Sri Indu
College of Engineering & Technology
UGC Autonomous Institution
Recognized under 2(f) & 12(B) of UGC Act 1956,
NAAC, Approved by AICTE &
Permanently Affiliated to JNTUH

JAVA PROGRAMMING LAB (R22CSI2228)

LAB MANUAL

II Year II Semester

DEPARTMENT OF INFORMATION TECHNOLOGY

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

B. TECH –INFORMATION TECHNOLOGY

INSTITUTION VISION

To be a premier Institution in Engineering & Technology and Management with competency, valuesand social consciousness.

INSTITUTION MISSION

IM1 Provide high quality academic programs, training activities and research facilities.

IM2 Promote Continuous Industry-Institute Interaction for Employability, Entrepreneurship, Leadership and Research aptitude among stakeholders.

IM3 Contribute to the Economical and technological development of the region, state and nation.

DEPARTMENT VISION

To be a recognized knowledge center in the field of Information Technology with self - motivated, employable engineers to society.

DEPARTMENT MISSION

The Department has following Missions:

DM1 To offer high quality student centric education in Information Technology.

DM2 To provide a conducive environment towards innovation and skills.

DM3 To involve in activities that provide social and professional solutions.

DM4 To impart training on emerging technologies namely cloud computing and IOT with involvement of stake holders.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: **Higher Studies:** Graduates with an ability to apply knowledge of Basic sciences and programming skills in their career and higher education.

PEO2: **Lifelong Learning:** Graduates with an ability to adopt new technologies for ever changing IT industry needs through Self-Study, Critical thinking and Problem solving skills.

PEO3: **Professional skills:** Graduates will be ready to work in projects related to complex problems involving multi-disciplinary projects with effective analytical skills.

PEO4: **Engineering Citizenship:** Graduates with an ability to communicate well and exhibitsocial, technical and ethical responsibility in process or product.

PROGRAM OUTCOMES (POs) & PROGRAM SPECIFIC OUTCOMES (PSOs)

PO	Description
PO 1	**Engineering Knowledge:** Apply knowledge of mathematics, natural science, computing, engineering fundamentals and an engineering specialization as specified in WK1 to WK4 respectively to develop to the solution of complex engineering problems.
PO 2	**Problem Analysis:** Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development. (WK1 to WK4)
PO 3	**Design/Development of Solutions:** Design creative solutions for complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required. (WK5)
PO 4	**Conduct Investigations of Complex Problems:** Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions. (WK8).
PO 5	**Engineering Tool Usage:** Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. (WK2 and WK6)
PO 6	**The Engineer and The World:** Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. (WK1, WK5, and WK7).
PO 7	**Ethics:** Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. (WK9)
PO 8	**Individual and Collaborative Team work:** Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.
PO 10	**Project Management and Finance:** Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.
PO 11	**Life-Long Learning:** Recognize the need for, and have the preparation and ability for i) independent and life-long learning ii) adaptability to new and emerging technologies and iii) critical thinking in the broadest context of technological change. (WK8)
Program Specific Outcomes	
PSO 1	**Software Development:** To apply the knowledge of Software Engineering, Data Communication, Web Technology and Operating Systems for building IOT and Cloud Computing applications.
PSO 2	**Industrial Skills Ability:** Design, develop and test software systems for world-wide network of computers toprovide solutions to real world problems.
PSO 3	**Project implementation:** Analyze and recommend the appropriate IT Infrastructure required for the implementationof a project.

Course Code & Name: R22CSI2228 & JAVA PROGRAMMING LAB

Upon the completion of the course, Students will be able to:

Course Name	Course outcomes
C2228.1	Illustrate the concepts of layout managers, and applets for a given problem. (L3-Apply)
C2228.2	Examine the Exception handling mechanism and multithreading environment by using a java programs. (L3-Apply)
C2228.3	Justify the concept of OOPs as well as the purpose and usage principles of inheritance, polymorphism, encapsulation and method overloading. (L5-Evaluate).
C2228.4	Develop the database management system concepts like create insert delete update select for standalone applications. (L6-Create)
C2228.5	Create GUI based applications through the knowledge on event handling mechanism andApplets. (L6-Create)

Mapping of Course Outcomes (CO's) with PO's:

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	P10	P11	PSO1	PSO2	PSO3
C22L8 .1	3	1	2	1	3	-	-	-	-	-	-	-	1	-
C22L8 .2	3	1	1	1	3	-	-	-	-	-	-	-	1	-
C22L8 .3	1	1	2	3	3	-	-	-	-	-	-	-	1	1
C22L8 .4	1	-	1	1	3	-	-	-	-	-	-	1	1	1
C22L8.5	1	1	2	1	3	-	-	-	-	-	-	1	1	1
C22L8 .6	1	-	-	3	3	-	-	-	-	-	-	-	-	-
C22L8	1.67	1	1.6	1.67	3	-	-	-	-	-	-	1	1	1

JAVA PROGRAMMING LAB MANUAL
R22CSI2228

DEPARTMENT OF INFORMATION TECHNOLOGY

(II- B. Tech. – II– Semester)

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), thosewho come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis / program / experiment details.
3. Student should enter into the laboratory with:
   a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure,Program, Expected Output, etc.,) filled in for the lab session.
   b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed inthe lab.
   c. Proper Dress code and Identity card.

