ADVANCED JAVA PROGRAMMING LAB

(Course Code: 22UPCSC1C16)

A programming laboratory record submitted to Periyar University, Salem

In partial fulfillment of the requirements for the degree of

MASTER OF COMPUTER APPLICATIONS

By

ELANCHEZHIAN M.

[Reg. No.: U22PG507CAP006]

DEPARTMENT OF COMPUTER SCIENCE

PERIYAR UNIVERSITY

(NAAC `A++` Grade with CGPA 3.61) – NIRF RANK 59 – ARIIA RANK 10

PERIYAR PALKALAI NAGAR,

SALEM – 636 011.

(NOVEMBER - 2023)

CERTIFICATE

This is to certify that the Programming Laboratory entitled "**ADVANCED JAVA PROGRAMMING LAB (22UPCSC1C16)"** is a bonafide record work done by Mr. /Ms. _____ Register No: _____ as partial fulfillment of the requirements for the degree of Master of Computer Applications, in the Department of Computer Science, Periyar University, Salem, During the Academic Year 2023-2024.

Staff In-charge Head of the Department

Submitted for the practical examination held on……..……

Internal Examiner External Examiner

CONTENT

SOURCE CODE:

```
package exceptionhandling;

import java.util.Scanner;
import java.io.*;

public class ExceptionHandling {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      try {
         System.out.print("Enter a number: ");
         int number = scanner.nextInt();
         System.out.println("Result: " + (10 / number));
      } catch (ArithmeticException e) {
         System.out.println("Arithmetic error: Division by zero.");
      }

      try {
         System.out.print("Enter a string: ");
         String text = scanner.next();
         text = null;
         System.out.println("Length: " + text.length());
      } catch (NullPointerException e) {
         System.out.println("Null pointer error: Object is null. Caught
Exception: " + e.getMessage());
      }

      try {
         System.out.print("Enter an index: ");
         int index = scanner.nextInt();
         int[] numbers = { 1, 2, 3 };
         System.out.println("Value: " + numbers[index]);
      } catch (ArrayIndexOutOfBoundsException e) {
         System.out.println("Array index is out of bounds.");
      }
```

```
        try {
            System.out.print("Enter a number: ");
            int number = Integer.parseInt(scanner.next());
            System.out.println("Number: " + number);


        } catch (NumberFormatException e) {
            System.out.println("Number format error: Invalid input.");
        }

        try {
            System.out.print("Enter a filename: ");
            String filename = scanner.next();
            File file = new File(filename);
            try (Scanner fileScanner = new Scanner(file)) {
                System.out.println("File content: " + fileScanner.nextLine());
            } catch (FileNotFoundException e) {
                System.out.println("File not found: " + e.getMessage());
            }
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

OUTPUT:

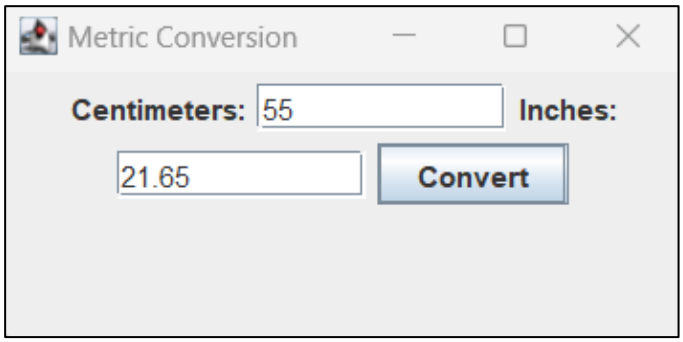
```
Enter a number: 0
Arithmetic error: Division by zero.
Enter a string: 1.5
Null pointer error: Object is null. Caught Exception: Cannot invoke "String.length()" because "text" is null
Enter an index: 5
Array index is out of bounds.
Enter a number: ela
Number format error: Invalid input.
Enter a filename: elan.txt
File not found: elan.txt (The system cannot find the file specified)
```

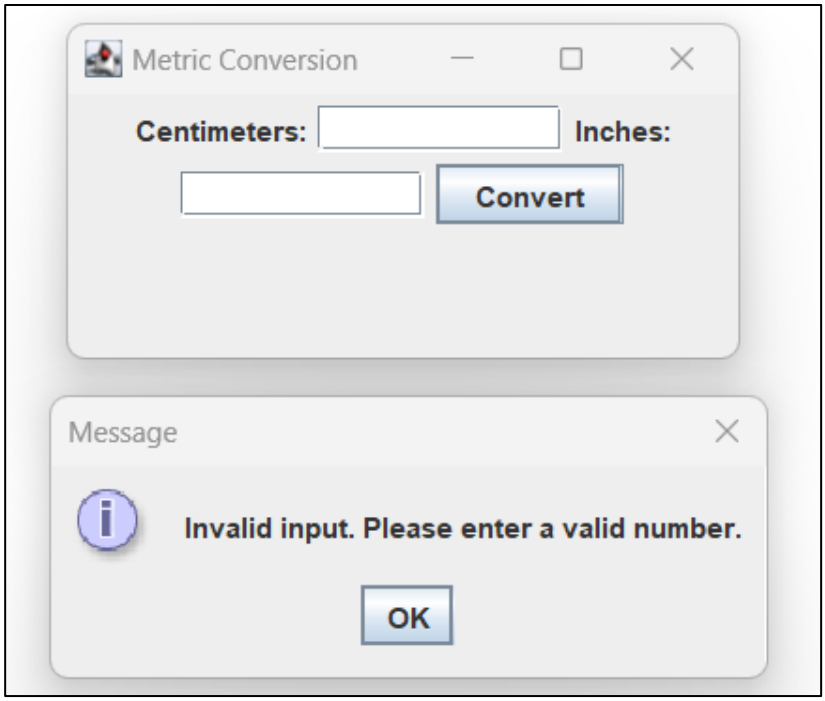
SOURCE CODE:

```
package swing_application;

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

public class MetricConversionApp extends JFrame {
   private JTextField cmTextField, inchTextField;

   public MetricConversionApp() {
      setTitle("Metric Conversion");
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setLayout(new FlowLayout());
      cmTextField = new JTextField(10);
      inchTextField = new JTextField(10);
      JButton convertButton = new JButton("Convert");

      convertButton.addActionListener(e -> {
         try {
            double inches = Double.parseDouble(cmTextField.getText()) /
2.54;
            inchTextField.setText(String.format("%.2f", inches));
         } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(null, "Invalid input. Please enter
a valid number.");
         }
      });

      add(new JLabel("Centimeters:"));
      add(cmTextField);
      add(new JLabel("Inches:"));
      add(inchTextField);
      add(convertButton);
      setSize(300, 150);
      setVisible(true);
   }
```

```
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new MetricConversionApp());
    }
}
```

OUTPUT:


Metric Conversion
Centimeters: 55
Inches:
21.65
Convert

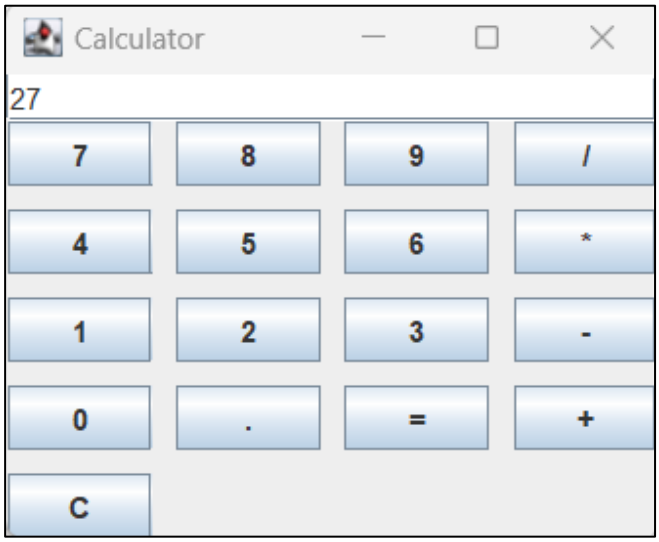

Metric Conversion
Centimeters:
Inches:
Convert
Message
Invalid input. Please enter a valid number.
OK

SOURCE CODE:

```
package simpleCal;

