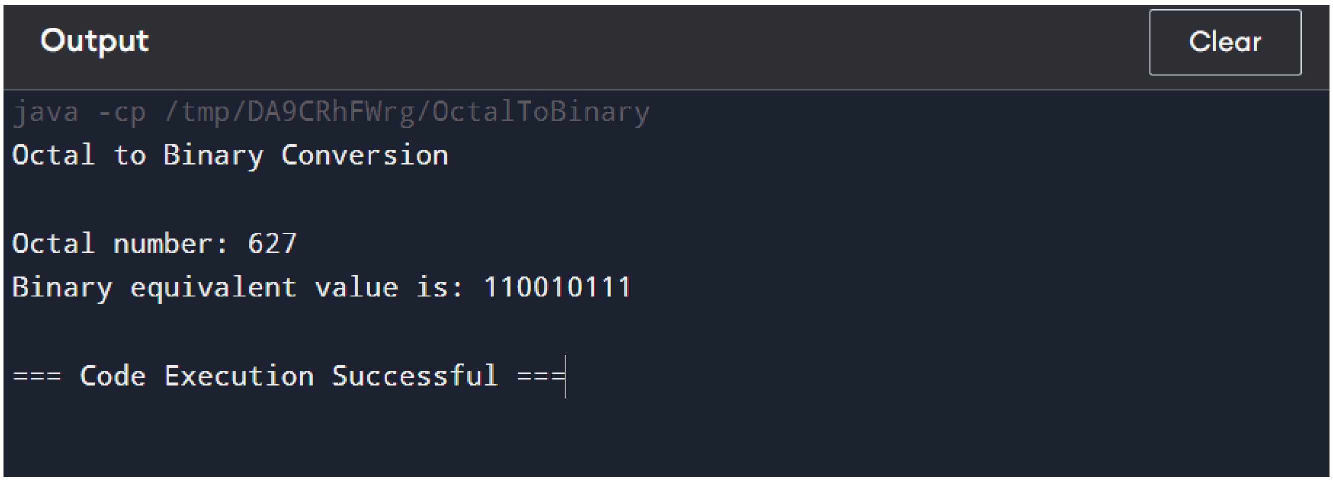
Experiment No. – 1

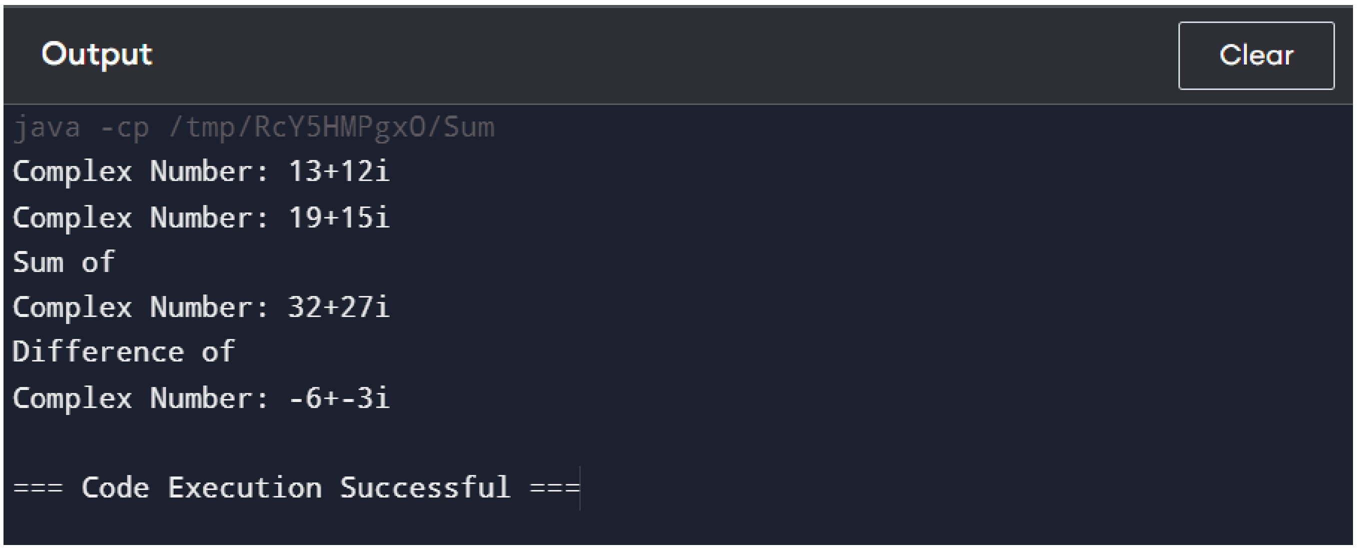
```
// package hello;
public class OctalToBinary {
    static String converter(String octalValue) {
        int i = 0;
        String binaryValue = "";
        while (i < octalValue.length()) {
            char c = octalValue.charAt(i);
            switch (c) {
                case '0':
                    binaryValue += "000";
                    break;
                case '1':
                    binaryValue += "001";
                    break;
                case '2':
                    binaryValue += "010";
                    break;
                case '3':
                    binaryValue += "011";
                    break;
                case '4':
                    binaryValue += "100";
                    break;
                case '5':
                    binaryValue += "101";
                    break;
                case '6':
                    binaryValue += "110";
                    break;
                case '7':
                    binaryValue += "111";
                    break;
                default:
                    System.out.println("\nInvalid Octal Digit " + c);
                    break;
            }
            i++;
        }
        return binaryValue;
    }
  public static void main(String args[]) {
        System.out.println("Octal to Binary Conversion\n");
        String octalNumber = "627";
        System.out.println("Octal number: " + octalNumber);
        String result = converter(octalNumber);
        System.out.println("Binary equivalent value is: " + result);
    }
}
```



```
java -cp /tmp/DA9CRhFWrg/OctalToBinary
Octal to Binary Conversion

Octal number: 627
Binary equivalent value is: 110010111

=== Code Execution Successful ===
```