4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allottedto you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book,and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should beutilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severepunishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody foundloitering outside the lab / class without permission during working hours will be treated seriously andpunished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

Head of the Department **Principal**

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC, New Delhi)

B.Tech. - II Year – II Semester

L	T	P	C
0	0	2	1

JAVA PROGRAMMING LAB

Course Objectives:

- To understand OOP principles.
- To understand the Exception Handling mechanism.
- To understand Java collection framework.
- To understand multithreaded programming.
- To understand swing controls in Java.

Course Outcomes:

- Able to write the programs for solving real world problems using Java OOP principles.
- Able to write programs using Exceptional Handling approach.
- Able to write multithreaded applications.
- Able to write GUI programs using swing controls in Java.

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 to demonstrate the OOP principles. [i.e., Encapsulation, Inheritance, Polymorphism and Abstraction]

3. Write a Java program to handle checked and unchecked exceptions. Also, demonstrate the usage of custom exceptions in real time scenario.

4. Write a Java program on Random Access File class to perform different read and write operations.

5. Write a Java program to demonstrate the working of different collection classes. [Use package structure to store multiple classes].

6. Write a program to synchronize the threads acting on the same object. [Consider the example of any reservations like railway, bus, movie ticket booking, etc.]

7. Write a program to perform CRUD operations on the student table in a database using JDBC.

8. 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.

9. 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]

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.

JAVA PROGRAMMING LAB
(R22CSI2228)

EXPERIMENT 1:

Aim:

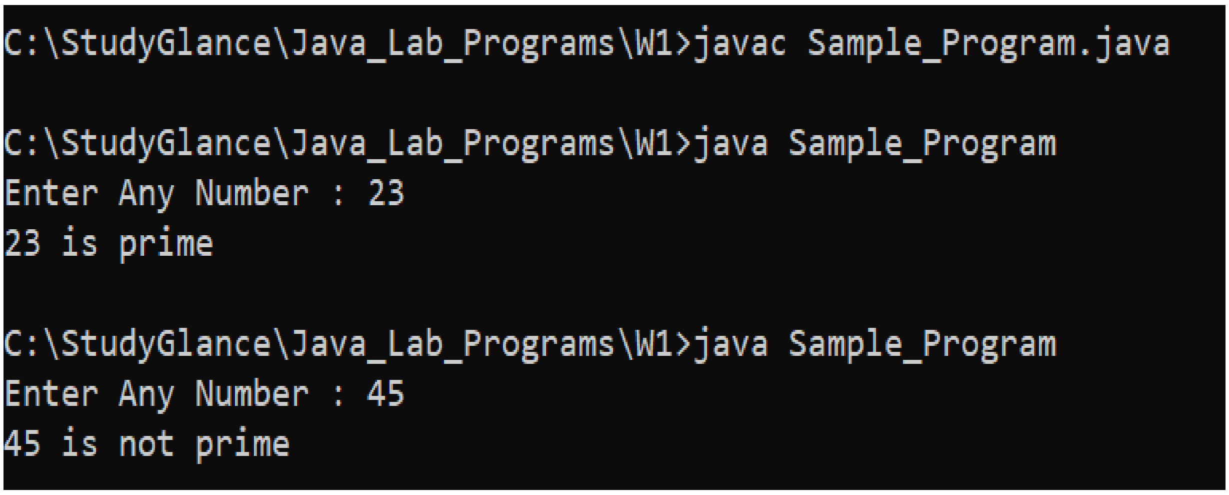
Use eclipse or Netbean platform and acquaint with the various menus, create a test project, add a test class and run it see how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods and classes. Try debug step by step with a small program of 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 */

import java.lang.System;
import java.util.Scanner;
class Sample_Program
{
    public static void main(String args[])
   {
        int i,count=0,n;
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter Any Number : ");
        n=sc.nextInt();
        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
```

Viva Questions:

1. List the features of the Java Programming language
2. Write the syntax for if..else and else..if statements.
3. Write the difference between while loop and do..while loop.
4. Define Break and Continue statement.
5. Write the syntax for Switch statement.

EXPERIMENT 2:

Write a Java program to demonstrate the OOP principles. [i.e., Encapsulation, Inheritance,Polymorphism and Abstraction]

2.i) Encapsulation.

Aim:

Write a Java program to demonstrate the OOP principle of Encapsulation.

Source Code:

File Name : Encapsulation.java

```
class Account
{

        private long acc_no;
        private String name,email;
        private float amount;

        public long getAcc_no()
        {
                return acc_no;
        }
        public void setAcc_no(long acc_no)
        {
                this.acc_no = acc_no;
        }
        public String getName()
        {
                return name;
        }
        public void setName(String name)
        {
                this.name = name;
        }
        public float getAmount()
        {
                return amount;
        }
        public void setAmount(float amount)
        {
                this.amount = amount;
        }

}
public class Encapsulation
{
        public static void main(String[] args)
        {
                Account acc=new Account();
```

```
                    acc.setAcc_no(7560504000L);
                    acc.setName("Sonoo Jaiswal");
                    acc.setAmount(500000f);

