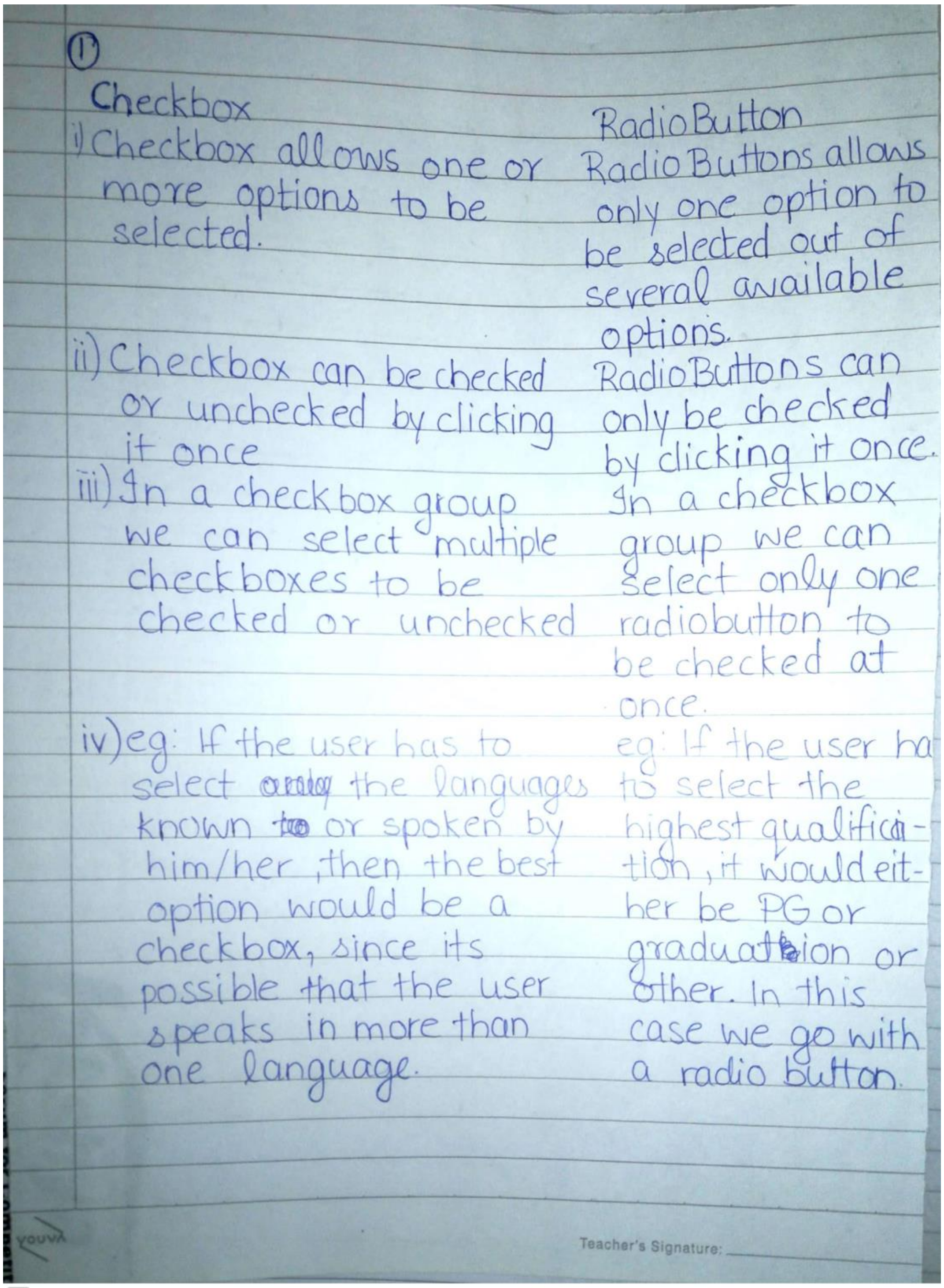
Experiment 1



①

Checkbox	RadioButton
i) Checkbox allows one or more options to be selected.	RadioButtons allows only one option to be selected out of several available options.
ii) Checkbox can be checked or unchecked by clicking it once	RadioButtons can only be checked by clicking it once.
iii) In a checkbox group we can select multiple checkboxes to be checked or unchecked	In a checkbox group we can select only one radiobutton to be checked at once.
iv) eg: If the user has to select ~~any~~ the languages known ~~to~~ or spoken by him/her, then the best option would be a checkbox, since its possible that the user speaks in more than one language.	eg: If the user ha to select the highest qualification, it would either be PG or graduation or other. In this case we go with a radio button.

Teacher's Signature:

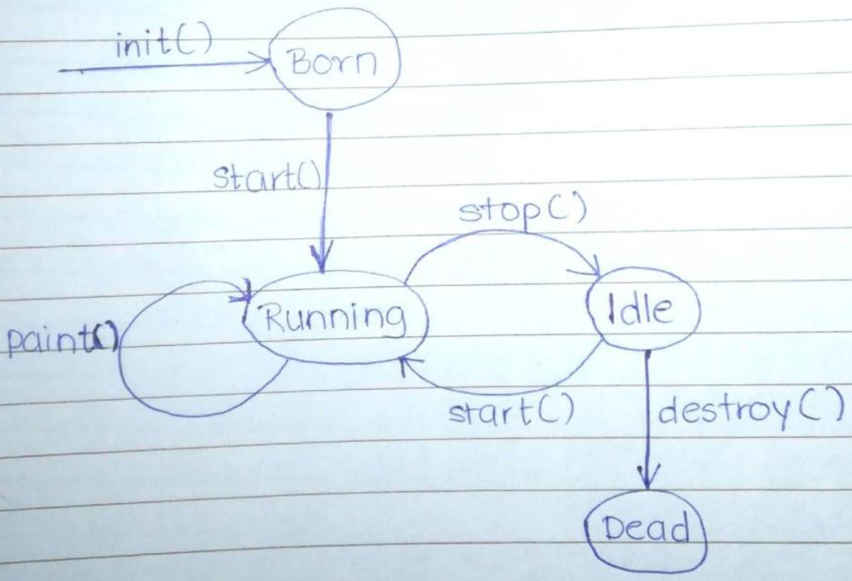
2

i) setEnabled() method is ~~a~~ ~~[illegible]~~ used on AWT or Swing components.

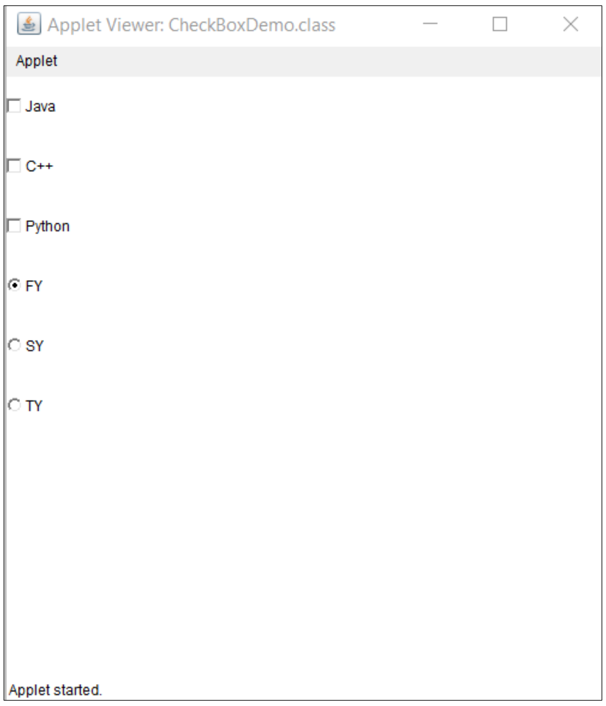
ii) By default all components in swing & AWT have setEnabled(true).

iii) If setEnabled(false) is used on a component that component will be disabled. The component cannot be selected by the user, data in the component cannot be copied by the user and the contents in the component cannot be changed directly.

3

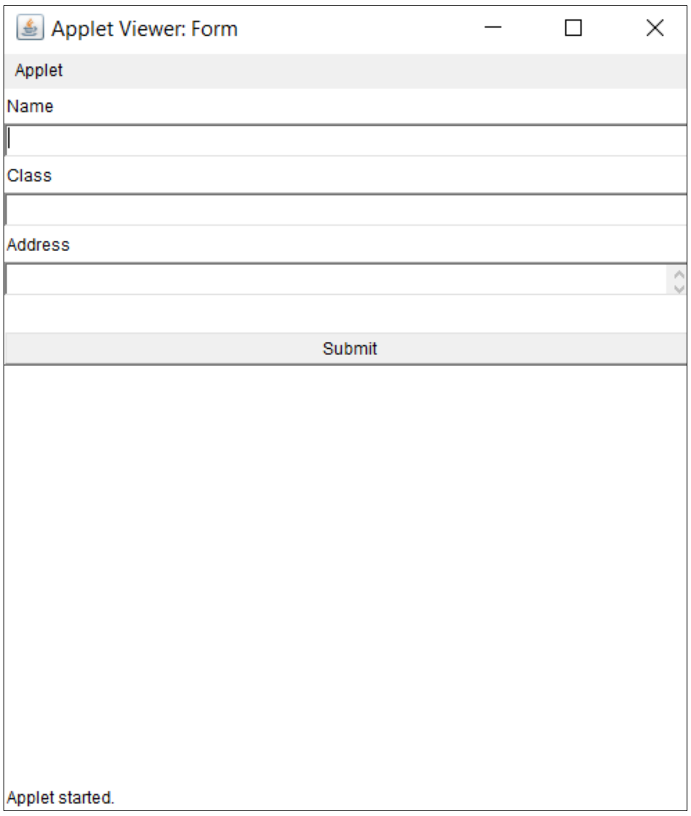


```java
import java.awt.*;
import java.applet.*;

public class CheckBoxDemo extends Applet{
    // Component Declaration
    CheckboxGroup cb= new CheckboxGroup();
    Checkbox c01,c02,c03,c11,c12,c13;
    public void init(){
    // Setting Layouts
        setLayout(new GridLayout(10,1));

    // CheckBox
        // Create Checkboxes
        c01=new Checkbox("Java");
        c02=new Checkbox("C++");
        c03=new Checkbox("Python");

        // Create RadioButtons
        c11=new Checkbox("FY",cb,true);
        c12=new Checkbox("SY",cb,false);
        c13=new Checkbox("TY",cb,false);

        // Add Checkboxes
        add(c01);
        add(c02);
        add(c03);

        // Add RadioButtons
        add(c11);
        add(c12);
        add(c13);

    }
}
/* <applet code="CheckBoxDemo.class" width=500 height=500></applet> */
```



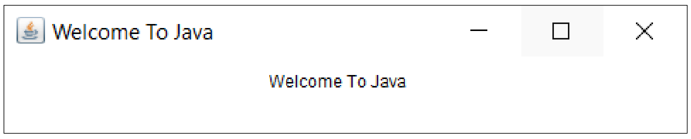
Applet started.

```java
import java.awt.*;
import java.applet.*;

public class Form extends Applet{
    // Component Declaration
    Label l1,l2,l3,l4;
    TextField tf1,tf2;
    TextArea ta1;
    Button b1;
    public void init(){
    // Layout
        setLayout(new GridLayout(20,2));

    // Label
        l1=new Label("Name");
        l2=new Label("Class");
        l3=new Label("Address");
        l4=new Label();

    // TextField
        tf1=new TextField();
        tf2=new TextField();

    // TextArea
        ta1=new TextArea();

    // Button
        b1=new Button("Submit");

    // Adding Components to Applet
        add(l1);
        add(tf1);
        add(l2);
        add(tf2);
        add(l3);
        add(ta1);
        add(l4);
        add(b1);
    }
}
/* <applet code="Form" width=500 height=500></applet> */
```



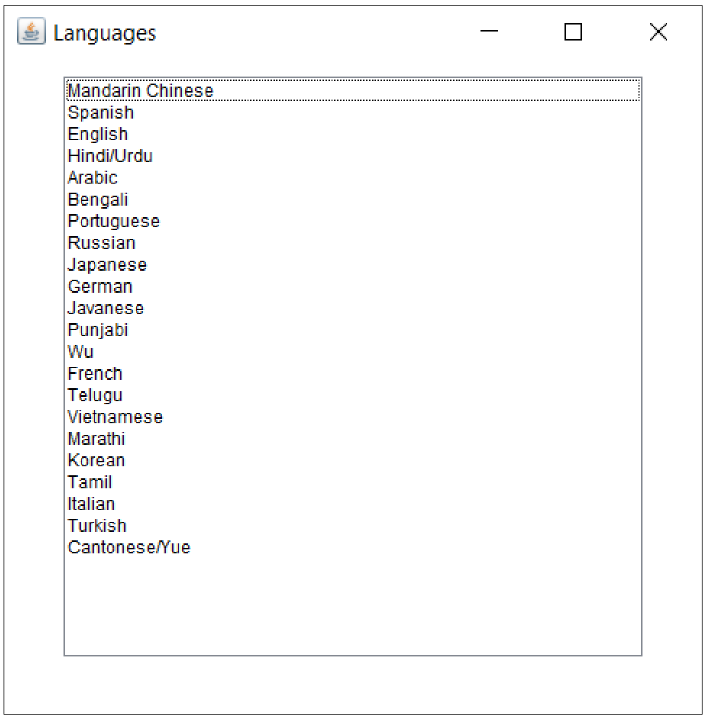
Applet started.

```
import java.awt.*;
import java.awt.event.*;

public class WelcomeToJava extends Frame{
    WelcomeToJava(){
        setLayout(new FlowLayout(FlowLayout.CENTER));

        Label l = new Label("Welcome To Java");
        add(l);

        setTitle("Welcome To Java");
        setVisible(true);
        setSize(500,100);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        WelcomeToJava w= new WelcomeToJava();
    }
}
```



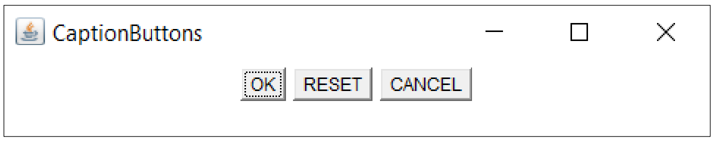
```
Welcome To Java

Welcome To Java
```

```
import java.awt.*;
import java.awt.event.*;

class Languages extends Frame{
    Languages(){
        setLayout(null);
        int x=500,y=500;

        String languages[]={"Mandarin
Chinese","Spanish","English","Hindi/Urdu","Arabic","Bengali","Portug
uese","Russian","Japanese","German","Javanese","Punjabi","Wu","Frenc
h","Telugu","Vietnamese","Marathi","Korean","Tamil","Italian","Turki
sh","Cantonese/Yue"
        };

        List l = new List(10,true);
        l.setBounds(x-450,y-450,x-100,y-100);

        for (String language: languages){
            l.add(language);
        }

        add(l);

        setTitle("Languages");
        setVisible(true);
        setSize(x,y);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        Languages w= new Languages();
    }
}
```