Experiment No. – 2

```
import java.util.*;
class Complex {
    int real, imaginary;
    Complex() {
    }
    Complex(int tempReal, int tempImaginary) {
        real = tempReal;
        imaginary = tempImaginary;
    }
    Complex addComp(Complex C1, Complex C2) {
        Complex temp = new Complex();
        temp.real = C1.real + C2.real;
        temp.imaginary = C1.imaginary + C2.imaginary;
        return temp;
    }

    Complex subtractComp(Complex C1, Complex C2) {
        Complex temp = new Complex();
        temp.real = C1.real - C2.real;
        temp.imaginary = C1.imaginary - C2.imaginary;
        return temp;
    }

        void printComplexNumber() {
        System.out.println("Complex Number: " + real + "+" + imaginary + "i");
    }
}

public class Sum {
    public static void main(String[] args) {
        Complex c1 = new Complex(13, 12);
        c1.printComplexNumber();
        Complex c2 = new Complex(19, 15);
        c2.printComplexNumber();

        Complex c3 = new Complex();
        c3 = c3.addComp(c1, c2);
        System.out.println("Sum of");
        c3.printComplexNumber();

        c3 = c3.subtractComp(c1, c2);
        System.out.println("Difference of");
        c3.printComplexNumber();
    }
}
```

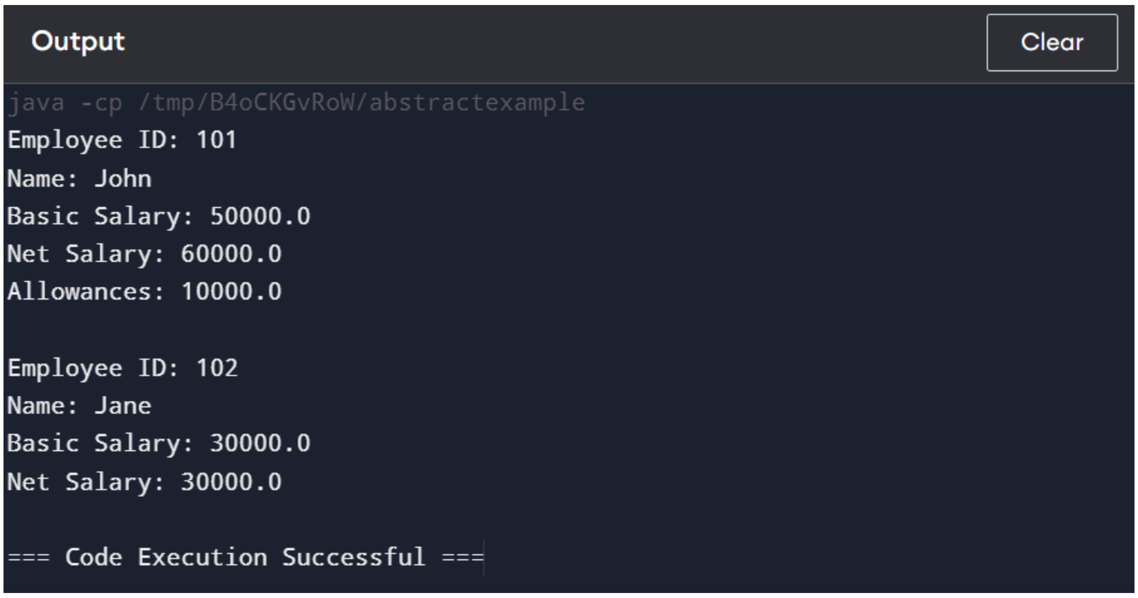


```
java -cp /tmp/RcY5HMPgxO/Sum
Complex Number: 13+12i
Complex Number: 19+15i
Sum of
Complex Number: 32+27i
Difference of
Complex Number: -6+-3i

=== Code Execution Successful ===
```