            System.out.println("Acc.No : "+acc.getAcc_no()+"\nName : "+acc.getName()+"\nAmount :
                    "+acc.getAmount());
        }
}
```

Output:

```
C:\jdk.19\bin>javac Encapsulation.java
C:\jdk.19\bin>java Encapsulation
Acc.No : 7560504000
Name : Sonoo Jaiswal
Amount : 500000.0
```

2.ii) Inheritance

Aim:

Write a Java program to demonstrate the OOP principle of Inheritance.

Source Code:

File Name : My_Calculation.java

```
class Calculation
{
        int z;

        public void addition(int x, int y)
        {
                z = x + y;
                System.out.println("The sum of the given numbers:"+z);
        }

        public void Subtraction(int x, int y)
        {
              z = x - y;
              System.out.println("The difference between the given numbers:"+z);
        }
}

public class My_Calculation extends Calculation
{
         public void multiplication(int x, int y)
        {
              z = x * y;
              System.out.println("The product of the given numbers:"+z);
         }
        public static void main(String args[])
        {
                int a = 20, b = 10;
                My_Calculation demo = new My_Calculation();
                demo.addition(a, b);
                demo.Subtraction(a, b);
                demo.multiplication(a, b);
        }
}
```

Output :

```
C:\jdk.19\bin>javac My_Calculation.java
C:\jdk.19\bin>java My_Calculation
The sum of the given numbers:30
The difference between the given numbers:10
The product of the given numbers:200
```

2.iii) Polymorphism

Aim:

Write a Java program to demonstrate the OOP principle of Polymorphism.

Source Code:

File Name : Shape.java

```
class Polygon
{
      public void render()
      {
            System.out.println("Rendering Polygon...");
       }
}

class Square extends Polygon
{
       public void render()
      {
            System.out.println("Rendering Square...");
       }
}


class Circle extends Polygon
{
      public void render()
     {
            System.out.println("Rendering Circle...");
      }
}
```

```
class Shape
{
        public static void main(String[] args)
        {
                Polygon p1 = new Polygon();
                p1.render();

                Square s1 = new Square();
                s1.render();

                Circle c1 = new Circle();
                c1.render();
        }
}
```

Output:

```
C:\jdk.19\bin>javac Shape.java
C:\jdk.19\bin>java Shape
Rendering Polygon...
Rendering Square...
Rendering Circle...
```

2.iv) Abstract class

Aim:

Write a Java program to demonstrate the OOP principle of Abstraction.

Source Code:

File Name : Abstraction.java

```
abstract class Bike
{
     Bike()
    {
         System.out.println("bike is created");
    }
    abstract void run();
    void changeGear()
    {
         System.out.println("gear changed");
    }
}
```

```
class Honda extends Bike
{
      void run()
      {
             System.out.println("running safely..");
      }
}

class Abstraction
{
       public static void main(String args[])
      {
            Bike obj = new Honda();
            obj.run();
            obj.changeGear();
      }
}
```

Output:

```
C:\jdk.19\bin>javac Abstraction.java
C:\jdk.19\bin>java Abstraction
bike is created
running safely..
gear changed
```

Viva Questions:

1. Why do we need to use OOPs?
2. What is the difference between overloading and overriding?
3. What is compile time polymorphism?
4. List the types of Inheritance with neat diagram.
5. How is encapsulation different from data abstraction.

EXPERIMENT 3:

Write a Java program to handle checked and unchecked exceptions. Also, demonstrate the usage of custom exceptions in real time scenario.

3.i) Checked Exception

Aim:

Write a Java program to handle checked and unchecked exceptions. Also, demonstrate the usage of custom exceptions in real time scenario.

Source Code:

File Name : Checked.java

```
import java.io.*;
class Checked
{
        public static void main(String args[])
       {
               FileInputStream fis = null;
               try
               {
                      fis = new FileInputStream("wel.txt");
               }
               catch(FileNotFoundException fnfe)
              {
                      System.out.println("The specified file is not " +"present at the given path");
              }
              int k;
              try
             {
                    while(( k = fis.read() ) != -1)
                    {
                           System.out.print((char)k);
                    }
                    fis.close();
             }
             catch(IOException ioe)
             {
                    System.out.println("I/O error occurred: "+ioe);
            }
      }
}
```

File Name : Wel.txt

Welcome to sri indu college

Output:

```
C:\jdk.19\bin>javac Checked.java
C:\jdk.19\bin>java Checked
welcome to sri indu college
```

3.ii) Unchecked exceptions

```
class Unchecked
{
      public static void main(String args[])
      {
        try
        {
             int arr[] ={1,2,3,4,5};
             System.out.println(arr[7]);
        }
         catch(ArrayIndexOutOfBoundsException e)
         {
              System.out.println("The specified index does not exist ");
        }
    }
}
```

Output:

```
C:\jdk.19\bin>javac Unchecked.java
C:\jdk.19\bin>java Unchecked
The specified index does not exist
```

3.iii) Program for User-Defined (custom)

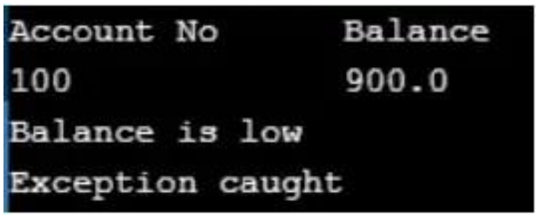
Exception in JavaSource Code:

File Name : User.java

