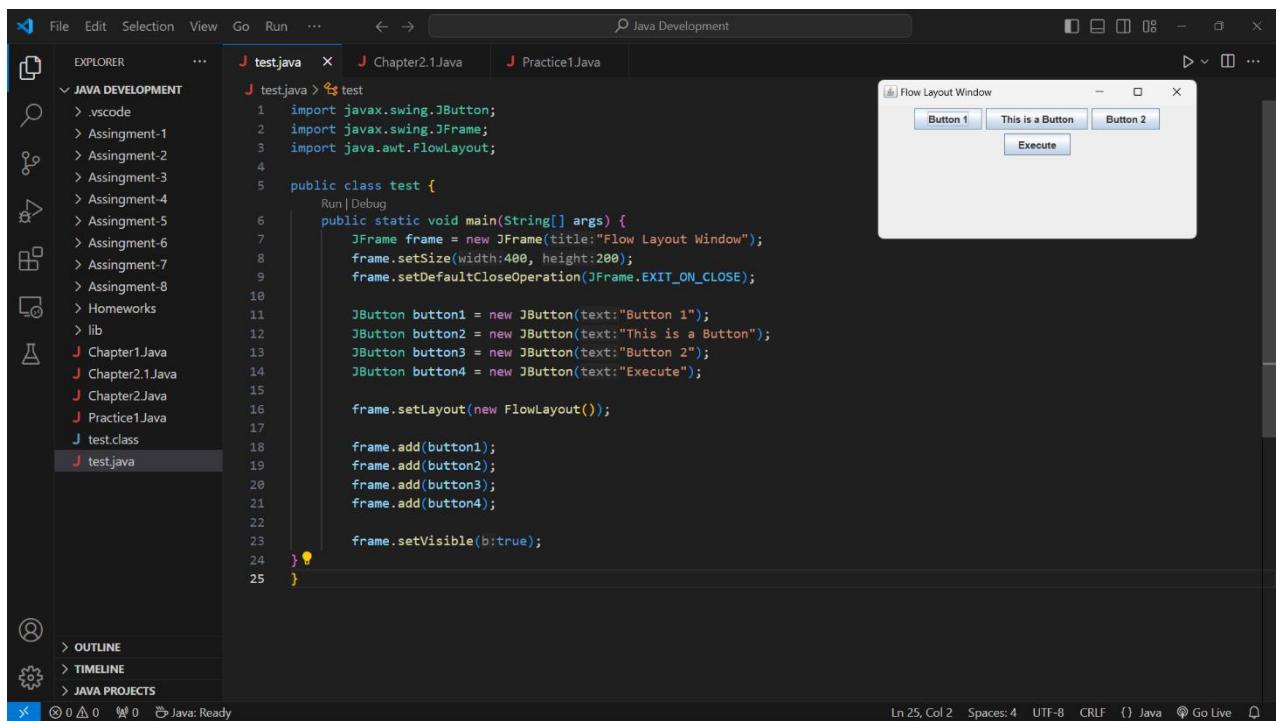
- File Explorer:** Shows a project structure under "JAVA DEVELOPMENT" with files like ".vscode", "Assingment-1" through "Assingment-8", "Homeworks", and several Java files: "Chapter1Java", "Chapter2.1Java", "Chapter2.1Java", "Practice1Java", "test.class", and "test.java".
- Code Editor:** The active file is "test.java", which contains Java code for creating a JFrame with a BorderLayout. It adds two JLabels to the NORTH and SOUTH positions.
- Output Window:** A separate window titled "Border Layout Window" displays the resulting application. It has a title bar "Border Layout Window" and two labels: "North Label" in the top-left position and "South Label" in the bottom position.
- Status Bar:** Shows "Ln 16, Col 1" and other standard status information.

Q4: - INPUT/OUTPUT

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a project structure under "JAVA DEVELOPMENT" with files like ".vscode", "Assingment-1" through "Assingment-8", "Homeworks", and several Java files: "Chapter1Java", "Chapter2.1Java", "Chapter2.1Java", "Practice1Java", "test.class", and "test.java".
- Code Editor:** The active file is "test.java", which contains Java code for creating a JFrame with a BorderLayout. It adds two JLabels to the NORTH and SOUTH positions and sets their fonts to bold serif at size 30.
- Output Window:** A separate window titled "Border Layout Window" displays the resulting application. It has a title bar "Border Layout Window" and two labels: "North Label" in the top-left position and "South Label" in the bottom position, both displayed in a bold serif font.
- Status Bar:** Shows "Ln 23, Col 32" and other standard status information.

Q5: - INPUT/OUTPUT



A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a file tree under 'EXPLORER' with several Java files and folders. The main editor window displays a Java class named 'test'. The code creates a JFrame titled 'Flow Layout Window' with a width of 400 and height of 200. It sets the default close operation to EXIT_ON_CLOSE. Inside the frame, four JButton objects are added: 'Button 1', 'This is a Button', 'Button 2', and 'Execute'. The frame's layout is set to new FlowLayout(). The code ends with frame.setVisible(true). On the right side of the interface, there is a preview window titled 'Flow Layout Window' showing the resulting window with the four buttons arranged horizontally.

```
File Edit Selection View Go Run ... <- > Java Development
EXPLORER J testjava J Chapter2.1.Java J Practice1Java
JAVA DEVELOPMENT J testjava > test
> .vscode
> Assingment-1
> Assingment-2
> Assingment-3
> Assingment-4
> Assingment-5
> Assingment-6
> Assingment-7
> Assingment-8
> Homeworks
> lib
J Chapter1Java
J Chapter2.1.Java
J Chapter2Java
J Practice1Java
J test.class
J test.java
Run | Debug
public class test {
    public static void main(String[] args) {
        JFrame frame = new JFrame(title:"Flow Layout Window");
        frame.setSize(width:400, height:200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

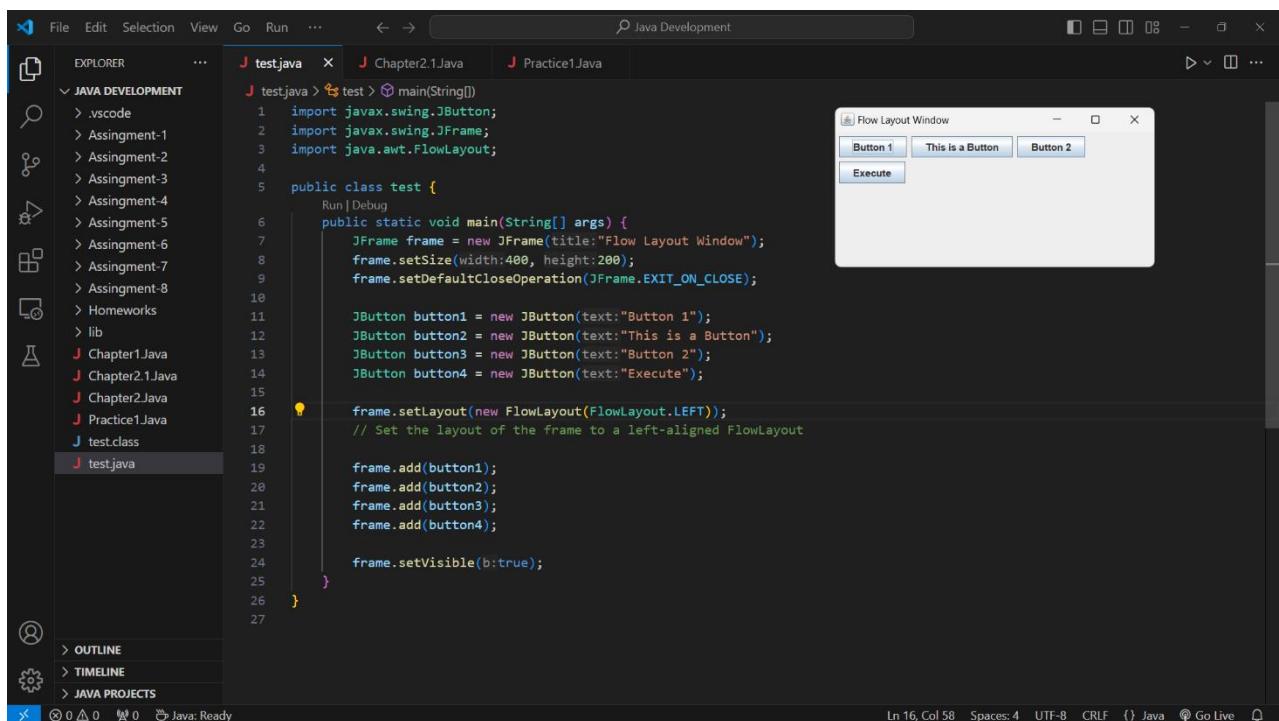
        JButton button1 = new JButton(text:"Button 1");
        JButton button2 = new JButton(text:"This is a Button");
        JButton button3 = new JButton(text:"Button 2");
        JButton button4 = new JButton(text:"Execute");

        frame.setLayout(new FlowLayout());

        frame.add(button1);
        frame.add(button2);
        frame.add(button3);
        frame.add(button4);

        frame.setVisible(b:true);
    }
}
Ln 25, Col 2 Spaces: 4 UTF-8 CRLF {} Java Go Live
```