Experiment No. – 3

```
abstract class Employee {
    int empId;
    String name;
    double basicSalary;
    public Employee(int empId, String name, double basicSalary) {
        this.empId = empId;
        this.name = name;
        this.basicSalary = basicSalary;
    }
    public abstract double calculateNetSalary();
    public void displayInformation() {
        System.out.println("Employee ID: " + empId);
        System.out.println("Name: " + name);
        System.out.println("Basic Salary: " + basicSalary);
        System.out.println("Net Salary: " + calculateNetSalary());
    } }
class Manager extends Employee {
    private double allowances;
    public Manager(int empId, String name, double basicSalary, double allowances) {
        super(empId, name, basicSalary);
        this.allowances = allowances;
    }
    @Override
    public double calculateNetSalary() {
        return basicSalary + allowances;
    }
    @Override
    public void displayInformation() {
        super.displayInformation();
        System.out.println("Allowances: " + allowances);
    }}
```

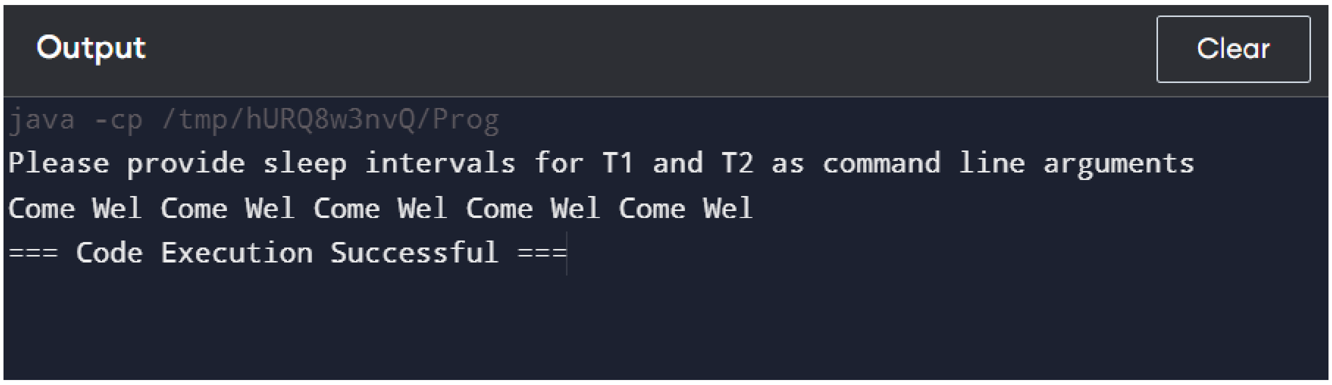
```
class Clerk extends Employee {
    public Clerk(int empId, String name, double basicSalary) {
        super(empId, name, basicSalary);
    }
    @Override
    public double calculateNetSalary() {
        return basicSalary;
    }
    @Override
    public void displayInformation() {
        super.displayInformation();
    }
}
public class abstractexample {
    public static void main(String[] args) {
        Manager manager = new Manager(101, "John", 50000, 10000);
        Clerk clerk = new Clerk(102, "Jane", 30000);
        manager.displayInformation();
        System.out.println();
        clerk.displayInformation();
    }
}
```



```
java -cp /tmp/B4oCKGvRoW/abstractexample
Employee ID: 101
Name: John
Basic Salary: 50000.0
Net Salary: 60000.0
Allowances: 10000.0

Employee ID: 102
Name: Jane
Basic Salary: 30000.0
Net Salary: 30000.0

=== Code Execution Successful ===
```