```
class MinBalanceException extends Exception
{
        public MinBalanceException ()
        {
                System.out.println ("Balance is low");
        }
}
public class User
{
        public static void main (String[]args)
        {
                try
```

```
                  {

                           int acc[] = { 100, 101, 102, 103, 104, 105 };
                           double balance[] = { 900, 2000, 1500, 1560, 1765.50 };
                           System.out.println ("Account No\t" + "Balance\t");
                           for (int i = 0; i < 5; i++)
                           {
                                    System.out.println (acc[i] + "\t\t" + balance[i] + "\t");
                                    if (balance[i] < 1000)
                                    {
                                             throw new MinBalanceException ();
                                    }
                           }
                  }
                  catch (MinBalanceException e)
                  {
                           System.out.println ("Exception caught");
                  }
         }
}
```

Output:



```
Account No      Balance
100             900.0
Balance is low
Exception caught
```

Viva Questions:

1. List the reasons why an exception occurs?
2. Define the types of exception.
3. Illustrate the difference between error and exception.
4. Write the difference between checked and unchecked exception.
5. Write the syntax for try…catch().

EXPERIMENT 4:

Aim :

Write a Java program on Random Access File class to perform different read and write operations.

Source Code:

File Name : Randomfile.java

```
import java.io.IOException;
import java.io.RandomAccessFile;
public class Randomfile
{
        static final String FILEPATH ="myfile.txt";
        public static void main(String[] args)
        {
                 try
                {
                        System.out.println(new String(readFromFile(FILEPATH, 0, 18)));
                        writeToFile(FILEPATH, "I love my country and my people", 31);
                }
                catch (IOException e)
                {
                        e.printStackTrace();
                }
        }
        private static byte[] readFromFile(String filePath, int position, int size)
        throws IOException
        {
                 RandomAccessFile file = new RandomAccessFile(filePath, "r");
                 file.seek(position);
                byte[] bytes = new byte[size];
                file.read(bytes);
                file.close();
                return bytes;
        }
```

```
          private static void writeToFile(String filePath, String data, int position)
          throws IOException
          {
                    RandomAccessFile file = new RandomAccessFile(filePath, "rw");
                    file.seek(position);
                    file.write(data.getBytes());
                    file.close();
           }
}
```

The myfile.txt contains text "This class is used for reading and writing to random access file."

after running the program it will contains

This class is used for reading I love my country and my peoplele.

Output:

```
C:\jdk.19\bin>java Randomfile.java
C:\jdk.19\bin>java Randomfile
welcome to sri ind
```

Viva Questions:

1. Define FileInputStream() and FileOutputStream().
2. Differentiate FileReader() and FileWriter().
3. Define RandomAccessFile.
4. List the file access modes.
5. Write the syntax for open a file for read mode.

EXPERIMENT 5:

Aim:

Write a Java program to demonstrate the working of different collection classes. [Use package structure to store multiple classes].

Source Code:

i). ArrayList

```
import java.util.*;
class TestJavaCollection1
{
        public static void main(String args[]){
        ArrayList<String> list=new ArrayList<String>();//Creating arraylist
        list.add("Ravi");//Adding object in arraylist
        list.add("Vijay");
        list.add("Ravi");
        list.add("Ajay");
        //Traversing list through Iterator
        Iterator itr=list.iterator();
        while(itr.hasNext())
        {
                System.out.println(itr.next());
        }
    }
}
```

Output:

```
Ravi
Vijay
Ravi
Ajay
```

ii). LinkedList

```
import java.util.*;
public class TestJavaCollection2
{
        public static void main(String args[]){
        LinkedList<String> al=new LinkedList<String>();
        al.add("Ravi");
        al.add("Vijay");
        al.add("Ravi");
        al.add("Ajay");
        Iterator<String> itr=al.iterator();
        while(itr.hasNext())
        {
                System.out.println(itr.next());
        }
     }
}
```

Output:

```
Ravi
Vijay
Ravi
Ajay
```

iii). Vector

```
import java.util.*;
public class TestJavaCollection3
{
        public static void main(String args[])
        {
                Vector<String> v=new Vector<String>();
                v.add("Ayush");
                v.add("Amit");
                v.add("Ashish");
                v.add("Garima");
                Iterator<String> itr=v.iterator();
                while(itr.hasNext())
                {
                        System.out.println(itr.next());
                }
        }
}
```

Output:

```
Ayush
Amit
Ashish
Garima
```

iv). Stack

```
import java.util.*;
public class TestJavaCollection4
{
        public static void main(String args[])
        {
                Stack<String> stack = new Stack<String>();
                stack.push("Ayush");
                stack.push("Garvit");
                stack.push("Amit");
                stack.push("Ashish");
                stack.push("Garima");
                stack.pop();
                Iterator<String> itr=stack.iterator();
                while(itr.hasNext())
                {
                        System.out.println(itr.next());
                }
        }
}
```

Output:

```
Ayush
Garvit
Amit
Ashish
```

Viva Questions:

1. Which package contain all the collection classes.
2. What is Collection in java?
3. Differentiate Class & Interface.
4. Define Collection Framework.
5. List any 5 java collection classes.

EXPERIMENT 6:

Aim:

Write a program to synchronize the threads acting on the same object. [Consider the example of any reservations like railway, bus, movie ticket booking, etc.]

Source Code:

File Name : SeatReservation.java

```
class SeatReservation
{
        public static void main(String[] args)
        {
                Reservation reserve = new Reservation(); // LINE A
                Person thread1 = new Person(reserve, 5); // LINE B
                thread1.start();
                Person thread2 = new Person(reserve, 4);
                thread2.start();
                Person thread3 = new Person(reserve, 2);
                thread3.start();
        }
}

class Reservation
{

        static int availableSeats = 10;

        synchronized void reserveSeat(int requestedSeats) // LINE D
        {
                System.out.println(Thread.currentThread().getName() + " entered.");
                System.out.println("Availableseats : " + availableSeats + " Requestedsetas : " +
                                requestedSeats);
                if (availableSeats >= requestedSeats)
                 {
                System.out.println("Seat Available. Reserve now :-)");
                 try
                {
                        Thread.sleep(100);     // LINE E
                }
                catch (InterruptedException e)
                {
                        System.out.println("Thread interrupted");
                }
                System.out.println(requestedSeats + " seats reserved.");
                availableSeats = availableSeats - requestedSeats;
            }
            else
            {
                System.out.println("Requested seats not available :-(");
            }
```

```
                System.out.println(Thread.currentThread().getName() + " leaving.");
                System.out.println("..........................................................");
            }
        }

        class Person extends Thread
        {

                Reservation reserve;
                int requestedSeats;

                public Person(Reservation reserve, int requestedSeats)
                {
                        this.reserve = reserve;
                        this.requestedSeats = requestedSeats;
                }

                @Override
                public void run() // LINE C
                {
                        reserve.reserveSeat(requestedSeats);
                }
        }
```

Output:

```
Thread-0 entered.
Availableseats : 10 Requestedsetas : 5
Seat Available. Reserve now :-)
5 seats reserved.
Thread-0 leaving.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Thread-2 entered.
Availableseats : 5 Requestedsetas : 2
Seat Available. Reserve now :-)
2 seats reserved.
Thread-2 leaving.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Thread-1 entered.
Availableseats : 3 Requestedsetas : 4
Requested seats not available :-(
Thread-1 leaving.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Viva Questions:

1. What is Multithreading?
2. Define Multitasking?
3. Differentiate MultiProcessing & MultiThreading?
4. List thread methods in java.
5. Define thread sleep() and run() methods.

EXPERIMENT 7:

Aim:

Develop a java stand alone application that connects with the database (Oracle / mySql) and perform the CRUD operation on the database tables.

MySQL Code:

```
sql> CREATE TABLE IF NOT EXISTS student (
s_id INT PRIMARY KEY,
s_nameVARCHAR(255),
s_addressVARCHAR(255)
);
```

This SQL code creates a table with three columns: s_id for student ID (primary key), s_name for student name, and s_address for student address.

// Create / Insert Data in Database

InsertData.java

```
import java.sql.*;
import java.util.Scanner;

public class InsertData {
    public static void main(String[] args) {
        try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost/mydb", "root", "");
             Statements = con.createStatement()) {

            Scanner sc = new Scanner(System.in);
            System.out.println("Inserting Data into student table:");
            System.out.println("________________________________________");

            System.out.print("Enter student id: ");
            int sid = sc.nextInt();
            System.out.print("Enter student name: ");
            String sname = sc.next();
            System.out.print("Enter student address: ");
            String saddr = sc.next();

            String insertQuery = "INSERT INTO student VALUES(" + sid + ",'" + sname + "','" + saddr +
             "')";
            s.executeUpdate(insertQuery);

            System.out.println("Data inserted successfully into student table");

        } catch (SQLException err) {
            System.out.println("ERROR: " + err);
        }
    }
}
```

Output :

```
Inserting Data into student table:
________________________________
Enter student id: 101
Enter student name: John Doe
Enter student address: 123 Main Street
Data inserted successfully into student table
```

// Retrieve / Display / Select Data in Database

DisplayData.java

```
import java.sql.*;

public class DisplayData {
    public static void main(String[] args) {
        try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost/mydb", "root", "");
             Statement s = con.createStatement()) {

            ResultSet rs = s.executeQuery("SELECT * FROM student");
            if (rs != null) {
                System.out.println("SID \t STU_NAME \t ADDRESS");
                System.out.println("__________________________________________");

                while (rs.next()) {
                    System.out.println(rs.getString("s_id") + " \t " + rs.getString("s_name") + " \t " +
                    rs.getString("s_address"));
                    System.out.println("__________________________________________");
                }
            }

        } catch (SQLException err) {
            System.out.println("ERROR: " + err);
        }
    }
}
```

Output :

```
SID STU_NAME ADDRESS
__________________________________________
102 Alice Smith 789 Oak Avenue
__________________________________________
103 Bob Johnson 567 Pine Road
__________________________________________
```

