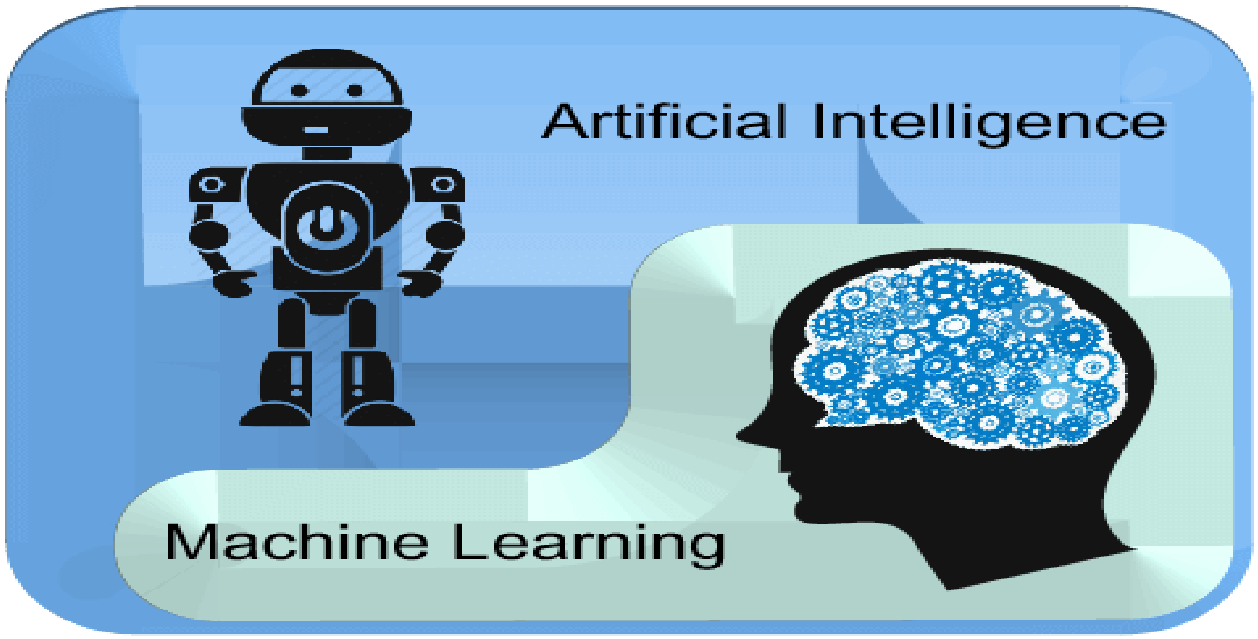
DAYANANDA SAGAR UNIVERSITY
SCHOOL OF ENGINEERING

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING
Artificial Intelligence and Machine Learning

OOP WITH JAVA Laboratory Manual
20AMXXXX



Dayananda Sagar University
Innovation City Campus, Hosur Main Road, Kudlu Gate, Bangalore, India,
Karnataka -560068

DAYANANDA SAGAR UNIVERSITY
SCHOOL OF ENGINEERING
Hosur Rd, Kudlu Gate, Srinivasa Nagar, Hal Layout, Singasandra, Bengaluru, Karnataka 560068

School of Engineering

Vision and Mission

Vision

To be a centre of excellence in education, research & training, innovation & entrepreneurship and to produce citizens with exceptional leadership qualities to serve national and global needs.

Mission

To achieve our objectives in an environment that enhances creativity, innovation and scholarly pursuits while adhering to our vision.

Values

The Pursuit of Excellence

A commitment to strive continuously to improve ourselves and our systems with the aim of becoming the best in our field.

Fairness

A commitment to objectivity and impartiality, to earn the trust and respect of society.

Leadership

A commitment to lead responsively and creatively in educational and research processes.

Integrity and Transparency

A commitment to be ethical, sincere and transparent in all activities and to treat all individuals with dignity and respect.

DAYANANDA SAGAR UNIVERSITY
SCHOOL OF ENGINEERING
Hosur Rd, Kudlu Gate, Srinivasa Nagar, Hal Layout, Singasandra, Bengaluru, Karnataka 560068

Computer Science and Engineering

(Artificial Intelligence and Machine Learning)

Vision

- Develop highly competent engineers in the field of AI & ML contributing globally to the benefit of industry and society.

Mission

- To develop state-of-the-art academic and infrastructural facilities with the latest tools and other learning resources supported by curriculum that can produce self-sustainable professionals.
- To emerge as a research centre that interacts with industry on a regular basis for imparting wholistic curriculum to the students.
- To impart emerging skill sets that the industry require, apart from ensuring that the soft skills of the learners are given adequate thrust.

DAYANANDA SAGAR UNIVERSITY
SCHOOL OF ENGINEERING
Hosur Rd, Kudlu Gate, Srinivasa Nagar, Hal Layout, Singasandra, Bengaluru, Karnataka 560068

Computer Science and Engineering

(Artificial Intelligence and Machine Learning)

Program Educational Objectives (PEO's)

- **PEO1.** Promote design, research, product implementation and services in the field of Artificial Intelligence Engineering through strong technical, communication and entrepreneurial skills
- **PEO2.** Engage to work productively as design and development Engineers, cater to supportive and leadership roles in multidisciplinary domains.
- **PEO3.** Learn and advance their careers by attaining professional certification and seeking higher education.
- **PEO4.** Possess skill in AI & ML expertise ready to serve the society locally and internationally.

Programme Outcome (PO's)

- **PO1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- **PO2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- **PO3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- **PO4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

- **PO5. 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.
- **PO6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
- **PO7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
- **PO8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- **PO9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- **PO10. 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 and design documentation, make effective presentations, and give and receive clear instructions.
- **PO11. Project management and finance:** Demonstrate knowledge and understanding of the 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.
- **PO12. 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's)

- **PSO1.** Apply the principal concepts of AI Engineering to design, develop, deploy and prototype AI Subsystems
- **PSO2.** Apply the knowledge gained pertaining to data storage, data analytics and AI concepts to solve real world business problems.
- **PSO3.** Apply, analyse, design, develop, and test principles of AI concepts on Intelligent Systems

COURSE NAME			
#	**Sem / Year**	**Course Code**	**Title of the Course**
1	**V/III**	**20AMXXXX**	**OOP WITH JAVA LAB**

COURSE OBJECTIVES:

- To learn an object-oriented way of solving problems using Java.
- To write Java programs using multithreading concepts and handle exceptions.
- To write Java programs that connect to a database and be able to perform various operations.
- To create the Graphical User Interface using AWT Components & Swing Components.