Q6: - INPUT/OUTPUT



A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows a file tree under 'EXPLORER' with several Java files and folders. The main editor window displays a Java class named 'test'. The code creates a JFrame titled 'Flow Layout Window' with a width of 400 and height of 200. It sets the default close operation to EXIT_ON_CLOSE. Inside the frame, four JButton objects are added: 'Button 1', 'This is a Button', 'Button 2', and 'Execute'. The code changes the frame's layout to new FlowLayout(FlowLayout.LEFT);. The code ends with frame.setVisible(true). On the right side of the interface, there is a preview window titled 'Flow Layout Window' showing the resulting window with the four buttons arranged horizontally, aligned to the left.

```
File Edit Selection View Go Run ... <- > Java Development
EXPLORER J testjava J Chapter2.1.Java J Practice1Java
JAVA DEVELOPMENT J testjava > test > main(String[])
> .vscode
> Assingment-1
> Assingment-2
> Assingment-3
> Assingment-4
> Assingment-5
> Assingment-6
> Assingment-7
> Assingment-8
> Homeworks
> lib
J Chapter1Java
J Chapter2.1.Java
J Chapter2Java
J Practice1Java
J test.class
J test.java
Run | Debug
public static void main(String[] args) {
    JFrame frame = new JFrame(title:"Flow Layout Window");
    frame.setSize(width:400, height:200);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton button1 = new JButton(text:"Button 1");
    JButton button2 = new JButton(text:"This is a Button");
    JButton button3 = new JButton(text:"Button 2");
    JButton button4 = new JButton(text:"Execute");

    frame.setLayout(new FlowLayout(FlowLayout.LEFT));
    // Set the layout of the frame to a left-aligned FlowLayout

    frame.add(button1);
    frame.add(button2);
    frame.add(button3);
    frame.add(button4);

    frame.setVisible(b:true);
}
}
Ln 16, Col 58 Spaces: 4 UTF-8 CRLF {} Java Go Live
```

Q7: - INPUT/OUTPUT

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a Java Development project structure with files like .vscode, Assingment-1 through -8, Homeworks, Chapter1Java, Chapter2Java, Practice1Java, test.class, and test.java.
- Code Editor:** The active file is `test.java`, containing Java code to create a `JFrame` with a `FlowLayout`. The code includes imports for `java.awt.FlowLayout`, `java.awt.JButton`, and `java.awt.JFrame`. It creates four buttons labeled "Button 1", "This is a Button", "Button 2", and "Execute".
- Output:** A preview window titled "Flow Layout Window" shows the resulting frame with the four buttons arranged horizontally.
- Status Bar:** Shows "Ln 27, Col 1" and other standard status bar information.

Q8: - INPUT/OUTPUT

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a Java Development project structure with files like .vscode, Assingment-1 through -8, Homeworks, Chapter1Java, Chapter2Java, Practice1Java, test.class, and test.java.
- Code Editor:** The active file is `test.java`, containing Java code to create a `JFrame` with a `GridLayout`. The code includes imports for `java.awt.GridLayout`, `java.awt.JButton`, and `java.awt.JFrame`. It creates four buttons labeled "Button 1", "Button 2", "Button 3", and "Button 4" arranged in a 2x2 grid.
- Output:** A preview window titled "Grid Layout Window" shows the resulting frame with the four buttons arranged in a 2x2 grid.
- Status Bar:** Shows "Ln 10, Col 1" and other standard status bar information.

Q9: - INPUT: -

File Edit Selection View Go Run ... ⏪ ⏹ Java Development

EXPLORER JAVA DEVELOPMENT

- > vscode
- > Assingment-1
- > Assingment-2
- > Assingment-3
- > Assingment-4
- > Assingment-5
- > Assingment-6
- > Assingment-7
- > Assingment-8

> GUI

- > Homeworks
- > lib
- J CalculatorGUI.class
- J Chapter1.java
- J Chapter2.java
- J Practice1.java
- J test.class
- J testjava 1

> OUTLINE > TIMELINE

29°C Haze

```
1 package GUI;
2 import javax.swing.*;
3 import java.awt.*;
4 import java.awt.event.ActionEvent;
5 import java.awt.event.ActionListener;
6 class CalculatorGUI extends JFrame implements ActionListener {
7     private JTextField displayField;
8     private double currentValue;
9     private char currentOperator;
10    public CalculatorGUI() {
11        super(title:"Simple Calculator");
12        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
13        setSize(width:300, height:300);
14        displayField = new JTextField();
15        displayField.setFont(new Font(name:"Arial", Font.PLAIN, size:20));
16        displayField.setHorizontalAlignment(JTextField.RIGHT);
17        add(displayField, BorderLayout.NORTH);
18        JPanel buttonPanel = new JPanel(new GridLayout(rows:4, cols:4));
19        String[] buttonLabels = {"7", "8", "9", "+",
20        "4", "5", "6", "-",
21        "1", "2", "3", "=",
22        "0", ".", "="};
23        for (String label : buttonLabels) {
24            JButton button = new JButton(label);
25            button.addActionListener(this);
26            buttonPanel.add(button);
27        }
28        add(buttonPanel, BorderLayout.CENTER);
29        setVisible(b:true);
30    }
31    @Override
```

21.03 ENG US 01-04-2024

File Edit Selection View Go Run ... ⏪ ⏹ Java Development

EXPLORER JAVA DEVELOPMENT

- > vscode
- > Assingment-1
- > Assingment-2
- > Assingment-3
- > Assingment-4
- > Assingment-5
- > Assingment-6
- > Assingment-7
- > Assingment-8

> GUI

- > Homeworks
- > lib
- J CalculatorGUI.class
- J Chapter1.java
- J Chapter2.java
- J Practice1.java
- J test.class
- J testjava 1

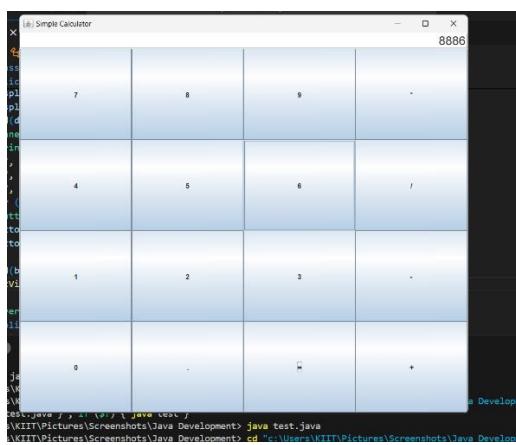
> OUTLINE > TIMELINE

29°C Haze

```
1 package GUI;
2 import javax.swing.*;
3 import java.awt.*;
4 import java.awt.event.ActionEvent;
5 import java.awt.event.ActionListener;
6 class CalculatorGUI extends JFrame implements ActionListener {
7     public CalculatorGUI() {
8         "1", "2", "3", "-",
9         "0", ".", "=",
10        for (String label : buttonLabels) {
11            JButton button = new JButton(label);
12            button.addActionListener(this);
13            buttonPanel.add(button);
14        }
15        add(buttonPanel, BorderLayout.CENTER);
16        setVisible(b:true);
17    }
18    @Override
19    public void actionPerformed(ActionEvent e) {
20        String command = e.getActionCommand();
21        if (Character.isDigit(command.charAt(index:0))) {
22            displayField.setText(displayField.getText() + command);
23        } else if (command.equals(anObject:"."))
24            displayField.setText(displayField.getText() + command);
25        } else if (command.equals(anObject)="")
26            displayField.setText(displayField.getText() + command);
27        }
28        Run | Debug
29        public static void main(String[] args) {
30            SwingUtilities.invokeLater(() -> new CalculatorGUI());
31        }
32    }
```

21.03 ENG US 01-04-2024

OUTPUT: -

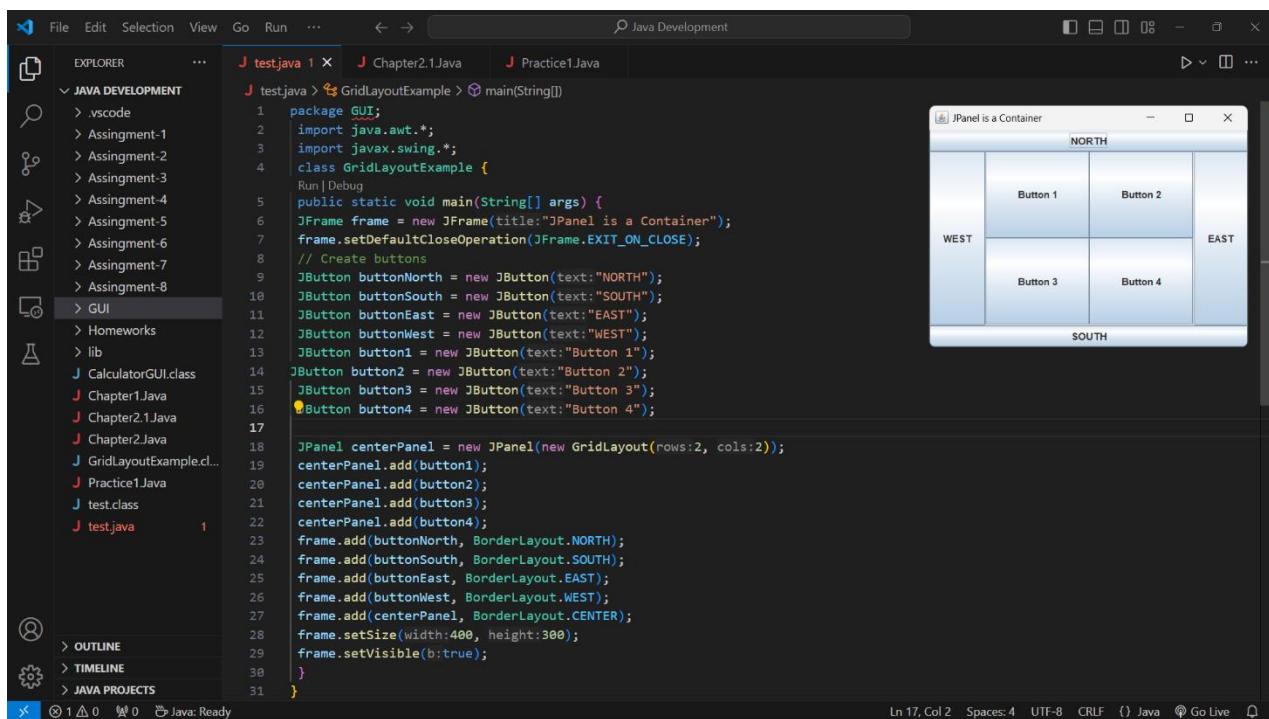


Q10: - The given program creates a JFrame with a JPanel inside it. The JPanel uses the default layout, which is Flow Layout. Flow Layout arranges components in a horizontal row, with each component's width set to its preferred width. If the horizontal space in the container is too small to fit all the components in one row, Flow Layout will wrap the extra components to the next row.