Experiment No. – 04

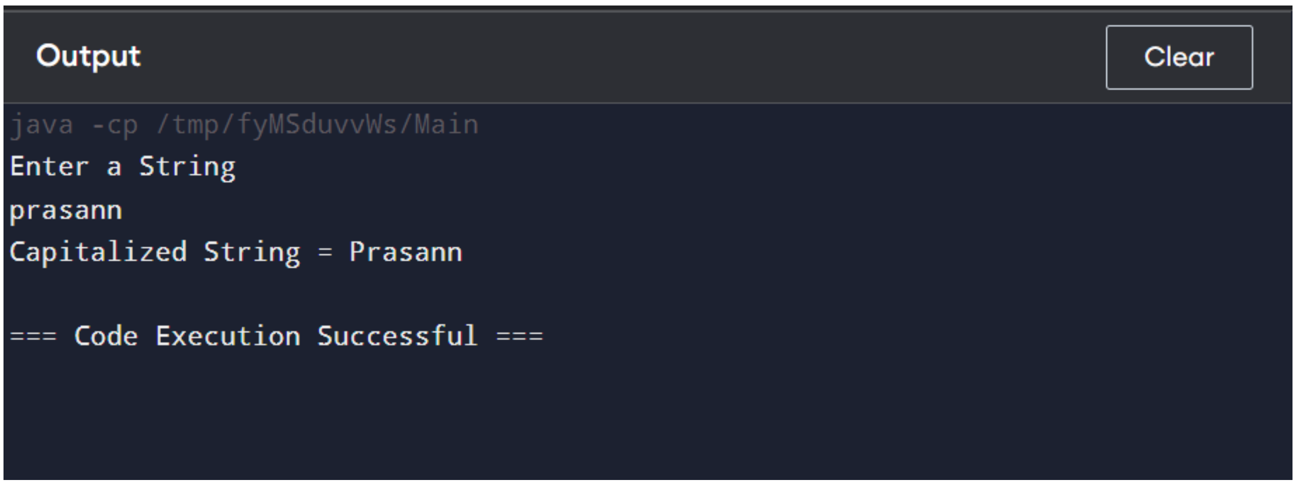
```
class SharedObject {
}
class PrintMessage implements Runnable {
    private SharedObject sharedObject;
    private String message;
    private int sleepInterval;
    public PrintMessage(SharedObject sharedObject, String message, int sleepInterval) {
        this.sharedObject = sharedObject;
        this.message = message;
        this.sleepInterval = sleepInterval;
    }
    public void run() {
        synchronized (sharedObject) {
            for (int i = 0; i < 5; i++) {
                System.out.print(message + " ");
                try {
                    Thread.sleep(sleepInterval);


                    sharedObject.notify();
                    if (i < 4) {
                        sharedObject.wait();
                    }
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
            sharedObject.notify();        }
    }
}

public class Prog {
    public static void main(String[] args) {
        int sleepIntervalT1 = 100;
        int sleepIntervalT2 = 200;
        if (args.length < 2) {
            System.out.println("Please provide sleep intervals for T1 and T2 as command line
arguments");
        } else {
            sleepIntervalT1 = Integer.parseInt(args[0]);
            sleepIntervalT2 = Integer.parseInt(args[1]);
        }

        SharedObject sharedObject = new SharedObject();

        Thread t1 = new Thread(new PrintMessage(sharedObject, "Wel", sleepIntervalT1));
        Thread t2 = new Thread(new PrintMessage(sharedObject, "Come", sleepIntervalT2));

        t1.start();
        t2.start();
    }
}
```



```
java -cp /tmp/hURQ8w3nvQ/Prog
Please provide sleep intervals for T1 and T2 as command line arguments
Come Wel Come Wel Come Wel Come Wel Come Wel
=== Code Execution Successful ===
```

Experiment No. – 05

```
import java.util.Scanner;
class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter a String");
      String input = scanner.nextLine();
      String capitalized = CapitalizeString(input);
      System.out.println("Capitalized String = " + capitalized);
   }
   public static String CapitalizeString(String input) {
      String[] words = input.split(" ");
      StringBuilder Capitalized = new StringBuilder();
      for (int i = 0; i < words.length; i++) {
         if (!words[i].isEmpty()) {
            String firstLetter = words[i].substring(0, 1);
            String restOfWord = words[i].substring(1);
            String CapitalizedWord = firstLetter.toUpperCase() + restOfWord;
            Capitalized.append(CapitalizedWord);
            if (i < words.length - 1) {
               Capitalized.append(" ");
            }}}
      return Capitalized.toString();
   }}
```

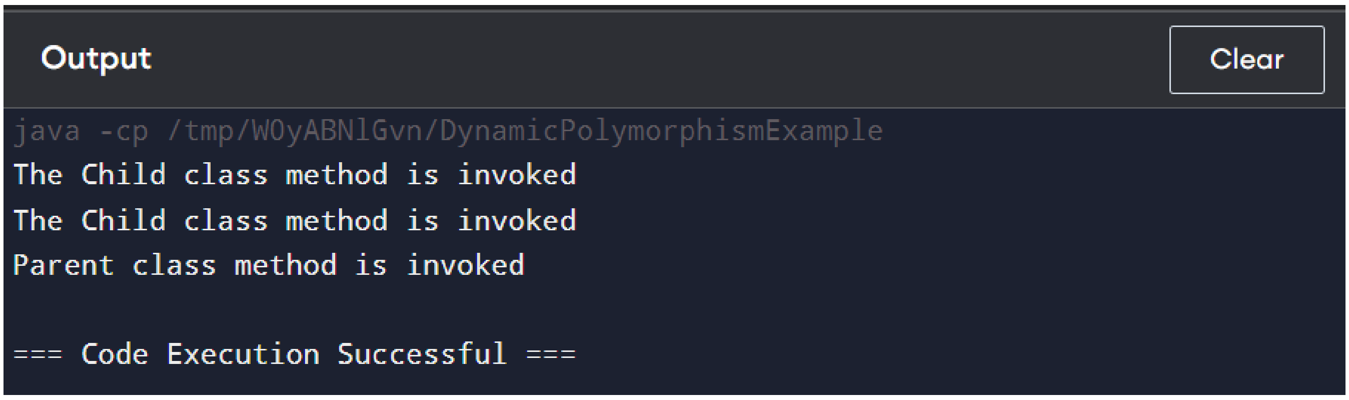


```
java -cp /tmp/fyMSduvvWs/Main
Enter a String
prasann
Capitalized String = Prasann

=== Code Execution Successful ===
```