COURSE OUTCOMES:

CO No.	Outcomes	Bloom's TaxonomyLevel
1	Develop simple Java programs that make use of classes and objects	L6
2	Make use of inheritance and interfaces to develop Java application	L6
3	Model exception handling, multi-threading concepts in Java	L4
4	Create the Graphical User Interface based application programs by utilizing event handling features and Swing in Java	L6
5	Develop a Java program that connects to a database and is able to perform various operations	L6

1. Table: Mapping Levels of COs to POs / PSOs

COs	Program Outcomes (POs)												PSOs		
	Engineering knowledge	Problem analysis	Design	Conduct investigations of complex problems	tool usage	The engineer and society	Environment and sustainability	Ethics	team work	Communication	Life-long learning	Project management and finance	Apply the principal concepts of AI Engineering	Apply the knowledge gained pertaining to data storage, data analytics and AI concepts	develop, and test principles of AI concepts on Intelligent Systems
	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3
CO1	**3**	**2**	**2**	**2**	3	-	-	-	-	-	1	-	**1**	-	**1**
CO2	**3**	**2**	2	2	3	-	-	-	-	-	1	-	**1**	-	**1**
CO3	**3**	**2**	**1**	-	3	-	-	-	-	-	1	-	**1**	-	**1**
CO4	**3**	**1**	**3**	-	**3**	-	-	-	3	-	1	-	**1**	**1**	**1**
CO5	**3**	**1**	**3**	-	**3**	-	-	-	3	-	1	-	**1**	**1**	**1**

Instructions for Laboratory Exercises:

1. The programs with comments are listed for your reference. Write the programs in the observation book.
2. Create your own subdirectory in the computer. Edit (type) the programs with program numbers and place them in your subdirectory.
3. Execute the programs as per the steps discussed earlier and note the results in your observation book
4. Initially you will start with notepad editor and compile, execute from command prompt.
5. Later you can use Eclipse IDE for execution of the program.
6. All the programs will be handwritten.
7. Please include program output screen for every program.

List of Experiments

Exp No	Experiment Name	Date	Marks	Sign
1	Basic programs using data types, operators, and control statements in Java			
2	Basic programs using Arrays			
3	Basic Programs involving Strings			
4	Problem on the use of constructors, inheritance			
5	Basic programs of Method overloading			
6	Problem with the use of Garbage collection			
7	Basic programs of Polymorphism			
8	Programs involving: Exception handling			
9	Programs involving: multi-threading			
10	Programs involving: Packages			
11	Programs involving: Interfaces			
12	GUI Programming in Java			
13	Programs involving: Database connectivity in Java			

Internals (Lab Exam, Lab Record, Performance) (40M)	Mini project (20M)	Total Marks (60M)

EXPERIMENT: 1

AIM: Basic programs using data types, operators, and control statements in Java.

Develop a Java application to generate Electricity bills. Create a class with the following members: Consumer no., consumer name, previous month reading, current month reading, and type of EB connection (i.e. domestic or commercial). Compute the bill amount using the following tariff.

If the type of the EB connection is domestic, calculate the amount to be paid as follows:

- First 100 units - Rs. 1 per unit
- 101-200 units - Rs. 2.50 per unit
- 201 -500 units - Rs. 4 per unit
- >501 units - Rs. 6 per unit

If the type of the EB connection is commercial, calculate the amount to be paid as follows:

- First 100 units - Rs. 2 per unit
- 101-200 units - Rs. 4.50 per unit
- 201 -500 units - Rs. 6 per unit
- > 501 units - Rs. 7 per unit

PROGRAM:

```
import java.util.*;
public class Ebill
{
        public static void main (String args[])
        {
        Customerdata ob = new Customerdata();
        ob.getdata();
        ob.calc();
        ob.display();
        }
}
        class Customerdata
```

```
{
        Scanner in = new Scanner(System.in);
        Scanner ins = new Scanner(System.in);
        String cname,type;
        int bn;
        double current,previous,tbill,units;
        void getdata()
        {
        System.out.print ("\n\t Enter consumer number ");
        bn = in.nextInt();
        System.out.print ("\n\t Enter Type of connection (D for Domestic or C for
        Commercial) ");
        type = ins.nextLine();
        System.out.print ("\n\t Enter consumer name ");
        cname = ins.nextLine();
        System.out.print ("\n\t Enter previous month reading ");
        previous= in.nextDouble();
        System.out.print ("\n\t Enter current month reading ");
        current= in.nextDouble();
        }
        void calc()
        {
        units=current-previous;
        if(type.equals("D"))
        {
        if (units<=100)
        tbill=1 * units;
        else if (units>100 && units<=200)
        tbill=2.50*units;
        else if(units>200 && units<=500)
        tbill= 4*units;
        else
        tbill= 6*units;
        }
        else
        {
        if (units<=100)
        tbill= 2 * units;
        else if(units>100 && units<=200)
        tbill=4.50*units;
        else if(units>200 && units<=500)
        tbill= 6*units;
        else
        tbill= 7*units;
        }
}
        void display()
{
        System.out.println("\n\t Consumer number = "+bn);
        System.out.println ("\n\t Consumer name = "+cname);
        if(type.equals("D"))
        System.out.println ("\n\t type of connection = DOMESTIC ");
        else
        System.out.println ("\n\t type of connection = COMMERCIAL ");
        System.out.println ("\n\t Current Month Reading = "+current);
        System.out.println ("\n\t Previous Month Reading = "+previous);
        System.out.println ("\n\t Total units = "+units);
        System.out.println ("\n\t Total bill = RS "+tbill);

}
}
```

INPUT DATA:

OUTPUT DATA:

EXPERIMENT: 2

AIM: Objective: Basic programs using Arrays

Write a Java program to move all 0's to the end of an array. Maintain the relative order of the other (non-zero) array elements.

