VINAYAKA MISSION'S RESEARCH FOUNDATION
(Deemed to be University under section 3 of the UGC Act 1956)

VINAYAKA MISSION'S
KIRUPANANDA VARIYAR
ENGINEERING COLLEGE

VINAYAKA MISSIONS KIRUPANANDA VARIYAR ENGINEERING COLLEGE

A constituent college of

VINAYAKA MISSION'S RESEARCH FOUNDATION

(DEEMED TO BE UNIVERSITY), SALEM

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB MANUAL

JAVA PROGRAMMING LAB

Programme:

B.E / B.Tech. COMPUTER SCIENCE AND ENGINEERING
B.E / B.Tech. ARTIFICIAL INTELLIGENCE & DATA SCIENCE

REGULATIONS 2021

List of Experiments

1. Search the smallest and largest element - Array.
2. Sort the strings in an alphabetical order - String.
3. Extract a portion of a character string and to print the extracted portion and the remaining portion of the string in the given range - String.
4. Addition of numbers - Method overloading.
5. Implementation of multiple inheritance - Interface
6. Basic mathematical operations - Packages.
7. Counts the number of digits in a given number. If an alphabet is entered instead of a number, the program should not terminate - Exception Handling
8. Program to move the text "JAVA PROGRAMMING LAB" - Thread.
9. Program Displaying the Message - Applet.
10. Program to animate the face image - Applet.
11. Program to create and display student details - Abstract Window Toolkit
12. Java Swing | Create a simple text editor

1. Search the smallest and largest element - Array.

Aim:

To Write a JAVA program to search the smallest and largest element from the given array.

Algorithm:

1. Input the array elements.
2. Initialize small = large = arr[0]
3. Repeat from i = 2 to n
4. if(arr[i] > large)
5. large = arr[i]
6. if(arr[i] < small)
7. small = arr[i]
8. Print small and large.

Program:

```
import java.util.*;
public class ArrayMaxMinElement {
  public static void main(String args[]) {
    int count, max, min, i;
    int[] inputArray = new int[500];
    Scanner in = new Scanner(System.in);
    System.out.println("Enter number of elements");
    count = in.nextInt();
    System.out.println("Enter " + count + " elements");
    for(i = 0; i < count; i++) {
      inputArray[i] = in.nextInt();
    }
   max = min = inputArray[0];
   for(i = 1; i < count; i++) {
      if(inputArray[i] > max)
        max = inputArray[i];
      else if (inputArray[i] < min)
        min = inputArray[i];
  }
      System.out.println("Largest Number : " + max);
        System.out.println("Smallest Number : " + min);
   }
}
```

OUTPUT:

```
Enter number of elements
6
Enter 6 elements
7 2 5 1 9 3
Largest Number: 9
Smallest Number: 1
```

Result:

Thus the program is executed successfully and output is verified

2. Sort the strings in an alphabetical order – String

Aim:

To Write a JAVA program to sort the strings in an alphabetical order.

Algorithm:

- Get the required string.
- Compare the first two elements of the array compareTo() method.
- If the first element is greater than the second swap them.
- Then, compare 2nd and 3rd elements if the second element is greater than the 3rd swap them.
- Repeat this till the end of the array.

Program:

```
import java.util.Scanner;
public class Alphabetical_Order
{
  public static void main(String[] args)
  {
    int n;
    String temp;
    Scanner s = new Scanner(System.in);
    System.out.print("Enter number of names you want to enter:");
    n = s.nextInt();
    String names[] = new String[n];
    Scanner s1 = new Scanner(System.in);
    System.out.println("Enter all the names:");
    for(int i = 0; i < n; i++)
    {
      names[i] = s1.nextLine();
    }
    for (int i = 0; i < n; i++)
    {
      for (int j = i + 1; j < n; j++)
```

```
            {
                if (names[i].compareTo(names[j])>0)
                {
                    temp = names[i];
                    names[i] = names[j];
                    names[j] = temp;
                }
            }
        }
        System.out.print("Names in Sorted Order:");
        for (int i = 0; i < n - 1; i++)
        {
            System.out.print(names[i] + ",");
        }
        System.out.print(names[n - 1]);
    }
}
```

Output:

```
Enter number of names you want to enter:5
Enter all the names:
bryan
adam
rock
chris
scott
Names in Sorted Order: adam, bryan ,chris, rock, scott
```

Result:

Thus the program is executed successfully and output is verified

3. Extract a portion of a character string and to print the extracted portion and the remaining portion of the string in the given range - String.

