JOGINPALLY B.R. ENGINEERING COLLEGE
UGC AUTONOMOUS
Approved by AICTE, Affiliated to JNTUH, Hyderabad& ISO 9001:2015 Certified
Accredited by NAAC with A+ Grade, Recognized under Section 2(f) of UGC Act.1956
Yenkapally (V), Moinabad (M), P.O. Himayathnagar, R.R. District, Hyderabad – 500 075.
Website: www.jbrec.edu.in**e-mail:**principal@jbrec.edu.in

DEPARTMENT
OF
CSE (DATA SCIENCE)

NAME OF THE LABORATORY	:	**OOPS THROUGH JAVA LAB**
YEAR AND SEM	:	**II B.TECH II SEM (R23)**
REGULATION / LAB CODE	:	**R23/DS305PC**
LAB INCHARGE	:	**Elijah Francis**
		Asst. Professor, CSE (DS)

OBJECT ORIENTED PROGRAMMING THROUGH JAVA LABORATORY MANUAL

HOD **PRINCIPAL**

JBREC
Joginpally BR Engineering College
UGC AUTONOMOUS

DEPARTMENT OF CSE (DATA SCIENCE)

VISION

To produce **competent,** data science professionals who possess **exceptional expertise to address industrial, real-world challenges** and contribute to **societal needs**

MISSION

- To provide students with a strong mathematical foundation, empowering them to analyze complex data sets and master advanced data science techniques, to become **highly competent in the field.**
- To gain expertise in Computer Science by using advanced tools of Data Science, AI, and ML, to **solve real-world challenges through research and collaboration with industry and government**.
- To foster innovation and an entrepreneurial mindset in students, through a high-quality education grounded in ethical values and leadership, addressing **society's evolving needs.**

PROGRAM EDUCATIONAL OBJECTIVES

- **PEO 1:** To address complex and challenging problems in their profession by applying a strong foundation in mathematics, core scientific principles, and engineering theory.
- **PEO 2:** To engage in lifelong learning to adapt to the dynamic advancements in a data-driven world, ensuring successful careers in computer science and engineering or excelling in their pursuit of advanced studies.
- **PEO 3:** To demonstrate effective communication, collaboration, professionalism, and a strong sense of moral and ethical responsibility for addressing real-world challenges with a broad perspective on sustainable development and social impact.

PROGRAM OUTCOMES

- **Engineering Knowledge**
  Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

- **Problem Analysis**
  Identify, formulate, review research literature, and analyze complex engineering problems using principles of mathematics, natural sciences, and engineering sciences.

- **Design/Development of Solutions**
  Design solutions for complex engineering problems and design system components or processes that meet specified needs with appropriate consideration for public health, safety, cultural, societal, and environmental factors.

- **Conduct Investigations of Complex Problems**
  Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of information to provide valid conclusions.

- **Modern Tool Usage**
  Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modeling, to complex engineering activities with an understanding of the limitations.

- **The Engineer and Society**
  Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to professional engineering practice.

- **Environment and Sustainability**
  Understand the impact of professional engineering solutions in societal and environmental contexts, and demonstrate knowledge of, and need for, sustainable development.

- **Ethics**
  Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

- **Individual and Team Work**

Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

- **Communication**
  Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports, design documentation, make effective presentations, and give and receive clear instructions.

- **Project Management and Finance**
  Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

- **Life-long Learning**
  Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES

- **PSO 1: Professional Skill:** Comprehend, design, analyze, and apply statistical models, mathematical tools, and management principles in the development of intelligent systems with advanced computational capabilities, specifically in the domain of data science.
- **PSO 2: Problem-Solving Skills:** Master adaptive algorithms and appropriate techniques to conduct data analysis and visualization, addressing complex problems and facilitating effective decision-making across interdisciplinary domains.
- **PSO 3: Successful Research:** Demonstrate domain expertise and proficiency to advance research capabilities, translating innovative ideas into solutions for societal challenges, while upholding ethical standards, fostering entrepreneurial skills, and pursuing higher studies.

SYLLABUS OF OBJECT ORIENTED PROGRAMMING THROUGH JAVA LAB (R23 / DS305PC)

B.Tech. II Year II Sem.

L	T	P	C
0	0	3	1.5

List of Experiments:

1. Use Eclipse or Net bean platform and acquaint yourself with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.
2. Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.
3. A) Develop an applet in Java that displays a simple message.
   B) Develop an applet in Java that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named “Compute” is clicked.
4. Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.
5. Write a Java program that implements a multi-thread application that has three threads. First thread generates a random integer every 1 second and if the value is even, the second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of the cube of the number.
6. Write a Java program for the following:
   - Create a doubly linked list of elements.
   - Delete a given element from the above list.
   - Display the contents of the list after deletion.
7. Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with “Stop” or “Ready” or “Go” should appear above the buttons in the selected color. Initially, there is no message shown.
8. Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle,

Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.

9. Suppose that a table named Table.txt is stored in a text file. The first line in the file is the header, and the remaining lines correspond to rows in the table. The elements are separated by commas. Write a java program to display the table using Labels in Grid Layout.
10. Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired (Use Adapter classes).
11. Write a Java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t). It takes a name or phone number as input and prints the corresponding other value from the hash table (hint: use hash tables).
12. Write a Java program that correctly implements the producer – consumer problem using the concept of inter thread communication.
13. Write a Java program to list all the files in a directory including the files present in all its subdirectories

TEXT BOOKS:

1. Java for Programmers, P. J. Deitel and H. M. Deitel, 10th Edition Pearson education.
2. Thinking in Java, Bruce Eckel, Pearson Education.
3. Java Programming, D. S. Malik and P. S. Nair, Cengage Learning.
4. Core Java, Volume 1, 9th edition, Cay S. Horstmann and G Cornell, Pearson.

OBJECT ORIENTED PROGRAMMING THROUGH JAVA LAB

Prerequisites: A Course on "Programming for problem solving".

Course Objectives:

- To write programs using abstract classes.
- To write programs for solving real world problems using the java collection framework.
- To write multithreaded programs.
- To write GUI programs using swing controls in Java.
- To introduce java compiler and eclipse platform.
- To impart hands-on experience with java programming.

Course Outcomes:

- Able to write programs for solving real world problems using the java collection framework.
- Able to write programs using abstract classes.
- Able to write multithreaded programs.
- Able to write GUI programs using swing controls in Java.

Note:

1. Use LINUX and MySQL for the Lab Experiments. Though not mandatory, encourage the use of the Eclipse platform.
2. The list suggests the minimum program set. Hence, the concerned staff is requested to add more problems to the list as needed.

DEPARTMENT OF CSE (DATA SCIENCE)

Safety instructions of Computer Lab

Electrical & Equipment Safety

Do not touch cables, plugs, or sockets unless instructed.
Do not switch off power supply while systems are running.
Report any damaged wires, sparks, or loose connections immediately.
Ensure your hands are dry before touching any electrical equipment.

System Usage

Do not install or uninstall any software without permission.
Use only authorized login credentials; never share your password.
Avoid unnecessary keyboard or mouse usage during class sessions.
Do not attempt to fix hardware/software issues yourself—report to the lab technician.

Cyber & Data Safety

Do not access inappropriate or unauthorized websites.
Never insert unknown USB drives or devices into the computers.
Save work regularly and take backups of important files.
Log out of your account after use to protect your data.

Ergonomics & Health

Sit in a correct posture to avoid neck and back strain.
Take short breaks to rest your eyes if working for long periods.
Adjust monitor brightness to a comfortable level.

Emergency Protocols

In case of a fire or electrical fault, shut down the computer, leave the lab immediately, and inform authorities.
Know the location of the nearest fire extinguisher and emergency exit.

Before Leaving the Lab

Close all applications and log off properly.
Push in your chair and keep the workspace tidy.
Leave the lab only after permission is granted by the instructor.

JBREC
Joginpally BR Engineering College
UGC AUTONOMOUS

DEPARTMENT OF CSE (DATA SCIENCE)

Do's and Don'ts of Computer Lab

Do's in a Computer Lab

- Enter quietly and maintain silence to avoid disturbing others.
- Follow all instructions given by the instructor or lab assistant.
- No food, drinks, or water bottles allowed near computers to prevent spills.
- Keep bags and personal belongings away from walkways to avoid tripping hazards.
- Log in with your own credentials; keep your password confidential in the same computer allotted.
- Handle all equipment with care.
- Report any hardware or software issues immediately to the instructor or technician.
- Keep your workspace clean and organized.
- Use the computers only for academic and authorized purposes.
- Back up your work regularly on external storage or cloud services.
- Maintain proper posture while working to avoid strain.
- Shut down or log off the system properly after use.
- Respect other users; maintain silence and discipline in the lab.

Don'ts in a Computer Lab

- Don't eat, drink, or chew gum near computers.
- Don't install or uninstall software without permission.
- Don't access social media, games, or unrelated websites during lab sessions.
- Don't insert unknown or personal USB devices without scanning for viruses.
- Don't tamper with system settings or network cables.
- Don't move or swap any hardware components.
- Don't share your login details with anyone.
- Don't attempt to fix or open hardware yourself.
- Don't leave the workstation without saving your work.
- Don't disturb others—avoid loud talking, music, or any kind of distraction.

JBREC
Joginpally BR Engineering College
UGC AUTONOMOUS

DEPARTMENT OF CSE (DATA SCIENCE)

CONTENTS

Course Objectives:

- To write programs using abstract classes.
- To write programs for solving real world problems using the java collection framework.
- To write multithreaded programs.
- To write GUI programs using swing controls in Java.
- To introduce java compiler and eclipse platform.
- To impart hands-on experience with java programming.

Course Outcomes:

- Able to write programs for solving real world problems using the java collection framework.
- Able to write programs using abstract classes.
- Able to write multithreaded programs.
- Able to write GUI programs using swing controls in Java.