PROGRAM:

```
import java.util.*;
 public class Exercise26
 {
    public static void main(String[] args) throws Exception
 {
        int[] array_nums = {0,0,1,0,3,0,5,0,6};
        int i = 0;
        System.out.print("\nOriginal array: \n");
        for (int n : array_nums)
        System.out.print(n+"  ");

        for(int j = 0, l = array_nums.length; j < l;)
        {
        if(array_nums[j] == 0)
                j++;
        else
         {
                int temp = array_nums[i];
                array_nums[i] = array_nums[j];
                array_nums[j] = temp;
                i ++;
                j ++;
         }
        }
        while (i < array_nums.length)
        array_nums[i++] = 0;
        System.out.print("\nAfter moving 0's to the end of the array: \n");
        for (int n : array_nums)
        System.out.print(n+"  ");
                System.out.print("\n");
   }
}
```

INPUT DATA:

OUTPUT DATA;

EXPERIMENT: 3

AIM: Basic programs using Strings

Write a Java program to find the first non-repeating character in a string

PROGRAM:

```
import java.util.*;
public class Main
{
        public static void main(String[] args)
{
        String str1 = "gibblegabbler";
        System.out.println("The given string is: " + str1);
        for (int i = 0; i < str1.length(); i++)
        {
        boolean unique = true;
        for (int j = 0; j < str1.length(); j++)
        {
        if (i != j && str1.charAt(i) == str1.charAt(j))
        {
        unique = false;
        break;
         }
        }
   if (unique)
         {
         System.out.println("The first non repeated character in String is: " +
        str1.charAt(i));
    break;
        }
        }
 }
 }
```

INPUT DATA:

OUTPUT DATA:

EXPERIMENT: 4

AIM: Object Oriented Programming Concepts: Problem on the use of constructors, inheritance

Declare a class employee having emp_id and empname as members. Extend class employee (inheritance) to have a subclass called salary having designation and monthly_salary as members. Define the following:

– Required constructors.

– A method to find and display all details of employee drawing salary more than 20000/-.

– Method main for creating an array for storing these details given as command line argument and showing usage of above methods.text.

PROGRAM:

```
import java.util.Scanner;
class Employee
{
 String[] employee_id;
 String[] employee_name;
}

class Salary extends Employee
{
 String[] Designation;
 double[] monthly_salary;

 Salary(int j)
 {
  /*initialization of array */
  employee_name=new String[j];
  employee_id=new String[j];
  Designation=new String[j];
  monthly_salary= new double[j];

 }
 void display(int j)
 {


   System.out.println("-------------------------------------------------------------");
   System.out.println("-------------------------------------------------------------");
   System.out.println("\t Details of employee who have salary above 20000");
   System.out.println("-------------------------------------------------------------");
   System.out.println("-------------------------------------------------------------\n
\n");
  System.out.format("%-15s %-15s %-25s %-10s %n","employee id","employee
  name","employee Designation","Monthly Salary");
   System.out.println("---------------------------------------------------------------------
------");
  for(int i=0;i<j;i++)
  {

   if(monthly_salary[i]>=20000)
   {
    System.out.format("%-15s %-15s %-25s %-10s
%n",employee_id[i],employee_name[i],Designation[i],monthly_salary[i]);

   }
  }
 }
 public static void main(String [] args)
 {
  Scanner jaimin=new Scanner(System.in);
  int length=args.length;

  Salary obj = new Salary(length);


  if(length==0)
  {
   System.out.println("please enter employee id");
  }

  for(int i=0;i<length;i++)
  {
    obj.employee_id[i]=args[i];
    System.out.println("\n\n enter the details of \""+args[i]+"\" employee id");

    System.out.print("\n name of employee -->");
    obj.employee_name[i]=jaimin.next();

    System.out.print("\n Designation of employee -->");
    obj.Designation[i]=jaimin.next();

    System.out.print("\nMonthly salary of employee -->");
    obj.monthly_salary[i]=jaimin.nextDouble();


  }

  obj.display(length);
 }
}
```

INPUT DATA:

OUTPUT DATA:

EXPERIMENT: 5

AIM: Object Oriented Programming Concepts: Problem on the use of Method Overloading and Overriding

Write a JAVA program to represent Method Overloading and Overriding.

PROGRAM:

```
package com.techvidvan.methodoverriding;
public class Addition
{
int add(int a, int b)
{
return (a + b);
}
int add(int a , int b , int c)
{
return (a + b + c) ;
}
double add(double a , double b)
{
return (a + b);
}
double add(int a , double b)
{
return (a + b);
}
public static void main( String args[])
{
Addition ob = new Addition();
System.out.println("Calling add method with two int parameters: " +ob.add(17,
25));
System.out.println("Calling add method with three int parameters: "
+ob.add(55, 27, 35));
System.out.println("Calling add method with two double parameters: "
+ob.add(36.5, 42.8));
System.out.println("Calling add method with one int and one double
parameter: " +ob.add(11, 24.5));
}
```

```
package com.techvidvan. methodoverriding;
//Base Class
class Parent
{
        void view()
{
        System.out.println("This is a parent class method");
}
}
class Child extends Parent
{
        @Override
        void view()
{
        System.out.println("This is a child class method");
}
}
        //Driver class
        public class MethodOverriding
{
         public static void main(String args[])
{
         Parent obj = new Parent();
         obj.view();
         Parent obj1 = new Child();
         obj1.view();
}
}
```

<u>INPUT DATA:</u>

<u>OUTPUT DATA:</u>

EXPERIMENT: 6

AIM: Object Oriented Programming Concepts: Problem on the use of Garbage collection

Write a JAVA program to represent Garbage Collection

PROGRAM:

```
class Employee
{

    private int ID;
    private String name;
    private int age;
    private static int nextId = 1;

    // it is made static because it
    // is keep common among all and
    // shared by all objects
    public Employee(String name, int age)
    {
        this.name = name;
        this.age = age;
        this.ID = nextId++;
    }
    public void show()
    {
        System.out.println("Id=" + ID + "\nName=" + name
                        + "\nAge=" + age);
    }
    public void showNextId()
    {
        System.out.println("Next employee id will be="
                        + nextId);
    }
    protected void finalize()
    {
        --nextId;
        // In this case,
        // gc will call finalize()
        // for 2 times for 2 objects.
    }
}

public class UseEmployee
 {
    public static void main(String[] args)
    {
        Employee E = new Employee("GFG1", 56);
        Employee F = new Employee("GFG2", 45);
        Employee G = new Employee("GFG3", 25);
        E.show();
        F.show();
        G.show();
        E.showNextId();
        F.showNextId();
        G.showNextId();

        {
            // It is sub block to keep
            // all those interns.
            Employee X = new Employee("GFG4", 23);
            Employee Y = new Employee("GFG5", 21);
            X.show();
            Y.show();
            X.showNextId();
            Y.showNextId();
            X = Y = null;
            System.gc();
            System.runFinalization();
        }
        E.showNextId();
    }
}
```

<u>INPUT DATA:</u>

<u>OUTPUT DATA</u>

EXPERIMENT: 7

AIM: Object Oriented Programming Concepts: Problem on the use of Polymorphism

Write a JAVA program to represent the concept of polymorphism.