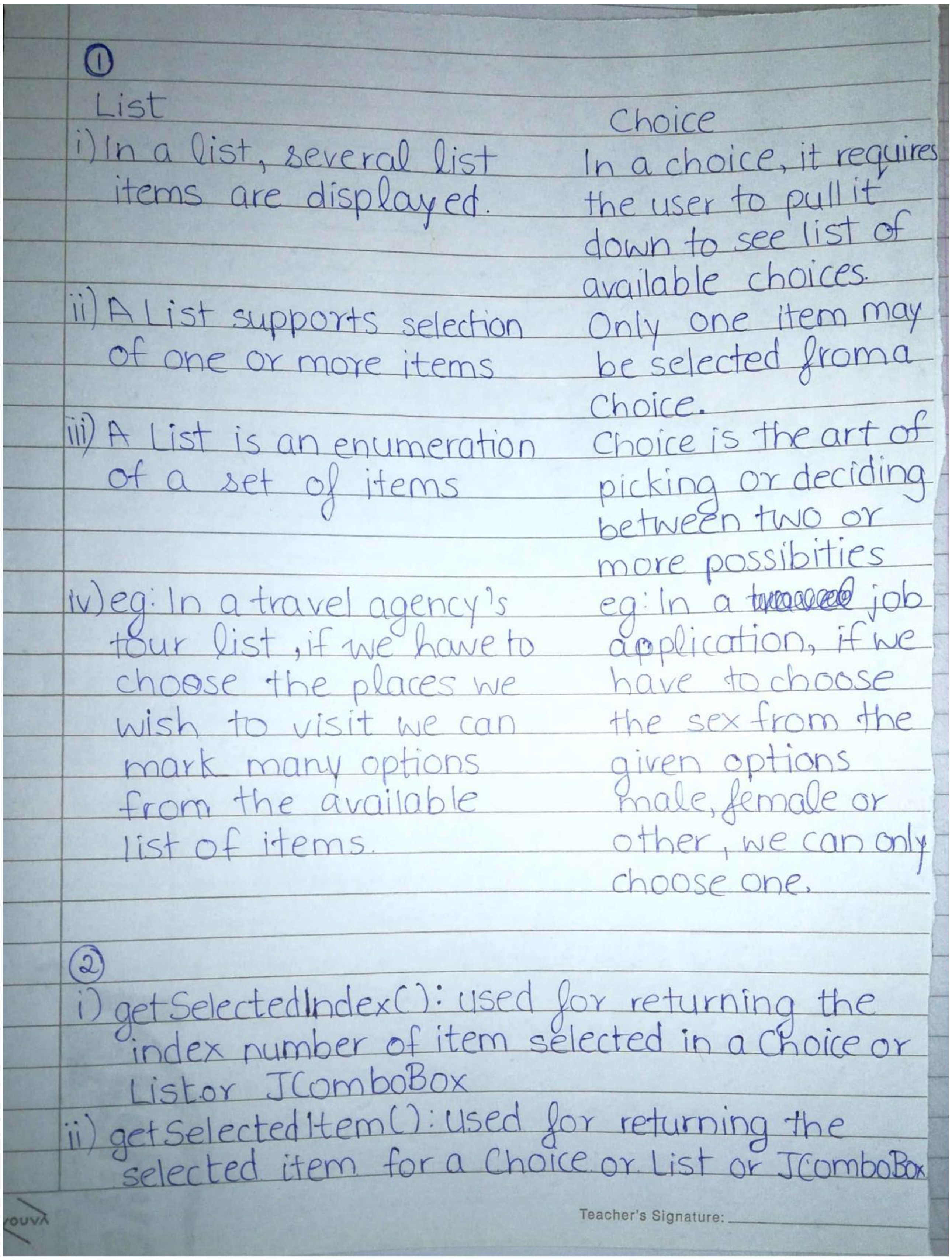


```
Languages

Mandarin Chinese
Spanish
English
Hindi/Urdu
Arabic
Bengali
Portuguese
Russian
Japanese
German
Javanese
Punjabi
Wu
French
Telugu
Vietnamese
Marathi
Korean
Tamil
Italian
Turkish
Cantonese/Yue
```

```java
import java.awt.*;
import java.awt.event.*;

class CaptionButtons extends Frame{
    CaptionButtons(){
        setLayout(new FlowLayout(FlowLayout.CENTER));
        int x=500,y=100;

        Button ok,reset,cancel;

        ok=new Button("OK");
        reset=new Button("RESET");
        cancel=new Button("CANCEL");

        add(ok);
        add(reset);
        add(cancel);

        setTitle("CaptionButtons");
        setVisible(true);
        setSize(x,y);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        CaptionButtons w= new CaptionButtons();
    }
}
```



Experiment 2



①

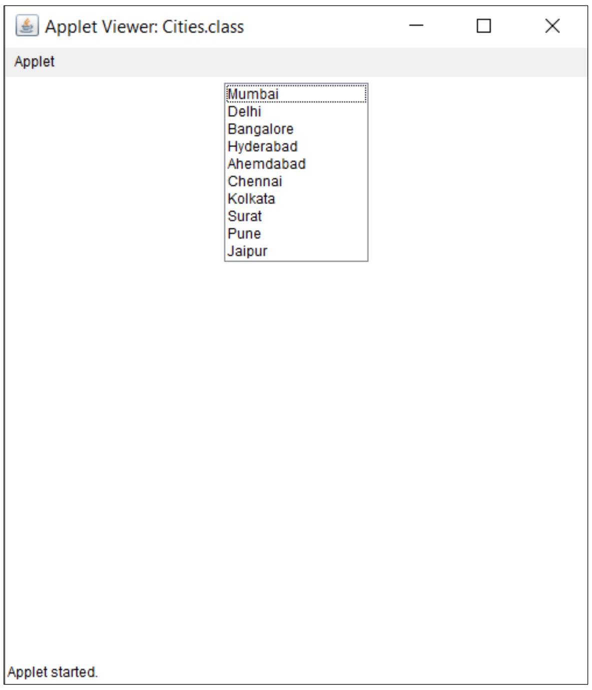
List	Choice
i) In a list, several list items are displayed.	In a choice, it requires the user to pull it down to see list of available choices.
ii) A List supports selection of one or more items	Only one item may be selected from a Choice.
iii) A List is an enumeration of a set of items	Choice is the art of picking or deciding between two or more possibilities
iv) eg: In a travel agency's tour list, if we have to choose the places we wish to visit we can mark many options from the available list of items.	eg: In a ~~travel~~ job application, if we have to choose the sex from the given options male, female or other, we can only choose one.

②

i) getSelectedIndex(): used for returning the index number of item selected in a Choice or List or JComboBox

ii) getSelectedItem(): used for returning the selected item for a Choice or List or JComboBox

Teacher's Signature: __________

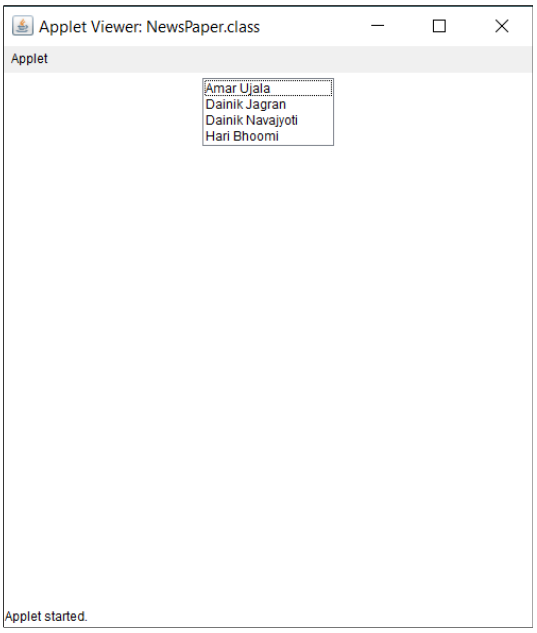

Applet Viewer: Cities.class

Applet

```
Mumbai
Delhi
Bangalore
Hyderabad
Ahemdabad
Chennai
Kolkata
Surat
Pune
Jaipur
```

Applet started.

```
import java.awt.*;
import java.applet.*;

public class Cities extends Applet{
    List l;
    public void init(){
        setLayout(new FlowLayout(FlowLayout.CENTER));
        l=new List();
        String items[]=
{"Mumbai","Delhi","Bangalore","Hyderabad","Ahemdabad",
"Chennai","Kolkata","Surat","Pune","Jaipur"};
        for(String item:items){
            l.add(item);
        }
        add(l);
    }
}
/*<applet code="Cities.class" width=500 height=500 ></applet> */
```

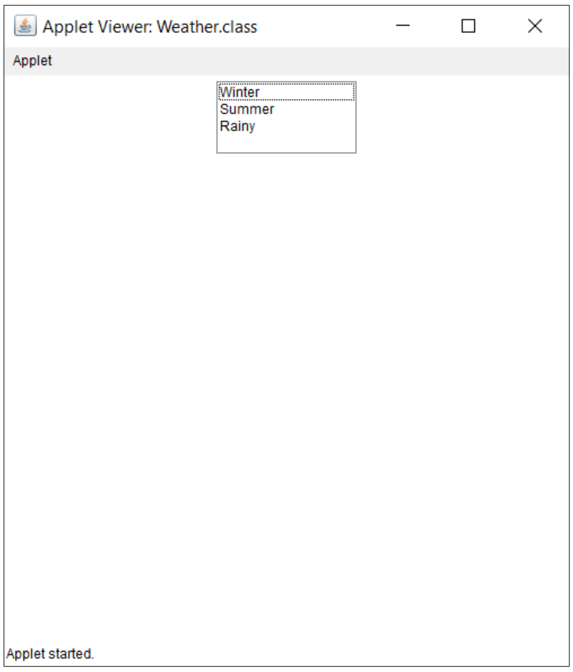

Applet Viewer: NewsPaper.class

Applet

```
Amar Ujala
Dainik Jagran
Dainik Navajyoti
Hari Bhoomi
```

Applet started.

```
import java.awt.*;
import java.applet.*;

public class NewsPaper extends Applet{
    List l;
    public void init(){
        setLayout(new FlowLayout(FlowLayout.CENTER));
        l=new List(4,true);
        String items[]={"Amar Ujala","Dainik Jagran",
"Dainik Navajyoti","Hari Bhoomi"};
        for(String item:items){
            l.add(item);
        }
        add(l);
    }
}
/*<applet code="NewsPaper.class" width=500 height=500 ></applet> */
```


Applet Viewer: Weather.class

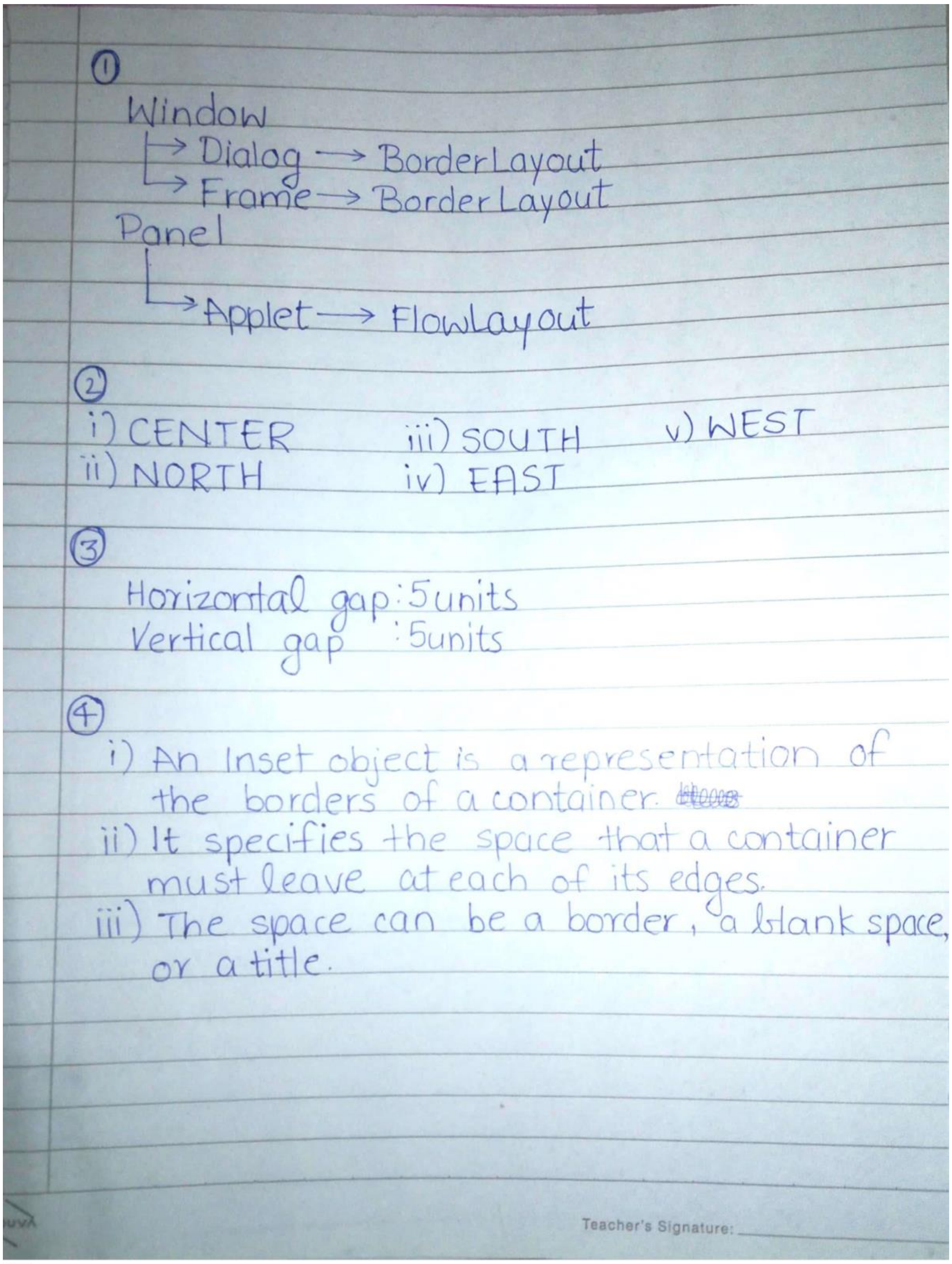
Applet

```
Winter
Summer
Rainy
```

Applet started.

```
import java.awt.*;
import java.applet.*;

public class Weather extends Applet{
    List l;
    public void init(){
        setLayout(new FlowLayout(FlowLayout.CENTER));
        l=new List();
        String items[]={"Winter","Summer","Rainy"};
        for(String item:items){
            l.add(item);
        }
        add(l);
    }
}
/*<applet code="Weather.class" width=500 height=500 ></applet> */
```

Experiment 3



①

$$\begin{tikzcd}
\text{Window} \arrow[d] \arrow[dd] & \\
\text{Dialog} \arrow[r] & \text{BorderLayout} \\
\text{Frame} \arrow[r] & \text{BorderLayout} \\
\text{Panel} \arrow[d] & \\
\text{Applet} \arrow[r] & \text{FlowLayout}
\end{tikzcd}$$

②

i) CENTER	iii) SOUTH	v) WEST
ii) NORTH	iv) EAST	

③

Horizontal gap	: 5 units
Vertical gap	: 5 units

④

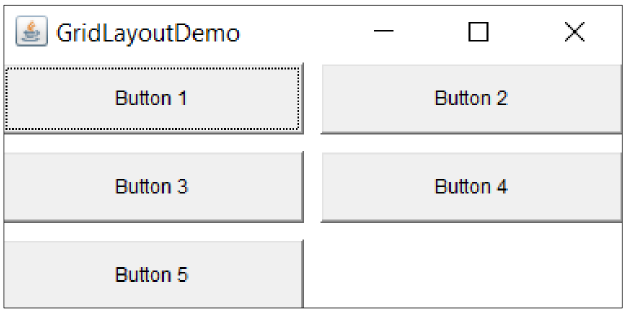
i) An Inset object is a representation of the borders of a container.

ii) It specifies the space that a container must leave at each of its edges.

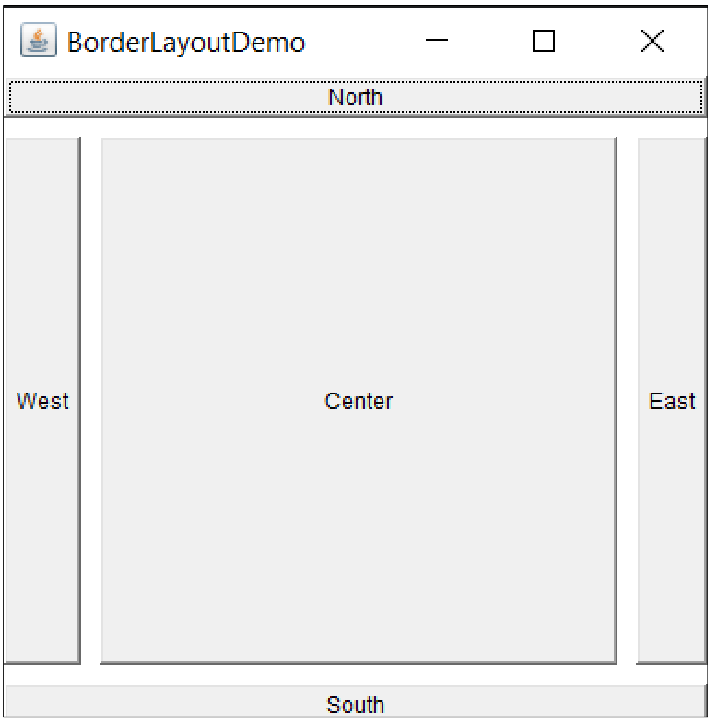
iii) The space can be a border, a blank space, or a title.