Note:

1. Use LINUX and MySQL for the Lab Experiments. Though not mandatory, encourage the use of the Eclipse platform.
2. The list suggests the minimum program set. Hence, the concerned staff is requested to add more problems to the list as needed.

List of Experiments:

1. Use Eclipse or Net bean platform and acquaint yourself with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.
2. Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.
3. A) Develop an applet in Java that displays a simple message.
   B) Develop an applet in Java that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named "Compute" is clicked.
4. Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.
5. Write a Java program that implements a multi-thread application that has three threads. First thread generates a random integer every 1 second and if the value is even, the

second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of the cube of the number.

6. Write a Java program for the following:
   A. Create a doubly linked list of elements.
   B. Delete a given element from the above list.
   C. Display the contents of the list after deletion.
7. Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with "Stop" or "Ready" or "Go" should appear above the buttons in the selected color. Initially, there is no message shown.
8. Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle, Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.
9. Suppose that a table named Table.txt is stored in a text file. The first line in the file is the header, and the remaining lines correspond to rows in the table. The elements are separated by commas. Write a java program to display the table using Labels in Grid Layout.
10. Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired (Use Adapter classes).
11. Write a Java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t). It takes a name or phone number as input and prints the corresponding other value from the hash table (hint: use hash tables).
12. Write a Java program that correctly implements the producer – consumer problem using the concept of inter thread communication.
13. Write a Java program to list all the files in a directory including the files present in all its subdirectories.

REFERENCE BOOKS:

1. Java for Programmers, P. J. Deitel and H. M. Deitel, 10th Edition Pearson education.
2. Thinking in Java, Bruce Eckel, Pearson Education.
3. Java Programming, D. S. Malik and P. S. Nair, Cengage Learning.
4. Core Java, Volume 1, 9th edition, Cay S. Horstmann and G Cornell, Pearson.

1. **Use Eclipse or Net bean platform and acquaint yourself with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.**

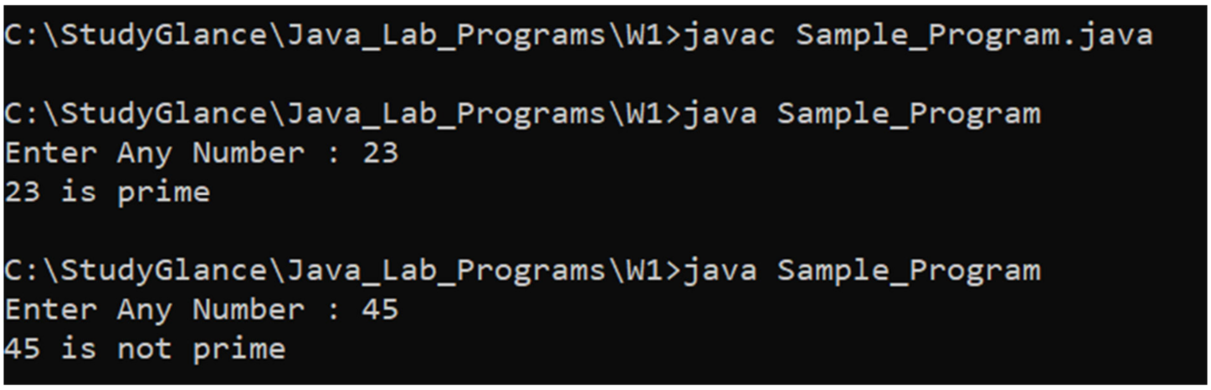
Source Code:

Sample_Program.java

```
/* Sample java program to check given number is prime or not */
//Importing packages
import java.lang.System;
import java.util.Scanner;
// Creating Class
class Sample_Program {
   // main method
   public static void main(String args[]) {
      int i,count=0,n;
      // creating scanner object
      Scanner sc=new Scanner(System.in);
      // get input number from user
      System.out.print("Enter Any Number : ");
      n=sc.nextInt();
      // logic to check prime or not
      for(i=1;i<=n;i++) {
         if(n%i==0) {
               count++;
         }
      }
      if(count==2)
         System.out.println(n+" is prime");
      else
```

```
            System.out.println(n+" is not prime");
        }
    }
```

Output:



```
C:\StudyGlance\Java_Lab_Programs\W1>javac Sample_Program.java

C:\StudyGlance\Java_Lab_Programs\W1>java Sample_Program
Enter Any Number : 23
23 is prime

C:\StudyGlance\Java_Lab_Programs\W1>java Sample_Program
Enter Any Number : 45
45 is not prime
```

2. **Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -,*, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.**

Source Code:

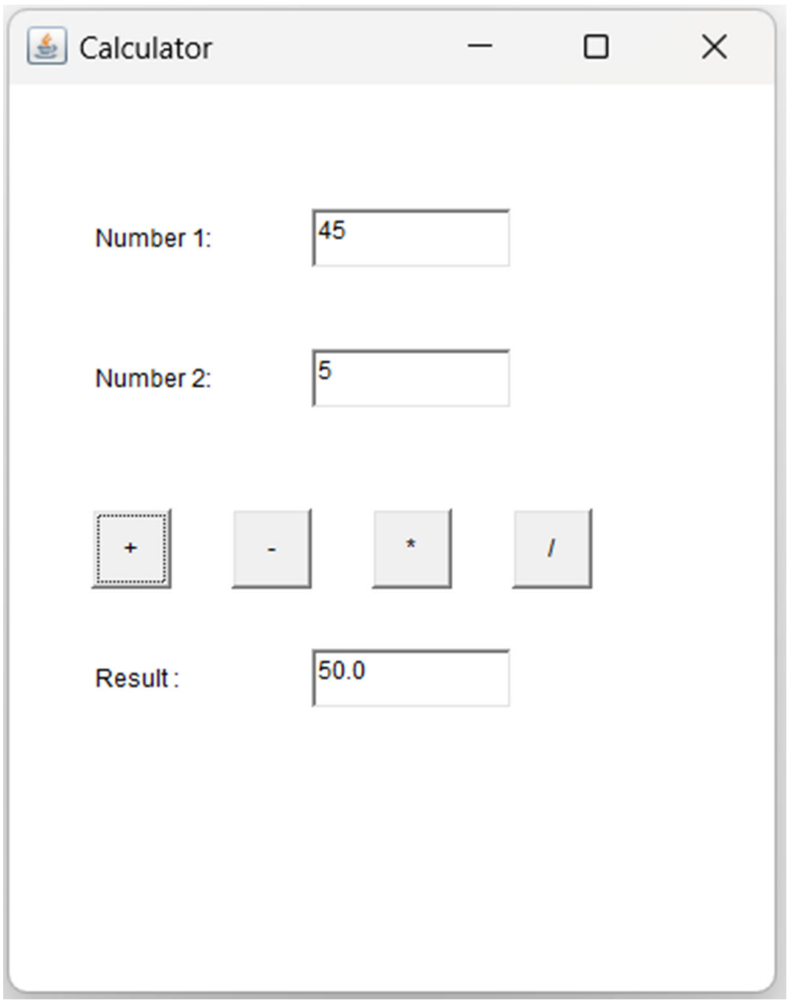
MyCalculator.java

```
/* Program to create a Simple Calculator */
import java.awt.*;
import java.awt.event.*;
public class MyCalculator extends Frame implements ActionListener {
   double num1,num2,result;
   Label lbl1,lbl2,lbl3;
   TextField tf1,tf2,tf3;
   Button btn1,btn2,btn3,btn4;
   char op;
   MyCalculator() {
      lbl1=new Label("Number 1: ");
      lbl1.setBounds(50,100,100,30);

      tf1=new TextField();
      tf1.setBounds(160,100,100,30);

      lbl2=new Label("Number 2: ");
      lbl2.setBounds(50,170,100,30);

      tf2=new TextField();
      tf2.setBounds(160,170,100,30);

      btn1=new Button("+");
      btn1.setBounds(50,250,40,40);
```

```
btn2=new Button("-");
btn2.setBounds(120,250,40,40);

btn3=new Button("*");
btn3.setBounds(190,250,40,40);

btn4=new Button("/");
btn4.setBounds(260,250,40,40);

lbl3=new Label("Result : ");
lbl3.setBounds(50,320,100,30);

tf3=new TextField();
tf3.setBounds(160,320,100,30);

btn1.addActionListener(this);
btn2.addActionListener(this);
btn3.addActionListener(this);
btn4.addActionListener(this);

add(lbl1); add(lbl2); add(lbl3);
add(tf1);  add(tf2);  add(tf3);
add(btn1); add(btn2); add(btn3); add(btn4);

setSize(400,500);
setLayout(null);
setTitle("Calculator");
setVisible(true);
```

```
    }

    public void actionPerformed(ActionEvent ae) {

        num1 = Double.parseDouble(tf1.getText());
        num2 = Double.parseDouble(tf2.getText());

        if(ae.getSource() == btn1)
        {
            result = num1 + num2;
            tf3.setText(String.valueOf(result));
        }
        if(ae.getSource() == btn2)
        {
            result = num1 - num2;
            tf3.setText(String.valueOf(result));
        }
        if(ae.getSource() == btn3)
        {
            result = num1 * num2;
            tf3.setText(String.valueOf(result));
        }
        if(ae.getSource() == btn4)
        {
                result = num1 / num2;
                tf3.setText(String.valueOf(result));
        }
    }
    public static void main(String args[]) {
        MyCalculator calc=new MyCalculator();
    }
}
```

Output:



3. **A) Develop an applet in Java that displays a simple message.**
   B) Develop an applet in Java that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named "Compute" is clicked.