PROGRAM:

```
import java.util.*;
public class ExceptionDemo
{
    static void func(int a,int b) throws ArithmeticException,
    ArrayIndexOutOfBoundsException
    {
    System.out.println(10/a);
     int[] arr={1,2,3};
    System.out.println(arr[b]);
    }
public static void main (String[] args)
{
            Scanner in=new Scanner(System.in);
            for(int i=0;i<3;i++)
            {
            Try
            {
                func(in.nextInt(),in.nextInt());
            }
            catch(ArithmeticException e)
            {
              System.out.println("can't divide by zero");
            }
              catch(ArrayIndexOutOfBoundsException e)
            {
                System.out.println("Out of bounds!");
            }
            }
        }
    }
```

INPUT DATA:

OUTPUT DATA:

EXPERIMENT: 8

AIM: Programs involving: Exception handling

Write a Java program to create multiple Exceptions.

PROGRAM:

```
import java.util.*;
public class ExceptionDemo
{
        static void func(int a,int b) throws ArithmeticException,
        ArrayIndexOutOfBoundsException
        {
                System.out.println(10/a);
                int[] arr={1,2,3};
                System.out.println(arr[b]);
        }
        public static void main (String[] args)
 {
                Scanner in=new Scanner(System.in);
                for(int i=0;i<3;i++){
                try
                {
                    func(in.nextInt(),in.nextInt());
                }
                catch(ArithmeticException e)
                {
                  System.out.println("can't divide by zero");
                }
                catch(ArrayIndexOutOfBoundsException e)
                {
                  System.out.println("Out of bounds!");
                }
            }
        }
}
```

INPUT DATA:

OUTPUT DATA:

EXPERIMENT: 9

AIM: Programs involving: Threads and Multiple threads

Write a Java program to create multiple threads in Java. Explain all thread methods with examples.

PROGRAM:

```
class ThreadTest extends Thread
{
      private Thread thread;
      private String threadName;

      ThreadTest( String msg)
      {
            threadName = msg;
            System.out.println("Creating thread: " +  threadName );
      }
      public void run()
      {
            System.out.println("Running thread: " +  threadName );
            try
            {
                  for(int i = 0; i < 5; i++)
                  {
                        System.out.println("Thread: " + threadName + ", " + i);
                        Thread.sleep(50);
                  }
            }
            catch (InterruptedException e)
            {
                  System.out.println("Exception in thread: " +  threadName);
            }
            System.out.println("Thread " +  threadName + " continue...");
      }
      public void start ()
      {
           System.out.println("Start method " +  threadName );
           if (thread == null)
```

```
          {
               thread = new Thread (this, threadName);
               thread.start ();
          }
     }
}
public class MultipleThread
{
     public static void main(String args[])
     {
          ThreadTest thread1 = new ThreadTest( "First Thread");
          thread1.start();

          ThreadTest thread2 = new ThreadTest( "Second Thread");
          thread2.start();
     }
}
```

INPUT DATA:

OUTPUT DATA:

Experiment 10

AIM: Programs involving: Packages in Java

Program 1

```
import java.io.File;
import java.io.IOException;

import java.util.Scanner;
class Progpackage
{
        public static void main(String[] args)
{
                try
                {
                        File r=new
                        File("C:\\Users\\LENOVO\\Desktop\\DSU\\DSU
                        data\\scanner.txt");
                        Scanner sc=new Scanner(r);
                        while(sc.hasNextLine())
                        {
                                System.out.println("this is my first \" \"program");
                                System.out.println(sc.nextLine());
                        }
                }
                catch(IOException e)
                {
                        System.out.println(e);
                }
        }
}
```

INPUT DATA:

OUTPUT DATA:

Program 2

```
package OODJ;

class A
{
	void show()
	{
		System.out.println("Java Programming");
	}
}

class proguserdefinepackage
{
	public static void main(String[] args)
	{
		A r=new A();
		r.show();
	}
}
```

INPUT DATA:

OUTPUT DATA:

Experiment 11

AIM: Programs involving: Interfaces in Java

PROGRAM 1:

```
interface Customer
{
        int amt=5;    //public+static+final
        void purchase();  //public+abstract

}
class seller implements Customer
{
        @Override
        public void purchase()
        {
                System.out.println("final amount"+""+amt);

        }
}
class Intefacevariable
{
                public static void main(String[] args)
{
                Customer c=new seller();
                c.purchase();
                System.out.println(Customer.amt);

}
}
```

INPUT DATA:

OUTPUT DATA

Program 2:

```
interface A
{
        void add();
}
interface B extends A
{
        void sub();
}
class java implements B
{
        //@override
        public void add()
        {
                int a=10, b=20,c;
                c=a+b;
                System.out.println("Addition"+c);
        }
        //@override
        public void sub()
        {
                int a=10, b=20,c;
                c=a-b;
                System.out.println("Subtraction"+c);
        }
        class main
        {
                public static void main(String[] args) {
                        B r=new java();
                        r.add();
                        r.sub();
        }

        }
}
```

INPUT DATA:

OUTPUT DATA:

Experiment 12

AIM: GUI Programming in Java

PROGRAM 1:

Create a program using java swing to demonstrate a table with the following fields: name, roll number, department. Use JTable

The JTable class is a part of Java Swing Package and is generally used to display or edit two-dimensional data that is having both rows and columns. It is similar to a spreadsheet. This arranges data in a tabular form.
Constructors in JTable:

1. JTable(): A table is created with empty cells.
2. JTable(int rows, int cols): Creates a table of size rows * cols.
3. JTable(Object[][] data, Object []Column): A table is created with the specified name where []Column defines the column names.