In this case, the JPanel contains four JButton components. Each JButton has a preferred width that is determined by the text and the font size of the button. The JPanel's width is set to 200, which is not enough to fit all the buttons in one row. Therefore, the Flow Layout will wrap the extra buttons to the next row.

The output of the program will be a window with a JPanel that contains four JButton components. The buttons will be arranged in two rows, with two buttons in the first row and two buttons in the second row. The window will be resizable, and the user can close it.

Q11: - INPUT/OUTPUT



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows Java Development projects and files like test.java, Chapter2.1Java, Practice1Java, and various assignment files.
- Code Editor:** Displays the Java code for `GridLayoutExample`. The code creates a JFrame with a center panel using a GridLayout(2, 2). It adds four buttons to the center panel and four more buttons to the frame's border using BorderLayout.
- Preview:** A window titled "JPanel is a Container" shows a 2x2 grid of buttons labeled Button 1 through Button 4. The grid is centered in the frame, and the frame has borders labeled NORTH, SOUTH, EAST, and WEST.
- Status Bar:** Shows file statistics (Ln 17, Col 2) and encoding (UTF-8).

Q12: - In the given JPanel example, the JFrame contains a JPanel named "buttonPanel". The JPanel uses the default layout, which is FlowLayout. FlowLayout arranges components in a horizontal row, with each component's width set to its preferred width.

The "buttonPanel" contains three JButton components. Each JButton has a preferred width that is determined by the text and the font size of the button. The JPanel's width is

determined by the width of the JFrame, which is 300 pixels.

The "buttonPanel" is added to the JFrame using the method add ("Center", buttonPanel). This method adds the "buttonPanel" to the JFrame's content pane with the constraint "Center". The constraint "Center" tells the layout manager to place the "buttonPanel" in the center of the JFrame.

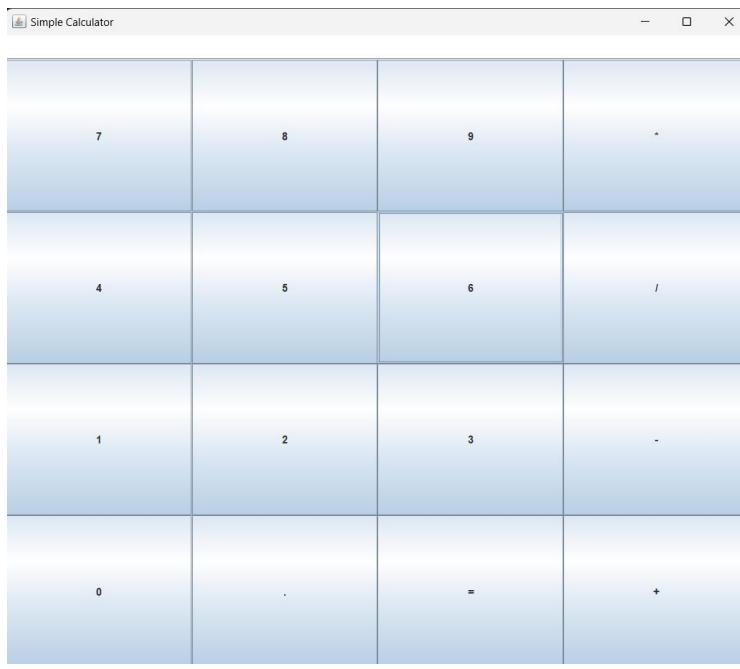
The JFrame's layout manager is a BorderLayout by default. BorderLayout places

Components in five regions: North, South, East, West, and Center. The "buttonPanel" is Placed in the Center region of the JFrame.

The output of the program will be a window with a JPanel that contains three JButton components. The buttons will be arranged in a single row, with each button's width set to its preferred width. The window will be resizable, and the user can close it.

Q13: - INPUT: -

OUTPUT: -



Q14: - INPUT/OUTPUT

The screenshot shows the Visual Studio Code (VS Code) interface with the following details:

- File Explorer:** Shows a project structure under "JAVA DEVELOPMENT" with files like ".vscode", "Assingment-1", "Assingment-2", "Assingment-3", "Assingment-4", "Assingment-5", "Assingment-6", "Assingment-7", "Assingment-8", "GUI", "Homeworks", "lib", "CalculatorGUI.class", "CalculatorUI.class", "Chapter1.java", "Chapter2.1.java", "Chapter2.2.java", "GridLayoutExample.cl...", "Practice1.java", "test.class", and "test.java".
- Code Editor:** Displays the "CalculatorUI.java" file with the following code:

```
1 package GUI;
2
3 import javax.swing.*;
4 import java.awt.*;
5
6 class CalculatorUI {
7     Run | Debug
8     public static void main(String[] args) {
9
10         JFrame frame = new JFrame(title:"Calculator UI");
11         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12         frame.setSize(width:300, height:200);
13
14         JTextField textField = new JTextField();
15         textField.setHorizontalAlignment(JTextField.RIGHT);
16
17         textField.setFont(new Font(name:"Arial", Font.BOLD, size:20));
18
19         frame.add(textField, BorderLayout.NORTH);
20
21         frame.setVisible(b:true);
22     }
23 }
```
- Terminal:** Shows the output "Niorth...." from the terminal.
- Status Bar:** Shows "Ln 17, Col 1" and "Java: Ready".

Q15: - The layout of the given GUI can be explained as follows:

1. The JFrame's layout manager is set to a FlowLayout. FlowLayout arranges components in a horizontal row, with each component's width set to its preferred width.
2. The JFrame contains a JTextField (num1textField), a JLabel (operatorLabel),

another JTextField (num2textField), a JButton (equalsButton), and a JLabel (answerLabel).

3. The JTextField (num1textField) is used to input the first number. It has a preferred width of 5 characters.
4. The JLabel (operatorLabel) displays the operator (+) between the two numbers.
5. The JTextField (num2textField) is used to input the second number. It also has a preferred width of 5 characters.
6. The JButton (equalsButton) is used to calculate the sum of the two numbers.
7. The JLabel (answerLabel) displays the result of the calculation.
8. The JFrame's size is set to 300x300 pixels using the setSize() method.
9. The JFrame's title is set to "Calculator" using the setTitle() method.
10. The JFrame is made resizable and closable using the setDefaultCloseOperation() method with the argument EXIT_ON_CLOSE.
11. The JFrame is centered on the screen using the setLocationRelativeTo() method with the argument null.
12. The JFrame is made visible using the setVisible() method with the argument true.
13. The pack() method is called to adjust the size of the JFrame to fit the preferred sizes of its components.

Q16: - The output of the program will be a window with a JTextField for inputting the first number, a JLabel for displaying the operator, another JTextField for inputting the second number, a JButton for calculating the sum, and a JLabel for displaying the result. The window will be resizable and closable.

Q17: - The output of the program will be a window with a JLabel displaying the text "This is a JLabel" and an image of a Facebook logo. The JLabel is horizontally centered and uses the font "Batang" with a size of 20. The window will be resizable and closable. If the file "facebook.png" is not in the working directory, the program will throw an exception and the window will not display the image.

Q19: - The output of the program will be a window with a JComboBox displaying a dropdown list of colors: RED, GREEN, BLACK, and BLUE. The user can select a color from the dropdown list. The selected color will be displayed in the JComboBox.

Q20: - INPUT: -

The screenshot shows a Java Development environment with the following details:

- File Explorer:** Shows a project structure under "JAVA DEVELOPMENT" with files like .vscode, Assingment-1, Assingment-2, Assingment-3, Assingment-4, Assingment-5, Assingment-6, Assingment-7, Assingment-8, GUI, Homeworks, lib, CalculatorGUI.class, CalculatorUI.class, Chapter1.java, Chapter2.java, GridLayoutExample.cl..., Practice1.java, test.class, and test.java.
- Code Editor:** The active file is "test.java" (line 1). The code defines a class "StudentDetailForm" with a main method creating a JFrame window and adding various JLabel and JTextField components.
- Status Bar:** Shows "In 1, Col 13" and "Java Ready".