Teacher's Signature: ____________

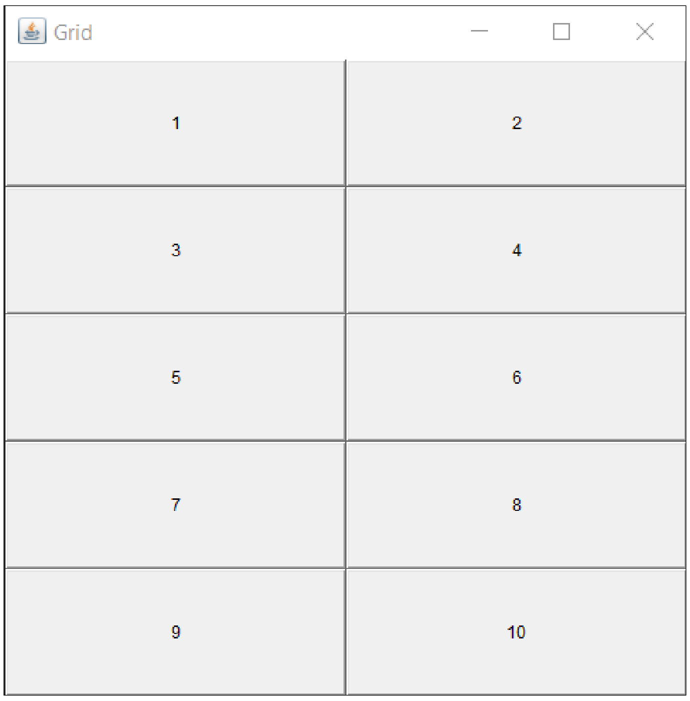
```
import java.awt.*;
import java.awt.event.*;

class GridLayoutDemo extends Frame{
    GridLayoutDemo(){
        setLayout(new GridLayout(3,2,10,10));
        int n=5;
        for(int i=1; i<=n; i++){
                Button b=new Button("Button "+Integer.toString(i));
                add(b);
        }


        setTitle("GridLayoutDemo");
        setVisible(true);
        setSize(400,200);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }

    public static void main(String[] args){
        GridLayoutDemo gd=new GridLayoutDemo();
    }
}
```



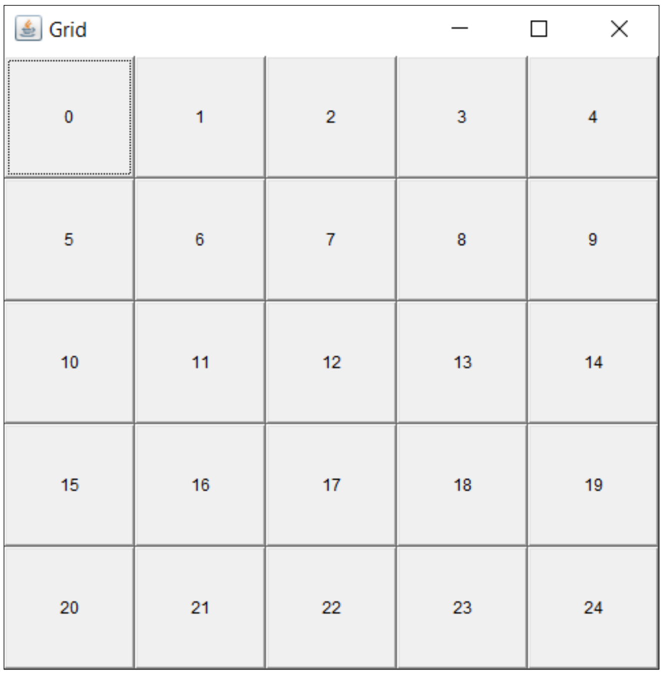
```
import java.awt.*;
import java.awt.event.*;

class BorderLayoutDemo extends Frame{
    BorderLayoutDemo(){
        setLayout(new BorderLayout(10,10));

        Button north=new Button("North");
        Button south=new Button("South");
        Button east=new Button("East");
        Button west=new Button("West");
        Button center=new Button("Center");

        add(north, BorderLayout.NORTH);
        add(south, BorderLayout.SOUTH);
        add(east, BorderLayout.EAST);
        add(west, BorderLayout.WEST);
        add(center, BorderLayout.CENTER);

        setTitle("BorderLayoutDemo");
        setVisible(true);
        setSize(400,400);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        BorderLayoutDemo bd=new BorderLayoutDemo();
    }
}
```



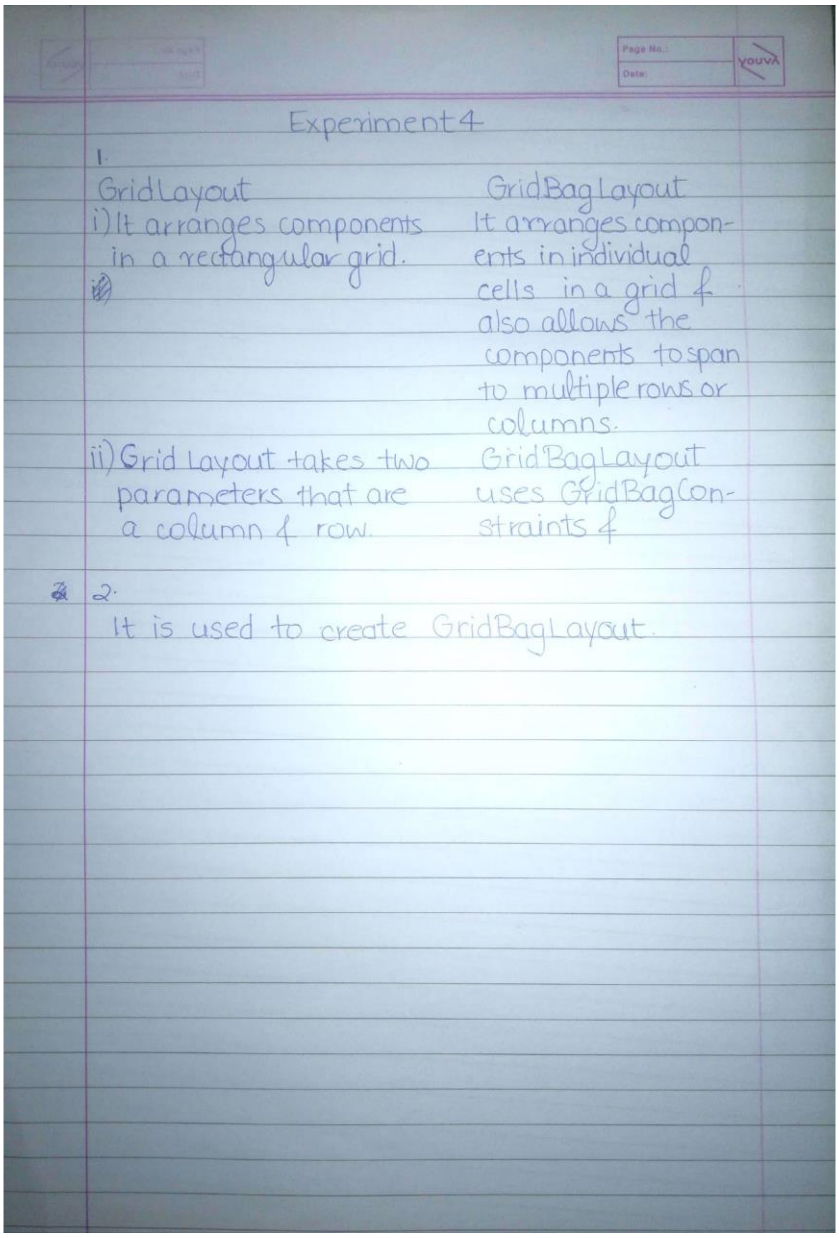
```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class Button1to9 extends Frame{
    Button1to9(){


        setLayout(new GridLayout(5,5));
        int count=0;
        for(int i=1; i<=10; i++){
                Button b = new Button(Integer.toString(i));
                add(b);
        }
        setTitle("Grid");
        setVisible(true);
        setSize(500,500);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        Button1to9 g= new Button1to9();
    }

}
```



```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class Grid5x5 extends Frame{
    Grid5x5(){


        setLayout(new GridLayout(5,5));
        int count=0;
        for(int i=0; i<5; i++){
            for(int j=0; j<5; j++){
                Button b = new Button(Integer.toString(count));
                count++;
                add(b);
            }
        }
        setTitle("Grid");
        setVisible(true);
        setSize(500,500);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });

    }
    public static void main(String[] args){
        Grid5x5 g= new Grid5x5();
    }

}
```



Experiment 4



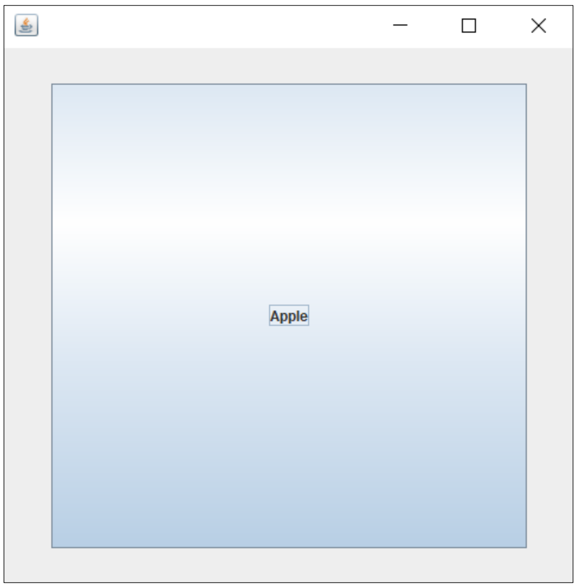
Experiment 4

1.

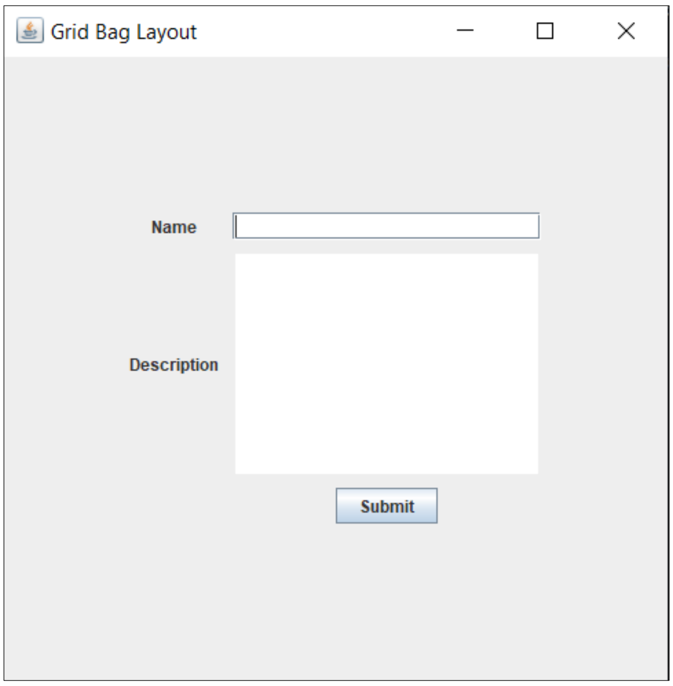
GridLayout	GridBagLayout
i) It arranges components in a rectangular grid.	It arranges components in individual cells in a grid & also allows the components to span to multiple rows or columns.
ii) Grid Layout takes two parameters that are a column & row.	GridBagLayout uses GridBagConstraints &

2.

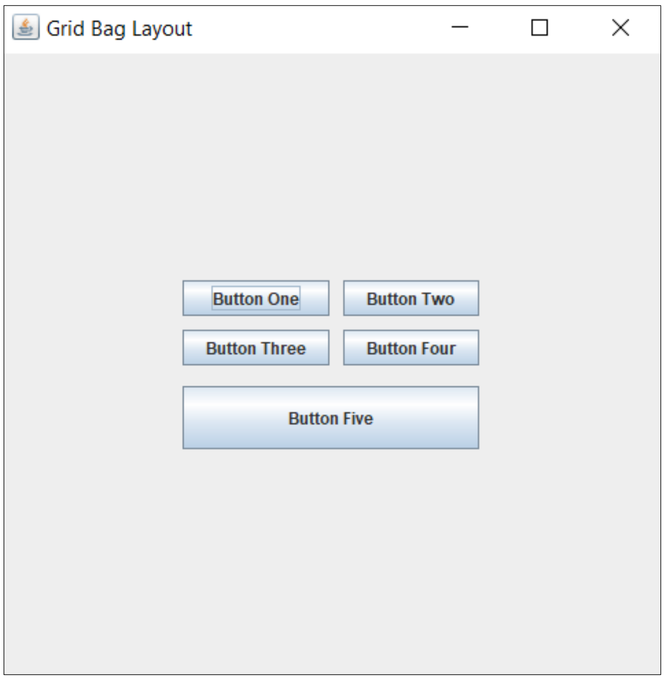
It is used to create GridBagLayout.



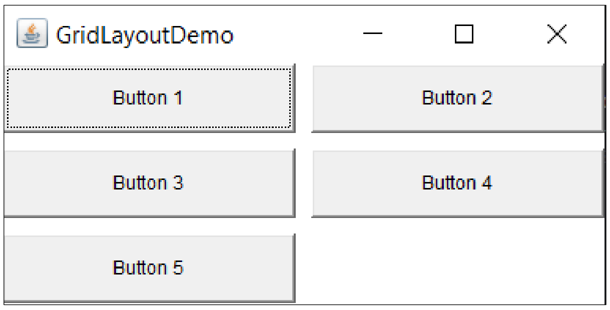
```java
import java.util.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class GridBagLayoutDemo1 extends JFrame {
    int buttons[][] = { { 0, 0 }, { 1, 0 }, { 0, 1 }, { 1, 1 } };
    String buttonLabels[] = { "One", "Two", "Three", "Four" };

    GridBagLayoutDemo1() {
        Container co = getContentPane();

        GridBagConstraints gbc = new GridBagConstraints();
        setLayout(new GridBagLayout());

        gbc.fill = GridBagConstraints.BOTH;
        gbc.insets = new Insets(5, 5, 5, 5);

        for (int i = 0; i < buttons.length; i++) {
            gbc.gridx = buttons[i][0];
            gbc.gridy = buttons[i][1];
            co.add(new JButton("Button " + buttonLabels[i]), gbc);
        }

        gbc.insets = new Insets(10, 5, 5, 5);
        gbc.ipady = 20;
        gbc.gridx = 0;
        gbc.gridy = 2;
        gbc.gridwidth = 3;
        gbc.gridheight = 3;
        co.add(new JButton("Button Five"), gbc);

        setVisible(true);
        setSize(500, 500);
        setTitle("Grid Bag Layout");
    }

    public static void main(String[] args) {
        new GridBagLayoutDemo1();
    }
}
```



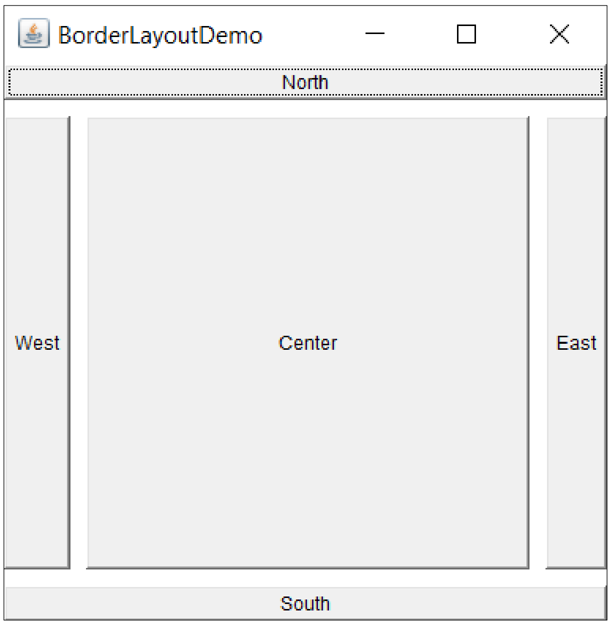
```java
import java.util.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class GridBagLayoutDemo1 extends JFrame {
    int buttons[][] = { { 0, 0 }, { 1, 0 }, { 0, 1 }, { 1, 1 } };
    String buttonLabels[] = { "One", "Two", "Three", "Four" };

    GridBagLayoutDemo1() {
        Container co = getContentPane();

        GridBagConstraints gbc = new GridBagConstraints();
        setLayout(new GridBagLayout());

        gbc.fill = GridBagConstraints.BOTH;
        gbc.insets = new Insets(5, 5, 5, 5);

        for (int i = 0; i < buttons.length; i++) {
            gbc.gridx = buttons[i][0];
            gbc.gridy = buttons[i][1];
            co.add(new JButton("Button " + buttonLabels[i]), gbc);
        }

        gbc.insets = new Insets(10, 5, 5, 5);
        gbc.ipady = 20;
        gbc.gridx = 0;
        gbc.gridy = 2;
        gbc.gridwidth = 3;
        gbc.gridheight = 3;
        co.add(new JButton("Button Five"), gbc);

        setVisible(true);
        setSize(500, 500);
        setTitle("Grid Bag Layout");
    }

    public static void main(String[] args) {
        new GridBagLayoutDemo1();
    }
}
```