Functions in JTable:

1. addColumn(TableColumn []column) : adds a column at the end of the JTable.
2. clearSelection() : Selects all the selected rows and columns.
3. editCellAt(int row, int col) : edits the intersecting cell of the column number col and row number row programmatically, if the given indices are valid and the corresponding cell is editable.
4. setValueAt(Object value, int row, int col) : Sets the cell value as 'value' for the position row, col in the JTable.

Below is the program to illustrate the various methods of JTable:

- Java

```
// Packages to import
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;

public class JTableExamples
{
    // frame
    JFrame f;
    // Table
    JTable j;

    // Constructor
    JTableExamples()
    {
        // Frame initialization
        f = new JFrame();

        // Frame Title
        f.setTitle("JTable Example");

        // Data to be displayed in the JTable
        String[][] data =
   {
            { "Kundan Kumar Jha", "4031", "CSE" },
            { "Anand Jha", "6014", "IT" }
   };

        // Column Names
        String[] columnNames = { "Name", "Roll Number", "Department" };

        // Initializing the JTable
        j = new JTable(data, columnNames);
        j.setBounds(30, 40, 200, 300);

        // adding it to JScrollPane
        JScrollPane sp = new JScrollPane(j);
        f.add(sp);
        // Frame Size
        f.setSize(500, 200);
        // Frame Visible = true
        f.setVisible(true);
    }

    // Driver  method
    public static void main(String[] args)
    {
        new JTableExamples();
    }
}
```

Program 2:

Create a program using java swing, to demonstrate how the basic registration form looks like.

Swing is a part of the JFC (Java Foundation Classes). Building Graphical User Interface in Java requires the use of Swings. Swing Framework contains a large set of components which allow a high level of customization and provide rich functionalities, and is used to create window-based applications. Java swing components are lightweight, platform-independent, provide powerful components like tables, scroll panels, buttons, list, color chooser, etc.
In this article, we'll see how to make a Registration form which includes all the buttons and field in one Form.
Steps:
1. Create a Java file that contains the main class – Registration. This class will only contain the main method to invoke the required methods.

```
class Registration {

    public static void main(String[] args)
                        throws Exception
    {
        MyFrame f = new MyFrame();
    }
}
```

2. Create another class MyFrame, which will contain the form.
3. In this MyFrame Class, the methods to be made are:

- Components like JLabel, JTextField, JRadioButton, ButtonGroup, JComboBox, and JTextArea. These components will collectively form the Registration form.
- A constructor, to initialize the components with default values.
- A method actionPerformed() to get the action performed by the user and act accordingly.
- 4. Copy the code of MyFrame class from below.
- 5. Save the file as Registration.java

6. Compile the file by using javac command.

```
javac Registration.java
```

7. Run the program by calling the main class

java Registration
Below is the code to implement the Simple Registration Form using Java Swing:

- Java

```
// Java program to implement
// a Simple Registration Form
// using Java Swing

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

class MyFrame
    extends JFrame
    implements ActionListener {

    // Components of the Form
    private Container c;
    private JLabel title;
    private JLabel name;
    private JTextField tname;
    private JLabel mno;
    private JTextField tmno;
    private JLabel gender;
    private JRadioButton male;
    private JRadioButton female;
    private ButtonGroup gengp;
    private JLabel dob;
    private JComboBox date;
    private JComboBox month;
    private JComboBox year;
    private JLabel add;
    private JTextArea tadd;
    private JCheckBox term;
    private JButton sub;
    private JButton reset;
    private JTextArea tout;
    private JLabel res;
    private JTextArea resadd;

    private String dates[]
        = { "1", "2", "3", "4", "5",
        "6", "7", "8", "9", "10",
        "11", "12", "13", "14", "15",
        "16", "17", "18", "19", "20",
        "21", "22", "23", "24", "25",
        "26", "27", "28", "29", "30",
        "31" };
private String months[]
    = { "Jan", "feb", "Mar", "Apr",
        "May", "Jun", "July", "Aug",
        "Sup", "Oct", "Nov", "Dec" };
private String years[]
    = { "1995", "1996", "1997", "1998",
        "1999", "2000", "2001", "2002",
        "2003", "2004", "2005", "2006",
        "2007", "2008", "2009", "2010",
        "2011", "2012", "2013", "2014",
        "2015", "2016", "2017", "2018",
        "2019" };

// constructor, to initialize the components
// with default values.
public MyFrame()
{
    setTitle("Registration Form");
    setBounds(300, 90, 900, 600);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setResizable(false);

    c = getContentPane();
    c.setLayout(null);

    title = new JLabel("Registration Form");
    title.setFont(new Font("Arial", Font.PLAIN, 30));
    title.setSize(300, 30);
    title.setLocation(300, 30);
    c.add(title);

    name = new JLabel("Name");
    name.setFont(new Font("Arial", Font.PLAIN, 20));
    name.setSize(100, 20);
    name.setLocation(100, 100);
    c.add(name);
```

```
tname = new JTextField();
tname.setFont(new Font("Arial", Font.PLAIN, 15));
tname.setSize(190, 20);
tname.setLocation(200, 100);
c.add(tname);

mno = new JLabel("Mobile");
mno.setFont(new Font("Arial", Font.PLAIN, 20));
mno.setSize(100, 20);
mno.setLocation(100, 150);
c.add(mno);

tmno = new JTextField();
tmno.setFont(new Font("Arial", Font.PLAIN, 15));
tmno.setSize(150, 20);
tmno.setLocation(200, 150);
c.add(tmno);

gender = new JLabel("Gender");
gender.setFont(new Font("Arial", Font.PLAIN, 20));
gender.setSize(100, 20);
gender.setLocation(100, 200);
c.add(gender);

male = new JRadioButton("Male");
male.setFont(new Font("Arial", Font.PLAIN, 15));
male.setSelected(true);
male.setSize(75, 20);
male.setLocation(200, 200);
c.add(male);

female = new JRadioButton("Female");
female.setFont(new Font("Arial", Font.PLAIN, 15));
female.setSelected(false);
female.setSize(80, 20);
female.setLocation(275, 200);
c.add(female);

gengp = new ButtonGroup();
gengp.add(male);
gengp.add(female);
```