The screenshot shows a Java Development environment with the following details:

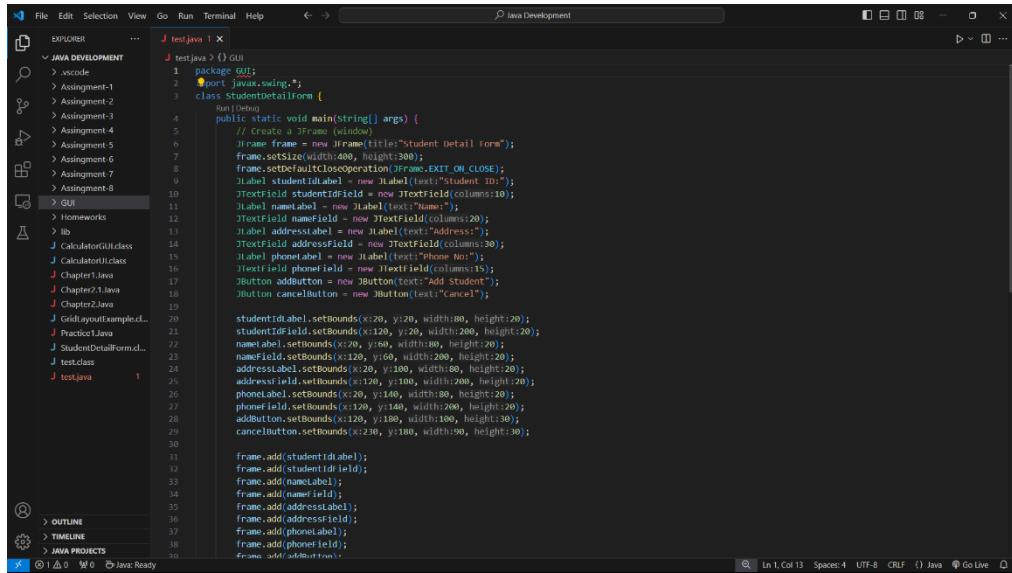
- File Explorer:** Shows a project structure under "JAVA DEVELOPMENT" with files like .vscode, Assingment-1, Assingment-2, Assingment-3, Assingment-4, Assingment-5, Assingment-6, Assingment-7, Assingment-8, GUI, Homeworks, lib, CalculatorGUI.class, CalculatorUI.class, Chapter1.java, Chapter2.java, GridLayoutExample.cl..., Practice1.java, test.class, and test.java.
- Code Editor:** The active file is "test.java" (line 1). The code defines a class "StudentDetailForm" with a main method creating a JFrame window and adding various JLabel and JTextField components.
- Status Bar:** Shows "In 1, Col 13" and "Java Ready".

OUTPUT: -

 Student Detail Form

Student ID:	<input type="text"/>
Name:	<input type="text"/>
Address:	<input type="text"/>
Phone No:	<input type="text"/>

Q21: - INPUT

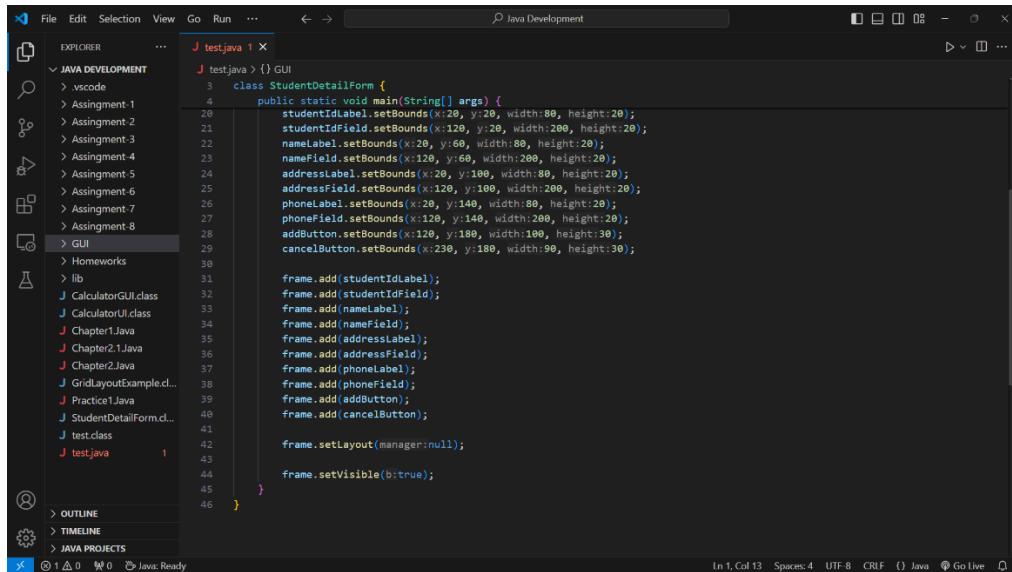


```

File Edit Selection View Go Run Terminal Help ⌘ Java Development
EXPLORER ... J test.java 1 ×
JAVA DEVELOPMENT > .vscode > Assignment-1 > Assignment-2 > Assignment-3 > Assignment-4 > Assignment-5 > Assignment-6 > Assignment-7 > Assignment-8 > GUI > Homeworks > lib J CalculatorGUI.class J CalculatorUI.class J Chapter1.java J Chapter2.1.java J Chapter2.java J GridLayoutExample.cl... J Practice.java J StudentDetailForm.cl... J test.class J test.java 1
J test.java > {} GUI
1 package GUI;
2 import javax.swing.*;
3 class StudentDetailForm {
4     public static void main(String[] args) {
5         // Create frame
6         JFrame frame = new JFrame("Student Detail Form");
7         frame.setSize(width:600, height:300);
8         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9         JLabel studentIdLabel = new JLabel(text:"Student ID:");
10        JTextField studentIdField = new JTextField(columns:10);
11        JLabel nameLabel = new JLabel(text:"Name:");
12        JTextField nameField = new JTextField(columns:20);
13        JLabel addressLabel = new JLabel(text:"Address:");
14        JTextField addressField = new JTextField(columns:30);
15        JLabel phoneLabel = new JLabel(text:"Phone:");
16        JTextField phoneField = new JTextField(columns:15);
17        JButton addButton = new JButton(text:"Add Student");
18        JButton cancelButton = new JButton(text:"Cancel");
19
20        studentIdLabel.setBounds(x:20, y:20, width:80, height:20);
21        studentIdField.setBounds(x:120, y:20, width:200, height:20);
22        nameLabel.setBounds(x:20, y:60, width:80, height:20);
23        nameField.setBounds(x:120, y:60, width:200, height:20);
24        addressLabel.setBounds(x:20, y:100, width:80, height:20);
25        addressField.setBounds(x:120, y:100, width:200, height:20);
26        phoneLabel.setBounds(x:20, y:140, width:80, height:20);
27        phoneField.setBounds(x:120, y:140, width:200, height:20);
28        addButton.setBounds(x:120, y:180, width:100, height:30);
29        cancelButton.setBounds(x:230, y:180, width:90, height:30);
30
31        frame.add(studentIdLabel);
32        frame.add(studentIdField);
33        frame.add(nameLabel);
34        frame.add(nameField);
35        frame.add(addressLabel);
36        frame.add(addressField);
37        frame.add(phoneLabel);
38        frame.add(phoneField);
39        frame.add(addButton);
40        frame.add(cancelButton);
41
42        frame.setLayout(manager:null);
43
44        frame.setVisible(b:true);
45    }
46 }

```

In 1, Col 13 Spaces: 4 UTF-8 CR/LF {} Java ⌘ Go Live



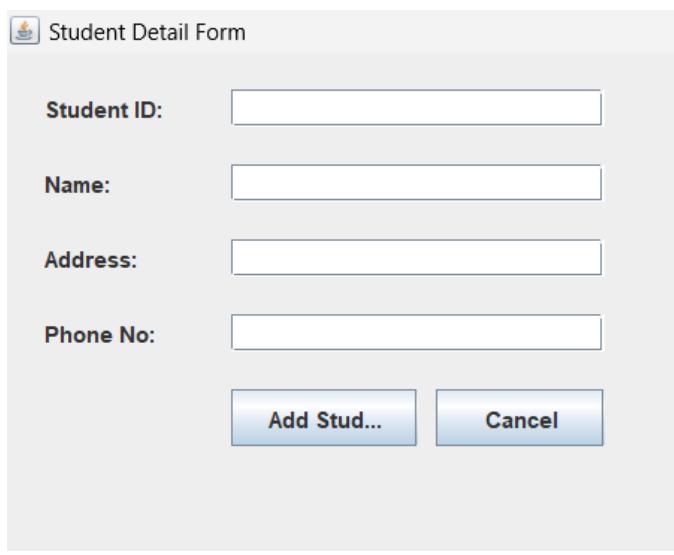
```

File Edit Selection View Go Run ... ⌘ Java Development
EXPLORER ... J test.java 1 ×
JAVA DEVELOPMENT > .vscode > Assignment-1 > Assignment-2 > Assignment-3 > Assignment-4 > Assignment-5 > Assignment-6 > Assignment-7 > Assignment-8 > GUI > Homeworks > lib J CalculatorGUI.class J CalculatorUI.class J Chapter1.java J Chapter2.1.java J Chapter2.java J GridLayoutExample.cl... J Practice.java J StudentDetailForm.cl... J test.class J test.java 1
J test.java > {} GUI
3 class StudentDetailForm {
4     public static void main(String[] args) {
5         studentIdLabel.setBounds(x:20, y:20, width:80, height:20);
6         studentIdField.setBounds(x:120, y:20, width:200, height:20);
7         nameLabel.setBounds(x:20, y:60, width:80, height:20);
8         nameField.setBounds(x:120, y:60, width:200, height:20);
9         addressLabel.setBounds(x:20, y:100, width:80, height:20);
10        addressField.setBounds(x:120, y:100, width:200, height:20);
11        phoneLabel.setBounds(x:20, y:140, width:80, height:20);
12        phoneField.setBounds(x:120, y:140, width:200, height:20);
13        addButton.setBounds(x:120, y:180, width:100, height:30);
14        cancelButton.setBounds(x:230, y:180, width:90, height:30);
15
16        frame.add(studentIdLabel);
17        frame.add(studentIdField);
18        frame.add(nameLabel);
19        frame.add(nameField);
20        frame.add(addressLabel);
21        frame.add(addressField);
22        frame.add(phoneLabel);
23        frame.add(phoneField);
24        frame.add(addButton);
25        frame.add(cancelButton);
26
27        frame.setLayout(manager:null);
28
29        frame.setVisible(b:true);
30    }
31
32 }

```

In 1, Col 13 Spaces: 4 UTF-8 CR/LF {} Java ⌘ Go Live

OUTPUT: -



Q22: - A toggle button, represented by the `JToggleButton` class in Java, is a button that can be either selected or deselected. When a toggle button is selected, it appears pressed in. When a toggle button is deselected, it appears unpressed. In the provided example, a `JToggleButton` is created with the text "Yes/No" and an initial status of true (selected). The toggle button is added to the `JFrame` using the `FlowLayout` layout manager. The user can click on the toggle button to change its state between selected and deselected. When the toggle button is selected, its text color changes to indicate that it is selected.