Source Code: A

FirstApplet.java

```
/* Develop a applet to display the simple message */
import java.awt.*;
import java.applet.*;
/*<applet code="FirstApplet" width=400 height=300></applet>*/
public class FirstApplet extends Applet {
public void paint(Graphics g) {
   g.setColor(Color.blue);
   Font font = new Font("Arial", Font.BOLD, 16);
   g.setFont(font);
   g.drawString("This is My First Applet",60,110);
   }
}
```

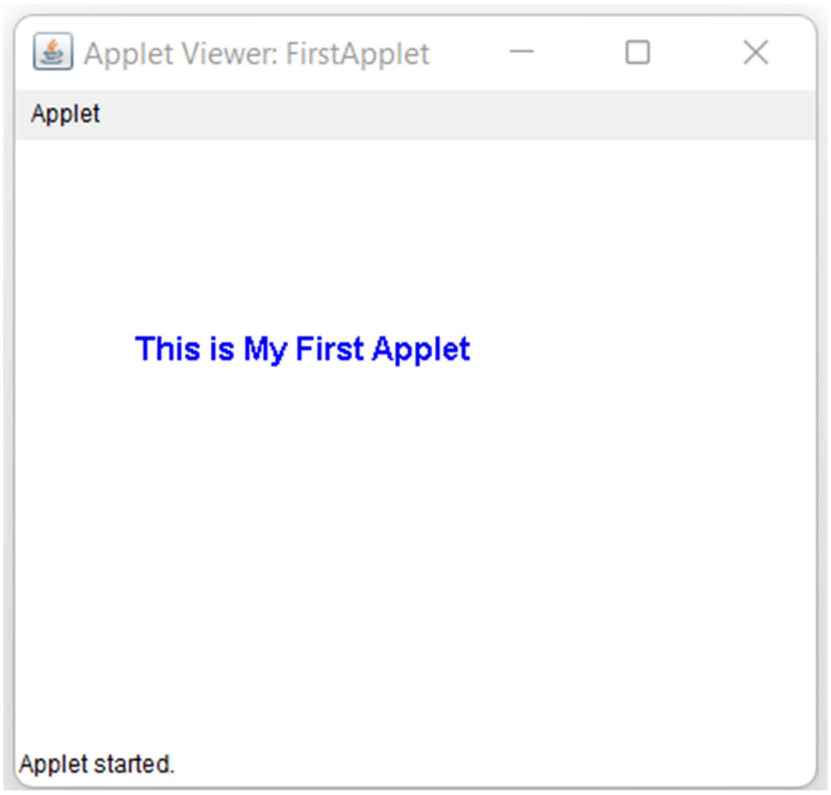
Source Code: B

FactorialApplet.java

```
/** Develop applet to find factorial of the given number */
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
/*<applet code="FactorialApplet" width=500 height=250>
</applet>*/
public class FactorialApplet extends Applet implements ActionListener {
   Label L1,L2;
   TextField T1,T2;
   Button B1;
```

```
    public void init() {
        L1=new Label("Enter any Number : ");
        add(L1);
        T1=new TextField(10);
        add(T1);
        L2=new Label("Factorial of Num : ");
        add(L2);
        T2=new TextField(10);
        add(T2);
        B1=new Button("Compute");
        add(B1);
        B1.addActionListener(this);
    }
    public void actionPerformed(ActionEvent e) {
        if(e.getSource()==B1)
        {
        int value=Integer.parseInt(T1.getText());
        int fact=factorial(value);
        T2.setText(String.valueOf(fact));
        }
    }
    int factorial(int n) {
        if(n==0)
            return 1;
        else
            return n*factorial(n-1);
    }
}
```

Output:



This is My First Applet

Applet started.

4. **Write a Java program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num 2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.**

Source Code:

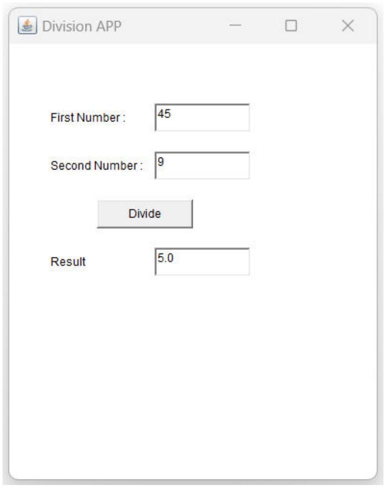
DivisionApp.java

```
/* Develop an java program to perform division */
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class DivisionApp extends Frame implements ActionListener {
   Label L1,L2,L3;
   TextField T1,T2,T3;
   Button B1;
   DivisionApp() {
      L1=new Label("First Number :");
      L1.setBounds(50,100,100,30);
      add(L1);

      T1=new TextField(10);
      T1.setBounds(160,100,100,30);
      add(T1);

      L2=new Label("Second Number :");
      L2.setBounds(50,150,100,30);
      add(L2);

      T2=new TextField(10);
      T2.setBounds(160,150,100,30);
```

```
        add(T2);

        B1=new Button("Divide");
        B1.setBounds(100,200,100,30);
        add(B1);
        B1.addActionListener(this);

        L3=new Label("Result");
        L3.setBounds(50,250,100,30);
        add(L3);

        T3=new TextField();
        T3.setBounds(160,250,100,30);
        add(T3);

        setSize(400,500);
        setLayout(null);
        setTitle("Division APP");
        setVisible(true);
    }
    public void actionPerformed(ActionEvent e) {
        if(e.getSource()==B1) {
            try {
                double value1=Double.parseDouble(T1.getText());
                double value2=Double.parseDouble(T2.getText());
                double result=value1 % value2;
                T3.setText(String.valueOf(result));
            }
            catch(NumberFormatException nfe) {
```

```
            JOptionPane.showMessageDialog(this,"Not a number");
         }
         catch(ArithmeticException ae) {
            JOptionPane.showMessageDialog(this,"Divided by Zero");
         }
      }
   }
   public static void main(String args[]) {
      DivisionApp div=new DivisionApp();
   }
}
```

Output:

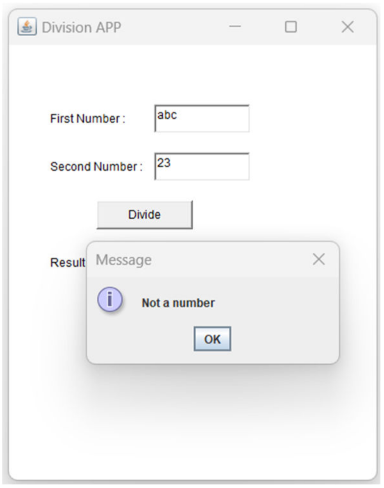

Division APP

First Number : 45

Second Number : 9

Divide

Result 5.0


Division APP

First Number : abc

Second Number : 23

Divide

Result

Message

Not a number

OK

5. **Write a Java program that implements a multi-thread application that has three threads. First thread generates a random integer every 1 second and if the value is even, the second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of the cube of the number.**

Source Code:

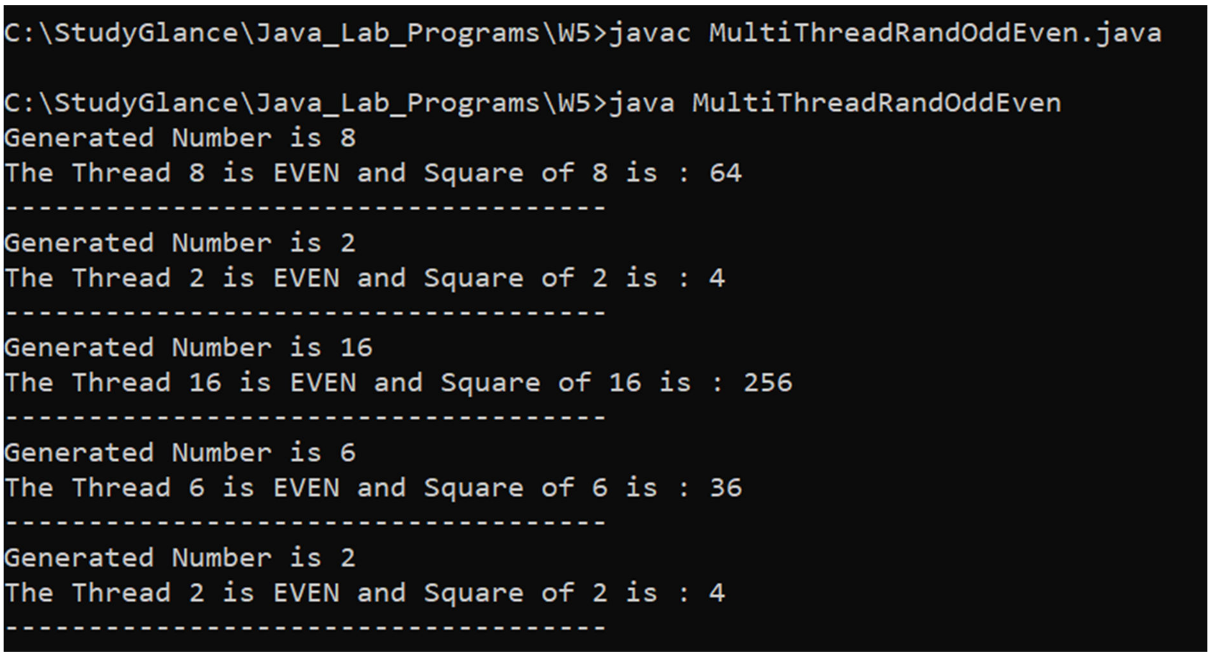
MultiThreadRandOddEven.java

```
import java.util.*;
// class for Even Number
class EvenNum implements Runnable {
   public int a;
   public EvenNum(int a) {
      this.a = a;
   }
   public void run() {
      System.out.println("The Thread "+ a +" is EVEN and Square of " + a + " is : " + a * a);
   }
} // class for Odd Number
class OddNum implements Runnable {
   public int a;
   public OddNum(int a)  {
      this.a = a;
   }
   public void run()  {
      System.out.println("The Thread "+ a +" is ODD and Cube of " + a + " is: " + a * a * a);
   }
}
// class to generate random number
class RandomNumGenerator extends Thread  {
   public void run() {
      int n = 0;
```