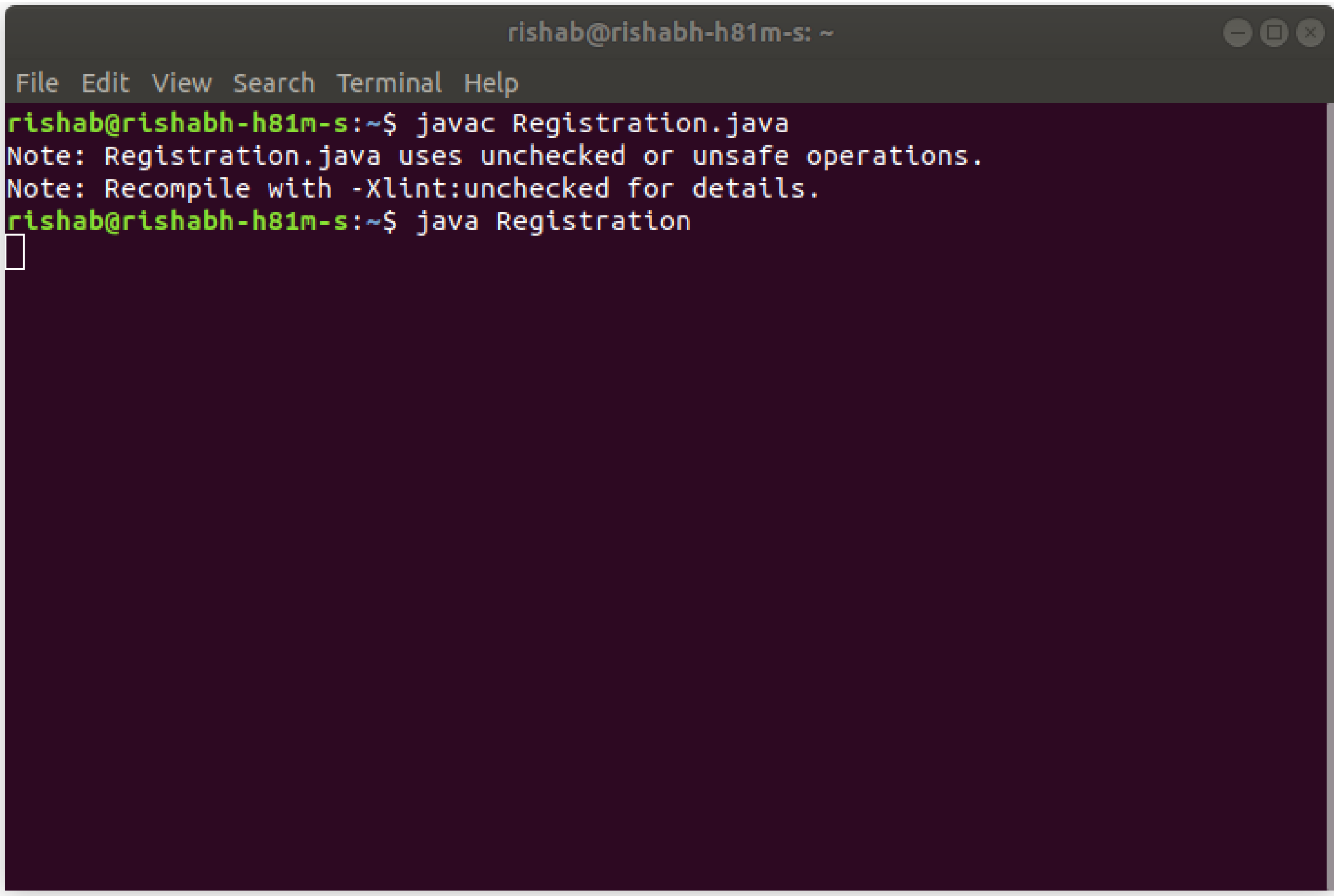
```
dob = new JLabel("DOB");
dob.setFont(new Font("Arial", Font.PLAIN, 20));
dob.setSize(100, 20);
dob.setLocation(100, 250);
c.add(dob);

date = new JComboBox(dates);
date.setFont(new Font("Arial", Font.PLAIN, 15));
date.setSize(50, 20);
date.setLocation(200, 250);
c.add(date);

month = new JComboBox(months);
month.setFont(new Font("Arial", Font.PLAIN, 15));
month.setSize(60, 20);
month.setLocation(250, 250);
c.add(month);

year = new JComboBox(years);
year.setFont(new Font("Arial", Font.PLAIN, 15));
year.setSize(60, 20);
year.setLocation(320, 250);
c.add(year);

add = new JLabel("Address");
add.setFont(new Font("Arial", Font.PLAIN, 20));
add.setSize(100, 20);
add.setLocation(100, 300);
c.add(add);

tadd = new JTextArea();
tadd.setFont(new Font("Arial", Font.PLAIN, 15));
tadd.setSize(200, 75);
tadd.setLocation(200, 300);
tadd.setLineWrap(true);
c.add(tadd);

term = new JCheckBox("Accept Terms And Conditions.");
term.setFont(new Font("Arial", Font.PLAIN, 15));
term.setSize(250, 20);
term.setLocation(150, 400);
```

```
        c.add(term);

        sub = new JButton("Submit");
        sub.setFont(new Font("Arial", Font.PLAIN, 15));
        sub.setSize(100, 20);
        sub.setLocation(150, 450);
        sub.addActionListener(this);
        c.add(sub);

        reset = new JButton("Reset");
        reset.setFont(new Font("Arial", Font.PLAIN, 15));
        reset.setSize(100, 20);
        reset.setLocation(270, 450);
        reset.addActionListener(this);
        c.add(reset);

        tout = new JTextArea();
        tout.setFont(new Font("Arial", Font.PLAIN, 15));
        tout.setSize(300, 400);
        tout.setLocation(500, 100);
        tout.setLineWrap(true);
        tout.setEditable(false);
        c.add(tout);

        res = new JLabel("");
        res.setFont(new Font("Arial", Font.PLAIN, 20));
        res.setSize(500, 25);
        res.setLocation(100, 500);
        c.add(res);

        resadd = new JTextArea();
        resadd.setFont(new Font("Arial", Font.PLAIN, 15));
        resadd.setSize(200, 75);
        resadd.setLocation(580, 175);
        resadd.setLineWrap(true);
        c.add(resadd);

        setVisible(true);
    }

    // method actionPerformed()
    // to get the action performed
```

```
// by the user and act accordingly
public void actionPerformed(ActionEvent e)
{
    if (e.getSource() == sub) {
        if (term.isSelected()) {
            String data1;
            String data
                = "Name : "
                  + tname.getText() + "\n"
                  + "Mobile : "
                  + tmno.getText() + "\n";
            if (male.isSelected())
                data1 = "Gender : Male"
                        + "\n";
            else
                data1 = "Gender : Female"
                        + "\n";
            String data2
                = "DOB : "
                  + (String)date.getSelectedItem()
                  + "/" + (String)month.getSelectedItem()
                  + "/" + (String)year.getSelectedItem()
                  + "\n";

            String data3 = "Address : " + tadd.getText();
            tout.setText(data + data1 + data2 + data3);
            tout.setEditable(false);
            res.setText("Registration Successfully..");
        }
        else {
            tout.setText("");
            resadd.setText("");
            res.setText("Please accept the"
                        + " terms & conditions..");
        }
    }

    else if (e.getSource() == reset) {
        String def = "";
        tname.setText(def);
        tadd.setText(def);
        tmno.setText(def);
```