To detect when the state of the toggle button changes, you can add an Action Listener to the `JToggleButton`. The Action Listener's `actionPerformed` method will be called whenever the state of the `JToggleButton` changes.

Q23: - INPUT: -

```
File Edit Selection View Go Run Terminal Help ⏎ → Java Development
EXPLORER ... J test.java 1 ×
JAVA DEVELOPMENT > .vscode > StudentDetailForm > main(String[] args) > actionPerformed(ActionEvent)
1 package GUI;
2 import javax.swing.*;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 class StudentDetailForm {
6     Run|Debug
7     public static void main(String[] args) {
8         JFrame frame = new JFrame("Student detail");
9         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10        frame.setSize(400, 200);
11
12        JLabel nameLabel = new JLabel("Name:");
13        JTextField nameField = new JTextField(20); // 20 columns wide
14        JLabel addressLabel = new JLabel("Address:");
15        JTextField addressField = new JTextField(20);
16        JLabel genderLabel = new JLabel("Gender:");
17        JRadioButton maleRadioButton = new JRadioButton("Male");
18        JRadioButton femaleRadioButton = new JRadioButton("Female");
19        ButtonGroup genderGroup = new ButtonGroup();
20        genderGroup.add(maleRadioButton);
21        genderGroup.add(femaleRadioButton);
22        JButton saveButton = new JButton("Save");
23        JButton cancelButton = new JButton("Cancel");
24
25        frame.setLayout(null);
26
27        nameLabel.setBounds(x:20, y:20, width:60, height:20);
28        nameField.setBounds(x:90, y:20, width:200, height:20);
29        addressLabel.setBounds(x:20, y:50, width:60, height:20);
30        addressField.setBounds(x:90, y:50, width:200, height:20);
31        genderLabel.setBounds(x:20, y:80, width:60, height:20);
32        maleRadioButton.setBounds(x:90, y:80, width:60, height:20);
33        femaleRadioButton.setBounds(x:160, y:80, width:60, height:20);
34        saveButton.setBounds(x:90, y:120, width:80, height:30);
35        cancelButton.setBounds(x:180, y:120, width:80, height:30);
36
37        frame.add(nameLabel);
38        frame.add(nameField);
39        frame.add(addressLabel);
40        frame.add(addressField);
41
42        frame.setVisible(true);
}
Ln 67, Col 1 Spaces: 4 UTF-8 CRLF {} Java Go Live
```

A screenshot of the Visual Studio Code interface, specifically the Java Development extension. The left sidebar shows a file tree with several Java files like "Assignment-1.java" through "Assignment-8.java", "CalculatorGUI.class", "CalculatorUI.class", "Chapter1.java", "Chapter2.1.java", "Chapter2.java", "GridLayoutExample.class", "PracticeJava", "StudentDetailForm.class", and "test.java". The main editor area displays the code for "StudentDetailForm.java". The code creates a JFrame titled "Student detail", sets its size to 400x200, and uses a null layout manager. It contains four JLabels ("Name:", "Address:", "Gender:"), four JTextField and JRadioButton components, and two JButton components ("Save" and "Cancel"). The code uses Java's AWT and Swing libraries for the UI components. The bottom status bar shows the line number (67), column number (1), and other development details.

File Edit Selection View Go Run Terminal Help

J test.java 1 x

```

J testjava > StudentDetailForm > main(String[]) > new ActionListener() {} > actionPerformed(ActionEvent)
5   class StudentDetailForm {
6     public static void main(String[] args) {
26
27       nameLabel.setBounds(x:20, y:20, width:60, height:20);
28       nameField.setBounds(x:90, y:20, width:200, height:20);
29       addressLabel.setBounds(x:20, y:50, width:60, height:20);
30       addressField.setBounds(x:90, y:50, width:200, height:20);
31       genderLabel.setBounds(x:20, y:80, width:60, height:20);
32       maleRadioButton.setBounds(x:90, y:80, width:60, height:20);
33       femaleRadioButton.setBounds(x:160, y:80, width:80, height:20);
34       saveButton.setBounds(x:90, y:120, width:80, height:30);
35       cancelButton.setBounds(x:180, y:120, width:80, height:30);
36
37       frame.add(nameLabel);
38       frame.add(nameField);
39       frame.add(addressLabel);
40       frame.add(addressField);
41       frame.add(genderLabel);
42       frame.add(maleRadioButton);
43       frame.add(femaleRadioButton);
44       frame.add(saveButton);
45       frame.add(cancelButton);
46
47       frame.setVisible(true);
48
49       saveButton.addActionListener(new ActionListener() {
50         @Override
51         public void actionPerformed(ActionEvent e) {
52
53           String name = nameField.getText();
54           String address = addressField.getText();
55           String gender = maleRadioButton.isSelected() ? "Male" :
56             "Female";
57
58           System.out.println("Name: " + name);
59           System.out.println("Address: " + address);
60           System.out.println("Gender: " + gender);
61         }
62       });
63
64       cancelButton.addActionListener(new ActionListener() {
65         @Override
66         public void actionPerformed(ActionEvent e) {
67
68           frame.dispose();
69           [ ]
70         }
71       });
72     }

```

Ln 67, Col 1 Spaces:4 UTF-8 CRLF {} Java Go Live

File Edit Selection View Go Run ...

J test.java 1 x

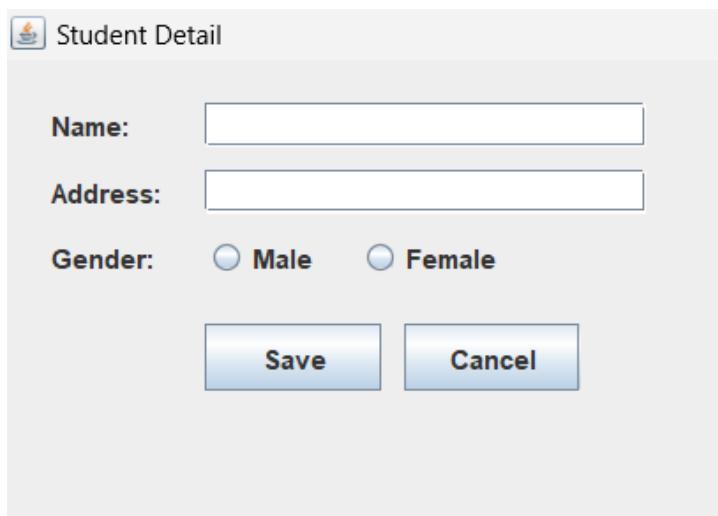
```

J testjava > StudentDetailForm > main(String[]) > new ActionListener() {} > actionPerformed(ActionEvent)
5   class StudentDetailForm {
6     public static void main(String[] args) {
49       saveButton.addActionListener(new ActionListener() {
50         public void actionPerformed(ActionEvent e) {
51
52           String address = addressField.getText();
53           String gender = maleRadioButton.isSelected() ? "Male" :
54             "Female";
55
56           System.out.println("Name: " + name);
57           System.out.println("Address: " + address);
58           System.out.println("Gender: " + gender);
59         }
60
61       );
62
63       cancelButton.addActionListener(new ActionListener() {
64         @Override
65         public void actionPerformed(ActionEvent e) {
66
67           frame.dispose();
68           [ ]
69         }
70       });
71     }
72   }

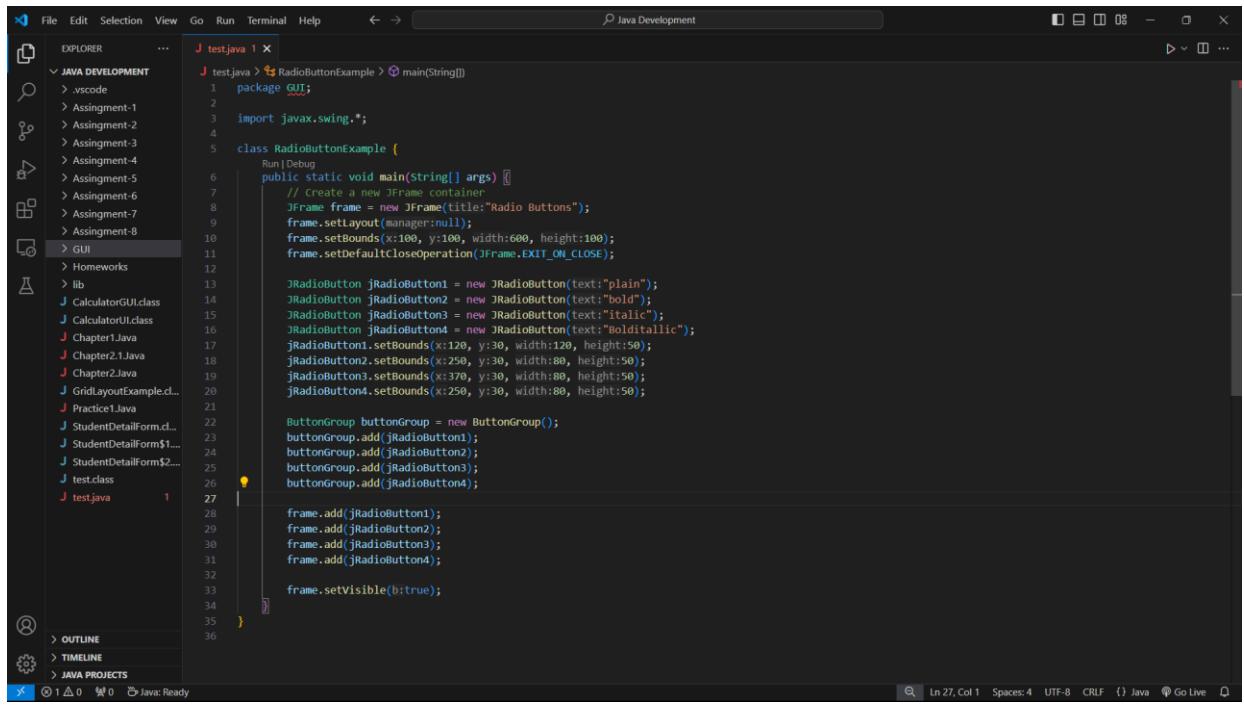
```