// Update Data in Database

UpdateData.java

```
import java.sql.*;
import java.util.Scanner;

public class UpdateData {
   public static void main(String[] args) {
      try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost/mydb", "root", "");
          Statements = con.createStatement()) {

         Scanner sc = new Scanner(System.in);
         System.out.println("Update Data in student table:");
         System.out.println("________________________________________");

         System.out.print("Enter student id: ");
         int sid = sc.nextInt();
         System.out.print("Enter student name: ");
         String sname = sc.next();
         System.out.print("Enter student address: ");
         String saddr = sc.next();

         String updateQuery = "UPDATE student SET s_name='" + sname + "', s_address='" + saddr + "'
         WHERE s_id=" + sid;
         s.executeUpdate(updateQuery);

         System.out.println("Data updated successfully");

      } catch (SQLException err) {
         System.out.println("ERROR: " + err);
      }
   }
}
```

Output :

```
Update Data in student table:
________________________________________
Enter student id: 101
Enter student name: Jane Doe
Enter student address: 456 Broad Street
Data updated successfully
```

// Delete Data in Database

DeleteData.java

```
import java.sql.*;
import java.util.Scanner;
```

```
public class DeleteData {
    public static void main(String[] args) {
        try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost/mydb", "root", "");
             Statements = con.createStatement()) {

            Scanner sc = new Scanner(System.in);
            System.out.println("Delete Data from student table:");
            System.out.println("________________________________________");

            System.out.print("Enter student id: ");
            int sid = sc.nextInt();

            String deleteQuery = "DELETE FROM student WHERE s_id=" + sid;
            s.executeUpdate(deleteQuery);

            System.out.println("Data deleted successfully");

        } catch (SQLException err) {
            System.out.println("ERROR: " + err);
        }
    }
}
```

Output :

```
Delete Data from student table:
________________________________________
Enter student id: 101
Data deleted successfully
```

Viva Questions:

1. Expand CRUD.
2. Define JDBC Driver.
3. Write the syntax for Connecting database.
4. Define Driver Manager.
5. Expand JDBC.

EXPERIMENT 8:

Aim:

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:

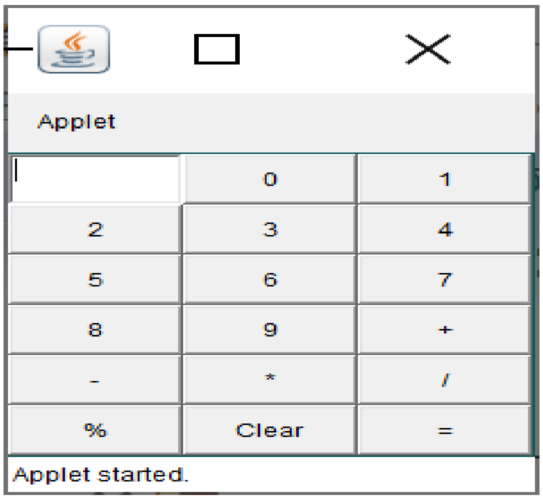
File Name : Calculator.java

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

/*<applet code="Calculator" width=500 height=500></applet>*/

public class Calculator extends Applet implements ActionListener
{
        String msg=" ";
        int v1,v2,result;
        TextField t1;
        Button b[]=new Button[10];
        Button add,sub,mul,div,clear,mod,EQ;
        char OP;
        public void init()
        {
                Color k=new Color(10,89,90); setBackground(k);
                t1=new TextField(50);
                GridLayout gl=new GridLayout(6,3); setLayout(gl);
                for(int i=0;i<10;i++)
                {
                        b[i]=new Button(""+i);
                }
                add=new Button("+");
                sub=new Button("-");
                mul=new Button("*");
                div=new Button("/");
                mod=new Button("%");
                clear=new Button("Clear");
                EQ=new Button("=");
                t1.addActionListener(this);
                add(t1);
                for(int i=0;i<10;i++)
                {
                        add(b[i]);
                }
                add(add);
                add(sub);
```

```
		add(mul);
		add(div);
		add(mod);
		add(clear);
		add(EQ);
		for(int i=0;i<10;i++)
		{
			b[i].addActionListener(this);
		}
		add.addActionListener(this);
		sub.addActionListener(this);
		mul.addActionListener(this);
		div.addActionListener(this);

		mod.addActionListener(this);
		clear.addActionListener(this);
		EQ.addActionListener(this);
	}
	public void actionPerformed(ActionEvent ae)
	{
		String str=ae.getActionCommand();
		char ch=str.charAt(0);

		if ( Character.isDigit(ch)) t1.setText(t1.getText()+str);
		else
		if(str.equals("+"))
		{
			v1=Integer.parseInt(t1.getText());
			OP='+';
			t1.setText("");
		}
		else if(str.equals("-"))
		{
			v1=Integer.parseInt(t1.getText());
			OP='-';
			t1.setText("");
		}
		else if(str.equals("*"))
		{
			v1=Integer.parseInt(t1.getText());
			OP='*';
			t1.setText("");
		}
		else if(str.equals("/"))
		{
			v1=Integer.parseInt(t1.getText());
			OP='/';
			t1.setText("");
		}
		else if(str.equals("%"))
		{
			v1=Integer.parseInt(t1.getText());
			OP='%';
			t1.setText("");
```