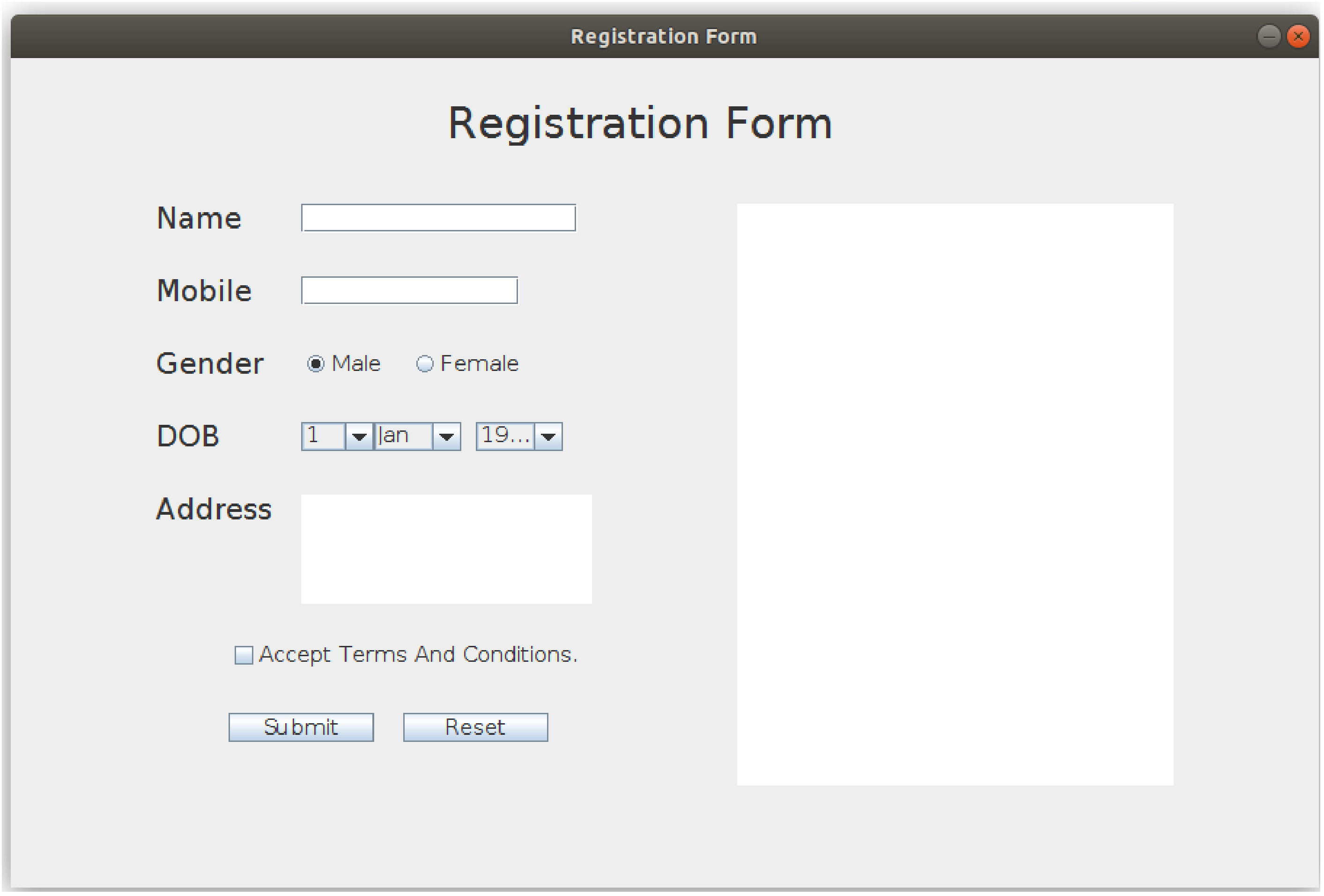
```
            res.setText(def);
            tout.setText(def);
            term.setSelected(false);
            date.setSelectedIndex(0);
            month.setSelectedIndex(0);
            year.setSelectedIndex(0);
            resadd.setText(def);
        }
    }
}

// Driver Code
class Registration {

    public static void main(String[] args) throws Exception
    {
        MyFrame f = new MyFrame();
    }
}
```

1. Compile:



```
rishab@rishabh-h81m-s:~$ javac Registration.java
Note: Registration.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
rishab@rishabh-h81m-s:~$ java Registration
```