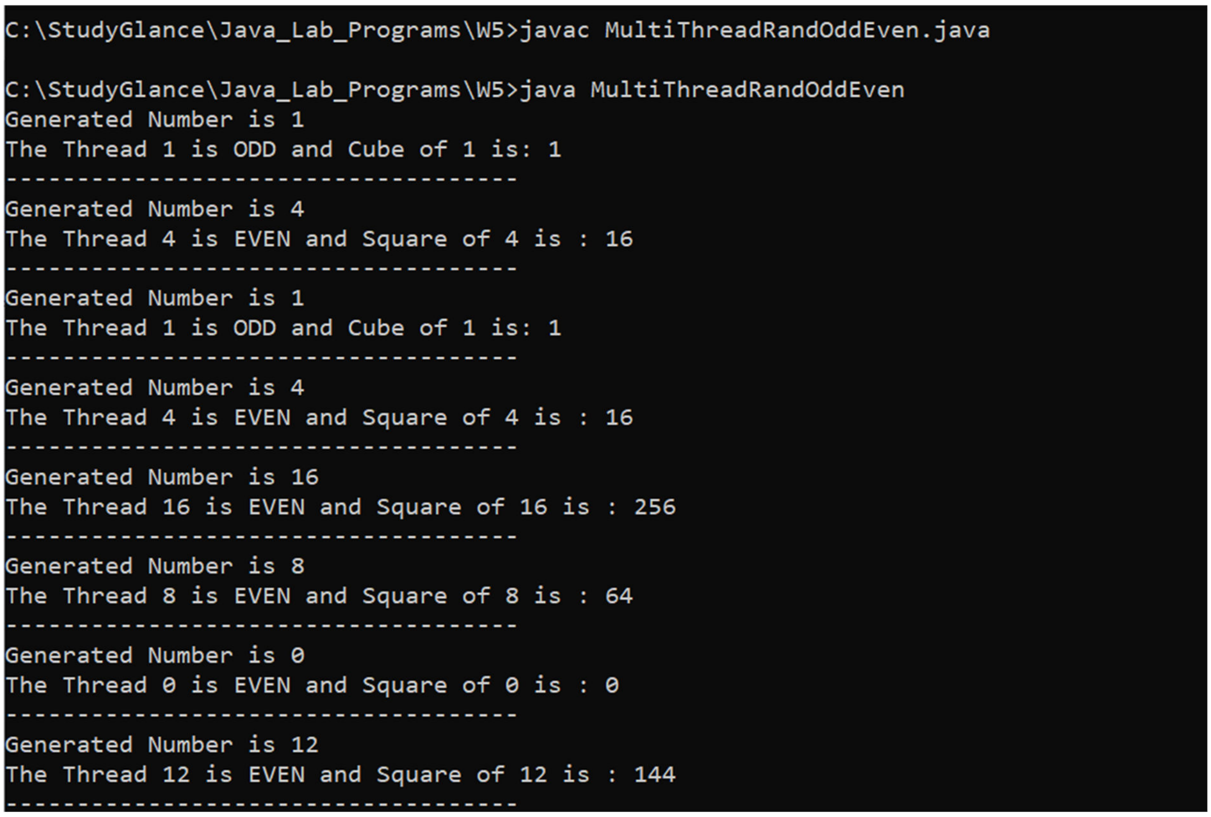
```
        Random rand = new Random();
        try  {
            for (int i = 0; i < 10; i++) {
                n = rand.nextInt(20);
                System.out.println("Generated Number is " + n);
                // check if random number is even or odd
                if (n % 2 == 0) {
                    Thread thread1 = new Thread(new EvenNum(n));
                    thread1.start();
                }
                else {
                    Thread thread2 = new Thread(new OddNum(n));
                    thread2.start();
                }
        // thread wait for 1 second
        Thread.sleep(1000);
        System.out.println("------------------------------------");
            }
        }
        catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    } }
// Driver class
public class MultiThreadRandOddEven {
    public static void main(String[] args) {
        RandomNumGenerator rand_num = new RandomNumGenerator();
        rand_num.start();
}
```

}

Output:



```
C:\StudyGlance\Java_Lab_Programs\W5>javac MultiThreadRandOddEven.java

C:\StudyGlance\Java_Lab_Programs\W5>java MultiThreadRandOddEven
Generated Number is 8
The Thread 8 is EVEN and Square of 8 is : 64
------------------------------------
Generated Number is 2
The Thread 2 is EVEN and Square of 2 is : 4
------------------------------------
Generated Number is 16
The Thread 16 is EVEN and Square of 16 is : 256
------------------------------------
Generated Number is 6
The Thread 6 is EVEN and Square of 6 is : 36
------------------------------------
Generated Number is 2
The Thread 2 is EVEN and Square of 2 is : 4
------------------------------------
```



```
C:\StudyGlance\Java_Lab_Programs\W5>javac MultiThreadRandOddEven.java

C:\StudyGlance\Java_Lab_Programs\W5>java MultiThreadRandOddEven
Generated Number is 1
The Thread 1 is ODD and Cube of 1 is: 1
------------------------------------
Generated Number is 4
The Thread 4 is EVEN and Square of 4 is : 16
------------------------------------
Generated Number is 1
The Thread 1 is ODD and Cube of 1 is: 1
------------------------------------
Generated Number is 4
The Thread 4 is EVEN and Square of 4 is : 16
------------------------------------
Generated Number is 16
The Thread 16 is EVEN and Square of 16 is : 256
------------------------------------
Generated Number is 8
The Thread 8 is EVEN and Square of 8 is : 64
------------------------------------
Generated Number is 0
The Thread 0 is EVEN and Square of 0 is : 0
------------------------------------
Generated Number is 12
The Thread 12 is EVEN and Square of 12 is : 144
------------------------------------
```

6. **Write a Java program for the following:**
   a) **Create a doubly linked list of elements.**
   b) **Delete a given element from the above list.**
   c) **Display the contents of the list after deletion.**

Source Code:

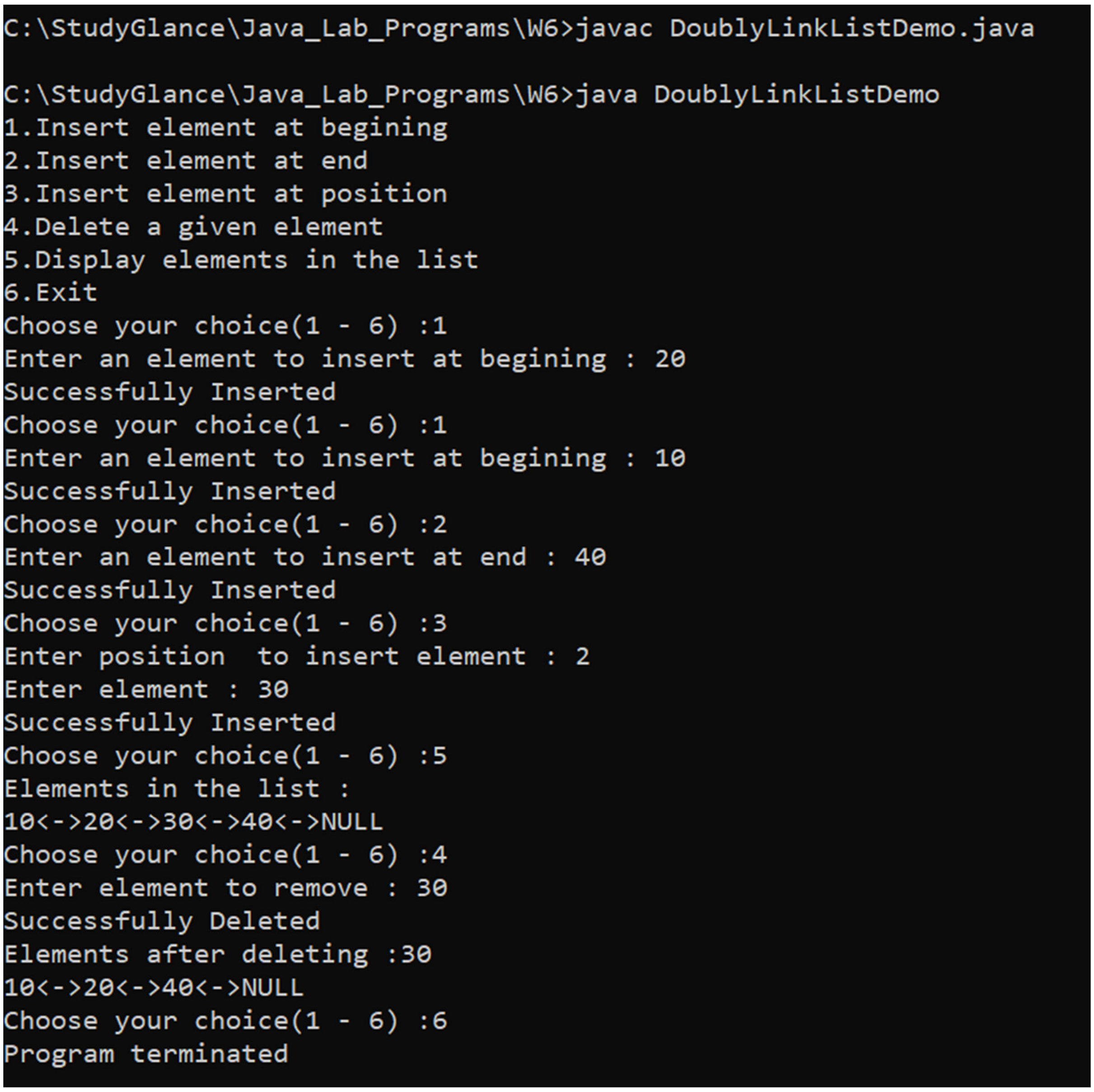
DoublyLinkListDemo.java

```
import java.util.*;

public class DoublyLinkListDemo {

  public static void main(String[] args) {

    int i,ch,element,position;

    LinkedList<Integer> dblList = new LinkedList<Integer>();

    System.out.println("1.Insert element at begining");

    System.out.println("2.Insert element at end");

    System.out.println("3.Insert element at position");

    System.out.println("4.Delete a given element");

    System.out.println("5.Display elements in the list");

    System.out.println("6.Exit");

    Scanner sc=new Scanner(System.in);

    do {

      System.out.print("Choose your choice(1 - 6) :");

      ch=sc.nextInt();

      switch(ch) {

        case 1: // To read element form the user

              System.out.print("Enter an element to insert at begining : ");

              element=sc.nextInt();

              // to add element to doubly linked list at begining

              dblList.addFirst(element);

              System.out.println("Successfully Inserted");

              break;

        case 2: // To read element form the user
```

```
          System.out.print("Enter an element to insert at end : ");
          element=sc.nextInt();
          // to add element to doubly linked list at end
          dblList.addLast(element);
          System.out.println("Successfully Inserted");
          break;
      case 3: // To read position form the user
          System.out.print("Enter position  to insert element : ");
          position=sc.nextInt();
          // checks if the position is lessthan or equal to list size.
          if(position<=dblList.size()) {
            // To read element
            System.out.print("Enter element : ");
            element=sc.nextInt();
            // to add element to doubly linked list at given position
            dblList.add(position,element);
            System.out.println("Successfully Inserted");
          }
          else {
            System.out.println("Enter the size between 0 to"+dblList.size());
          }
          break;
      case 4:    // To read element form the user to remove
          System.out.print("Enter element to remove : ");
          Integer ele_rm;
          ele_rm=sc.nextInt();
          if (dblList.contains(ele_rm)){
            dblList.remove(ele_rm);
            System.out.println("Successfully Deleted");
```

```
                    Iterator itr=dblList.iterator();
                    System.out.println("Elements after deleting :"+ele_rm);
                    while(itr.hasNext()) {
                        System.out.print(itr.next()+"<->");
                    }
                    System.out.println("NULL");
                }
                else {
                    System.out.println("Element not found");
                }
                break;

            case 5:    // To Display elements in the list
                Iterator itr=dblList.iterator();
                System.out.println("Elements in the list :");
                while(itr.hasNext()) {
                    System.out.print(itr.next()+"<->");
                }
                System.out.println("NULL");
                break;

            case 6:    System.out.println("Program terminated");
                break;
            default:System.out.println("Invalid choice");
        }
    }
    while(ch!=6);
  }
}
```