Ln 67, Col 1 Spaces:4 UTF-8 CRLF {} Java Go Live

OUTPUT: -



Q24: - INPUT: -

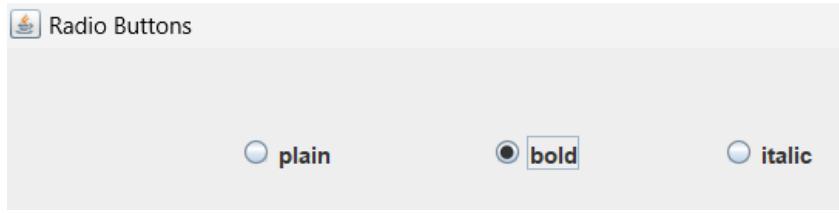


A screenshot of the Visual Studio Code interface. The title bar says "Java Development". The left sidebar shows a project structure under "JAVA DEVELOPMENT" with various Java files like "test.java", "CalculatorUI.class", and "Chapter1.java". The main editor area contains the following Java code:

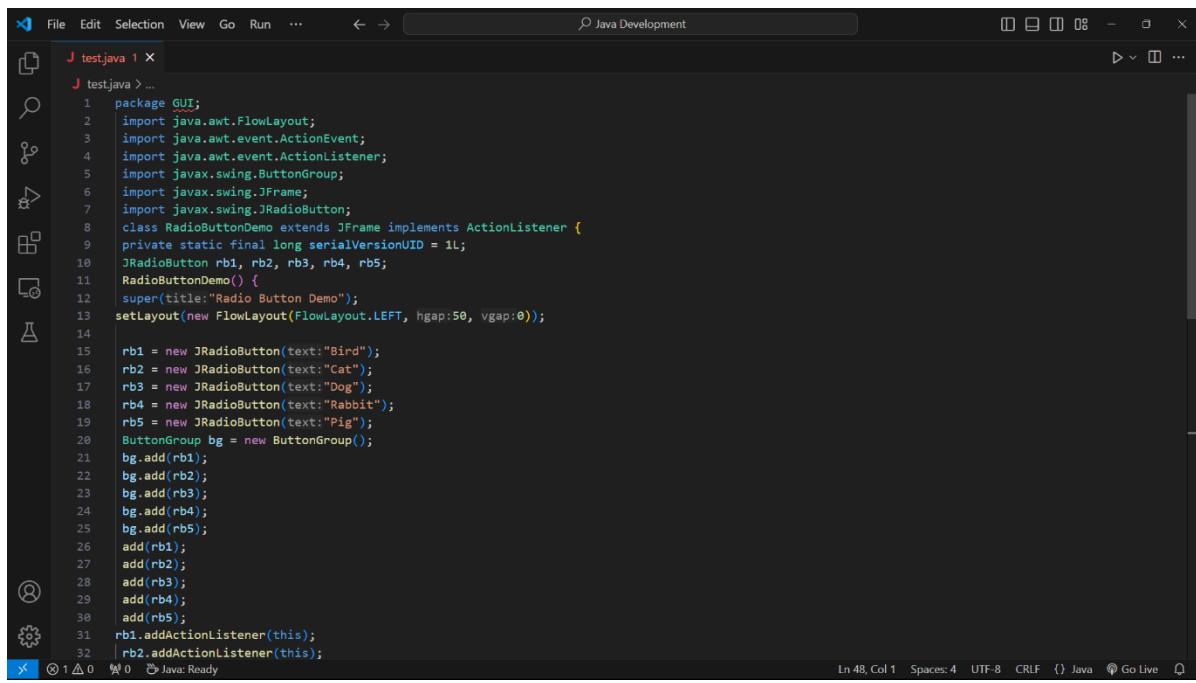
```
test.java 1 x
J test.java > RadioButtonItemExample > main(String[])
1 package GUI;
2
3 import javax.swing.*;
4
5 class RadioButtonExample {
6     public static void main(String[] args) {
7         // Create a new JFrame container
8         JFrame frame = new JFrame("Radio Buttons");
9         frame.setLayout(null);
10        frame.setBounds(x:100, y:100, width:600, height:100);
11        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12
13        JRadioButton jRadioButton1 = new JRadioButton(text:"plain");
14        JRadioButton jRadioButton2 = new JRadioButton(text:"bold");
15        JRadioButton jRadioButton3 = new JRadioButton(text:"italic");
16        JRadioButton jRadioButton4 = new JRadioButton(text:"bolditalic");
17        jRadioButton1.setBounds(x:120, y:30, width:120, height:50);
18        jRadioButton2.setBounds(x:250, y:30, width:80, height:50);
19        jRadioButton3.setBounds(x:370, y:30, width:80, height:50);
20        jRadioButton4.setBounds(x:250, y:30, width:80, height:50);
21
22        ButtonGroup buttonGroup = new ButtonGroup();
23        buttonGroup.add(jRadioButton1);
24        buttonGroup.add(jRadioButton2);
25        buttonGroup.add(jRadioButton3);
26        buttonGroup.add(jRadioButton4);
27
28        frame.add(jRadioButton1);
29        frame.add(jRadioButton2);
30        frame.add(jRadioButton3);
31        frame.add(jRadioButton4);
32
33        frame.setVisible(b:true);
34    }
35 }
```

The status bar at the bottom shows "Ln 27, Col 1 Spaces: 4 UTF-8 CRLF {} Java Go Live".

OUTPUT: -



Q28: - INPUT: -



A screenshot of the Visual Studio Code interface. The title bar says "Java Development". The left sidebar shows a project structure under "JAVA DEVELOPMENT" with various Java files like "test.java", "RadioButtonDemo.class", and "Chapter1.java". The main editor area contains the following Java code:

```
test.java 1 x
J test.java > ...
1 package GUI;
2
3 import java.awt.FlowLayout;
4 import java.awt.event.ActionEvent;
5 import java.awt.event.ActionListener;
6 import javax.swing.JButton;
7 import javax.swing.JFrame;
8 import javax.swing.JRadioButton;
9
10 class RadioButtonDemo extends JFrame implements ActionListener {
11     private static final long serialVersionUID = 1L;
12     JRadioButton rb1, rb2, rb3, rb4, rb5;
13
14     RadioButtonDemo() {
15         super(title:"Radio Button Demo");
16         setLayout(new FlowLayout(FlowLayout.LEFT, hgap:50, vgap:0));
17
18         rb1 = new JRadioButton(text:"Bird");
19         rb2 = new JRadioButton(text:"Cat");
20         rb3 = new JRadioButton(text:"Dog");
21         rb4 = new JRadioButton(text:"Rabbit");
22         rb5 = new JRadioButton(text:"Pig");
23
24         JButton bg = new JButton();
25         bg.add(rb1);
26         bg.add(rb2);
27         bg.add(rb3);
28         bg.add(rb4);
29         bg.add(rb5);
30
31         rb1.addActionListener(this);
32         rb2.addActionListener(this);
```