Experiment No.- 06

A] Dynamic Polymorphism Program

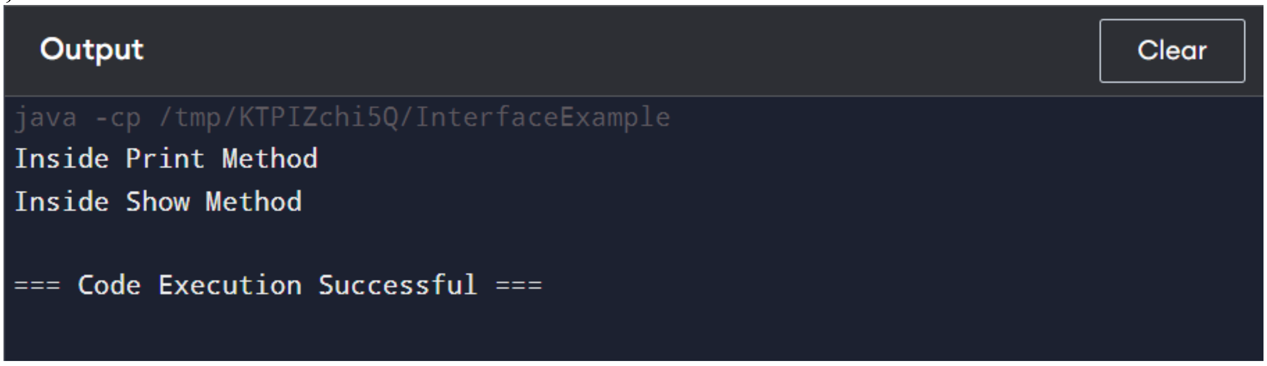
```
public class DynamicPolymorphismExample{

public static void main(String args[]){
Fruits fruits = new Mango(); fruits.Color();
Mango m = new Mango(); m.Color();
Fruits fruit = new Fruits(); fruit.Color();
}
}

class Fruits{
public void Color(){
System.out.println("Parent class method is invoked");
}
}

class Mango extends Fruits{
public void Color(){
System.out.println("The Child class method is invoked");
}
}
```



```
java -cp /tmp/WOyABNlGvn/DynamicPolymorphismExample
The Child class method is invoked
The Child class method is invoked
Parent class method is invoked

=== Code Execution Successful ===
```

B] Interfaces Program package hello;

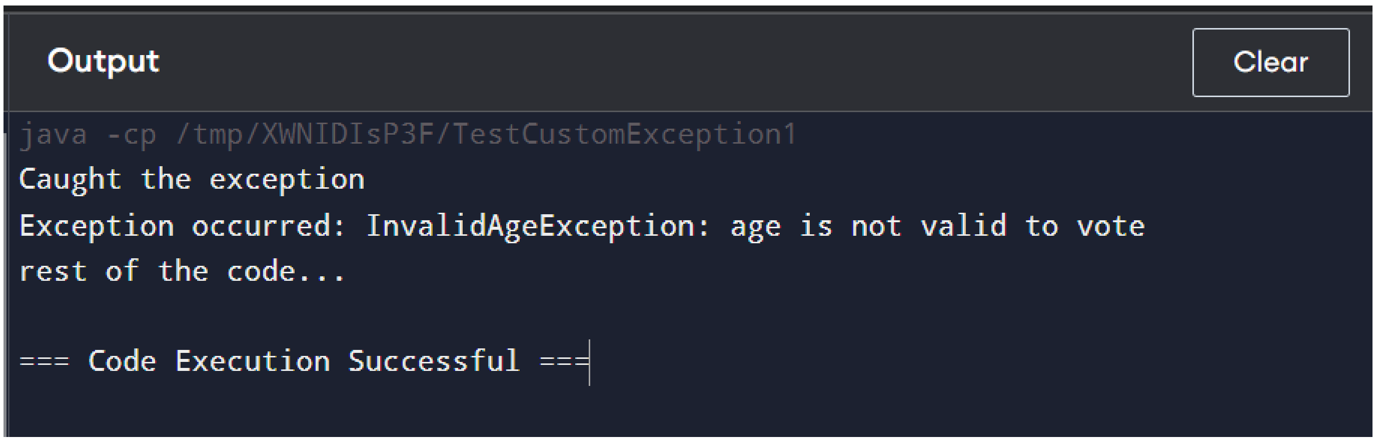
```
interface Printable{
void print();
}
 interface Showable
{
void show();
}
class InterfaceExample implements Printable, Showable
{
public void print()
{
System.out.println("Inside Print Method");
}
public void show()
{
System.out.println("Inside Show Method");
}
public static void main(String args[])
{
InterfaceExample obj = new InterfaceExample(); obj.print();
obj.show();
}
}
```



```
java -cp /tmp/KTPIZchi5Q/InterfaceExample
Inside Print Method
Inside Show Method

=== Code Execution Successful ===
```