Aim:

To Write a JAVA program to extract a portion of a character string and to print the extracted portion and there m portion of the string. Assume that m characters are extracted, starting with the nth character.

Algorithm:

- Get the required string.
- Get the required index of m and n for the characters to be extracted from string.
- Extract the string from the given range using substring() method.

Program:

```
/* Program to extract a portion of a character string and print the extracted string */
import java.io.DataInputStream;     // to load DataInputStream class
class P21
{
  public static void main(String args[ ])
  {
    String str1 = new String();
    String str2 = new String();
    int m=0,n=0;
    DataInputStream in = new DataInputStream(System.in);
    try
    {
      System.out.print("Enter String : ");
      str1 = in.readLine();
      System.out.println(" String is : "+str1);
      System.out.print("Enter no. of characters to be extracted from string : ");
      m = Integer.parseInt(in.readLine());
      System.out.print("Enter starting index : ");
```

```
            n = Integer.parseInt(in.readLine());
        }
        catch(Exception e) {  System.out.println("I/O Error");   }
        str2=str1.substring(n,(m+n));
        System.out.println(" Extracted String is : "+str2);
    }
}
```

OUTPUT:

```
Enter String: My name is Programming
String is: My name is Programming
Enter no. of characters to be extracted from string: 5
Enter starting index: 9
Extracted String is: s Pro
```

Result:

Thus the program is executed successfully and output is verified.

4. Addition of numbers - Method overloading.

Aim:

To Write a JAVA program for illustrating overloading and overriding methods in JAVA.

Algorithm:

- Create a class Addition with object
- Define a function with the
    - by changing the number of parameters
    - by changing the data type of parameters
    - int add(int a, int b)
    - int add(int a , int b , int c)
    - double add(double a , double b)
    - double add(int a , double b)
- Call the above mentioned functions for overloading. ob.add(17, 25), ob.add(55, 27, 35), ob.add(36.5, 42.8), ob.add(11, 24.5).

Program:

```
public class Addition
{
    int add(int a, int b)
    {
        return (a + b);
    }
    int add(int a , int b , int c)
    {
        return (a + b + c) ;
    }
    double add(double a , double b)
    {
        return (a + b);
    }
    double add(int a , double b)
    {
        return (a + b);
    }
    public static void main( String args[])
    {
        Addition ob = new Addition();

        System.out.println("Calling add method with two int parameters: " +ob.add(17,
25));
        System.out.println("Calling add method with three int parameters: " +ob.add(55,
27, 35));
        System.out.println("Calling add method with two double parameters: "
+ob.add(36.5, 42.8));
        System.out.println("Calling add method with one int and one double parameter: "
+ob.add(11, 24.5));
    }
}

class Animal {
  public void displayInfo() {
    System.out.println("I am an animal.");
  }
}
class Dog extends Animal {
  public void displayInfo() {
    super.displayInfo();
    System.out.println("I am a dog.");
  }
}
class Main {
  public static void main(String[] args) {
    Dog d1 = new Dog();
    d1.displayInfo();
  }
}
```

Output:

```
Calling add method with two int parameters: 42
Calling add method with three int parameters: 117
Calling add method with two double parameters: 79.3
Calling add method with one int and one double parameter: 35.5
```

Output:

```
I am an animal.
I am a dog.
```

Result:

Thus the program is executed successfully and output is verified.

5. Implementation of multiple inheritance – Interface

Aim:

To Write a JAVA program which illustrates the implementation of multiple inheritance using interfaces in JAVA.

Algorithm:

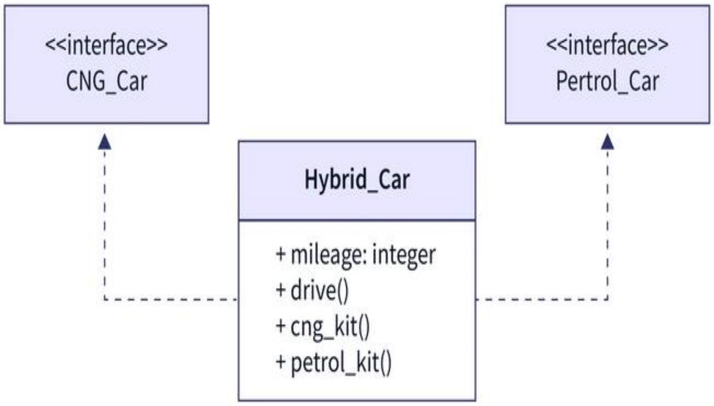
- Create the interface named as CNG_Car
- Create the interface named as Petrol_Car
- Implements the multiple inheritance through creating the class as Hybrid_car implements with CNG_Car, Petrol_Car

  class Hybrid_Car implements Petrol_Car, CNG_Car { }
- Create another class as Main (), create the object for the Hybrid_car and call the functions from the interfaces

  obj.drive();

  obj.cng_kit();

  obj.petrol_kit();