Output:



```
C:\StudyGlance\Java_Lab_Programs\W6>javac DoublyLinkListDemo.java

C:\StudyGlance\Java_Lab_Programs\W6>java DoublyLinkListDemo
1.Insert element at begining
2.Insert element at end
3.Insert element at position
4.Delete a given element
5.Display elements in the list
6.Exit
Choose your choice(1 - 6) :1
Enter an element to insert at begining : 20
Successfully Inserted
Choose your choice(1 - 6) :1
Enter an element to insert at begining : 10
Successfully Inserted
Choose your choice(1 - 6) :2
Enter an element to insert at end : 40
Successfully Inserted
Choose your choice(1 - 6) :3
Enter position  to insert element : 2
Enter element : 30
Successfully Inserted
Choose your choice(1 - 6) :5
Elements in the list :
10<->20<->30<->40<->NULL
Choose your choice(1 - 6) :4
Enter element to remove : 30
Successfully Deleted
Elements after deleting :30
10<->20<->40<->NULL
Choose your choice(1 - 6) :6
Program terminated
```

7. **Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green with radio buttons. On selecting a button, an appropriate message with “Stop” or “Ready” or “Go” should appear above the buttons in the selected color. Initially, there is no message shown.**

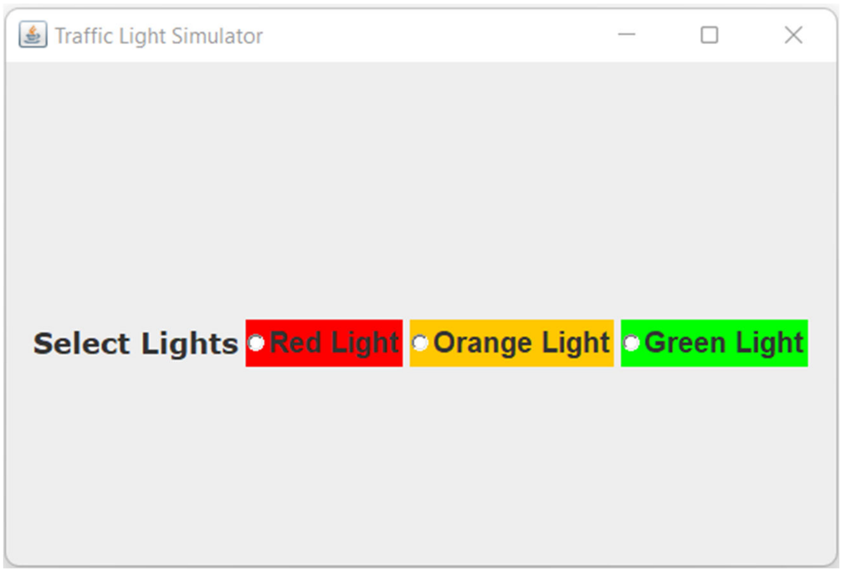
Source Code:

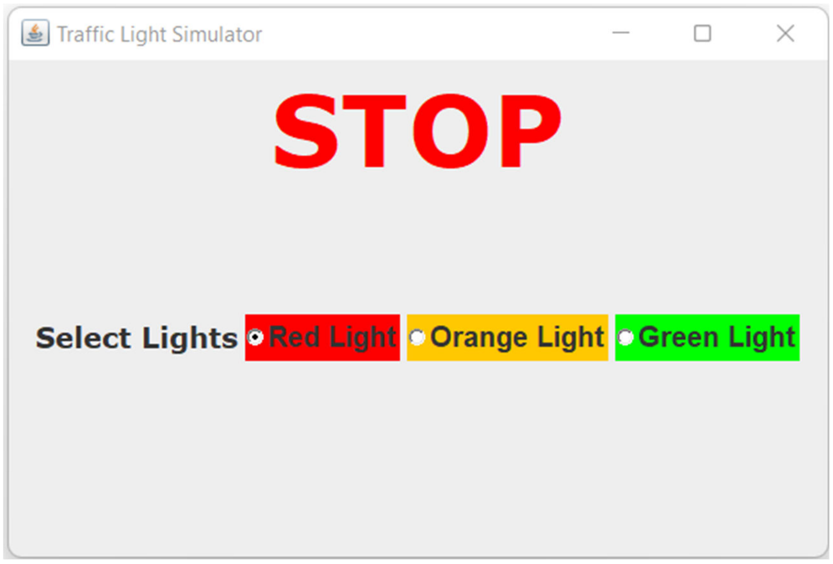
TrafficLights.java

```
import java.awt.*;

import javax.swing.*;

import java.awt.event.*;

class TrafficLights extends JFrame implements ActionListener {

   JLabel lbl1, lbl2;

   JRadioButton rl,gl,ol;

   ButtonGroup bg;

   TrafficLights() {

      Font fontL = new Font("Verdana", Font.BOLD, 70);

      lbl1 = new JLabel();

      lbl1.setBounds(170,30,300,200);

      lbl1.setFont(fontL);

      add(lbl1);


      Font fontR = new Font("Verdana", Font.BOLD, 16);

      lbl2 = new JLabel("Select Lights");

      lbl2.setBounds(20,250,130,50);

      lbl2.setFont(fontR);

      add(lbl2);


      rl = new JRadioButton("Red Light");

      rl.setBounds(160,250,120,50);

      rl.setBackground(Color.RED);

      rl.setFont(fontR);
```

```
        add(rl);
        rl.addActionListener(this);

        ol = new JRadioButton("Orange Light");
        ol.setBounds(300,250,150,50);
        ol.setBackground(Color.ORANGE);
        ol.setFont(fontR);
        add(ol);
        ol.addActionListener(this);

        gl = new JRadioButton("Green Light");
        gl.setBounds(460,250,140,50);
        gl.setBackground(Color.GREEN);
        gl.setFont(fontR);
        add(gl);
        gl.addActionListener(this);

        bg = new ButtonGroup();
        bg.add(rl);bg.add(ol);bg.add(gl);
        setTitle("Traffic Light Simulator");
        setSize(650,400);
        setLayout(null);
        setVisible(true);
    }
    // To read selected item
    public void actionPerformed(ActionEvent ae) {
        if(ae.getSource() == rl)
        {
            lbl1.setText("STOP");
            lbl1.setForeground(Color.RED);
        }
        if(ae.getSource() == ol)
        {
            lbl1.setText("READY");
            lbl1.setForeground(Color.ORANGE);
        }
        if(ae.getSource() == gl)
        {
            lbl1.setText("GO");
            lbl1.setForeground(Color.GREEN);
        }
    }
    // main method
    public static void main(String[] args) {
        new TrafficLights();
    }
}
```

Output:

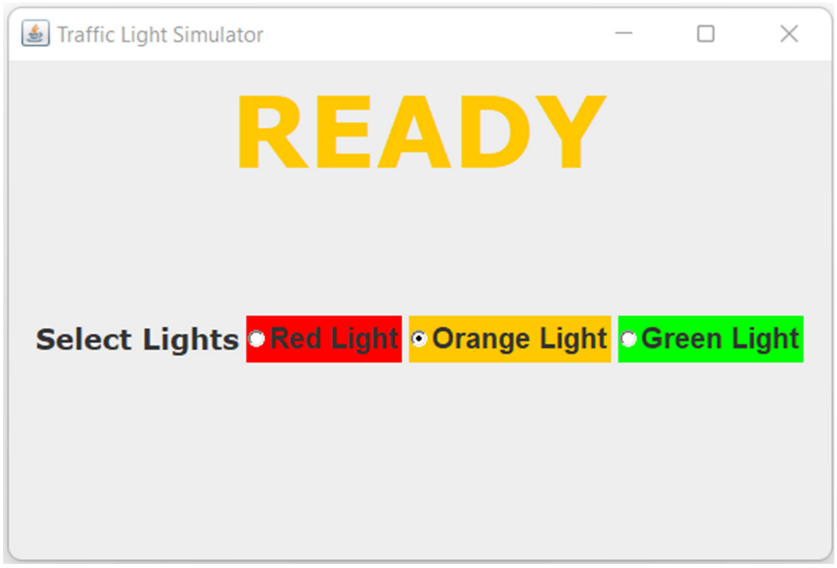




Traffic Light Simulator

STOP

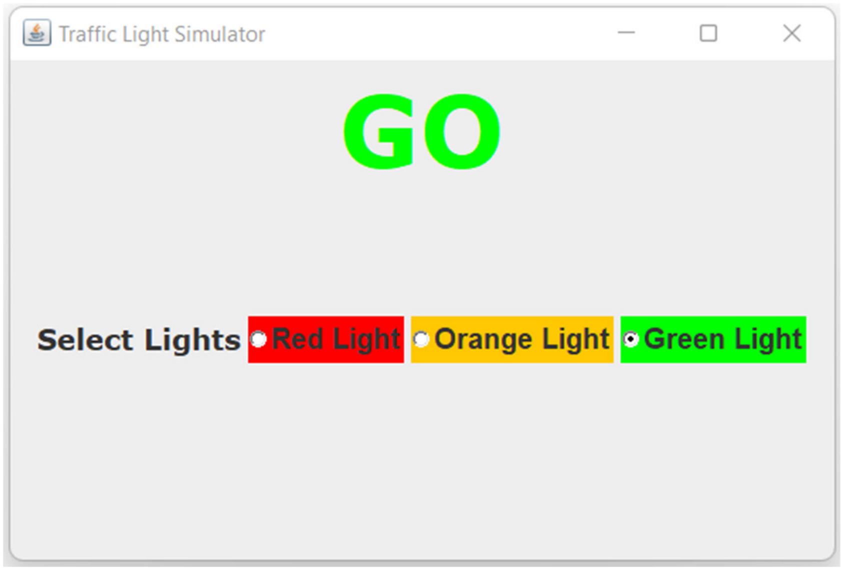
Select Lights [✓ Red Light] Orange Light Green Light



Traffic Light Simulator

READY

Select Lights Red Light [✓ Orange Light] Green Light



Traffic Light Simulator

GO

Select Lights Red Light Orange Light [✓ Green Light]

8. **Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle, Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.**

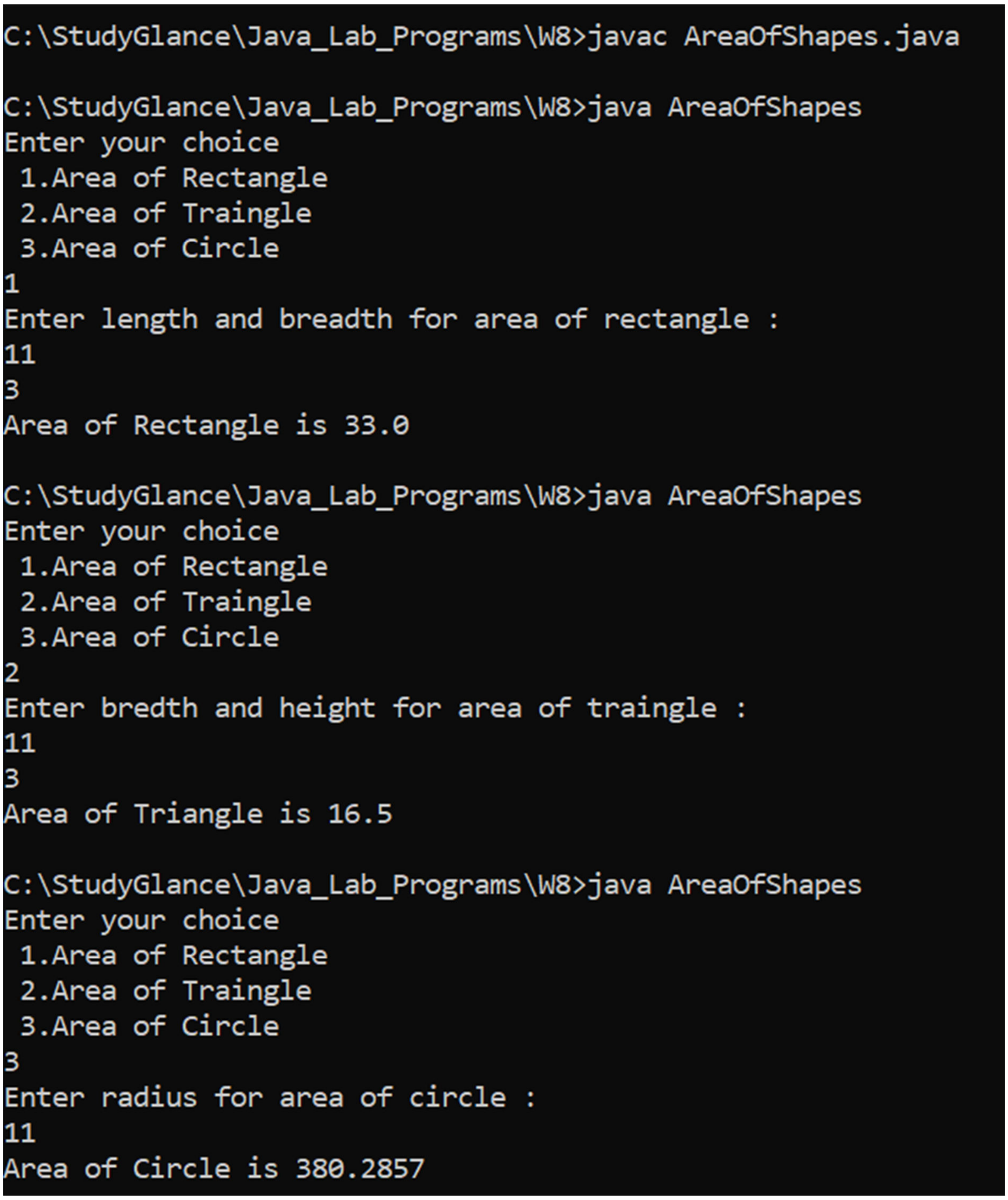
Source Code:

AreaOfShapes.java

```
import java.util.*;
abstract class Shape {
   public int x,y;
   public abstract void printArea();
}
class Rectangle1 extends Shape {
   public void printArea() {
      float area;
      area= x * y;
      System.out.println("Area of Rectangle is " +area);
   }
}
class Triangle extends Shape {
   public void printArea() {
      float area;
      area= (x * y) / 2.0f;
      System.out.println("Area of Triangle is " + area);
   }
}
class Circle extends Shape {
   public void printArea() {
      float area;
      area=(22 * x * x) / 7.0f;
```

```
        System.out.println("Area of Circle is " + area);
    }
}
public class AreaOfShapes {
    public static void main(String[] args) {
        int choice;
        Scanner sc=new Scanner(System.in);
        System.out.println("Menu \n 1.Area of Rectangle \n 2.Area of Traingle \n 3.Area of 
Circle ");
        System.out.print("Enter your choice : ");
        choice=sc.nextInt();
        switch(choice) {
            case 1: System.out.println("Enter length and breadth for area of rectangle : ");
                Rectangle1 r = new Rectangle1();
                r.x=sc.nextInt();
                r.y=sc.nextInt();
                r.printArea();
                break;
            case 2: System.out.println("Enter bredth and height for area  of traingle : ");
                Triangle t = new Triangle();
                t.x=sc.nextInt();
                t.y=sc.nextInt();
                t.printArea();
                break;
            case 3: System.out.println("Enter radius for area of circle : ");
                Circle c = new Circle();
                c.x = sc.nextInt();
                c.printArea();
                break;
            default:System.out.println("Enter correct choice");
        }
    }
}
```

Output:



```
C:\StudyGlance\Java_Lab_Programs\W8>javac AreaOfShapes.java

C:\StudyGlance\Java_Lab_Programs\W8>java AreaOfShapes
Enter your choice
 1.Area of Rectangle
 2.Area of Traingle
 3.Area of Circle
1
Enter length and breadth for area of rectangle :
11
3
Area of Rectangle is 33.0

C:\StudyGlance\Java_Lab_Programs\W8>java AreaOfShapes
Enter your choice
 1.Area of Rectangle
 2.Area of Traingle
 3.Area of Circle
2
Enter bredth and height for area of traingle :
11
3
Area of Triangle is 16.5

C:\StudyGlance\Java_Lab_Programs\W8>java AreaOfShapes
Enter your choice
 1.Area of Rectangle
 2.Area of Traingle
 3.Area of Circle
3
Enter radius for area of circle :
11
Area of Circle is 380.2857
```