Experiment No. -07

```
class InvalidAgeException extends Exception {
   InvalidAgeException(String str) {
      super(str);
   }
}
public class TestCustomException1 {
   static void validate(int age) throws InvalidAgeException {
      if (age < 18) {
         throw new InvalidAgeException("age is not valid to vote");
      } else {
         System.out.println("Welcome to vote");
      }
   }
   public static void main(String args[]) {
      try {
         validate(13);
      } catch (InvalidAgeException ex) {
         System.out.println("Caught the exception");
         System.out.println("Exception occurred: " + ex);
      }
      System.out.println("rest of the code...");
   }
}
```



```
java -cp /tmp/XWNIDIsP3F/TestCustomException1
Caught the exception
Exception occurred: InvalidAgeException: age is not valid to vote
rest of the code...

=== Code Execution Successful ===
```

Experiment No. – 9

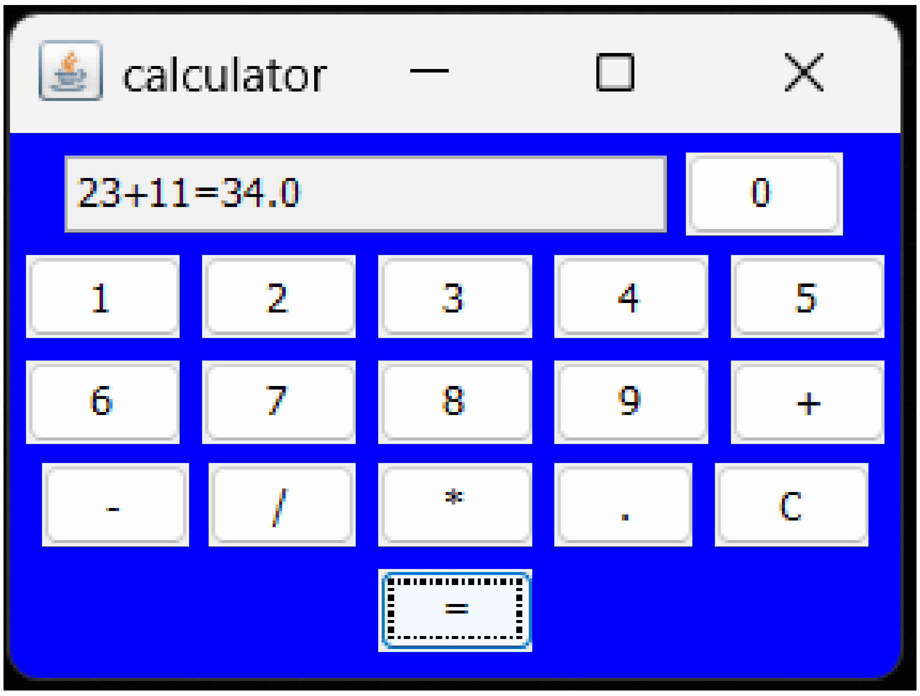
```
package hello;
import java.awt.event.*;
import javax.swing.*;
import java.awt.*;
class calculator extends JFrame implements ActionListener
{
static JFrame f;
static JTextField I;
String s0,s1,s2;
calculator()
{ s0=s1=s2="";
}
public static void main(String args[])
{
f=new JFrame("calculator");
try
{
UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

}
catch(Exception e)
{
System.err.println(e.getMessage());
}
calculator c = new calculator();
I = new JTextField(16);
I.setEditable(false);
JButton b0,b1,b2,b3,b4,b5,b6,b7,b8,b9,ba,bs,bd,bm,be,beq,beq1;
b0=new JButton("0");
b1=new JButton("1");
b2=new JButton("2");
```

```
b3=new JButton("3");
b4=new JButton("4");
b5=new JButton("5");
b6=new JButton("6");
b7=new JButton("7");
b8=new JButton("8");
b9=new JButton("9");
ba=new JButton("+");
bs=new JButton("-");
bd=new JButton("/");
bm=new JButton("*");
be=new JButton(".");
beq=new JButton("C");
beq1=new JButton("=");
JPanel p = new JPanel();
b0.addActionListener(c);
b1.addActionListener(c);
b2.addActionListener(c);
b3.addActionListener(c);
b4.addActionListener(c);
b5.addActionListener(c);
b6.addActionListener(c);
b7.addActionListener(c);
b8.addActionListener(c);
b9.addActionListener(c);
ba.addActionListener(c);
bs.addActionListener(c);
bd.addActionListener(c);
bm.addActionListener(c);
be.addActionListener(c);
beq.addActionListener(c);
beq1.addActionListener(c);
```

```
p.add(l);
p.add(b0);
p.add(b1);
p.add(b2);
p.add(b3);
p.add(b4);
p.add(b5);
p.add(b6);
p.add(b7);
p.add(b8);
p.add(b9);
p.add(ba);
p.add(bs);
p.add(bd);
p.add(bm);
p.add(be);
p.add(beq);
p.add(beq1);
p.setBackground(Color.blue);
f.add(p);
f.setSize(200,220);
f.show();
}
public void actionPerformed(ActionEvent e)
{

String s = e.getActionCommand();
if(s.charAt(0)>='0' && s.charAt(0)<='9'||s.charAt(0)=='.')
{
if(!s1.equals("")) s2=s2+s;
else
s0=s0+s;
```

```
I.setText(s0+s1+s2);
}
else if(s.charAt(0)=='C')
{ s0=s1=s2="";
I.setText(s0 +s1 + s2);
}
else if(s.charAt(0)=='=')
{
Double te;
if(s1.equals("+"))
te=(Double.parseDouble(s0)+ Double.parseDouble(s2));
else if(s1.equals("-"))
te=(Double.parseDouble(s0)- Double.parseDouble(s2));
else if(s1.equals("/"))
te=(Double.parseDouble(s0)/ Double.parseDouble(s2));
else
te=(Double.parseDouble(s0)* Double.parseDouble(s2));
I.setText(s0+s1+s2+'='+te);
s0=Double.toString(te);
s1=s2="";
}
else
{
if(s1.equals("")||s2.equals("")) s1=s;
else
{
Double te;
if(s1.equals("+"))
te=(Double.parseDouble(s0)+ Double.parseDouble(s2));
else if(s1.equals("-"))
te=(Double.parseDouble(s0)- Double.parseDouble(s2));
else if(s1.equals("/"))
te=(Double.parseDouble(s0)/ Double.parseDouble(s2));
else
te=(Double.parseDouble(s0)* Double.parseDouble(s2)); s0=Double.toString(te);
s1=s; s2="";
}
I.setText(s0+s1+s2);
}
}}
```