Program:

```
interface CNG_Car{
  // Abstract methods
  void drive();
  void cng_kit();
}
```

```
// Petrol Car interface
interface Petrol_Car{
  // Abstract methods
  void drive();
  void petrol_kit();
}
// Multiple Inheritance using Interface
class Hybrid_Car implements Petrol_Car, CNG_Car {
  public void drive(){
    System.out.println("Driving a Hybrid Car");
  }
  // Overridden method of CNG_Car Interface
  public void cng_kit(){
    System.out.println("Using the CNG kit for Hybrid Car");
  }
  // Overridden method of Petrol_Car Interface
  public void petrol_kit(){
    System.out.println("Using the Petrol kit for Hybrid Car");
  }
}
// Driver Code
class Main {
  public static void main(String args[]) {
    // Creating a new object of the Hybrid Car class
    Hybrid_Car obj = new Hybrid_Car();
    // Calling the methods of the Hybrid_Car class
    obj.drive();
    obj.cng_kit();
    obj.petrol_kit();
  }
}
```

OUTPUT:

```
Driving a Hybrid Car
Using the CNG kit for Hybrid Car
Using the Petrol kit for Hybrid Car
```

Result:

Thus the program is executed successfully and output is verified.

6. Basic mathematical operations - Packages

Aim:

To Write a JAVA program to create your package for basic mathematical operations such as add, subtract, multiply. Demonstrate the use of this package in another class.

Algorithm:

- Create the package named as arithemetic.
- Create the class as MyMath for defining the basic mathematical operations such as
- public int add(int x,int y)
- public int sub(int x,int y)
- public int mul(int x,int y)
- public double div(int x,int y)
- public int mod(int x,int y)
- import the arithemetic package in a new program class named as Test
- Create the object for the class which is defined in arithemetic package.
- MyMath m=new MyMath();
- Call the functions defined in the MyMath class using the defined object like
- m.add(8,5)

Program:

```
package arithmetic;
public class MyMath
{
public int add(int x,int y)
{
return x+y;
}
public int sub(int x,int y)
{
return x-y;
}
public int mul(int x,int y)
{
return x*y;
}
public double div(int x,int y)
{
return (double)x/y;
}
public int mod(int x,int y)
{
return x%y;
}
}
import arithmetic.*;
class Test
{
public static void main(String as[])
{
MyMath m=new MyMath();
System.out.println(m.add(8,5));
System.out.println(m.sub(8,5));
System.out.println(m.mul(8,5));
System.out.println(m.div(8,5));
System.out.println(m.mod(8,5));
}
}
```

Output:

Result:

Thus the program is executed successfully and output is verified.

7. Count the number of digit in a given number. If an alphabet is entered instead of a number, the program should not terminate - Exception Handling

Aim:

To Write a JAVA program that counts the number of digits in a given number. If an alphabet is entered instead of a number the program should not terminate. Instead it should display appropriate error message (Exception Handling).

Algorithm:

- Declare the variable number, temp and count. Define temp = 0 and count = 0.
- Get the input through scanner class.
- Create a function to check whether the given input is number or string.
- If the given input is string, it gives the exception InputMismatchException
- If the given input is integer, it displays the given input is an integer value and counts the number digits in the given input and displayed.

Program:

```
import java.util.Scanner;
class InvalidValueException extends Exception{
 InvalidValueException(String s){
  super(s);
 }
}
 class Main{
    static void validate(int int_val)throws InvalidValueException{
     if(int_val<=0)
      throw new InvalidValueException("Invalid value");
     else
      System.out.println("This is valid integer");
   }
   public static void main(String args[]){
      try{
        Scanner scanner;
        int number;
        int temp = 0, count = 0;
          scanner = new Scanner(System.in);
         System.out.println("Enter an Integer");
        number = scanner.nextInt();
      validate(number);
      temp = number;
        while (temp != 0) {
            temp = temp / 10;
            ++count;
        }
         System.out.format("Number of Digits in %d = %d",
           number, count);
      }
      catch(Exception m)
      {
        System.out.println("Exception occured: "+m);
      }
  }
}
```

OUTPUT:

```
Enter an Integer
 abcd
Exception occured: java.util.InputMismatchException
Enter an Integer
 45678902
This is valid integer
Number of Digits in 45678902 = 8
```

Result:

Thus the program is executed successfully and output is verified.