```
                    }

                    if(str.equals("="))
                    {
                              v2=Integer.parseInt(t1.getText());
                              if(OP=='+')
                              result=v1+v2;
                              else if(OP=='-')
                              result=v1-v2;
                              else if(OP=='*')
                              result=v1*v2;
                              else if(OP=='/')
                              result=v1/v2;
                              else if(OP=='%')
                              result=v1%v2;
                              t1.setText(""+result);
                    }
                    if(str.equals("Clear"))
                    {
                              t1.setText("");
                    }
          }
}
```

Output:



Viva Questions:

- What is an Applet ?
- What is the difference between an Applet and a Java Application ?
- Name three Component subclasses that support painting.
- What is the relationship between an event-listener interface and an event-adapterclass ?
- How can a GUI component handle its own events ?

EXPERIMENT 9:

Aim:

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:

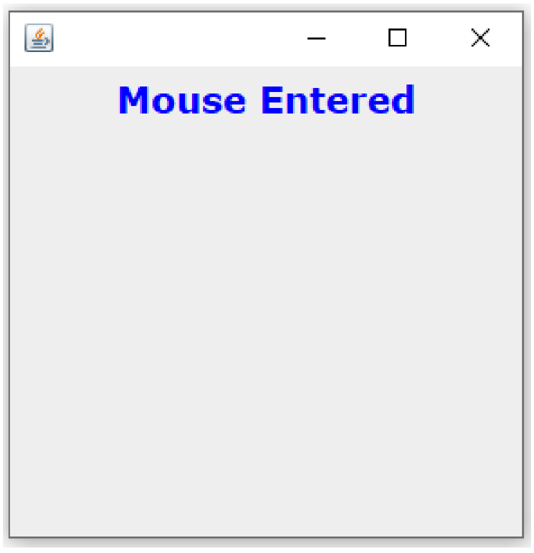
File Name : MouseEvent.java

```
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");
	}
```

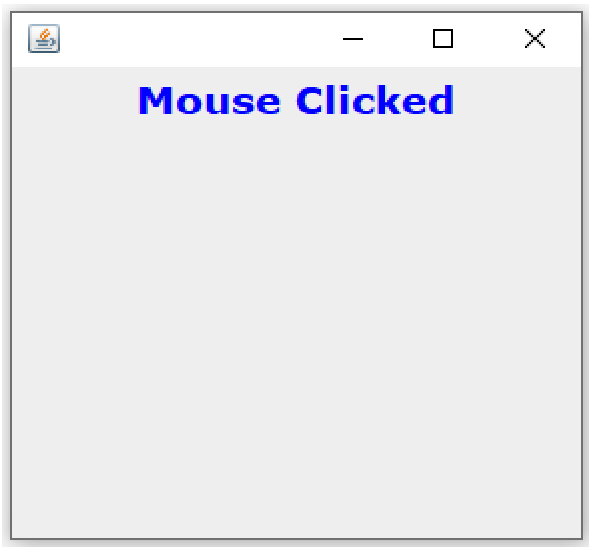
```
            public static void main(String[] args) {
            MouseEventPerformer mep = new MouseEventPerformer();
        }
    }
```

Output:

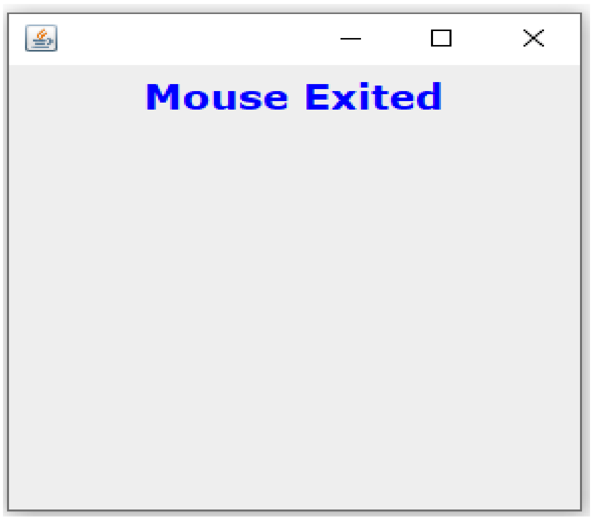
.



Mouse Entered



Mouse Clicked



Mouse Exited

Viva Questions:

1. Define event.
2. List the event handling methods.
3. Define Key event handling methods.
4. Define Mouse event handling methods.
5. Define Actionlistener.

Additional Programs

Exercise : 1

```
// Java Program to Swap Two values using third variable
// using temp variable
// Importing generic libraries

import java.util.*;
class GFG {
// Function to swap two numbers
// Using temporary variable
static void swapValuesUsingThirdVariable(int m, int n)
{
// Swapping the values
int temp = m;
m = n;
n = temp;
System.out.println("Value of m is " + m
+ " and Value of n is " + n);
}
// Main driver code
public static void main(String[] args)
{
// Random integer values
int m = 9, n = 5;
// Calling above function to
// reverse the numbers
swapValuesUsingThirdVariable(m, n);
}
}
```

Output

```
Value of m is 5 and Value of n is 9
```

Exercise : 2

```
// Java Program to print Multiplication of two floating point Number.
import java.io.*;
class GFG {
public static void main(String[] args)
{ // f is to ensures that numbers are float DATA TYPE
float f1 = 1.5f;
float f2 = 2.0f;
// to store the multiplied value
float p = f1 * f2;
// to print the product
System.out.println("The product is: " + p);
}
}
```

Output