OUTPUT -



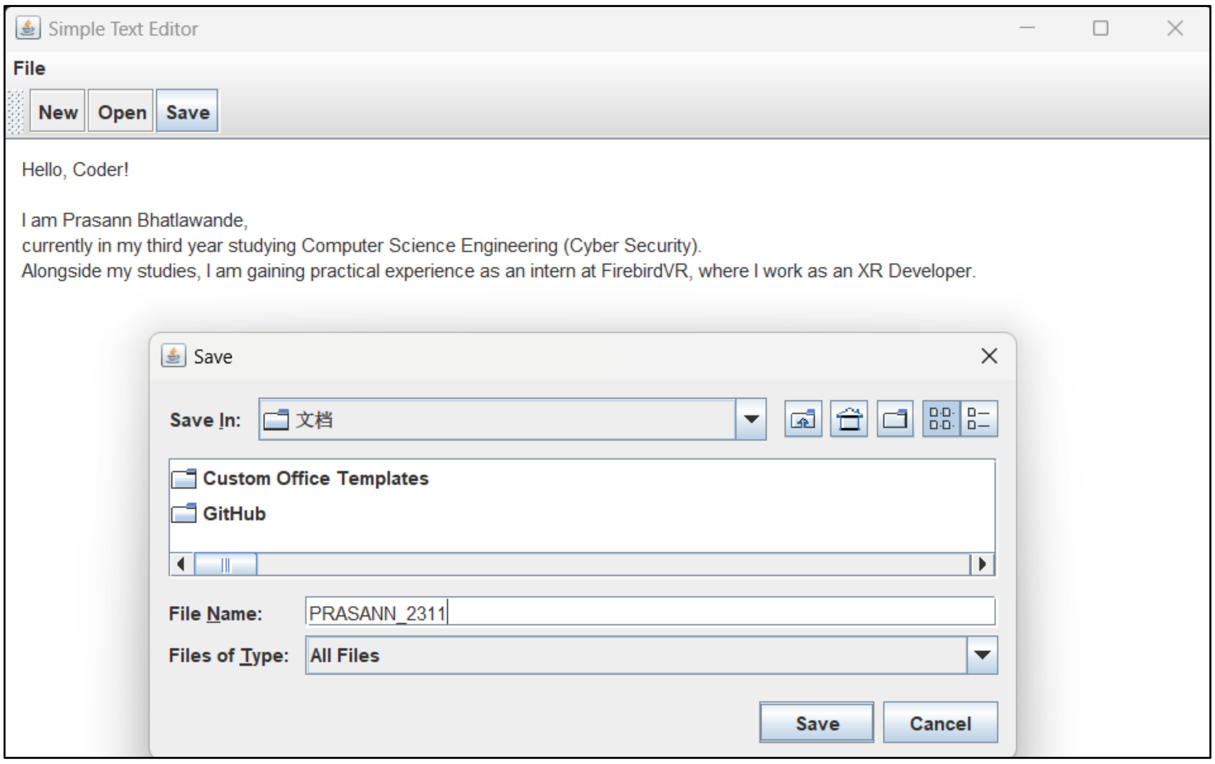
Experiment No. 10

```
import javax.swing.*;
import javax.swing.border.EmptyBorder;
import javax.swing.filechooser.FileNameExtensionFilter;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
public class SimpleTextEditor extends JFrame {
   private JTextArea textArea;
   private String currentFilePath = null;
   public SimpleTextEditor() {
      setTitle("Simple Text Editor");
      setSize(800, 600);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      setLocationRelativeTo(null);
      textArea = new JTextArea();
      textArea.setBorder(new EmptyBorder(10, 10, 10, 10));
      JScrollPane scrollPane = new JScrollPane(textArea);
      add(scrollPane, BorderLayout.CENTER);
      createMenuBar();
      createToolBar();
   }
   private void createMenuBar() {
      JMenuBar menuBar = new JMenuBar();
      JMenu fileMenu = new JMenu("File");
      JMenuItem newItem = new JMenuItem("New");
      JMenuItem openItem = new JMenuItem("Open");
      JMenuItem saveItem = new JMenuItem("Save");
      JMenuItem exitItem = new JMenuItem("Exit");
      newItem.addActionListener(e -> newFile());
      openItem.addActionListener(e -> openFile());
```

```
    saveItem.addActionListener(e -> saveFile());
    exitItem.addActionListener(e -> System.exit(0));
    fileMenu.add(newItem);
    fileMenu.add(openItem);
    fileMenu.add(saveItem);
    fileMenu.addSeparator();
    fileMenu.add(exitItem);
    menuBar.add(fileMenu);
    setJMenuBar(menuBar);
}
private void createToolBar() {
    JToolBar toolBar = new JToolBar();
    JButton newButton = new JButton("New");
    JButton openButton = new JButton("Open");
    JButton saveButton = new JButton("Save");
    newButton.addActionListener(e -> newFile());
    openButton.addActionListener(e -> openFile());
    saveButton.addActionListener(e -> saveFile());
    toolBar.add(newButton);
    toolBar.add(openButton);
    toolBar.add(saveButton);
    add(toolBar, BorderLayout.NORTH);
}
private void newFile() {