8. Program to move the text “JAVA PROGRAMMING LAB” - Thread.

Aim:

To Write a JAVA program to move the text “JAVA PROGRAMMING LAB” diagonally using Applet.

Algorithm:

- Import the packages awt and applet for GUI.
- Declare a String variable for displaying in the applet.
- Define the applet with the Runnable interface.
- Create a Thread with an object t.
- In initializing the applet define the value of string, and size and start the thread using the function
- t = new Thread(this, "MyThread");
- start the thread -   t.start();
- Displays the given string diagonally and update continuously using the thread.
- The exception is also founded as InterruptedException if that string is not diagonally moved.

Program:

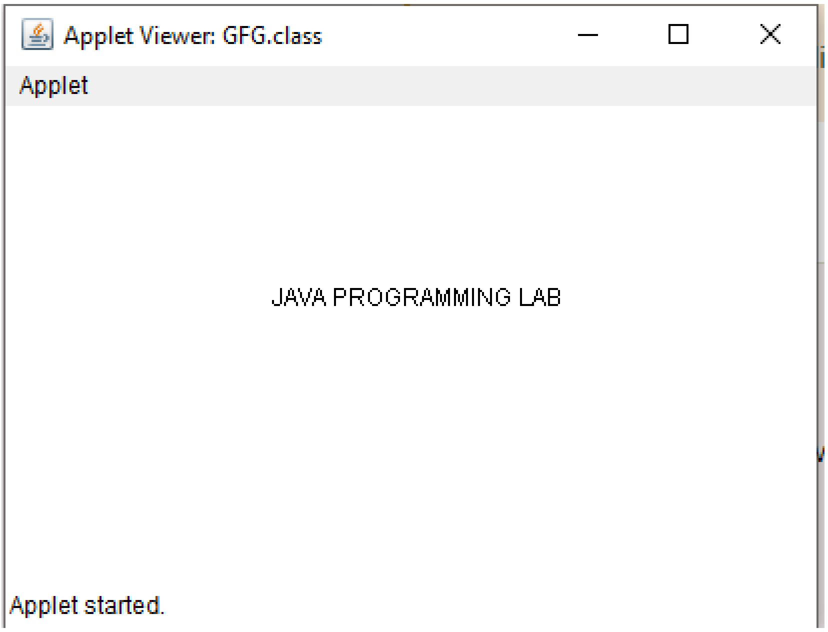
```
import java.awt.*;
import java.applet.*;
public class GFG extends Applet implements Runnable {
  private String display;
  private int x, y, flag;
  Thread t;
   // initializing called when the applet is started.
  public void init()
  {
    display = " JAVA PROGRAMMING LAB ";
    x = 100;
    y = 100;
    flag = 1;
    // creating thread
    t = new Thread(this, "MyThread");
    // start thread
    t.start();
  }
 // update the x co-ordinate
  public void update()
  {
     x = x + 10 * flag;
    if (x > 300)
      flag = -1;
    if (x < 100)
      flag = 1;
  }
 // run
  public void run()
  {
   while (true) {
       // Repainting the screen calls the paint function
      repaint();
      update();
      try {
         // creating a pause of 1 second so that the movement is recognizable
        Thread.sleep(1000);
      }
      catch (InterruptedException ie) {
        System.out.println(ie);
      }
    }
  }
  // drawString
```

```
  public void paint(Graphics g)
  {
    g.drawString(display, x, y);
  }
}
```

CFG.html

```
<html>
<body>
<applet code="GFG.class" width="300" height="300">
</applet>
</body>
</html>
```

OUTPUT:



```
JAVA PROGRAMMING LAB
```

```
Applet started.
```

Result:

Thus the program is executed successfully and output is verified.

9. Program Displaying the Message - Applet

Aim:

To Write a JAVA program to create an Applet with a label “Do you know car driving?” and two buttons Yes, No. When the user clicks “Yes” button, the message “Congrats” must be displayed. When the user clicks “NO” button, the message “Regrets” must be displayed.