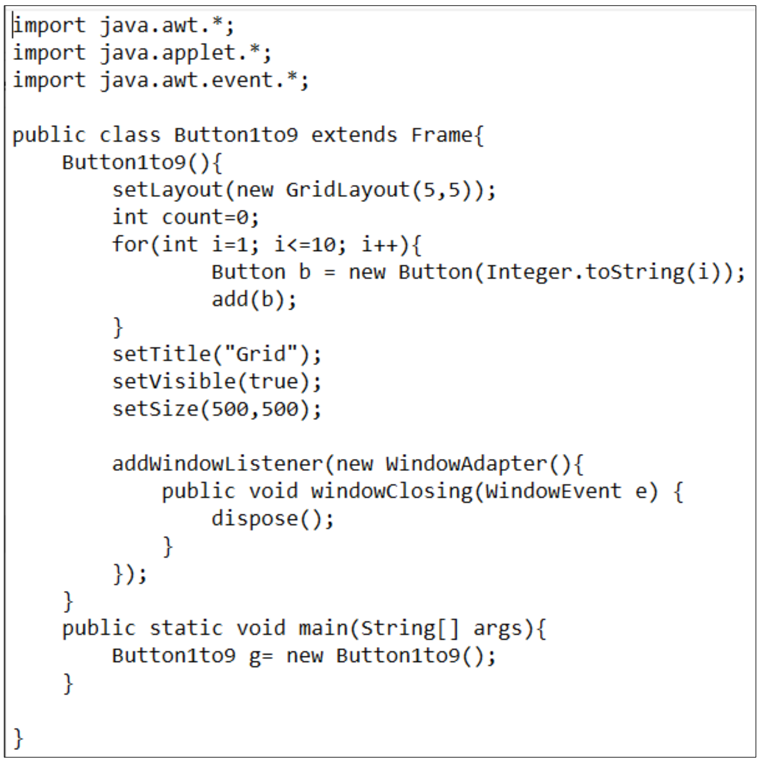


```
import java.awt.*;
import java.awt.event.*;

class GridLayoutDemo extends Frame{
    GridLayoutDemo(){
        setLayout(new GridLayout(3,2,10,10));
        int n=5;
        for(int i=1; i<=n; i++){
                Button b=new Button("Button "+Integer.toString(i));
                add(b);
        }


        setTitle("GridLayoutDemo");
        setVisible(true);
        setSize(400,200);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }

    public static void main(String[] args){
        GridLayoutDemo gd=new GridLayoutDemo();
    }
}
```

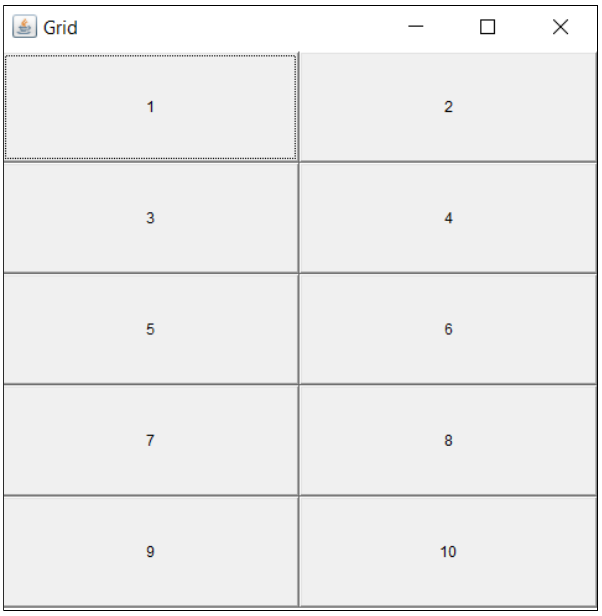


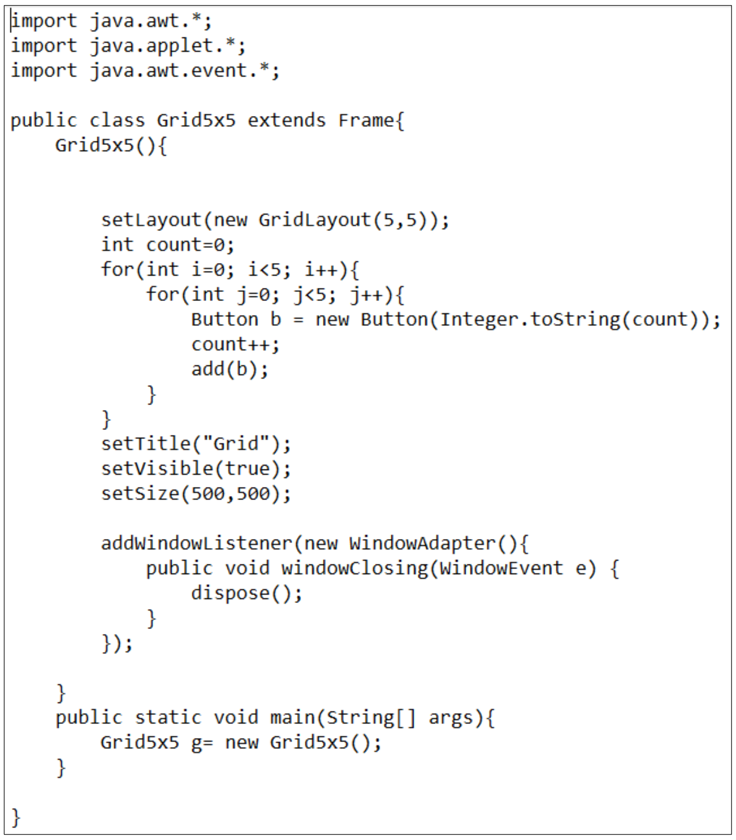
```
import java.awt.*;
import java.awt.event.*;

class BorderLayoutDemo extends Frame{
    BorderLayoutDemo(){
        setLayout(new BorderLayout(10,10));

        Button north=new Button("North");
        Button south=new Button("South");
        Button east=new Button("East");
        Button west=new Button("West");
        Button center=new Button("Center");

        add(north, BorderLayout.NORTH);
        add(south, BorderLayout.SOUTH);
        add(east, BorderLayout.EAST);
        add(west, BorderLayout.WEST);
        add(center, BorderLayout.CENTER);

        setTitle("BorderLayoutDemo");
        setVisible(true);
        setSize(400,400);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        BorderLayoutDemo bd=new BorderLayoutDemo();
    }
}
```



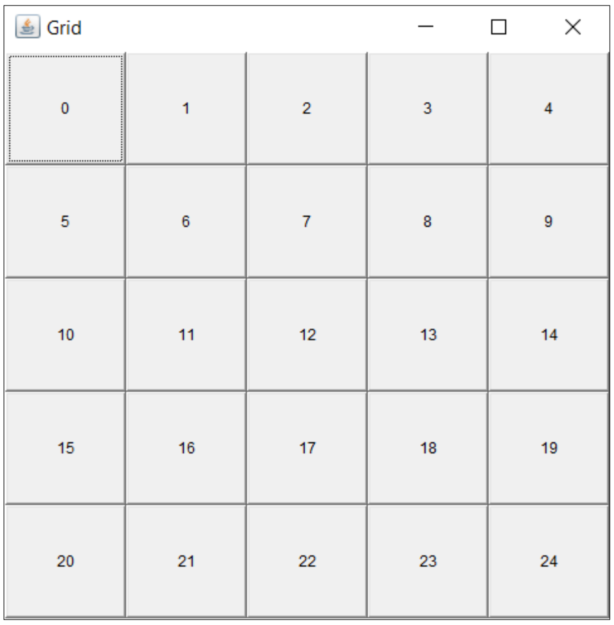


```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class Button1to9 extends Frame{
    Button1to9(){
        setLayout(new GridLayout(5,5));
        int count=0;
        for(int i=1; i<=10; i++){
                Button b = new Button(Integer.toString(i));
                add(b);
        }
        setTitle("Grid");
        setVisible(true);
        setSize(500,500);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        Button1to9 g= new Button1to9();
    }

}
```

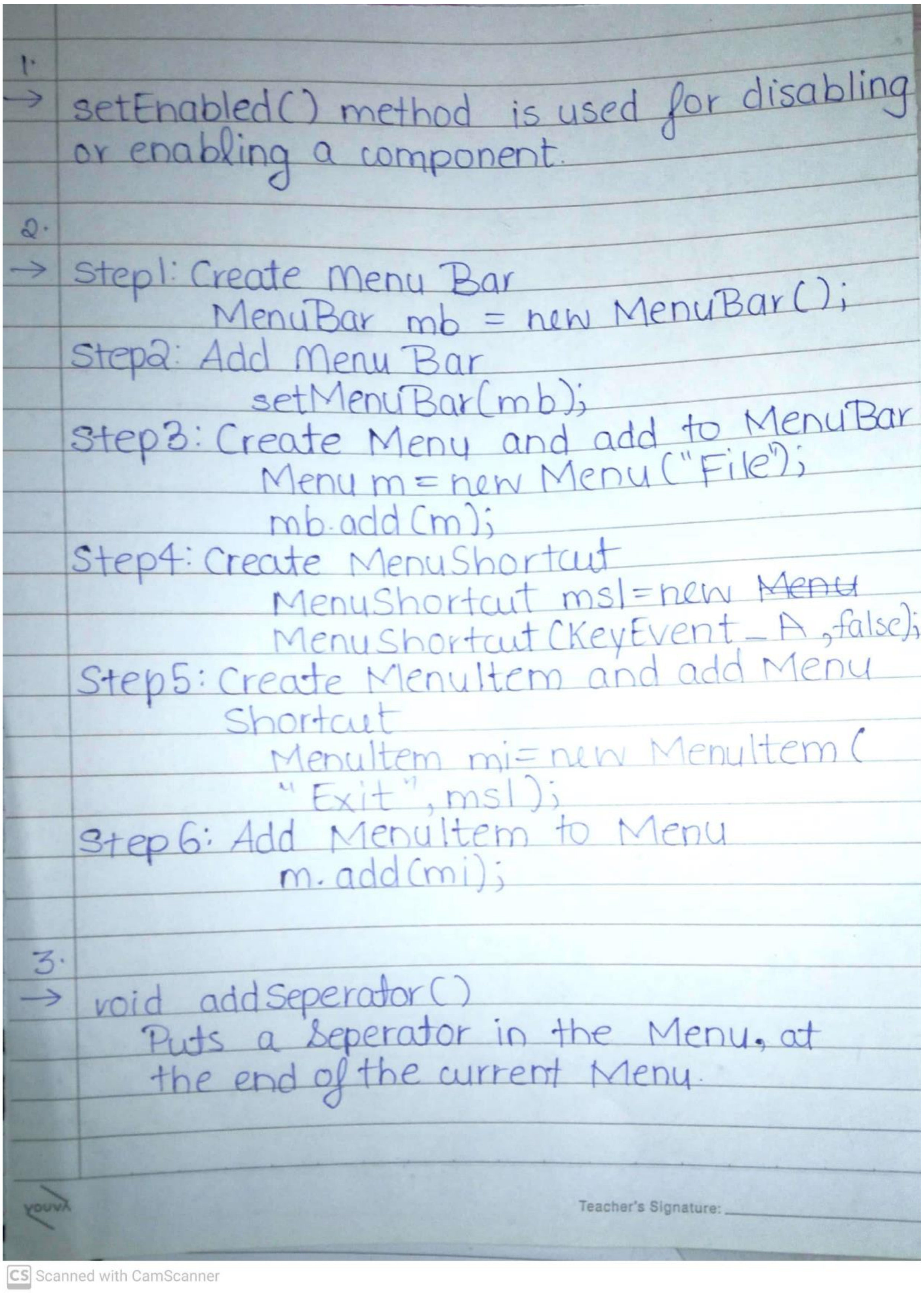




```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class Grid5x5 extends Frame{
    Grid5x5(){


        setLayout(new GridLayout(5,5));
        int count=0;
        for(int i=0; i<5; i++){
            for(int j=0; j<5; j++){
                Button b = new Button(Integer.toString(count));
                count++;
                add(b);
            }
        }
        setTitle("Grid");
        setVisible(true);
        setSize(500,500);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });

    }
    public static void main(String[] args){
        Grid5x5 g= new Grid5x5();
    }

}
```



Experiment 5



1.

→ setEnabled() method is used for disabling or enabling a component.

2.

→ Step1: Create Menu Bar

```
        MenuBar mb = new MenuBar();
```

Step2: Add Menu Bar

```
        setMenuBar(mb);
```

Step3: Create Menu and add to MenuBar

```
        Menu m=new Menu("File");
        mb.add(m);
```

Step4: Create MenuShortcut

```
        MenuShortcut ms1=new ~~Menu~~
        MenuShortcut(KeyEvent_A ,false);
```

Step5: Create MenuItem and add Menu Shortcut

```
        MenuItem mi=new MenuItem(
        "Exit", ms1);
```

Step6: Add MenuItem to Menu

```
        m.add(mi);
```

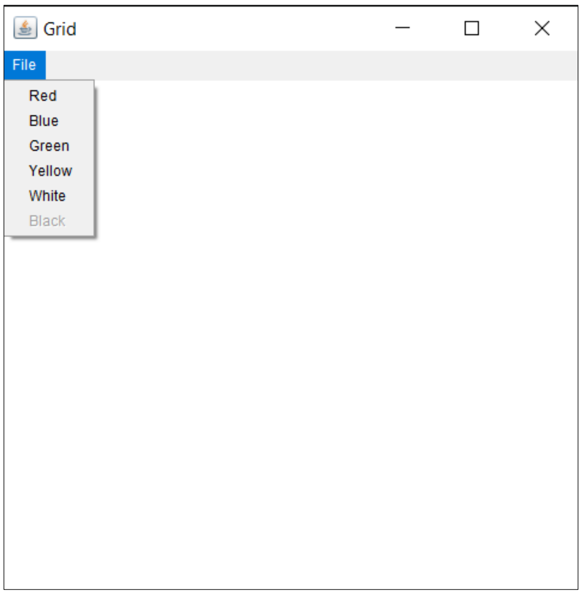
3.

→ void addSeperator()

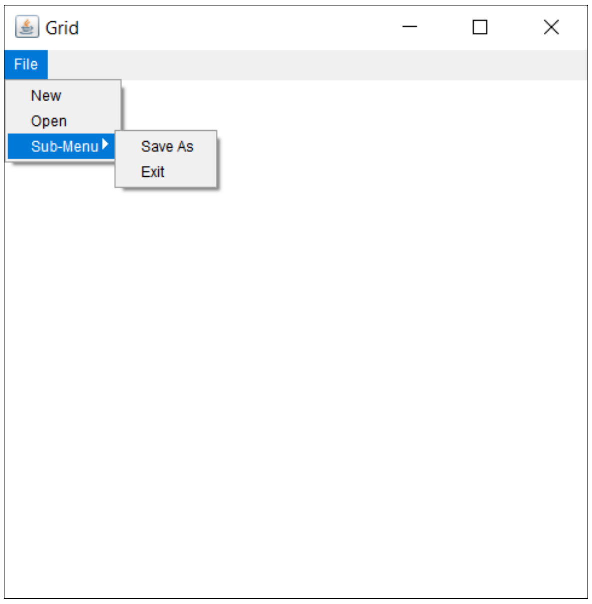
Puts a Seperator in the Menu, at the end of the current Menu.