9. **Suppose that a table named Table.txt is stored in a text file. The first line in the file is the header, and the remaining lines correspond to rows in the table. The elements are separated by commas. Write a java program to display the table using Labels in Grid Layout.**

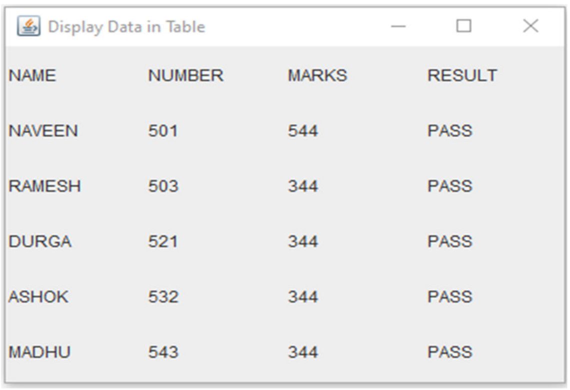
Source Code:

TableText.java

```
import java.io.*;
import java.util.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
class Text_To_Table extends JFrame
{
  public void convertTexttotable()
  {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setSize(400,300);
    GridLayout g = new GridLayout(0, 4);
    setLayout(g);
    try
    {
      FileInputStream fis = new FileInputStream("./Table.txt");
      Scanner sc = new Scanner(fis);
      String[] arrayList;
      String str;
      while (sc.hasNextLine())
      {
        str = sc.nextLine();
        arrayList = str.split(",");
```

```
                for (String i : arrayList)
                {
                    add(new Label(i));
                }
            }
        }
        catch (Exception ex) {
            ex.printStackTrace();
        }
        setVisible(true);
        setTitle("Display Data in Table");
    }
}
public class TableText
{
    public static void main(String[] args)
    {
        Text_To_Table tt = new Text_To_Table();
        tt.convertTexttotable();
    }
}
```

Output:



```
Display Data in Table

NAME        NUMBER      MARKS       RESULT
NAVEEN      501         544         PASS
RAMESH      503         344         PASS
DURGA       521         344         PASS
ASHOK       532         344         PASS
MADHU       543         344         PASS
```

10. Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired (Use Adapter classes).

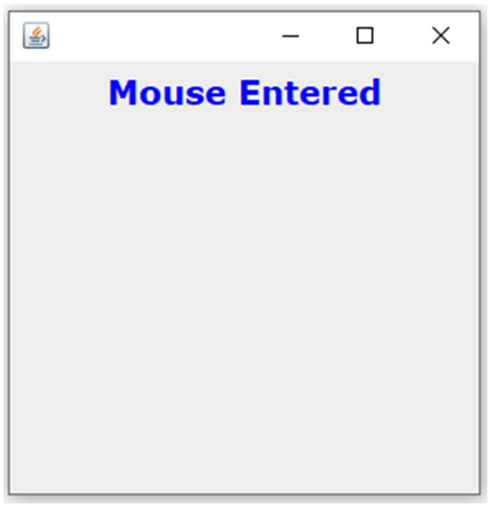
Source Code:

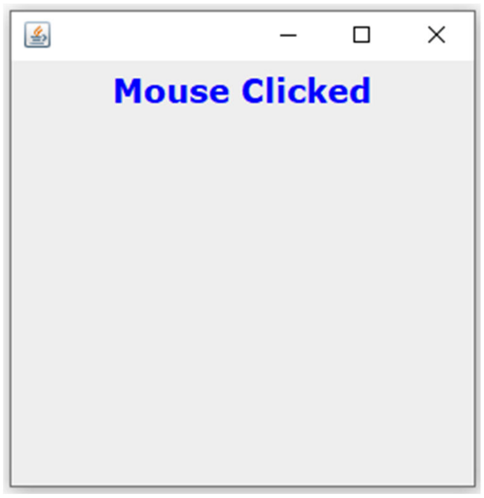
MouseEventPerformer.java

```
import javax.swing.*;
import java.awt.*;
import javax.swing.event.*;
import java.awt.event.*;
class MouseEventPerformer extends JFrame implements MouseListener
{
   JLabel l1;
   public MouseEventPerformer()
   {
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setSize(300,300);
      setLayout(new FlowLayout(FlowLayout.CENTER));
      l1 = new JLabel();
      Font f = new Font("Verdana", Font.BOLD, 20);
      l1.setFont(f);
      l1.setForeground(Color.BLUE);
      add(l1);
      addMouseListener(this);
      setVisible(true);
   }
   public void mouseExited(MouseEvent m)
   {
      l1.setText("Mouse Exited");
   }
   public void mouseEntered(MouseEvent m)
```

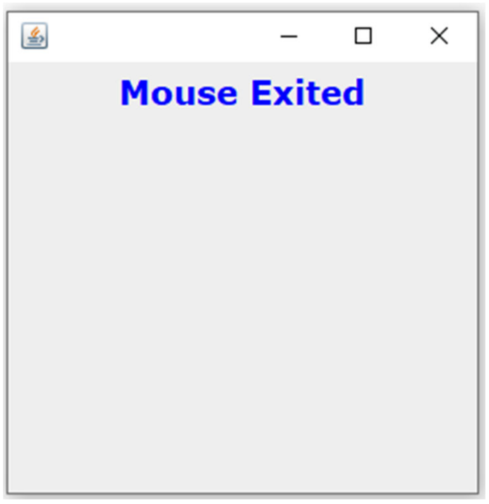
```
  {
    l1.setText("Mouse Entered");
  }
  public void mouseReleased(MouseEvent m)
  {
    l1.setText("Mouse Released");
  }
  public void mousePressed(MouseEvent m)
  {
    l1.setText("Mouse Pressed");
  }
  public void mouseClicked(MouseEvent m)
  {
    l1.setText("Mouse Clicked");
  }
  public static void main(String[] args) {
    MouseEventPerformer mep = new MouseEventPerformer();
  }
}
```

Output:





```
Mouse Clicked
```



```
Mouse Exited
```

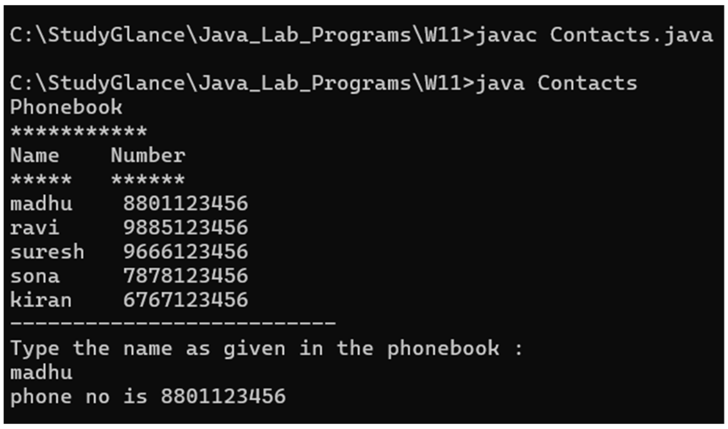
11. **Write a Java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by a tab (\t). It takes a name or phone number as input and prints the corresponding other value from the hash table (hint: use hash tables).**

Source Code:

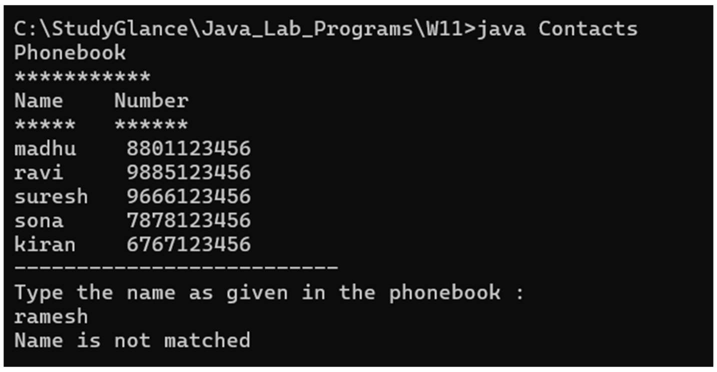
Contacts.java

```
import java.io.*;
import java.util.*;
public class Contacts
{
public static void main(String args[])
{
try
{
FileInputStream fin=new FileInputStream("myfile.txt");
Scanner sc=new Scanner(fin).useDelimiter("\t");
Hashtable<String,String> ht=new Hashtable<String,String> ();
String[] strary;
String record,name;
System.out.println("Phonebook");
System.out.println("***********");
System.out.println("Name \tNumber");
System.out.println("*****\t******");
while(sc.hasNext())
{
String[] ary;
record=sc.nextLine();
strary=record.split("\t");
ht.put(strary[0],strary[1]);
System.out.println(strary[0]+" \t "+strary[1]);
}
System.out.println("--------------------------");
Scanner s=new Scanner(System.in);
System.out.println("Type the name as given in the phonebook : ");
name=s.next();
name=name.toLowerCase();
if(ht.containsKey(name))
{
System.out.println("phone no is "+ht.get(name));
}
else
{
System.out.println("Name is not matched");
}
}
catch(Exception e)
{
System.out.println(e);
}
}
}
```