```
The product is: 3.0
```

Exercise : 3

```
// Java Program to Check if Given Integer is Odd or Even
// Using Brute Force Approach Importing required classes
import java.io.*;
import java.util.Scanner;
class GFG {
public static void main(String[] args)
{
int num = 10;
if (num % 2 == 0) {
// If remainder is zero then this number is even
System.out.println("Entered Number is Even");
}
else {
System.out.println("Entered Number is Odd");
}
}
}
```

Output

```
Entered Number is Even
```

Exercise : 4

```
// Java Program to Find the Biggest of 3 Numbers
// Importing generic Classes/Files
import java.io.*;
class GFG {
// Function to find the biggest of three numbers
static int biggestOfThree(int x, int y, int z)
{
return z > (x > y ? x : y) ? z : ((x > y) ? x : y);
}
// Main driver function
public static void main(String[] args)
{
// Declaring variables for 3 numbers
int a, b, c;
// Variable holding the largest number
int largest;
a = 5;
b = 10;
c = 3;
// Calling the above function in main
largest = biggestOfThree(a, b, c);
// Printing the largest number
System.out.println(largest
+ " is the largest number.");
}
}
```

Output

```
10 is the largest number.
```

Exercise : 5

```
// Java Program to find
// the LCM of two numbers
import java.io.*;
// Driver Class
class GFG {
// main function
public static void main(String[] args)
{
// Numbers
int a = 15, b = 25;
// Checking for the largest
// Number between them
int ans = (a > b) ? a : b;
// Checking for a smallest number that
// can de divided by both numbers
while (true) {
if (ans % a == 0 && ans % b == 0)
break;
ans++;
}
// Printing the Result
System.out.println("LCM of " + a + " and " + b
+ " : " + ans);
}
}
```

Output

```
LCM of 15 and 25 : 75
```

Exercise : 6

```
// Java program to find all the
// prime numbers from 1 to N
class gfg {
static void prime_N(int N)
{
int x, y, flg;
System.out.println("All the Prime numbers within 1 and " + N
+ " are:");
for (x = 1; x <= N; x++) {
if (x == 1 || x == 0)
continue;
flg = 1;
for (y = 2; y <= x / 2; ++y) {
if (x % y == 0) {
flg = 0;
break;
}
}
// If flag is 1 then x is prime but
// if flag is 0 then x is not prime
if (flg == 1)
System.out.print(x + " ");
}
}
// The Driver code
public static void main(String[] args)
{
int N = 45;
prime_N(N);
}
}
```

Output

```
All the Prime numbers within 1 and 45 are:
2 3 5 7 11 13 17 19 23 29 31 37 41 43
```

Exercise : 7

```
// Java program to find factorial
// of given number
class Test {
// Method to find factorial
// of given number
static int factorial(int n)
{
int res = 1, i;
for (i = 2; i <= n; i++)
res *= i;
return res;
}
public static void main(String[] args)
{
int num = 5;
System.out.println("Factorial of " + num + " is "
+ factorial(5));
}
}
```

Output

```
Factorial of 5 is 120
```

Exercise : 8

```
import java.io.*;
// Java code to demonstrate right star triangle
public class GeeksForGeeks {
// Function to demonstrate printing pattern
public static void StarRightTriangle(int n)
{
int a, b;
// outer loop to handle number of rows
// k in this case
for (a = 0; a < n; a++) {

// inner loop to handle number of columns
// values changing acc. to outer loop
for (b = 0; b <= a; b++) {
// printing stars
System.out.print("* ");
}
// end-line
System.out.println();

}
}

// Driver Function
public static void main(String args[])
{
int k = 5;
StarRightTriangle(k);
}
}
```

Output

```
*
* *
* * *
* * * *
* * * * *
```

Exercise : 9

```
// Java program to convert a decimal
// number to binary number
import java.io.*;
class GFG
{
// function to convert decimal to binary
static void decToBinary(int n)
{
// array to store binary number
int[] binaryNum = new int[1000];
// counter for binary array
int i = 0;
while (n > 0)
{
// storing remainder in binary array
binaryNum[i] = n % 2;
n = n / 2;
i++;
}
// printing binary array in reverse order
for (int j = i - 1; j >= 0; j--)
System.out.print(binaryNum[j]);
}
// driver program
public static void main (String[] args)
{
int n = 17;
System.out.println("Decimal - " + n);
System.out.print("Binary - ");
decToBinary(n);
}
}
```

Output

```
Decimal - 17
Binary - 10001
```

Exercise : 10

```
// java program to check whether input
// character is a vowel or consonant
import java.io.*;
public class geek {
// Function to find whether an input
// character is vowel or not

static void Vowel_Or_Consonant(char y)
{
if (y == 'a' || y == 'e' || y == 'i' || y == 'o'
|| y == 'u')
System.out.println("It is a Vowel.");
else
System.out.println("It is a Consonant.");
}

// The Driver code
static public void main(String[] args)
{
Vowel_Or_Consonant('b');
Vowel_Or_Consonant('u');
}
}
```

Output

```
It is a Consonant.
It is a Vowel.
```