Teacher's Signature: ____________

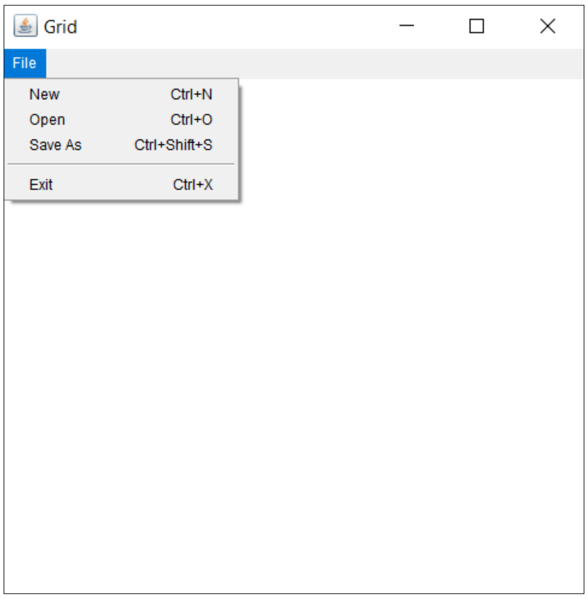
```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class ColorMenu extends Frame{
    ColorMenu(){
        setLayout(new GridLayout(5,5));
        MenuBar mb = new MenuBar();
        setMenuBar(mb);
        Menu file=new Menu("File");
        String colors[]=
{"Red","Blue","Green","Yellow","White","Black"};
        for(String color:colors){
            MenuItem mi=new MenuItem(color);
            if(color=="Black"){
                file.add(mi);
                mi.setEnabled(false);
            }
            else{
                file.add(color);
            }
        }
        mb.add(file);
        setTitle("Grid");
        setVisible(true);
        setSize(500,500);
        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        ColorMenu g= new ColorMenu();
    }

}
```



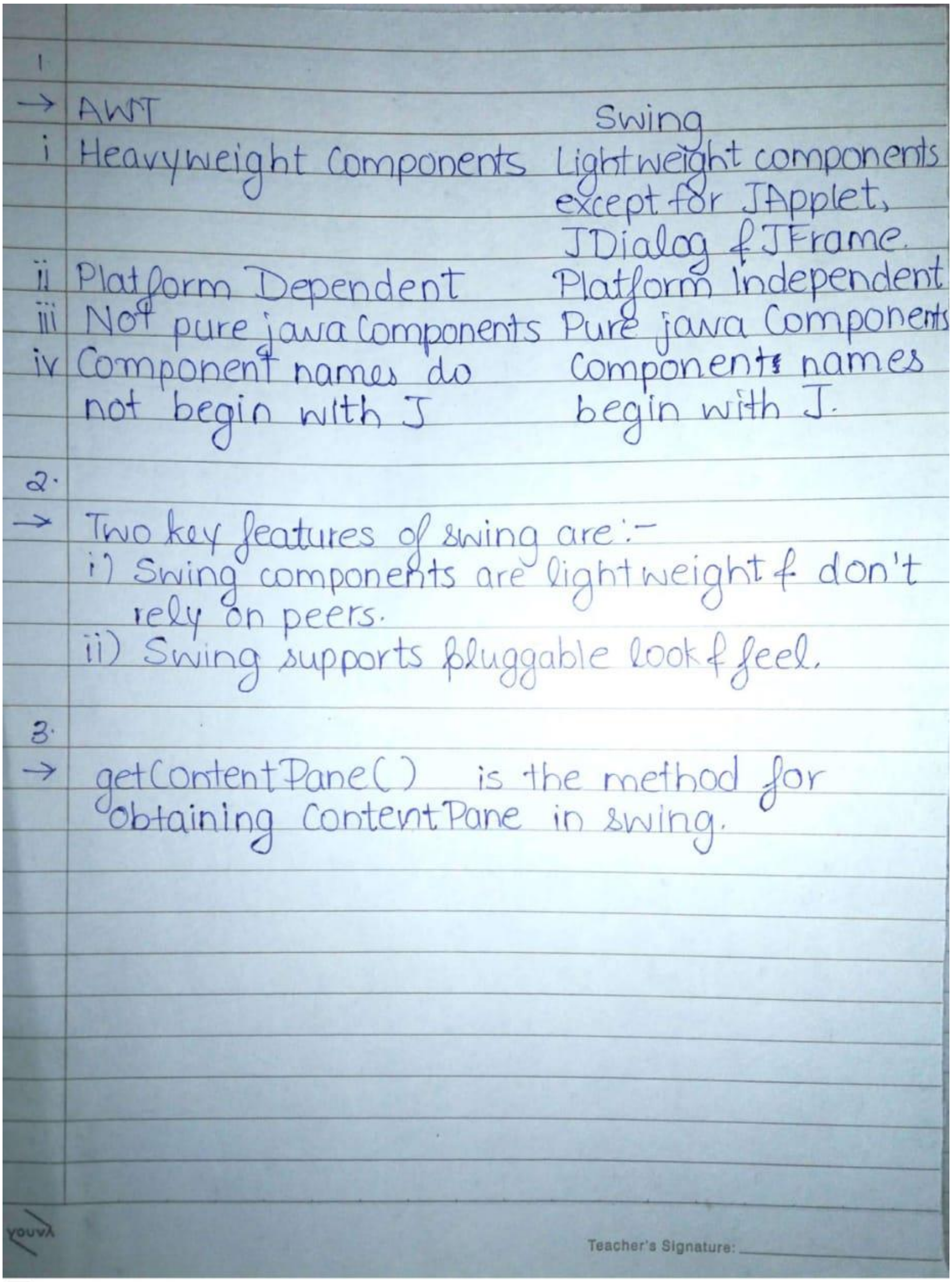
```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class MenuDemo extends Frame{
    MenuDemo(){
        setLayout(new GridLayout(5,5));
        MenuBar mb = new MenuBar();
        setMenuBar(mb);
        Menu file=new Menu("File");
        Menu sub=new Menu("Sub-Menu");
        MenuItem m1=new MenuItem("New");
        MenuItem m2=new MenuItem("Open");
        MenuItem subm1=new MenuItem("Save As");
        MenuItem subm2=new MenuItem("Exit");
        mb.add(file);
        file.add(m1);
        file.add(m2);
        file.add(sub);
        sub.add(subm1);
        sub.add(subm2);
        setTitle("Grid");
        setVisible(true);
        setSize(500,500);
        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });
    }
    public static void main(String[] args){
        MenuDemo g= new MenuDemo();
    }

}
```



```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
public class AnotherMenu extends Frame{
    AnotherMenu(){
        setLayout(new GridLayout(5,5));
        MenuBar mb = new MenuBar();
        setMenuBar(mb);
        Menu file=new Menu("File");
        MenuShortcut ms1=new MenuShortcut(KeyEvent.VK_X);
        MenuShortcut ms2=new MenuShortcut(KeyEvent.VK_S,true);
        MenuShortcut ms3=new MenuShortcut(KeyEvent.VK_O);
        MenuShortcut ms4=new MenuShortcut(KeyEvent.VK_N);

        MenuItem m1=new MenuItem("New",ms4);
        MenuItem m2=new MenuItem("Open",ms3);
        MenuItem m3=new MenuItem("Save As",ms2);
        MenuItem m4=new MenuItem("Exit",ms1);

        mb.add(file);
        file.add(m1);
        file.add(m2);
        file.add(m3);
        file.addSeparator();
        file.add(m4);

        setTitle("Grid");
        setVisible(true);
        setSize(500,500);

        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                dispose();
            }
        });

    }
    public static void main(String[] args){
        AnotherMenu g= new AnotherMenu();
    }
}
```



Experiment 6



1

→

	AWT	Swing
i	Heavyweight Components	Lightweight components except for JApplet, JDialog & JFrame.
ii	Platform Dependent	Platform Independent
iii	Not pure java Components	Pure java Components
iv	Component names do not begin with J	Components names begin with J.

2.

→ Two key features of swing are:-

i) Swing components are lightweight & don't rely on peers.

ii) Swing supports pluggable look & feel.

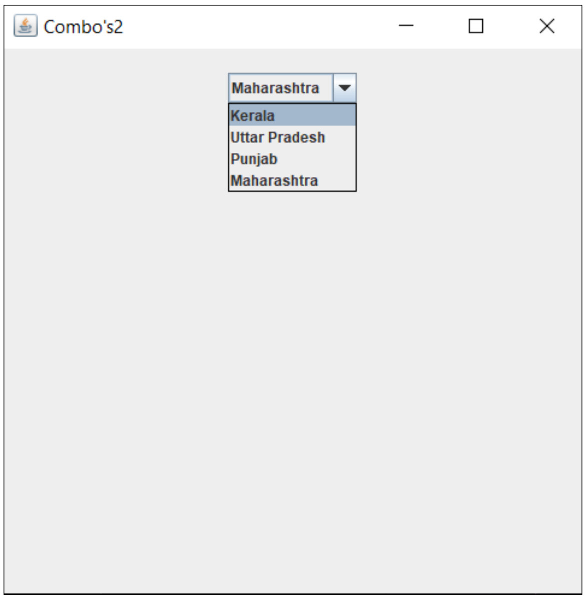
3.

→ getContentPane() is the method for obtaining ContentPane in swing.

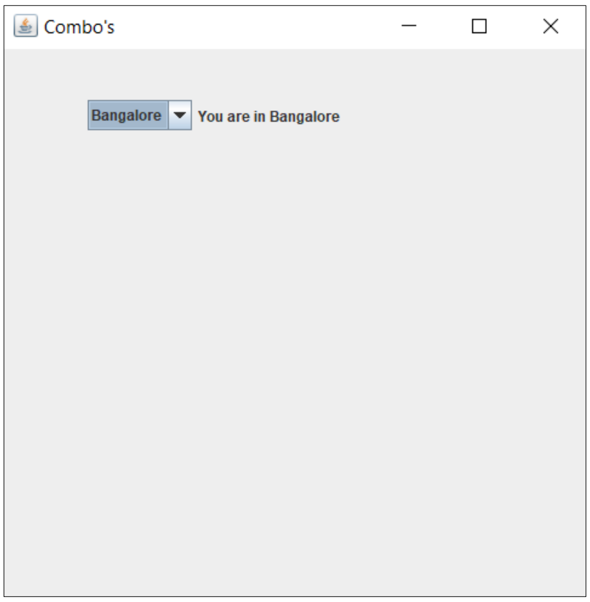
Teacher's Signature: ______

Scanned with CamScanner

```
import java.awt.*;
import java.util.*;
import javax.swing.*;

public class AnotherCombo extends JFrame{
// Declaration
    JComboBox jcb1,jcb2;

    AnotherCombo(){
    // Data Variables
        String[] subjects={"Kerala","Uttar
Pradesh","Punjab","Maharashtra"};


    // Get Container
        Container co= getContentPane();

    //  Create and Add Components
        jcb1 = new JComboBox(subjects);

    // Add Components to Container
        co.add(jcb1);

    // Customize Container
        co.setLayout(new FlowLayout(FlowLayout.CENTER,20,20));

        setVisible(true);
        setSize(500,500);
        setTitle("Combo's2");
    }
    public static void main(String[] args){
        AnotherCombo sd=new AnotherCombo();
    }

}
```

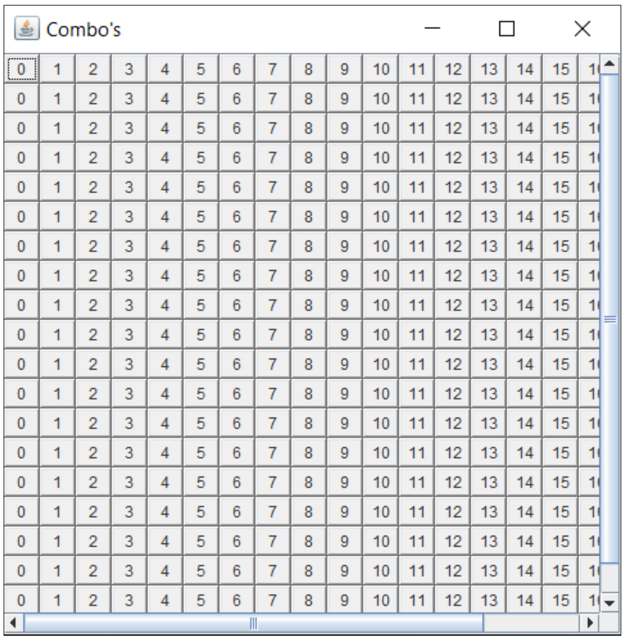


```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;

public class ComboDemo extends JFrame implements ItemListener{
    JComboBox jcb1,jcb2;
    JLabel l;
    ComboDemo(){
        String[] subjects={"Solapur","Pune","Mumbai","Bangalore"};
        Container co= getContentPane();
        jcb1 = new JComboBox(subjects);
        jcb1.addItemListener(this);
        l= new JLabel();
        l.setPreferredSize(new Dimension(250,100));
        co.add(jcb1);
        co.add(l);
        co.setLayout(new FlowLayout());
        setVisible(true);
        setSize(500,500);
        setTitle("Combo's");
    }

    public void itemStateChanged(ItemEvent e){
        l.setText("You are in "+jcb1.getSelectedItem());
    }

    public static void main(String[] args){
        ComboDemo sd=new ComboDemo();
    }

}
```

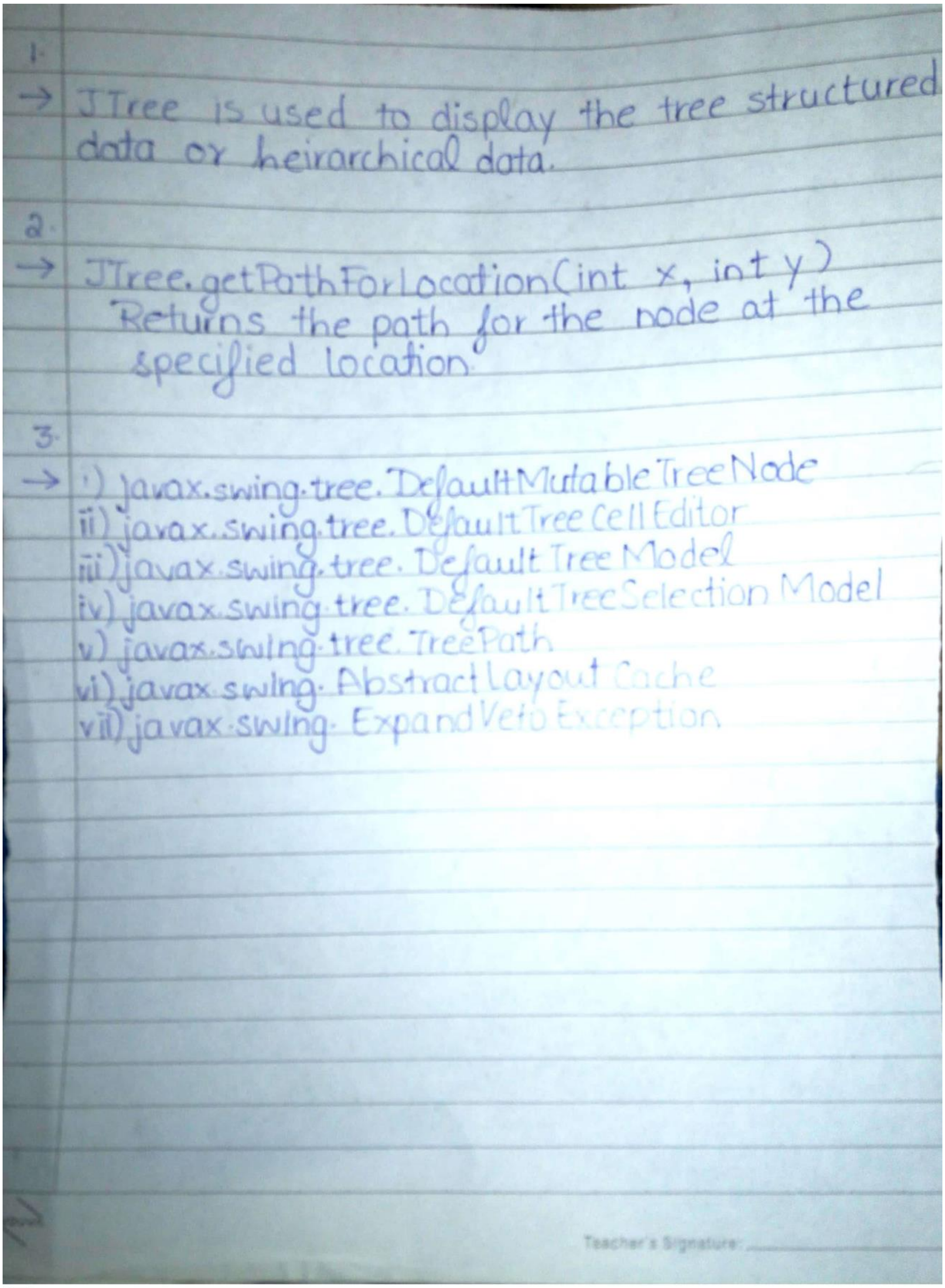


You are in Bangalore

```
import java.awt.*;
import java.util.*;
import javax.swing.*;

public class ScrollDemo extends JFrame{
    int vsb=ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
hsb=ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS;
    JComponent jc;
    JPanel jpanel;
    JScrollPane jsp;
    ScrollDemo(){
        Container co= getContentPane();

        jpanel = new JPanel();
        jpanel.setLayout(new GridLayout(20,20));

        jsp=new JScrollPane(jpanel,vsb,hsb);

        int val=1;
        for(int i=0; i<20;i++){
             for(int j=0;j<20;j++){
                 jpanel.add(new Button(Integer.toString(j)));
             }
        }
        co.add(jsp);
        co.setLayout(new GridLayout());
        setVisible(true);
        setSize(500,500);
        setTitle("Combo's");

    }
    public static void main(String[] args){
        ScrollDemo sd = new ScrollDemo();
    }

}
```



Experiment 7



1.