The status bar at the bottom shows "Ln 48, Col 1 Spaces: 4 UTF-8 CRLF {} Java Go Live".

```

 1  testjava 1 X
 2
 3  J testjava > ...
 4
 5      8   class RadioButtonDemo extends JFrame implements ActionListener {
 6
 7          11  RadioButtonDemo() {
 8
 9              21  bg.add(rb1);
10              22  bg.add(rb2);
11              23  bg.add(rb3);
12              24  bg.add(rb4);
13              25  bg.add(rb5);
14
15              26  add(rb1);
16              27  add(rb2);
17              28  add(rb3);
18              29  add(rb4);
19              30  add(rb5);
20
21              31  rb1.addActionListener(this);
22              32  rb2.addActionListener(this);
23              33  rb3.addActionListener(this);
24              34  rb4.addActionListener(this);
25              35  rb5.addActionListener(this);
26
27              36  setVisible(b:true);
28              37  setSize(width:200, height:180);
29
30
31
32
33
34
35
36
37
38
39  public void actionPerformed(ActionEvent e) {
40      JRadioButton jab = (JRadioButton) e.getSource();
41      String text = jab.getText();
42      System.out.println("Selected: " + text);
43
44
45  public static void main(String[] args) {
46      new RadioButtonDemo();
47  }
48

```

Ln 48, Col 1 Spaces: 4 UTF-8 CRLF {} Java Go Live

OUTPUT: -



Q30: - INPUT: -

```

 1  testjava 1 X
 2
 3  J testjava > {} GUI;
 4
 5
 6  package GUI;
 7
 8
 9  import javax.swing.*;
10
11  import java.awt.event.*;
12
13
14  class MenuBarToolBar extends JFrame implements ActionListener {
15
16      private JMenuBar menuBar;
17      private JMenu fileMenu, formatMenu, viewMenu, editMenu, helpMenu;
18
19      private JMenuItem newItem, openMenuItem, openMenuItem, saveMenuItem, exitMenuItem,
20          formatItem, viewItem, viewItem;
21
22      private JToolBar toolbar;
23      private JButton newButton, openButton, saveButton;
24
25
26  public MenuBarToolBar() {
27      super(title:"Database Application Example");
28
29      menuBar = new JMenuBar();
30
31      fileMenu = new JMenu(s:"File");
32      formatMenu = new JMenu(s:"Format");
33      viewMenu = new JMenu(s:"View");
34      editMenu = new JMenu(s:"Edit");
35      helpMenu = new JMenu(s:"Help");
36
37      newItem = new JMenuItem(text:"New");
38      openMenuItem = new JMenuItem(text:"Open");
39      saveMenuItem = new JMenuItem(text:"Save");
40      exitMenuItem = new JMenuItem(text:"Exit");
41
42      newItem.addActionListener(this);
43      openMenuItem.addActionListener(this);
44      saveMenuItem.addActionListener(this);
45      exitMenuItem.addActionListener(this);
46
47      fileMenu.add(newMenuItem);
48      fileMenu.add(openMenuItem);
49      fileMenu.add(saveMenuItem);
50      fileMenu.addSeparator();
51      fileMenu.add(exitMenuItem);
52
53
54
55
56
57
58
59
59

```

Ln 1, Col 13 Spaces: 4 UTF-8 CRLF {} Java Go Live

The screenshot shows the Java Development interface with the code for `MenuBarToolBar.java`. The code implements a `JFrame` with a menu bar and tool bar. It includes methods for creating menus like File, View, Format, Edit, and Help, and tool bars for New, Open, Save, and Exit. The code uses `ActionListener` and `ImageIcon` for button creation.

```
File Edit Selection View Go Run Terminal Help ← → ⚡ Java Development
J testJava 1 ×
J testJava > {} GUI
6 class MenubarToolBar extends JFrame implements ActionListener {
14 public MenubarToolBar() {
24
25     newMenuItem = new JMenuItem(text:"New");
26     openMenuItem = new JMenuItem(text:"Open");
27     saveMenuItem = new JMenuItem(text:"Save");
28     exitMenuItem = new JMenuItem(text:"Exit");
29
30     newMenuItem.addActionListener(this);
31     openMenuItem.addActionListener(this);
32     saveMenuItem.addActionListener(this);
33     exitMenuItem.addActionListener(this);
34
35     fileMenu.add(newMenuItem);
36     fileMenu.add(openMenuItem);
37     fileMenu.add(saveMenuItem);
38     fileMenu.addSeparator();
39     fileMenu.add(exitMenuItem);
40
41     formatItem = new JMenuItem(text:"Format");
42     formatMenu.add(formatItem);
43
44     viewItem1 = new JMenuItem(text:"View1");
45     viewItem2 = new JMenuItem(text:"View2");
46     viewMenu.add(viewItem1);
47     viewMenu.add(viewItem2);
48
49     menuBar.add(fileMenu);
50     menuBar.add(formatMenu);
51     menuBar.add(viewMenu);
52     menuBar.add(editMenu);
53     menuBar.add(helpMenu);
54
55     toolBar = new JToolBar();
56
57     newButton = new JButton(new ImageIcon(filename:"[Image of New Icon]"));
58     openButton = new JButton(new ImageIcon(filename:"[Image of Open Icon]"));
59
60     editButton = new JButton(new ImageIcon(filename:"[Image of Edit Icon]"));
61     saveButton = new JButton(new ImageIcon(filename:"[Image of Save Icon]"));
62
63     toolBar.add(newButton);
64     toolBar.add(openButton);
65     toolBar.add(editButton);
66     toolBar.add(saveButton);
67
68     add(menuBar, "North");
69     add(toolBar, "South");
70
71     pack();
72     setLayout(null);
73     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
74     setVisible(true);
75 }
76
77 public void actionPerformed(ActionEvent e) {
78     if (e.getSource() == newMenuItem) {
79         System.out.println("New item selected");
80     }
81     else if (e.getSource() == openMenuItem) {
82         System.out.println("Open item selected");
83     }
84     else if (e.getSource() == saveMenuItem) {
85         System.out.println("Save item selected");
86     }
87     else if (e.getSource() == exitMenuItem) {
88         System.out.println("Exit item selected");
89     }
90     else if (e.getSource() == formatItem) {
91         System.out.println("Format item selected");
92     }
93     else if (e.getSource() == viewItem1) {
94         System.out.println("View1 item selected");
95     }
96     else if (e.getSource() == viewItem2) {
97         System.out.println("View2 item selected");
98     }
99     else if (e.getSource() == editButton) {
100        System.out.println("Edit item selected");
101    }
102    else if (e.getSource() == saveButton) {
103        System.out.println("Save item selected");
104    }
105 }
106
107 public static void main(String[] args) {
108     new MenubarToolBar();
109 }
110 }
```

Ln 1, Col 13 Spaces: 4 UTF-8 CRLF ⓘ Java ⓘ Go Live

OUTPUT: -

The screenshot shows the Java Development interface with the terminal output and a running application window. The terminal shows the command to run the Java application and its execution. The application window displays a menu bar with options like File, Format, View, Edit, and Help, and a tool bar with buttons for New, Open, Save, and Exit.

```
File Edit Selection View Go Run ... ← → ⚡ Java Development
PROBLEMS 1 OUTPUT TEST RESULTS DEBUG CONSOLE PORTS TERMINAL
PS C:\Users\KIIT\Pictures\Screenshots\Java Development> cd "c:\Users\KIIT\Pictures\Screenshots\Java Development\" ; if ($?) { javac test.java } ;
PS C:\Users\KIIT\Pictures\Screenshots\Java Development> Java test.java
Menu item clicked: New
Menu item clicked: Open
Menu item clicked: Save
PS C:\Users\KIIT\Pictures\Screenshots\Java Development> Java test.java
[Database Application Example]
File Format View Edit Help
New
Open
Save
Exit
```

Ln 1, Col 12 Spaces: 4 UTF-8 CRLF ⓘ Java ⓘ Go Live

Q31: - INPUT: -

```

File Edit Selection View Go Run Terminal Help < > Java Development
J test.java ×
J test.java > $ SliderDemo > stopAnimation()
1 import javax.swing.*;
2 import javax.swing.event.ChangeEvent;
3 import javax.swing.event.ChangeListener;
4 import java.awt.*;
5 class SliderDemo extends JFrame implements ChangeListener {
6 private static final int FPS_MIN = 0;
7 private static final int FPS_MAX = 50;
8 private static final int FPS_INIT = 15;
9 private Timer timer;
10 private boolean frozen = false;
11 private int delay = 1000 / FPS_INIT;
12 public SliderDemo() {
13 setTitle("Slider Example");
14 setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
15 setLayout(new FlowLayout());
16 JSlider framesPerSecond = new JSlider(JSlider.HORIZONTAL, FPS_MIN,
17 FPS_MAX, FPS_INIT);
18 framesPerSecond.addChangeListener(this);
19 framesPerSecond.setMajorTickSpacing(10);
20 framesPerSecond.setMinorTickSpacing(1);
21 framesPerSecond.setPaintTicks(true);
22 framesPerSecond.setPaintLabels(true);
23 Font font = new Font("Serif", Font.ITALIC, 15);
24 framesPerSecond.setFont(font);
25 add(framesPerSecond);
26 timer = new Timer(delay, e -> {
27 });
28 timer.start();
29 pack();
30 setLocationRelativeTo(null);
31 }
32 public void stateChanged(ChangeEvent e) {
33 JSlider source = (JSlider) e.getSource();
34 if (!source.getValueIsAdjusting()) {
35 int fps = (int) source.getValue();
36 if (fps == 0) {
37 stopAnimation();
38 } else {
39 delay = 1000 / fps;
40 timer.setInitialDelay(delay * 10);
41 if (frozen) startAnimation();
42 }
43 }
44 private void startAnimation() {
45 frozen = false;
46 }
47 private void stopAnimation() {
48 frozen = true;
49 }
50 public static void main(String[] args) {
51 SwingUtilities.invokeLater(() -> new
52 SliderDemo().setVisible(true));
53 }
54 }

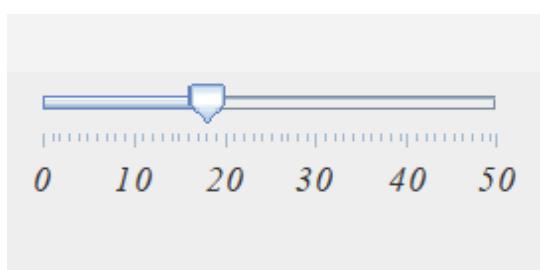
```

```

File Edit Selection View Go Run Terminal Help < > Java Development
J test.java ×
J test.java > $ SliderDemo > stopAnimation()
1 class SliderDemo extends JFrame implements ChangeListener {
2 public void stateChanged(ChangeEvent e) {
3 if (fps == 0) {
4 if (!frozen) stopAnimation();
5 } else {
6 delay = 1000 / fps;
7 timer.setInitialDelay(delay * 10);
8 if (frozen) startAnimation();
9 }
10 }
11 private void startAnimation() {
12 frozen = false;
13 }
14 private void stopAnimation() {
15 frozen = true;
16 }
17 Run | Debug
18 public static void main(String[] args) {
19 SwingUtilities.invokeLater(() -> new
20 SliderDemo().setVisible(true));
21 }
22 }