Algorithm:

- Import the packages related to the program.
- Create a class name DialogTest and extends Applet implements ActionListener to checks the event handling.
- Start the init() method in Applet class.
- Create an object for label as lab and pass the argument “Do you know car driving? in label and set the place where it should be displayed.
- Create an object for button class as b1, b2 and create the set bounds where the button is displayed in applet.
- While the user clicking the button yes or no, the even handling will be invoked. If the user clicks yes, it will displays “congratulations” otherwise it will displays “Regrets”.

Program:

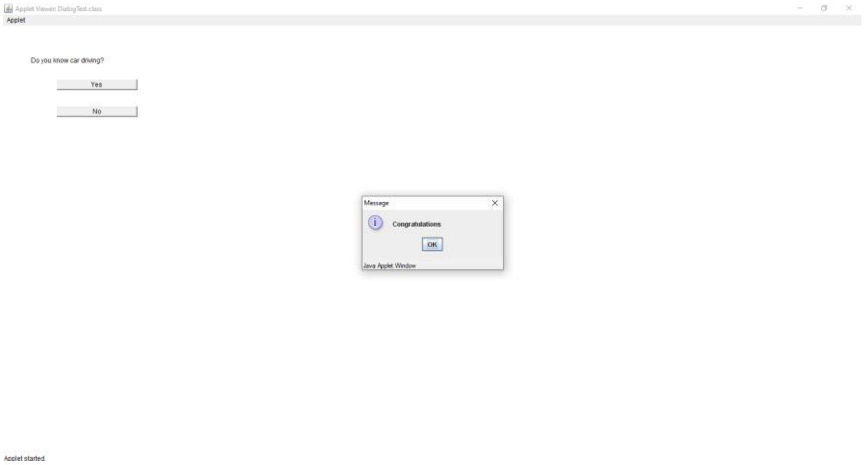
```
import java.applet.*;
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;

public class DialogTest extends Applet implements ActionListener
{
  public void init()
  {
    setLayout(null);
    setSize(500,500);
    Label lab = new Label ("Do you know car driving?");
    lab.setBounds(new Rectangle(50,50,200,30));
```

```
        add(lab,null);
        Button b1 = new Button("Yes");
        b1.setBounds(new Rectangle(100,100,150,20));
        b1.addActionListener(this);
        add(b1,null);
       Button b2 = new Button("No");
        b2.setBounds(new Rectangle(100,150,150,20));
        b2.addActionListener(this);
        add(b2,null);
    }
    public void actionPerformed(ActionEvent e)
    {
        String s = e.getActionCommand();
        if (s.equals("Yes"))
            JOptionPane.showMessageDialog(null,new String("Congratulations"));

else

            JOptionPane.showMessageDialog(null,new String("Regrets"));
}
}
```

DialogTest.html

```
<html>
<body>
<applet code="DialogTest.class" width="300" height="300">
</applet>
</body>
</html>
```

Output:



Do you know car driving?

Congratulations

Applet started

Result:

Thus the program is executed successfully and output is verified.

10. Write a JAVA program to animate the face image using Applet.

Aim:

To Write a JAVA program to animate the face image using Applet.

Algorithm:

1. Initially, import the packages for creating the applet.
2. Create a color which is similar to the human skin color, and then draw and fill the outer boundary of the face.
3. Then set the color to black.
4. Draw and fill the left eye and the right eye.
5. Draw the right eyebrow and the left eyebrow.
6. Draw the nose.
7. Draw the smile.

Program:

```
import java.applet.*;
import java.awt.*;
public class Human extends Applet
{
  //Initialize the applet
  public void init()
  {
        setBackground(Color.white);
  }
  //Draw the human face
  public void paint(Graphics g)
  {
    //Change color to cream
    Color clr=new Color(255,179,86);
    g.setColor(clr);
    //Draw and fill the face
    g.drawOval(100,100,250,300);
    g.fillOval(100,100,250,300);
```