→ JTree is used to display the tree structured data or heirarchical data.

2.

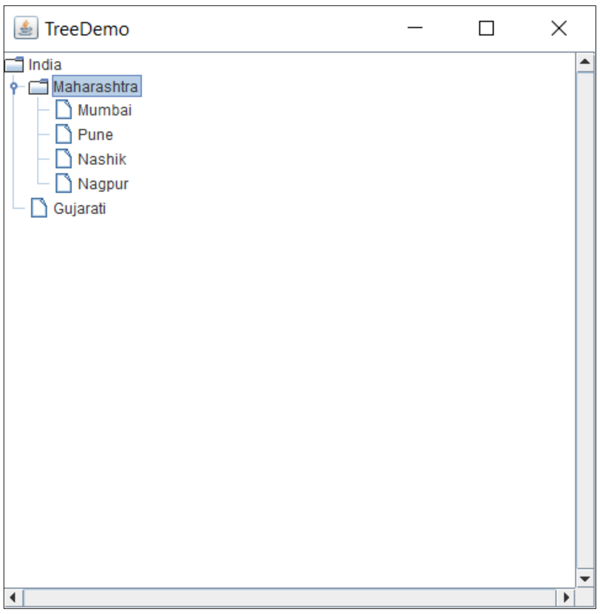
→ JTree.getPathForLocation(int x, int y)
Returns the path for the node at the specified location.

3.

→ i) javax.swing.tree.DefaultMutableTreeNode
ii) javax.swing.tree.DefaultTreeCellEditor
iii) javax.swing.tree.DefaultTreeModel
iv) javax.swing.tree.DefaultTreeSelectionModel
v) javax.swing.tree.TreePath
vi) javax.swing.AbstractLayoutCache
vii) javax.swing.ExpandVetoException

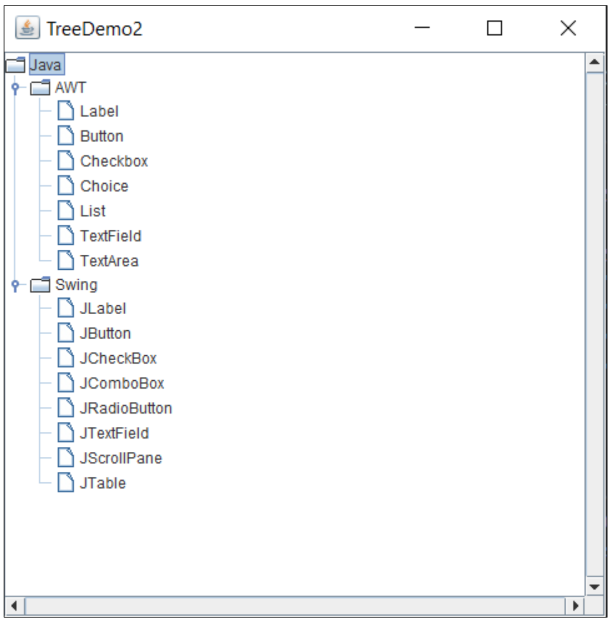
Teacher's Signature: ________

```
import javax.swing.*;
import java.awt.*;
import javax.swing.tree.*;
import java.util.*;

public class TreeDemo extends JFrame{
    int vsb=ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
hsb=ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS;
    Color teal= new Color(0,128,128);
    JTree jt;
    JScrollPane jsp;

    TreeDemo(){
        String states[]={"Maharashtra"};
        String cities[][]={{"Mumbai","Pune","Nashik","Nagpur"}};
        Container co=getContentPane();
        setLayout(new GridLayout(1,1));
            DefaultMutableTreeNode root=new DefaultMutableTreeNode("India");
            for (int i=0;i<states.length;i++){
                DefaultMutableTreeNode dmt=new DefaultMutableTreeNode(states[i]);
                root.add(dmt);
                for(int j=0; j<cities[i].length;j++){
                    dmt.add(new DefaultMutableTreeNode(cities[i][j]));
                }
            }
            root.add(new DefaultMutableTreeNode("Gujarati"));
        jt=new JTree(root);
        jsp=new JScrollPane(jt,vsb,hsb);

        co.add(jsp);
        co.setBackground(teal);

        setVisible(true);
        setSize(500,500);
        setTitle("TreeDemo");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }
    public static void main(String[] args){
        TreeDemo td=new TreeDemo();
    }
}
```

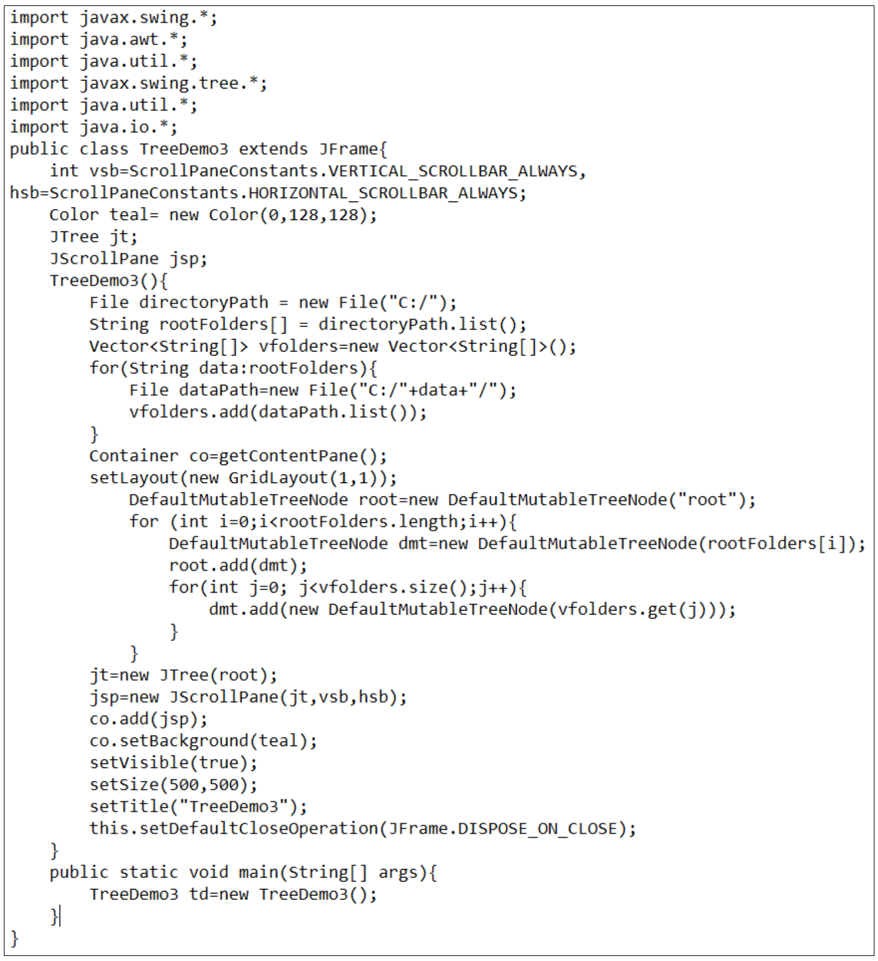


```
TreeDemo
India
  Maharashtra
    Mumbai
    Pune
    Nashik
    Nagpur
  Gujarati
```

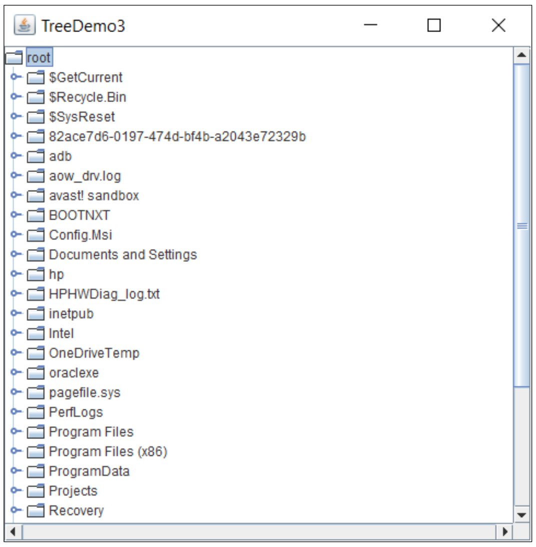
```
import javax.swing.*;
import java.awt.*;
import java.util.*;
import javax.swing.tree.*;
import java.util.*;

public class TreeDemo2 extends JFrame{
    int vsb=ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
hsb=ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS;
    Color teal= new Color(0,128,128);
    JTree jt;
    JScrollPane jsp;

    TreeDemo2(){
        String[] javaClasses={"AWT","Swing"};
        String[][] data=
{{"Label","Button","Checkbox","Choice","List","TextField","TextArea"},
{"JLabel","JButton","JCheckBox","JComboBox","JRadioButton","JTextField",
"JScrollPane","JTable"}};
        Container co=getContentPane();
        setLayout(new GridLayout(1,1));
            DefaultMutableTreeNode root=new DefaultMutableTreeNode("Java");
            for (int i=0;i<javaClasses.length;i++){
                DefaultMutableTreeNode dmt=new DefaultMutableTreeNode(javaClasses[i]);
                root.add(dmt);
                for(int j=0; j<data[i].length;j++){
                    dmt.add(new DefaultMutableTreeNode(data[i][j]));
                }
            }
        jt=new JTree(root);
        jsp=new JScrollPane(jt,vsb,hsb);
        co.add(jsp);
        co.setBackground(teal);
        setVisible(true);
        setSize(500,500);
        setTitle("TreeDemo2");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }
    public static void main(String[] args){
        TreeDemo2 td=new TreeDemo2();
    }

}
```



```
TreeDemo2
Java
  AWT
    Label
    Button
    Checkbox
    Choice
    List
    TextField
    TextArea
  Swing
    JLabel
    JButton
    JCheckBox
    JComboBox
    JRadioButton
    JTextField
    JScrollPane
    JTable
```

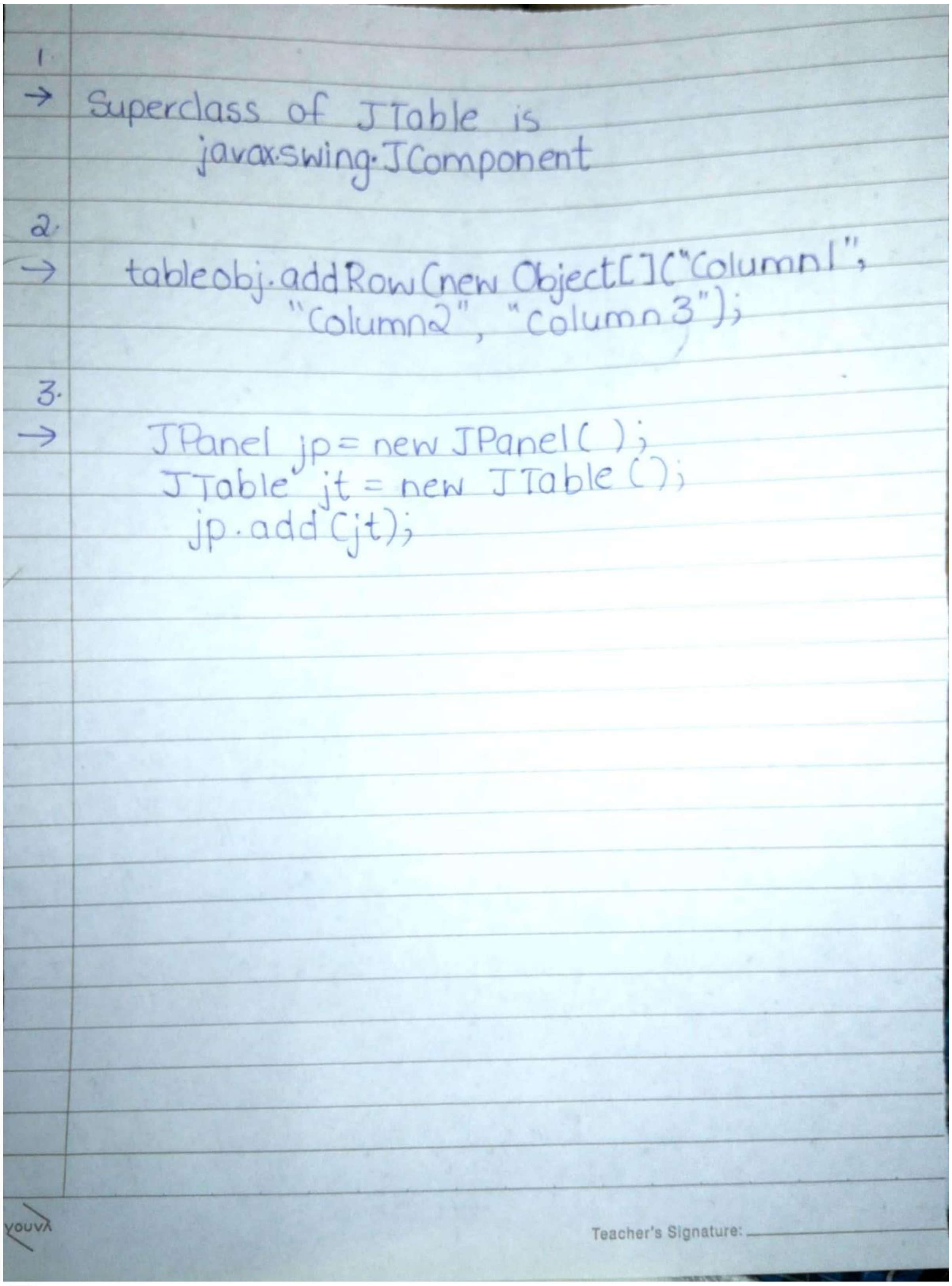


```java
import javax.swing.*;
import java.awt.*;
import java.util.*;
import javax.swing.tree.*;
import java.util.*;
import java.io.*;
public class TreeDemo3 extends JFrame{
    int vsb=ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS,
hsb=ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS;
    Color teal= new Color(0,128,128);
    JTree jt;
    JScrollPane jsp;
    TreeDemo3(){
        File directoryPath = new File("C:/");
        String rootFolders[] = directoryPath.list();
        Vector<String[]> vfolders=new Vector<String[]>();
        for(String data:rootFolders){
            File dataPath=new File("C:/"+data+"/");
            vfolders.add(dataPath.list());
        }
        Container co=getContentPane();
        setLayout(new GridLayout(1,1));
            DefaultMutableTreeNode root=new DefaultMutableTreeNode("root");
            for (int i=0;i<rootFolders.length;i++){
                DefaultMutableTreeNode dmt=new DefaultMutableTreeNode(rootFolders[i]);
                root.add(dmt);
                for(int j=0; j<vfolders.size();j++){
                    dmt.add(new DefaultMutableTreeNode(vfolders.get(j)));
                }
            }
        jt=new JTree(root);
        jsp=new JScrollPane(jt,vsb,hsb);
        co.add(jsp);
        co.setBackground(teal);
        setVisible(true);
        setSize(500,500);
        setTitle("TreeDemo3");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }
    public static void main(String[] args){
        TreeDemo3 td=new TreeDemo3();
    }
}
```



```
TreeDemo3
root
  $GetCurrent
  $Recycle.Bin
  $SysReset
  82ace7d6-0197-474d-bf4b-a2043e72329b
  adb
  aow_drv.log
  avast! sandbox
  BOOTNXT
  Config.Msi
  Documents and Settings
  hp
  HPHWDiag_log.txt
  inetpub
  Intel
  OneDriveTemp
  oraclexe
  pagefile.sys
  PerfLogs
  Program Files
  Program Files (x86)
  ProgramData
  Projects
  Recovery
```