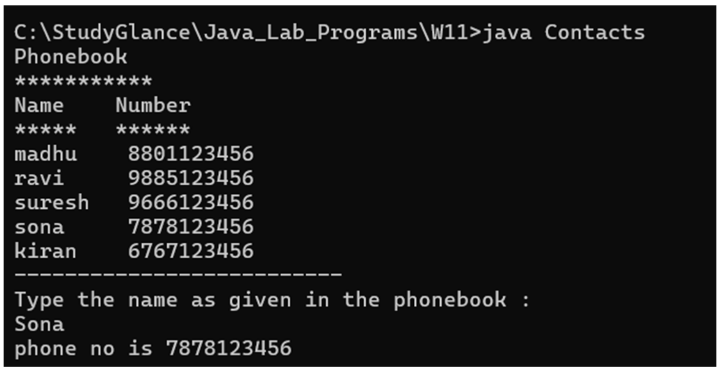
Output:



```
C:\StudyGlance\Java_Lab_Programs\W11>javac Contacts.java

C:\StudyGlance\Java_Lab_Programs\W11>java Contacts
Phonebook
***********
Name     Number
*****    ******
madhu     8801123456
ravi      9885123456
suresh    9666123456
sona      7878123456
kiran     6767123456
--------------------------
Type the name as given in the phonebook :
madhu
phone no is 8801123456
```



```
C:\StudyGlance\Java_Lab_Programs\W11>java Contacts
Phonebook
***********
Name    Number
*****   ******
madhu    8801123456
ravi     9885123456
suresh   9666123456
sona     7878123456
kiran    6767123456
--------------------------
Type the name as given in the phonebook :
ramesh
Name is not matched
```



```
C:\StudyGlance\Java_Lab_Programs\W11>java Contacts
Phonebook
***********
Name    Number
*****   ******
madhu    8801123456
ravi     9885123456
suresh   9666123456
sona     7878123456
kiran    6767123456
--------------------------
Type the name as given in the phonebook :
Sona
phone no is 7878123456
```

12. **Write a Java program that correctly implements the producer – consumer problem using the concept of inter thread communication.**

Source Code:

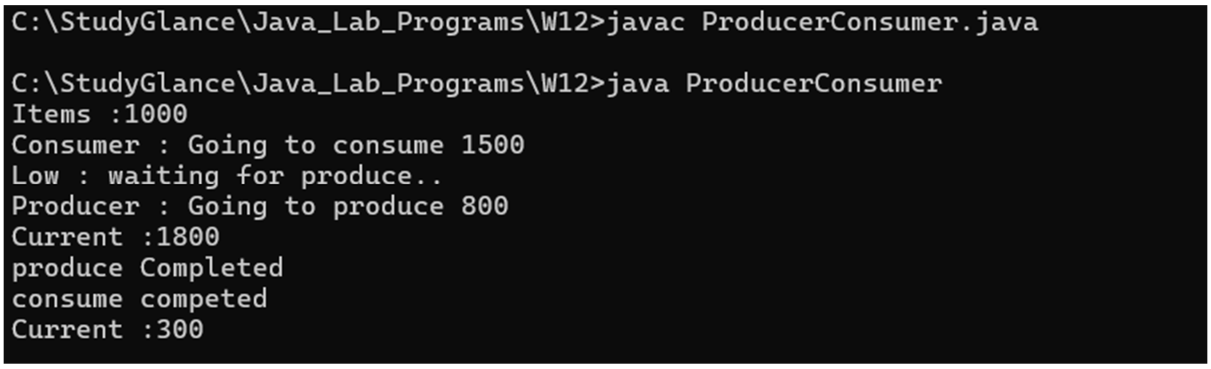
ProducerConsumer.java

```
// create a class
class Items
{
    int count = 1000;
    synchronized void consume(int cnt)
    {
        System.out.println("Items :"+this.count);
        System.out.println("Consumer : Going to consume "+cnt);
        if (this.count < cnt)
        {
            System.out.println("Low : waiting for produce..");
            try
            {
                Thread.sleep(10000);
                wait();
            }
            catch (InterruptedException e)
            { }
        }
        this.count = this.count - cnt;
        System.out.println("consume competed");
        System.out.println("Current :"+this.count);


    }
```

```
   synchronized void produce(int cnt)
   {
      System.out.println("Producer : Going to produce "+cnt);
      this.count = this.count + cnt;
      System.out.println("Current :"+this.count);
      System.out.println("produce Completed");
      notify();
   }
}
// creating a Thread1
class Consumer extends Thread
{
   Items pt;
   Consumer(Items pt)
   {
      this.pt = pt;
   }
   public void run()
   {
      pt.consume(1500);
   }
}
// creating a Thread2
class Producer extends Thread
{
   Items pt;
   Producer(Items pt)
   {
      this.pt = pt;
```

```
    }
    public void run()
    {
        pt.produce(800);
    }


}
// main class
class ProducerConsumer
{
    public static void main(String[] args)
    {
        Items obj = new Items();
        Consumer t1 = new Consumer(obj);
        Producer t2 = new Producer(obj);
        t1.start();
        t2.start();
    }
}
```

Output:



```
C:\StudyGlance\Java_Lab_Programs\W12>javac ProducerConsumer.java

C:\StudyGlance\Java_Lab_Programs\W12>java ProducerConsumer
Items :1000
Consumer : Going to consume 1500
Low : waiting for produce..
Producer : Going to produce 800
Current :1800
produce Completed
consume competed
Current :300
```

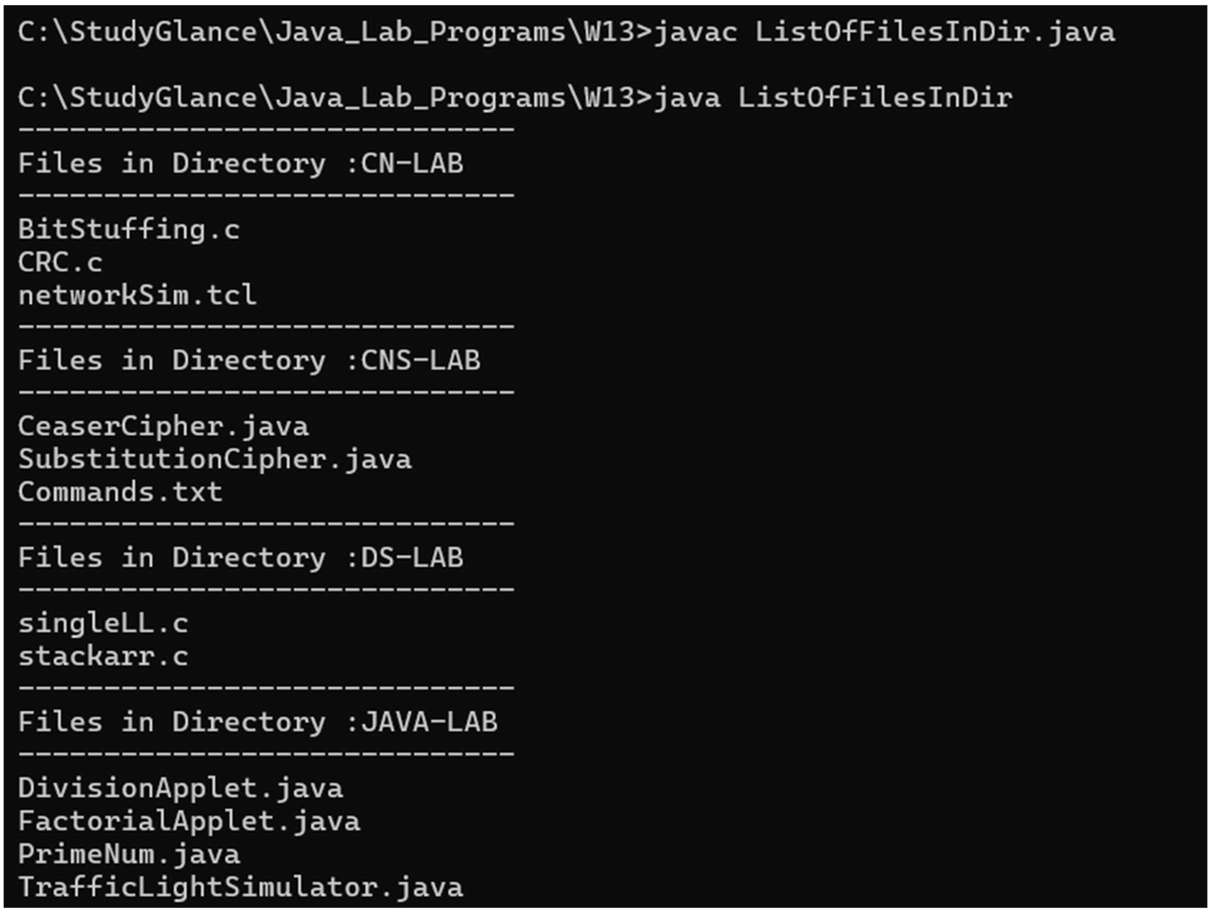
13. Write a Java program to list all the files in a directory including the files present in all its subdirectories.

Source Code:

ListOfFilesInDir.java

```
import java.io.*;
public class ListOfFilesInDir {
  public static void listOfFiles(File dirPath){
    File fileList[] = dirPath.listFiles();
    for(File file : fileList) {
      if(file.isFile()) {
        System.out.println(file.getName());
      } else {
         System.out.println("-----------------------------");
         System.out.println("Files in Directory :"+file.getName());
         System.out.println("-----------------------------");
         listOfFiles(file);
      }
    }
  }
  public static void main(String args[]) throws IOException {
    File file = new File("C:/Labs");
    listOfFiles(file);
  }
}
```

Output:



```
C:\StudyGlance\Java_Lab_Programs\W13>javac ListOfFilesInDir.java

C:\StudyGlance\Java_Lab_Programs\W13>java ListOfFilesInDir
-----------------------------
Files in Directory :CN-LAB
-----------------------------
BitStuffing.c
CRC.c
networkSim.tcl
-----------------------------
Files in Directory :CNS-LAB
-----------------------------
CeaserCipher.java
SubstitutionCipher.java
Commands.txt
-----------------------------
Files in Directory :DS-LAB
-----------------------------
singleLL.c
stackarr.c
-----------------------------
Files in Directory :JAVA-LAB
-----------------------------
DivisionApplet.java
FactorialApplet.java
PrimeNum.java
TrafficLightSimulator.java
```