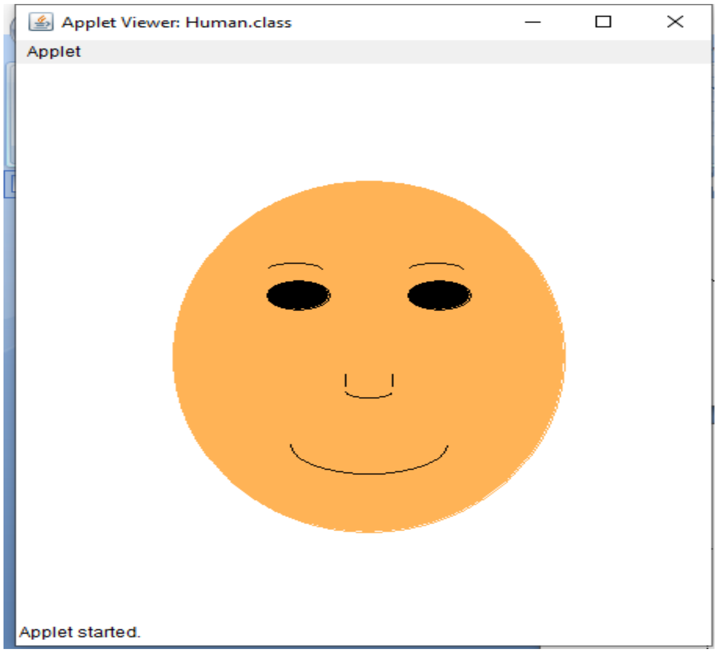
```
        //Change color to black
        g.setColor(Color.black);
        //Draw the left eye
        g.drawOval(160,185,40,25);
        g.fillOval(160,185,40,25);
        //Draw the right eye
        g.drawOval(250,185,40,25);
        g.fillOval(250,185,40,25);
        //Draw the Left Eyebrow
        g.drawArc(160,170,35,10,0,180);
        //Draw the Right Eyebrow
        g.drawArc(250,170,35,10,0,180);
        //Draw the Nose
        g.drawLine(210,265,210,275);
        g.drawLine(240,265,240,275);
        g.drawArc(210,275,30,10,0,-180);
       //Draw the smile
        g.drawArc(175,300,100,50,0,-180);
    }
}
```

Human.html

```
<html>
<body>
<applet code="Human.class" width="300" height="300">
</applet>
</body>
</html>
```

Output:

```
D:\javaprograming>javac Human.java
D:\javaprograming>appletviewer Human.html
```



Result:

Thus the program is executed successfully and output is verified.

11. Program to create and display student details - Abstract Window Toolkit

Aim:

To Write a JAVA program to accept and display student details

Algorithm:

- Import the packages for creating the graphical user interface for the student details.
- Create the class name as Student **extends** Frame **implements** ActionListener for accepting and displaying the student details in GUI.
- Create the objects for label, Checkbox, Choice, TextField, TextArea and Button.
- Create constructor to initialize the all objects and set the values for arranging the elements properly in the frame.
- Add all elements in the frame using add() method.
- Create the WindowClosing() method for Closing the window using WindowEvent Class.
- After Entered all the details in the Fields submit the details. After that it will displays the entered student details in the mentioned text area.

Program:

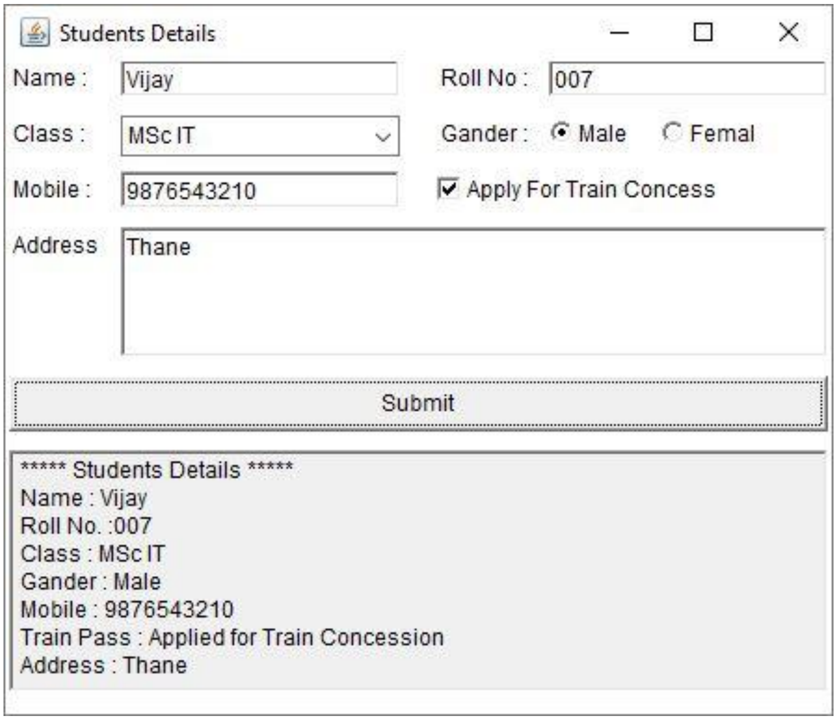
```
import java.awt.*;
import java.awt.event.*;
class Student extends Frame implements ActionListener
{
        Label lsname, lsrollno, lsclass, lgander, lsbg, lsmob, lsadrs;
        CheckboxGroup gander;
        Checkbox male, female, trainpass;
        Choice csclass;
        TextField tfsname, tfsrollno, tfsmob;
        TextArea tasadrs;
        Button submit;
        TextArea display_details;
        Student()
        {
```

```
        lsname   = new Label("Name : ");
        lsrollno = new Label("Roll No : ");
        lsclass  = new Label("Class : ");
        lgander  = new Label("Gander : ");
        lsbg     = new Label("Blood Group : ");
        lsmob    = new Label("Mobile : ");
        lsadrs   = new Label("Address : ");

gander = new CheckboxGroup();
male   = new Checkbox("Male", gander, false);
female = new Checkbox("Female", gander, false);
trainpass = new Checkbox("Apply For Train Concession");
csclass = new Choice();
csclass.add("BSc IT");
csclass.add("BSc CS");
csclass.add("BCA");
csclass.add("MSc IT");
csclass.add("MSc CS");
csclass.add("MCA");
        tfsname   = new TextField();
        tfsrollno = new TextField();
        tfsmob    = new TextField();
        tasadrs = new TextArea("", 2 , 100 , TextArea.SCROLLBARS_NONE);
        submit  = new Button("Submit");
        display_details = new TextArea("", 2 , 100 , TextArea.SCROLLBARS_NONE);
        display_details.setEditable(false);
        lsname.setBounds(10, 30, 50, 20);
        tfsname.setBounds(70, 30, 150, 20);
        lsrollno.setBounds(240, 30, 50, 20);
        tfsrollno.setBounds(300, 30, 150, 20);
        lsclass.setBounds(10, 60, 50, 20);
```

```
csclass.setBounds(70, 60, 150, 20);
lgander.setBounds(240, 60, 50, 20);
male.setBounds(300, 60, 50, 20);
female.setBounds(360, 60, 50, 20);
lsmob.setBounds(10, 90, 50, 20);
tfsmob.setBounds(70, 90, 150, 20);
trainpass.setBounds(240, 90, 150, 20);
lsadrs.setBounds(10, 120, 50, 20);
tasadrs.setBounds(70, 120, 380, 70);
submit.setBounds(10, 200, 440, 30);
display_details.setBounds(10, 240, 440, 130);
add(lsname);
add(lsrollno);
add(lsclass);
add(lgander);
add(lsbg);
add(lsadrs);
add(lsmob);
add(male);
add(female);
add(csclass);
add(tfsname);
add(tfsrollno);
add(tasadrs);
add(tfsmob);
add(trainpass);
add(submit);
add(display_details);
submit.addActionListener(this);
setTitle("Students Details");
setSize(460,390);
```

```
                setLayout(null);
                setVisible(true);
                addWindowListener(new WindowAdapter()
                {
       public void windowClosing(WindowEvent e)
       {
         dispose();
       }
     });
        }
        public void actionPerformed(ActionEvent e)
        {
                if(e.getSource()==submit)
                {
                        String tp = trainpass.getState() ? "Applied for Train Concession" : "Not
Applied for Train Concession";

                        String sdetails = " ***** Students Details *****" +"\n Name : " +
tfsname.getText() + "\n Roll No. :" + tfsrollno.getText() + "\n Class : " +
csclass.getSelectedItem() + "\n Gender : " + gander.getSelectedCheckbox().getLabel() + "\n
Mobile : " + tfsmob.getText() + "\n Train Pass : " + tp + "\n Address : " + tasadrs.getText();

                        display_details.setText(sdetails);
                }
        }
        public static void main(String[] args)
        {
                new Student();
        }
}
```

Output:



Name : Vijay

Roll No : 007

Class : MSc IT

Gander : ● Male ○ Femal

Mobile : 9876543210

☑ Apply For Train Concess

Address Thane

```
***** Students Details *****
Name : Vijay
Roll No. :007
Class : MSc IT
Gander : Male
Mobile : 9876543210
Train Pass : Applied for Train Concession
Address : Thane
```

Result:

Thus the program is executed successfully and output is verified.

Java Swing | Create a simple text editor

Aim:

To write a java program for simple text editor.