Experiment 8



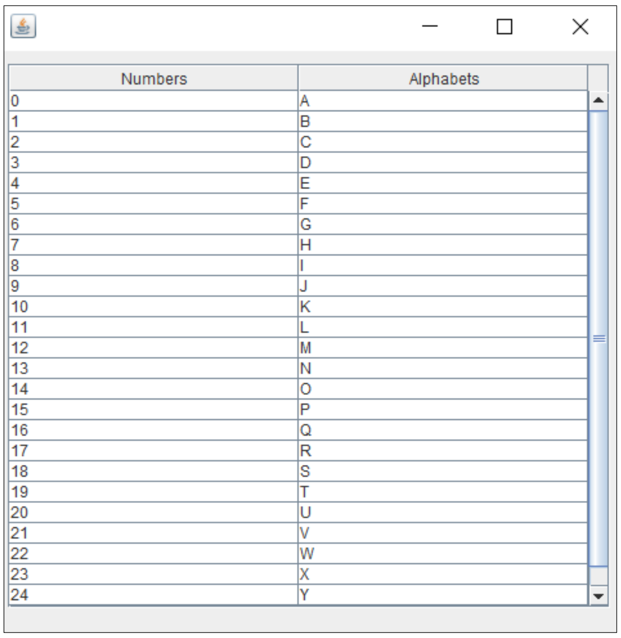
1.
→ Superclass of JTable is
javax.swing.JComponent

2.
→
```
tableobj.addRow(new Object[]("Column1",
        "Column2", "column3");
```

3.
→
```
JPanel jp= new JPanel( );
 JTable jt = new JTable ();
  jp.add (jt);
```

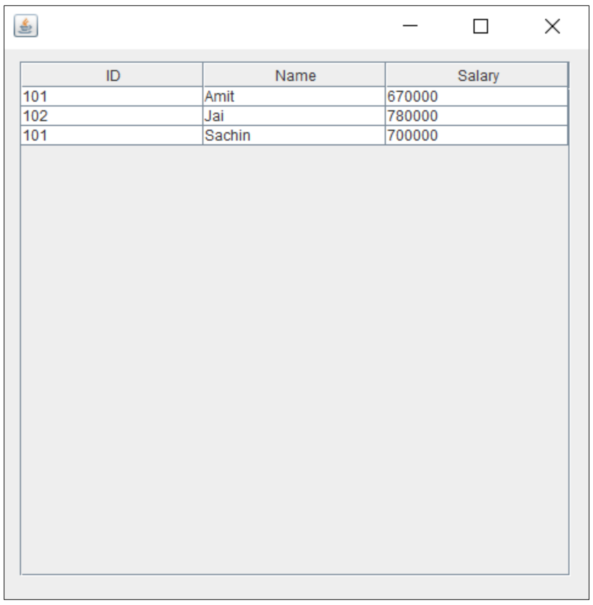
Teacher's Signature:

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.tree.*;
import java.util.*;

class TableDemo extends JFrame {
    JTable jt;
    JScrollPane js;

    TableDemo() {
        String colHeads[] = { "Numbers", "Alphabets" };
        String data[][] = new String[26][2];
        for (int i = 1; i < 26; i++) {
            char c = (char) (i + 64);
            data[i][0] = Integer.toString(i);
            data[i][1] = Character.toString(c);
        }
        Container co = getContentPane();
        setLayout(new FlowLayout(FlowLayout.CENTER, 10, 10));
        jt = new JTable(data, colHeads);
        js = new JScrollPane(jt);
        co.add(js);
        setVisible(true);
        setSize(500, 500);
    }

    public static void main(String[] args) {
        new TableDemo();
    }
}
```



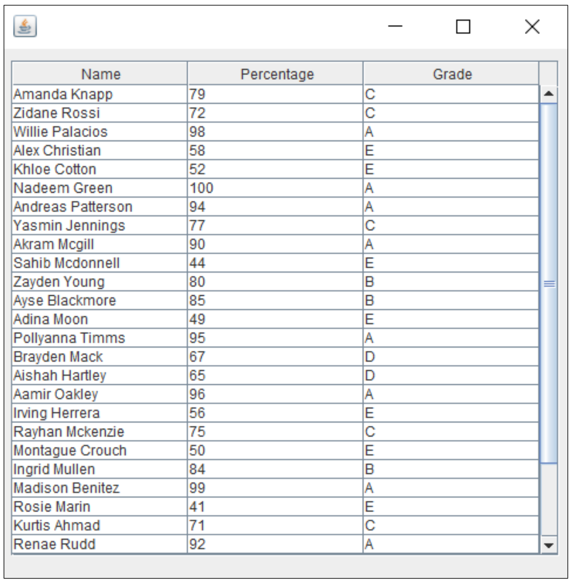
Numbers	Alphabets
0	A
1	B
2	C
3	D
4	E
5	F
6	G
7	H
8	I
9	J
10	K
11	L
12	M
13	N
14	O
15	P
16	Q
17	R
18	S
19	T
20	U
21	V
22	W
23	X
24	Y

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.tree.*;
import java.util.*;
class TableDemo extends JFrame{
    JTable jt;
    JScrollPane js;
    TableDemo(){
        String colHeads[]={"ID","Name","Salary"};
        String data[][]={{"101","Amit","670000"},{"102","Jai","780000"},
{"101","Sachin","700000"}};
        Container co=getContentPane();
        setLayout(new FlowLayout(FlowLayout.CENTER, 10,10));
        jt=new JTable(data,colHeads);
        js=new JScrollPane(jt);
        co.add(js);
        setVisible(true);
        setSize(500,500);
    }
    public static void main(String[] args){
        TableDemo sd=
        new TableDemo();
    }
}
```



ID	Name	Salary
101	Amit	670000
102	Jai	780000
101	Sachin	700000

```
import java.awt.*;
import javax.swing.*;
class TableDemo2 extends JFrame{
    JTable jt;
    JScrollPane js;
    TableDemo2(){
        String colHeads[]={"Name","Percentage","Grade"};
        String data[][]={{"Amanda Knapp","79","C"},{"Zidane Rossi","72","C"}
,{"Willie Palacios","98","A"},{"Alex Christian","58","E"},
{"Khloe Cotton","52","E"},{"Nadeem Green","100","A"},
{"Andreas Patterson","94","A"},{"Yasmin Jennings","77","C"},
{"Akram Mcgill","90","A"},{"Sahib Mcdonnell","44","E"},
{"Zayden Young","80","B"},{"Ayse Blackmore","85","B"},
{"Adina Moon","49","E"},{"Pollyanna Timms","95","A"},
{"Brayden Mack","67","D"},{"Aishah Hartley","65","D"},
{"Aamir Oakley","96","A"},{"Irving Herrera","56","E"},
{"Rayhan Mckenzie","75","C"},{"Montague Crouch","50","E"},
{"Ingrid Mullen","84","B"},{"Madison Benitez","99","A"},
{"Rosie Marin","41","E"},{"Kurtis Ahmad","71","C"},{"Renae Rudd","92","A"},{"Jo
Moyer","51","E"},{"Rupert Roche","93","A"},{"Arwen Whitehead","46","E"},{"Lance
Curran","89","B"},{"Rhianna Driscoll","55","E"}};
        Container co=getContentPane();
        setLayout(new FlowLayout(FlowLayout.CENTER, 10,10));
        jt=new JTable(data,colHeads);
        js=new JScrollPane(jt);
        co.add(js);
        setVisible(true);
        setSize(500,500);
    }
    public static void main(String[] args){
        TableDemo2 sd=
        new TableDemo2();
    }
}
```



Name	Percentage	Grade
Amanda Knapp	79	C
Zidane Rossi	72	C
Willie Palacios	98	A
Alex Christian	58	E
Khloe Cotton	52	E
Nadeem Green	100	A
Andreas Patterson	94	A
Yasmin Jennings	77	C
Akram Mcgill	90	A
Sahib Mcdonnell	44	E
Zayden Young	80	B
Ayse Blackmore	85	B
Adina Moon	49	E
Pollyanna Timms	95	A
Brayden Mack	67	D
Aishah Hartley	65	D
Aamir Oakley	96	A
Irving Herrera	56	E
Rayhan Mckenzie	75	C
Montague Crouch	50	E
Ingrid Mullen	84	B
Madison Benitez	99	A
Rosie Marin	41	E
Kurtis Ahmad	71	C
Renae Rudd	92	A

Experiment 9

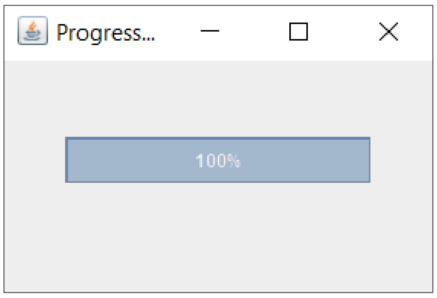
1.
→ Different orientations in Progress Bar :-
~~Swing Constants. Vertica~~
i) Swing Constants. VERTICAL
ii) Swing Constants. HORIZONTAL

2.
→ sets the progress bars current value.

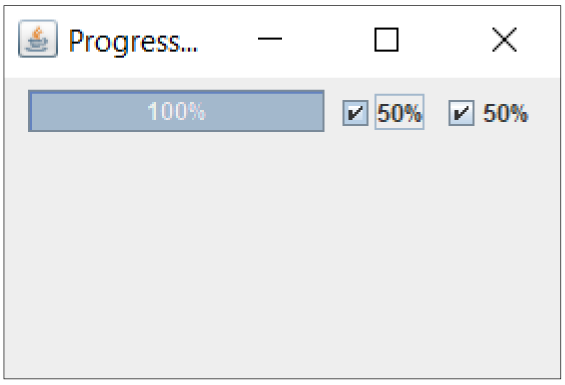
3.
→ i) The min value ~~from~~ of progress bar is the least value you can set ~~in~~ for the progress bar.
ii) The max value of progress bar is the maximum value you can set for the progress bar.

Teacher's Signature: __________

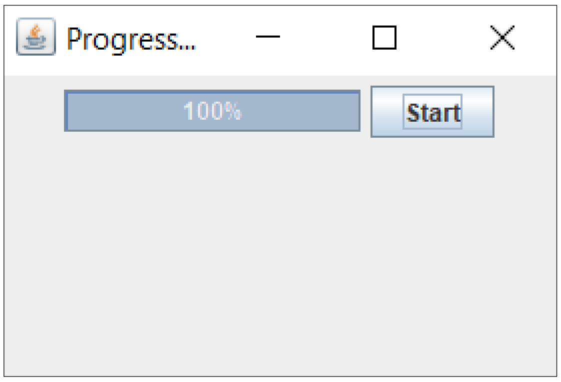
```java
import java.util.*;
import java.awt.*;
import javax.swing.*;
public class ProgressDemo extends JFrame{
    JProgressBar jp;
    void ProgressBarTimer(){
        try{
            int i=0;
            while(i<=100){
                jp.setValue(i);
                Thread.sleep(500);
                i+=20;
            }
        }
        catch(Exception e){
            System.out.println(e.getMessage());
        }
    }
    ProgressDemo(){
        Container co=getContentPane();
        setLayout(null);
        jp=new JProgressBar(0,100);
        jp.setStringPainted(true);
        jp.setBounds(40,50,200,30);
        co.add(jp);
        setVisible(true);
        setSize(300,200);
        setTitle("ProgressDemo");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }
    public static void main(String[] args){
        ProgressDemo td=new ProgressDemo();
        td.ProgressBarTimer();
    }

}
```



```java
import java.util.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class ProgressOnCheck extends JFrame implements ItemListener {
    JProgressBar jp;
    JCheckBox jcb1, jcb2;
    int i = 0;
    ProgressOnCheck() {
        Container co = getContentPane();
        setLayout(new FlowLayout(FlowLayout.CENTER));
        jp = new JProgressBar(0, 100);
        jcb1 = new JCheckBox("50%");
        jcb2 = new JCheckBox("50%");
        jp.setStringPainted(true);
        jp.setBounds(40, 50, 200, 30);
        co.add(jp);
        co.add(jcb1);
        co.add(jcb2);
        jcb1.addItemListener(this);
        jcb2.addItemListener(this);
        setVisible(true);
        setSize(300, 200);
        setTitle("ProgressOnCheck");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }
    public void itemStateChanged(ItemEvent ie) {
        if (jcb1.isSelected() && jcb2.isSelected()) {
            i = 100;
        } else if (jcb1.isSelected() || jcb2.isSelected()) {
            i = 50;
        } else {
            i = 0;
        }
        jp.setValue(i);
    }
    public static void main(String[] args) {
        new ProgressOnCheck();
    }
}
```



```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class ProgressOnButton extends JFrame implements ActionListener {
    JProgressBar jp;
    JButton b1;
    void progressBarBegin() {
        try {
            int i = 0;
            while (i <= 100) {
                jp.setValue(i);
                jp.paintImmediately(0, 0, 200, 25);
                Thread.sleep(500);
                i += 20;
            }
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
    ProgressOnButton() {
        Container co = getContentPane();
        setLayout(new FlowLayout(FlowLayout.CENTER));
        jp = new JProgressBar(0, 100);
        b1 = new JButton("Start");
        jp.setStringPainted(true);
        jp.setBounds(40, 50, 200, 30);
        co.add(jp);
        co.add(b1);
        b1.addActionListener(this);
        setVisible(true);
        setSize(300, 200);
        setTitle("ProgressOnButton");
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    }
    public void actionPerformed(ActionEvent ae) {
        progressBarBegin();
    }
    public static void main(String[] args) {
        new ProgressOnButton();
    }
}
```



Experiment 10

Experiment 10

XII

1. →i) ActionListener
   ii) ItemListener
   iii) KeyListener
   iv) WindowListener

2→i) keyPressed(KeyEvent ke)
   ii) keyReleased(KeyEvent ke)
   iii) keyTyped (KeyEvent ke)

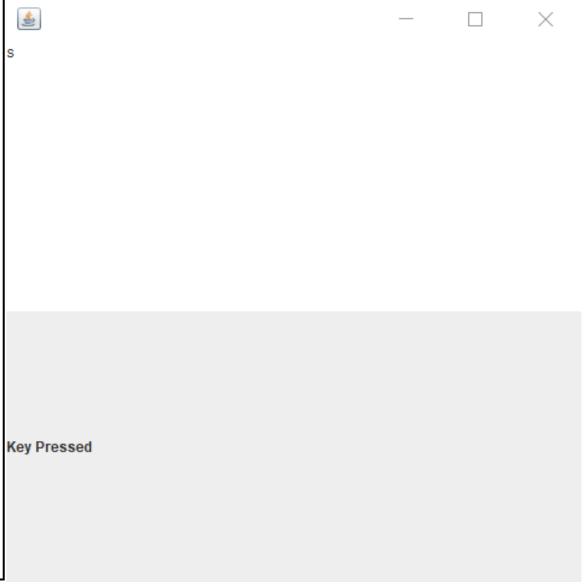
   * When a key is typed all 3methods are generated.

XIII

1. → i) Source object fires an event object when the user check or unchecks a Checkbox, if the source was a checkbox.
   ii) The event object contains information about its source object.
   iii) The Listener object is an interface that must be registered as a 'listener' by the source object to be able to respond when the event object was fired, & invoke the handler method of the listener object.