2. Registration Form unfilled:



Registration Form

Registration Form

Name

Mobile

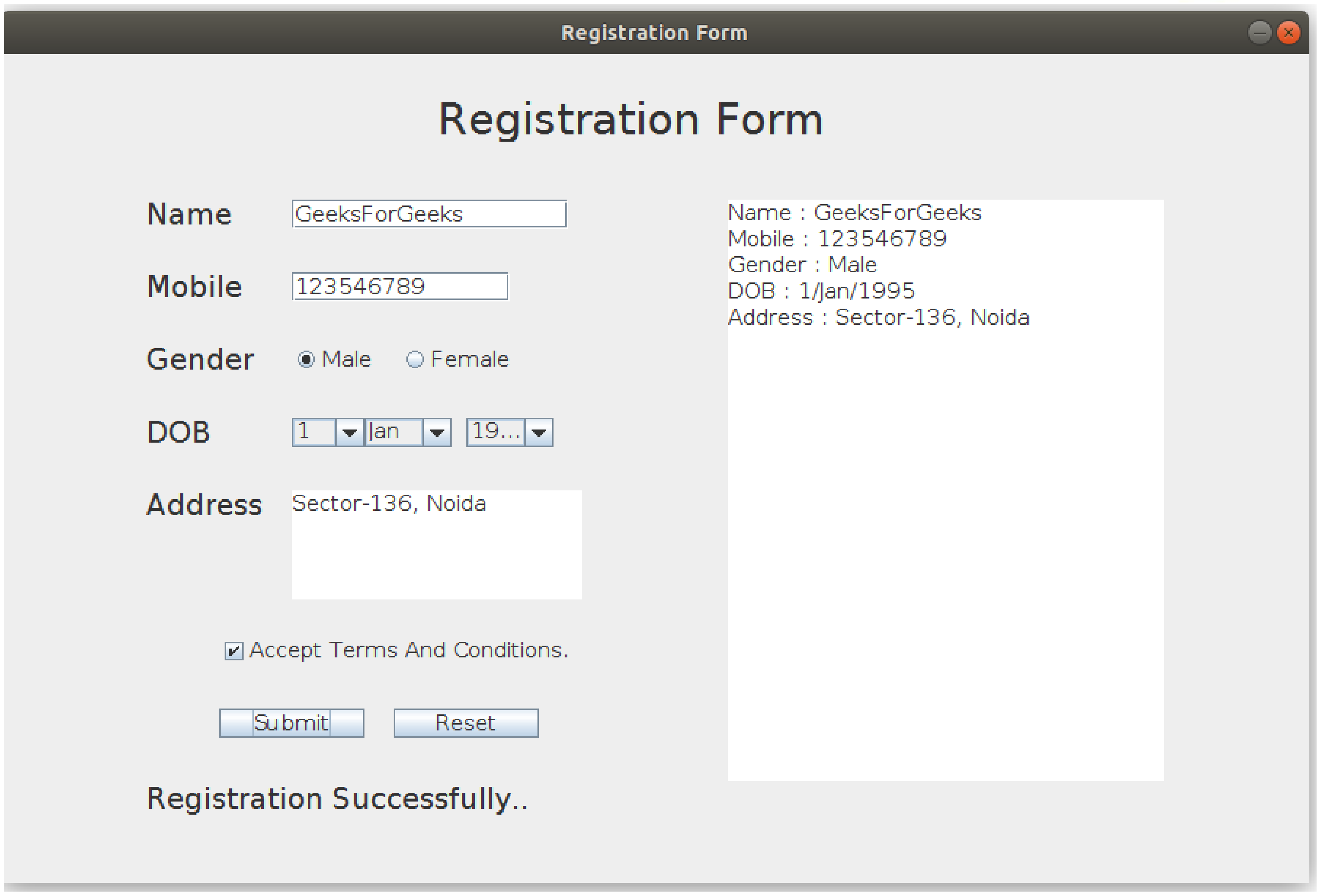
Gender (● Male) ○ Female

DOB 1 Jan 19...

Address

☐ Accept Terms And Conditions.

3. Registration Form filled:



Registration Form

Name GeeksForGeeks

Mobile 123546789

Gender [✓ Male] Female

DOB 1 Jan 19...

Address Sector-136, Noida

[✓] Accept Terms And Conditions.

Submit Reset

Registration Successfully..

```
Name : GeeksForGeeks
Mobile : 123546789
Gender : Male
DOB : 1/Jan/1995
Address : Sector-136, Noida
```

INPUT DATA:

OUTPUT DATA:

Experiment 13

AIM: Programs involving: Database connectivity in Java

Write a program to implement Create, and Retrieve operations for the registration of Student details using JDBC, HSQLDB.

PROGRAM 1:

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


public class JDBCcallablestatement
 {
        public static void main(String[] args)
        {
                try
                {
                        //load the driver
                        Class.forName("com.mysql.jdbc.Driver");

                        //creating a connection
                        String url="jdbc:mysql://localhost:3306/javasql";
                        String uname="root";
                        String pass="dsu@22";
                        Connection con=DriverManager.getConnection(url, uname,
pass);//cannot create the object for connection interface

                        CallableStatement cs=con.prepareCall("{call
callableStmtSP(?,?,?)}");

                        Scanner sc=new Scanner(System.in);

                        System.out.println("Enter tId");
                        int a=sc.nextInt();

                        System.out.println("Enter tName");
                        String b=sc.next();

                        System.out.println("Enter tCity");
```

```
                    String c=sc.next();

                    cs.setInt(1,a);
                    cs.setString(2, b);
                    cs.setString(3, c);

                    cs.execute();
                    con.close();

                }
                catch(Exception e)
                {
                    e.printStackTrace();
                }


        }

}
```

INPUT DATA:

OUTPUT DATA:

Program 2

```
import java.sql.*;
import java.io.*;

class JDBCdynamicstatement
 {
        public static void main(String[] args)
        {
                try
                {
                        //load the driver
                        Class.forName("com.mysql.jdbc.Driver"); //forname method
throws an exception

                        //creating a connection
                        String url="jdbc:mysql://localhost:3306/javasql";
                        String uname="root";
                        String pass="dsu@22";

                        Connection con=DriverManager.getConnection(url, uname,
pass);//cannot create the object for connection interface

                        //create a query
                        String q="insert into table1(tName, tCity) values (?,?)";

                        //get the PreparedStatement Object
                        PreparedStatement pstmt=con.prepareStatement(q);

                        BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
                        System.out.println("Enter name :");
                        String name=br.readLine();

                        System.out.println("Enter city :");
                        String city=br.readLine();

                        //Set the values to the query
                        pstmt.setString(1, name);
                        pstmt.setString(2, city);

                        pstmt.executeUpdate();

                        System.out.println("inserted...");
                        con.close();
                }
                catch(Exception e)
                {
                        e.printStackTrace();
                }


        }
}
```