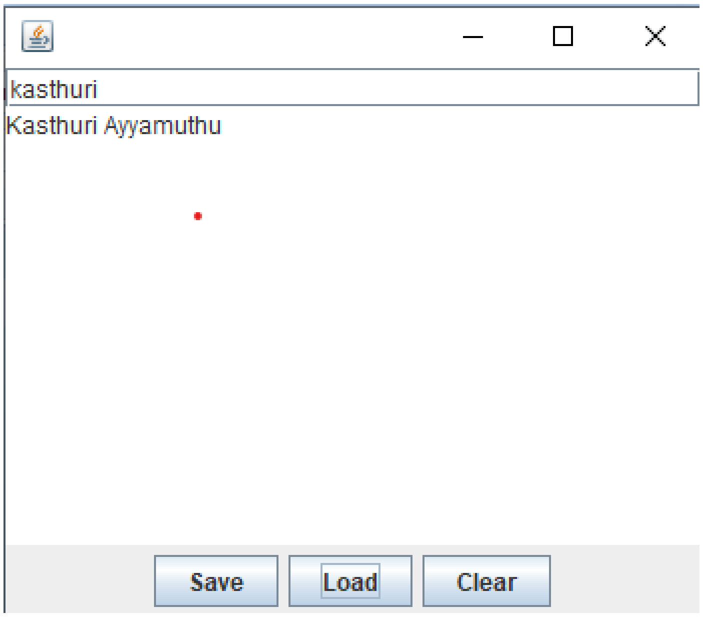
Program:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Scanner;

public class Notepad extends JFrame {
  private JTextField fileNameTextField;
  private JTextArea textArea;
  private JButton saveButton, loadButton, clearButton;
   public Notepad() {

        fileNameTextField = new JTextField();
        textArea = new JTextArea();

        JPanel buttonPanel = new JPanel();

        saveButton = new JButton("Save");
        loadButton = new JButton("Load");
        clearButton = new JButton("Clear");

        // Instantiate an action listener to listen for button click events
        NotepadButtonListener buttonListener = new NotepadButtonListener();

        // attach our action listener to the buttons
        saveButton.addActionListener(buttonListener);
        loadButton.addActionListener(buttonListener);
        clearButton.addActionListener(buttonListener);

        buttonPanel.add(saveButton);
        buttonPanel.add(loadButton);
        buttonPanel.add(clearButton);
```

```
      // set the layout of the JFrame
      this.setLayout(new BorderLayout());

     add(fileNameTextField, BorderLayout.NORTH);
      add(textArea, BorderLayout.CENTER);
      add(buttonPanel, BorderLayout.SOUTH);

      setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

      // show the frame
      setPreferredSize(new Dimension(400, 300));
      pack();
      setVisible(true);
   }

   private String getFileName() {
      return fileNameTextField.getText();
   }

   private void readFile(String fileName) {
      Scanner inFile = null;
      try {
         // file reader
         inFile = new Scanner(new FileReader(fileName));

         // clear the text area
         textArea.setText("");

         // copy file
         while (inFile.hasNextLine()) {
            textArea.append(inFile.nextLine());
         }

      } catch (IOException ioe) {
         ioe.printStackTrace();
         System.out.println("File not found");
      } finally {
         if (inFile != null) {
            inFile.close();
         }
      }
   }

   private void writeFile(String fileName) {
      PrintWriter outFile = null;

      try {
         // file writer
         outFile = new PrintWriter(new FileWriter(fileName));

         outFile.print(textArea.getText());

      } catch (IOException ioe) {
         ioe.printStackTrace();
         System.out.println("File not found");
      } finally {
         if (outFile != null) {
            outFile.close();
         }
      }
   }

   class NotepadButtonListener implements ActionListener {

      public void actionPerformed(ActionEvent e) {
         JButton sourceButton = (JButton) e.getSource();

         // Figure out which button was pressed
         if (sourceButton.equals(loadButton)) {
            System.out.println("Load button pressed");
            readFile(getFileName());
         } else if (sourceButton.equals(saveButton)) {
            System.out.println("Save button pressed");
            writeFile(getFileName());
         } else if (sourceButton.equals(clearButton)) {
            System.out.println("Clear button pressed");

            // clear the text area
            textArea.setText("");
```

```
            } else {
                System.out.println("Unknown button pressed");
            }
        }
    }

    public static void main(String[] args) {
        // Create the notepad instance
        new Notepad();
    }
}
```

Output;



kasthuri

Kasthuri Ayyamuthu

Result:

Thus the program is executed successfully and output is verified.