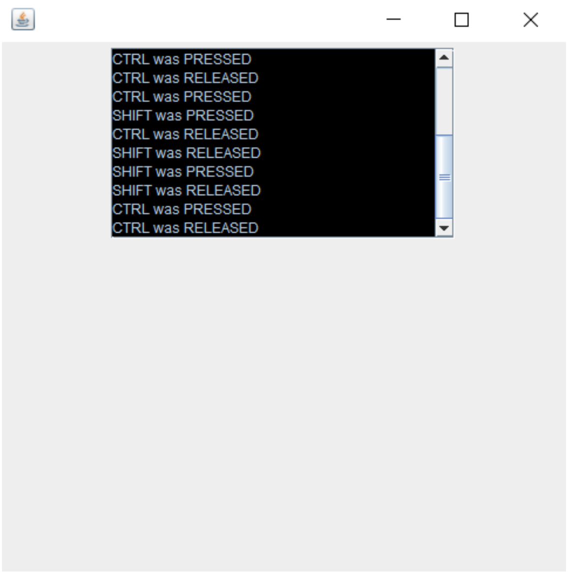
2→ public ~~void~~ abstract void
   actionPerformed (ActionEvent ae);

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class KeyEventDemo0 extends JFrame implements KeyListener {
   JLabel label;

   KeyEventDemo0() {
      Container co = getContentPane();

      label = new JLabel();
      JTextArea ta = new JTextArea();

      co.add(ta);
      co.add(label);

      ta.addKeyListener(this);

      setSize(500, 500);
      setLayout(new GridLayout(2, 1));
      setVisible(true);
   }

   public void keyPressed(KeyEvent e) {
      label.setText("Key Pressed");
   }

   public void keyReleased(KeyEvent e) {
   }

   public void keyTyped(KeyEvent e) {
   }

   public static void main(String[] args) {
      new KeyEventDemo0();
   }
}
```



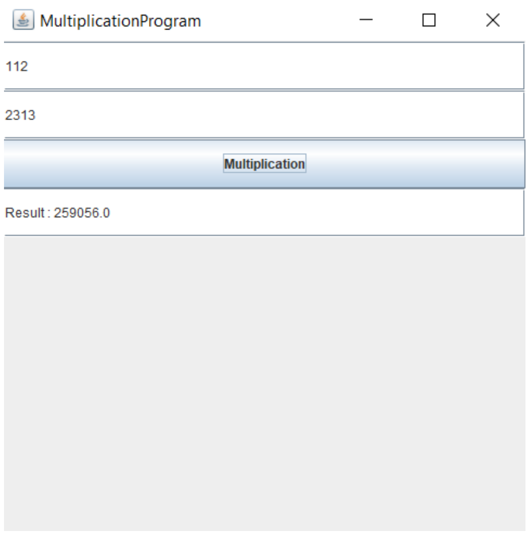
```
s

Key Pressed
```

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class KeyEventDemo1 extends JFrame implements KeyListener {
  int vsb = JScrollPane.VERTICAL_SCROLLBAR_ALWAYS;
  int hsb = JScrollPane.HORIZONTAL_SCROLLBAR_NEVER;
  Container co;
  JTextArea ta;

  KeyEventDemo1() {
    co = getContentPane();
    ta = new JTextArea(10, 25);
    ta.setEnabled(false);
    ta.setBackground(Color.black);
    JScrollPane jsp = new JScrollPane(ta, vsb, hsb);
    addKeyListener(this);
    co.add(jsp);
    setSize(500, 500);
    setLayout(new FlowLayout(FlowLayout.CENTER));
    setVisible(true);
  }
  public void keyReleased(KeyEvent ke) {
    String c;
    int k = ke.getKeyCode();
    switch (k) {
      case KeyEvent.VK_ALT: c = "ALT"; break;
      case KeyEvent.VK_CONTROL: c = "CTRL";break;
      case KeyEvent.VK_SHIFT: c = "SHIFT"; break;
      default:  c = "" + ke.getKeyChar(); break;
    }
    String str = ta.getText() + "\n" + c + " was RELEASED";
    ta.setText(str);
  }
  public void keyPressed(KeyEvent ke) {
    String c;
    int k = ke.getKeyCode();
    switch (k) {
      case KeyEvent.VK_ALT: c = "ALT"; break;
      case KeyEvent.VK_CONTROL:  c = "CTRL";  break;
      case KeyEvent.VK_SHIFT:  c = "SHIFT";  break;
      default:  c = "" + ke.getKeyChar();  break;
    }
    String str = ta.getText() + "\n" + c + " was PRESSED";
    ta.setText(str);
  }
  public void keyTyped(KeyEvent ke) {
    String c;
    int k = ke.getKeyCode();
    switch (k) {
      case KeyEvent.VK_ALT: c = "ALT";  break;
      case KeyEvent.VK_CONTROL:  c = "CTRL"; break;
      case KeyEvent.VK_SHIFT:  c = "SHIFT";  break;
      default:  c = "" + ke.getKeyChar(); break;
    }
    String str = ta.getText() + "\n" + c + " was TYPED";
    ta.setText(str);
  }

  public static void main(String[] args) {
    new KeyEventDemo1();
  }
}
```



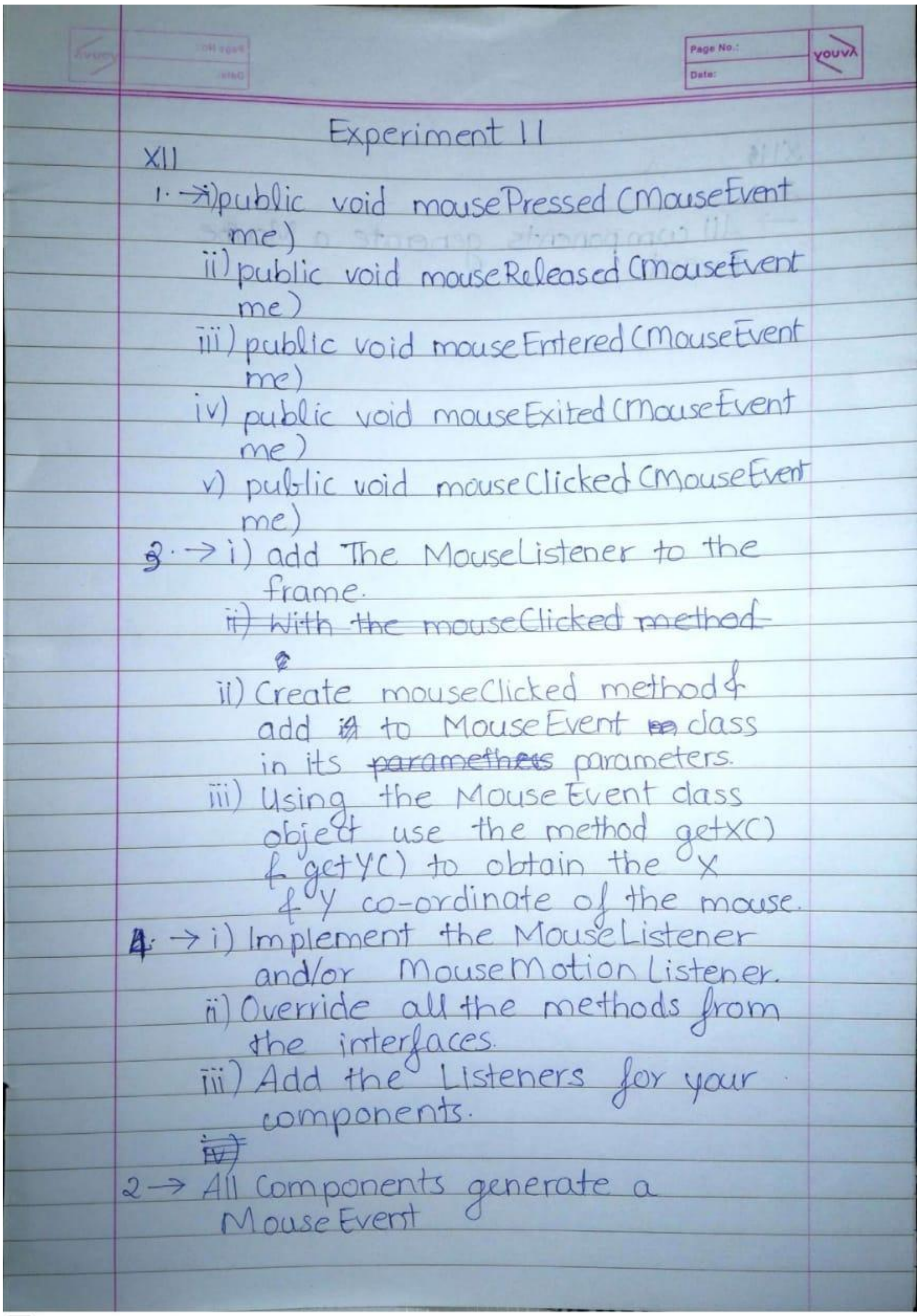
```
CTRL was PRESSED
CTRL was RELEASED
CTRL was PRESSED
SHIFT was PRESSED
CTRL was RELEASED
SHIFT was RELEASED
SHIFT was PRESSED
SHIFT was RELEASED
CTRL was PRESSED
CTRL was RELEASED
```

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class MultiplicationProgram extends JFrame implements
ActionListener {
   Container co;
   JTextField jtf1, jtf2, jtf3;
   JButton b1, b2;

   MultiplicationProgram() {
      co = getContentPane();

      b1 = new JButton("Multiplication");
      jtf1 = new JTextField();
      jtf2 = new JTextField();
      jtf3 = new JTextField();

      co.add(jtf1);
      co.add(jtf2);
      co.add(b1);
      co.add(jtf3);

      b1.addActionListener(this);

      setLayout(new GridLayout(10, 2));

      setTitle("MultiplicationProgram");
      setSize(500, 500);
      setVisible(true);
   }

   public void actionPerformed(ActionEvent ae) {
      float a = Float.parseFloat(jtf1.getText());
      float b = Float.parseFloat(jtf2.getText());
      jtf3.setText("Result : " + Float.toString((a * b)));
   }

   public static void main(String[] args) {
      new MultiplicationProgram();
   }
}
```



```
112
2313
Result : 259056.0
```

Experiment 11



Experiment 11

XII

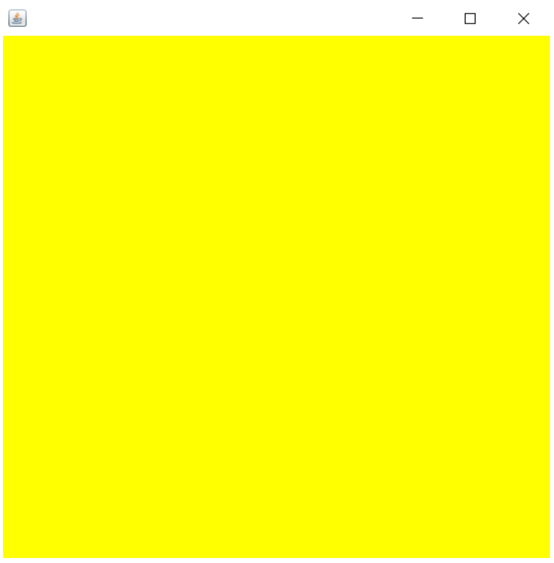
1. → i) public void mousePressed (MouseEvent me)
   ii) public void mouseReleased (MouseEvent me)
   iii) public void mouseEntered (MouseEvent me)
   iv) public void mouseExited (MouseEvent me)
   v) public void mouseClicked (MouseEvent me)

3. → i) add The MouseListener to the frame.
   ~~ii) With the mouseClicked method~~
   ii) Create mouseClicked method & add ~~it~~ to MouseEvent ~~an~~ class in its ~~paramethers~~ parameters.
   iii) Using the MouseEvent class object use the method getX() & getY() to obtain the X & Y co-ordinate of the mouse.

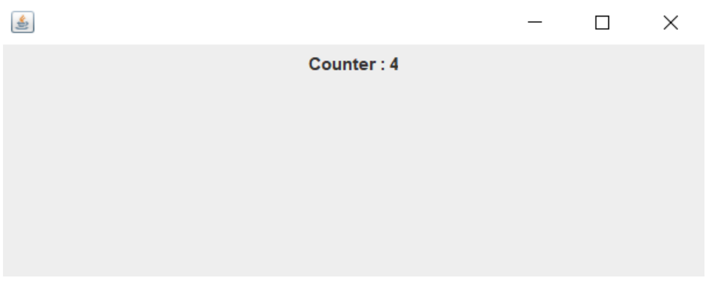
4. → i) Implement the MouseListener and/or MouseMotion Listener.
   ii) Override all the methods from the interfaces.
   iii) Add the Listeners for your components.
   ~~iv)~~

2 → All Components generate a MouseEvent

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class MouseDemo extends JFrame implements
MouseListener
{
   Container  co;
   MouseDemo()
   {
      co = getContentPane();
      co.addMouseListener(this);
      setVisible(true);
      setSize(500,500);
   }
   public void mousePressed(MouseEvent e)
   {
      co.setBackground(Color.red);
   }
   public void mouseReleased(MouseEvent e)
   {
      co.setBackground(Color.blue);
   }
   public void mouseEntered(MouseEvent e)
   {
      co.setBackground(Color.yellow);
   }
   public void mouseExited(MouseEvent e)
   {
      co.setBackground(Color.black);
   }
   public void mouseClicked(MouseEvent e)
   {
      co.setBackground(Color.green);
   }
   public static void main(String[] args) {
      new MouseDemo();
   }
}
```

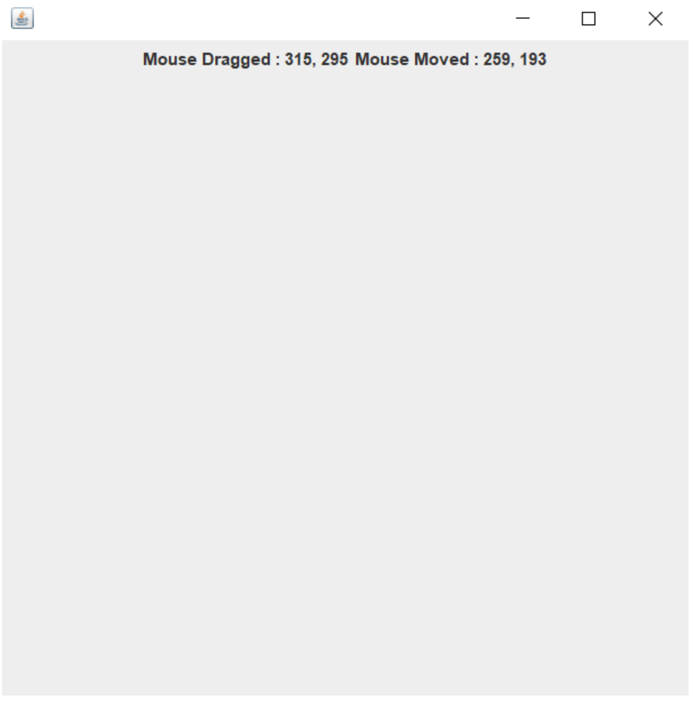


```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class MouseDemo1 extends JFrame implements
MouseListener
{
   Container  co;
   int counter = 0;
   JLabel label;
   MouseDemo1() {
      co = getContentPane();
      label = new JLabel("Counter : "  + counter);
      co.add(label);
      co.addMouseListener(this);
      co.setLayout(new FlowLayout(FlowLayout.CENTER
));
      setVisible(true);
      setSize(500,200);
   }
   public void mousePressed(MouseEvent e) {
   }
   public void mouseReleased(MouseEvent e) {
   }
   public void mouseEntered(MouseEvent e) {
   }
   public void mouseExited(MouseEvent e) {
   }
   public void mouseClicked(MouseEvent e) {
      counter++;
      label.setText("Counter : " + counter);
   }
   public static void main(String[] args) {
      new MouseDemo1();
   }
```



```
Counter : 4
```

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class MouseDemo2 extends JFrame implements
MouseMotionListener
{
   Container  co;
   JLabel l1;
   JLabel l2;
   MouseDemo2() {
      co = getContentPane();

      l1 = new JLabel("Mouse Moved : None");
      l2 = new JLabel("Mouse Dragged : None");

      co.add(l1);
      co.add(l2);
      co.addMouseMotionListener(this);
      setLayout(new FlowLayout(FlowLayout.CENTER));
      setVisible(true);
      setSize(500,500);
   }
   public void mouseDragged(MouseEvent e) {
      l1.setText("Mouse Dragged : " + e.getX() + ", " + e.g
etY());
   }
   public void mouseMoved(MouseEvent e) {
      l2.setText("Mouse Moved : " + e.getX() + ", " + e.ge
tY());
   }
   public static void main(String[] args) {
      new MouseDemo2();
   }
}
```



```
Mouse Dragged : 315, 295 Mouse Moved : 259, 193
```