```

OUTPUT: -



Q33: - INPUT

The screenshot shows an IDE interface with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, etc. A search bar is at the top right. The main area displays a Java file named test.java. The code creates a JFrame titled "Time Table" with a DefaultTableModel. The model has 5 columns (Monday through Friday) and 10 rows. A JTable is created from the model, and a JScrollPane is used to add the table to the frame. The frame is set to size 500x220 and has a default close operation. It is made visible.

```
test.java
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
class TimeTableExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Time Table");
        DefaultTableModel model = new DefaultTableModel(
            new Object[] { "Monday", "Tuesday", "Wednesday", "Thursday", "Friday" }, rowCount:0);
        for (int i = 0; i <= 9; i++) {
            model.addRow(new Object[] { null, null, null, null });
        }
        JTable table = new JTable(model);
        JScrollPane scrollPane = new JScrollPane(table);
        frame.add(scrollPane);
        frame.setSize(width:500, height:220);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(b:true);
    }
}
```

OUTPUT: -



Q25: - INPUT

The screenshot shows an IDE interface with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, etc. A search bar is at the top right. The main area displays a Java file named test.java. The code creates a JFrame titled "ComboBox Demo". Inside, it creates a JPanel with a FlowLayout. It adds three JComboBoxes: one for days (1-31), one for months (January-December), and one for years (1990-2020). It also adds two JLabels for "Day:" and "Year:". Finally, it packs and makes the frame visible.

```
test.java
class ComboBoxDemo {
    public static void main(String[] args) {
        // Create and set up the window.
        JFrame frame = new JFrame("ComboBox Demo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        //Create and set up the content pane.
        JPanel panel = new JPanel();
        frame.add(panel);
        panel.setLayout(new FlowLayout());

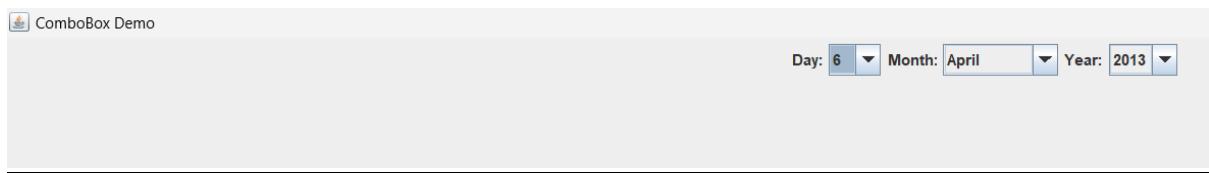
        // Create the combo boxes
        String[] days = new String[31];
        for (int i = 1; i <= 31; i++) {
            days[i - 1] = Integer.toString(i);
        }
        panel.add(new JLabel(text:"Day:"));
        JComboBox<String> dayBox = new JComboBox<>(days);
        panel.add(dayBox);

        String[] months = {"January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"};
        panel.add(new JLabel(text:"Month:"));
        JComboBox<String> monthBox = new JComboBox<>(months);
        panel.add(monthBox);

        String[] years = new String[31];
        for (int i = 1990; i <= 2020; i++) {
            years[i - 1990] = Integer.toString(i);
        }
        panel.add(new JLabel(text:"Year:"));
        JComboBox<String> yearBox = new JComboBox<>(years);
        panel.add(yearBox);

        //Display the window.
        frame.pack();
        frame.setVisible(b:true);
    }
}
```

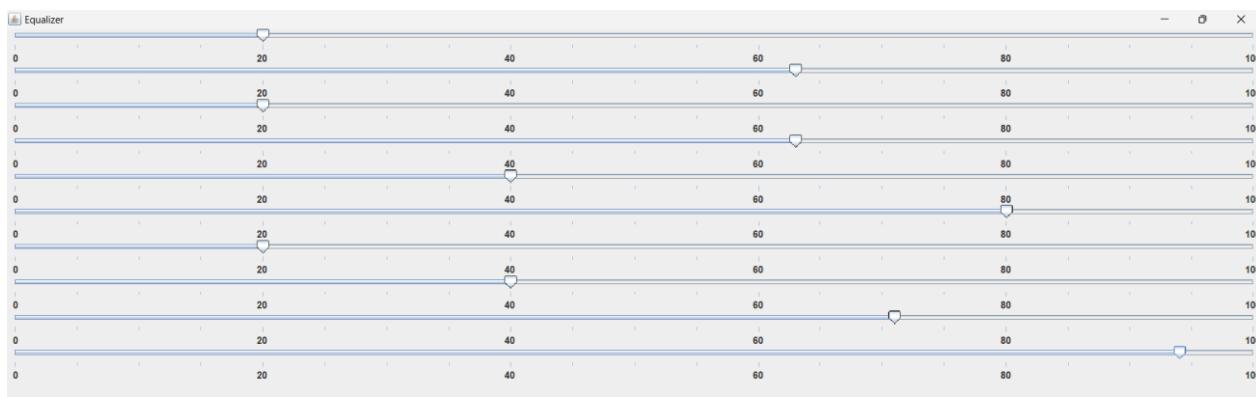
OUTPUT: -



Q26: - INPUT

```
testjava 1 x
testjava > ...
1 import javax.swing.*;
2 import java.awt.*;
3
4 class Equalizer {
5     Run|Debug
6     public static void main(String[] args) {
7
8         JFrame frame = new JFrame("Equalizer");
9         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10        frame.setSize(width:200, height:400);
11
12        JPanel panel = new JPanel();
13        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
14
15        // Create 10 sliders
16        for (int i = 0; i < 10; i++) {
17            JSlider slider = new JSlider(JSlider.HORIZONTAL, min:0, max:100, value:50);
18            slider.setMajorTickSpacing(n:20);
19            slider.setMinorTickSpacing(n:5);
20            slider.setPaintTicks(b:true);
21            slider.setPaintLabels(b:true);
22            panel.add(slider);
23        }
24
25        frame.add(panel);
26        frame.setVisible(b:true);
27    }
28 }
29
```

OUTPUT: -



Q27: - INPUT/OUTPUT

The screenshot shows a Java Development environment with a code editor and a preview window. The code editor displays a Java file named `test.java` containing the following code:

```
test.java
1 test.java > Login > main(String[])
2 import javax.swing.*;
3 import java.awt.*;
4 class loginUI {
5     Run|Debug
6     public static void main(String[] args) {
7         JFrame frame = new JFrame("Login");
8         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9         frame.setSize(300, height:300);
10        JPanel panel = new JPanel();
11        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
12        JLabel userLabel = new JLabel(text:"User Name:");
13        JTextField userField = new JTextField(text:"admin");
14        JLabel passLabel = new JLabel(text:"Password:");
15        JPasswordField passField = new JPasswordField();
16        JLabel dbLabel = new JLabel(text:"Database:");
17        String[] databases = {"Common"};
18        JComboBox<String> dbBox = new JComboBox<>(databases);
19        JCheckBox rememberCheck = new JCheckBox(text:"Remember password");
20        JLabel encLabel = new JLabel(text:"Encryption Status:");
21        JTextField encField = new JTextField(text:"Unencrypted");
22        encField.setEditable(b:false);
23        JButton okButton = new JButton(text:"OK");
24        JButton closeButton = new JButton(text:"Close");
25        JButton helpButton = new JButton(text:"Help");
26        panel.add(userLabel);
27        panel.add(userField);
28        panel.add(dbBox);
29        panel.add(encLabel);
30        panel.add(encField);
31        panel.add(dbLabel);
32        panel.add(okButton);
33        panel.add(closeButton);
34        panel.add(helpButton);
35        panel.add(Box.createRigidArea(new Dimension(width:0, height:2)));
36        panel.add(Box.createRigidArea(new Dimension(width:0, height:2)));
37        panel.add(Box.createRigidArea(new Dimension(width:0, height:2)));
38        panel.add(Box.createRigidArea(new Dimension(width:0, height:2)));
39        panel.add(Box.createRigidArea(new Dimension(width:0, height:2)));
40    }
}
```

The preview window shows a "Login" dialog box with the following fields and buttons:

- User Name: admin
- Password: (empty)
- Database: Common
- Encryption Status: Unencrypted
- Checkboxes: Remember password (unchecked), Encryption Status (unchecked)
- Buttons: OK, Close, Help