    textArea.setText("");
    currentFilePath = null;
    setTitle("Simple Text Editor - New File");
}
private void openFile() {
    JFileChooser fileChooser = new JFileChooser();
    FileNameExtensionFilter filter = new FileNameExtensionFilter("Text Files", "txt");
    fileChooser.setFileFilter(filter);
```

```
        if (fileChooser.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
            File file = fileChooser.getSelectedFile();
            currentFilePath = file.getAbsolutePath();
            setTitle("Simple Text Editor - " + file.getName());
            try (BufferedReader br = new BufferedReader(new FileReader(file))) {
                textArea.read(br, null);
            } catch (IOException e) {
                JOptionPane.showMessageDialog(this, "Error opening file", "Error",
JOptionPane.ERROR_MESSAGE);
            }
        }
    }
    private void saveFile() {
        if (currentFilePath == null) {
            JFileChooser fileChooser = new JFileChooser();
            if (fileChooser.showSaveDialog(this) == JFileChooser.APPROVE_OPTION) {
                currentFilePath = fileChooser.getSelectedFile().getAbsolutePath();
            } else {
                return;
            }
        }
        try (BufferedWriter bw = new BufferedWriter(new FileWriter(currentFilePath))) {
            textArea.write(bw);
            setTitle("Simple Text Editor - " + new File(currentFilePath).getName());
        } catch (IOException e) {
            JOptionPane.showMessageDialog(this, "Error saving file", "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            SimpleTextEditor editor = new SimpleTextEditor();
            editor.setVisible(true);
        });
    }
}
```

OUTPUT -



```
Hello, Coder!

I am Prasann Bhatlawande,
currently in my third year studying Computer Science Engineering (Cyber Security).
Alongside my studies, I am gaining practical experience as an intern at FirebirdVR, where I work as an XR Developer.
```

File Name: PRASANN_2311

Files of Type: All Files