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

public class CalculatorApp extends JFrame {
    private JTextField display = new JTextField(10);
    private String operator = "";
    private double firstOperand = 0;

    public CalculatorApp() {
        setTitle("Calculator");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        String[] buttonLabels = {
            "7", "8", "9", "/",
            "4", "5", "6", "*",
            "1", "2", "3", "-",
            "0", ".", "=", "+", "C"
        };

        JPanel buttonPanel = new JPanel(new GridLayout(5, 4, 10, 10));

        ActionListener buttonListener = e -> {
            String command = ((JButton) e.getSource()).getText();
            if ("0123456789.".contains(command)) {
                display.setText(display.getText() + command);
            } else if ("+-*/".contains(command)) {
                operator = command;
                firstOperand = Double.parseDouble(display.getText());
                display.setText("");
            } else if ("=".equals(command)) {
                double secondOperand = Double.parseDouble(display.getText());
                display.setText(String.valueOf(calculate(firstOperand,
secondOperand, operator)));
```

```
            } else if ("C".equals(command)) {
                display.setText("");
            }
        };

        for (String label : buttonLabels) {
            JButton button = new JButton(label);
            button.addActionListener(buttonListener);
            buttonPanel.add(button);
        }
        setLayout(new BorderLayout());
        add(display, BorderLayout.NORTH);
        add(buttonPanel, BorderLayout.CENTER);
        pack();
        setLocationRelativeTo(null);
    }
    private double calculate(double operand1, double operand2, String
operator) {
        switch (operator) {
            case "+":
                return operand1 + operand2;
            case "-":
                return operand1 - operand2;
            case "*":
                return operand1 * operand2;
            case "/":
                if (operand2 != 0) return operand1 / operand2;
                else return Double.NaN;
            default:
                return Double.NaN;
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new CalculatorApp().setVisible(true));
    }
}
```

OUTPUT:



SOURCE CODE:

index.html:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<html>
<head>
   <title>Exam Registration</title>
   <style>
      .container {
         text-align: center;
         margin-top: 100px;
      }
      .form-group {
         margin-bottom: 20px;
      }
      .form-group label {
         display: inline-block;
         width: 100px;
         text-align: right;
         margin-right: 10px;
      }
   </style>
</head>
<body>
   <div class="container">
      <h2>Exam Registration</h2>
      <form action="register" method="post">
         <div class="form-group">
            <label for="name">Name:</label>
            <input type="text" id="name" name="name" required>
         </div>
         <div class="form-group">
            <label for="rollNumber">Roll Number:</label>
            <input type="text" id="rollNumber" name="rollNumber" required>
         </div>
         <br>
```

```
            <input type="submit" value="Register">
        </form>
    </div>
</body>
</html>
```

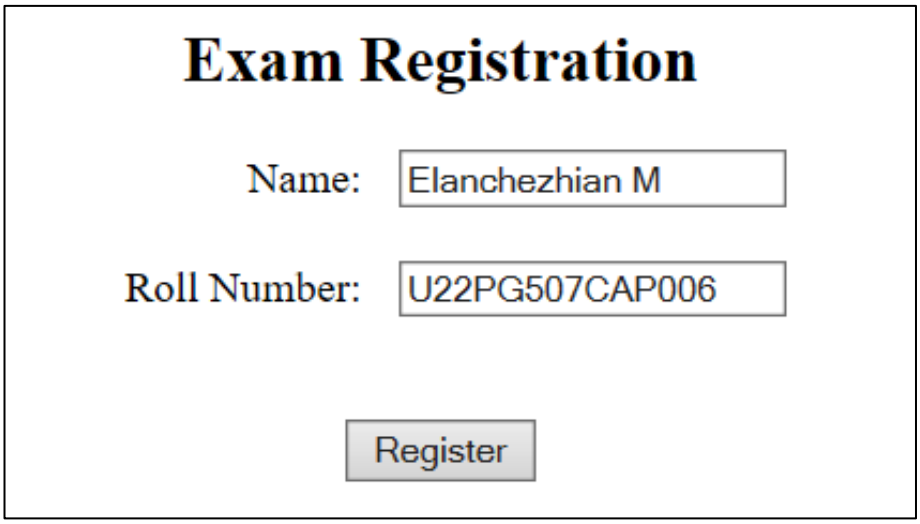
ExamRegistrationServlet.java:

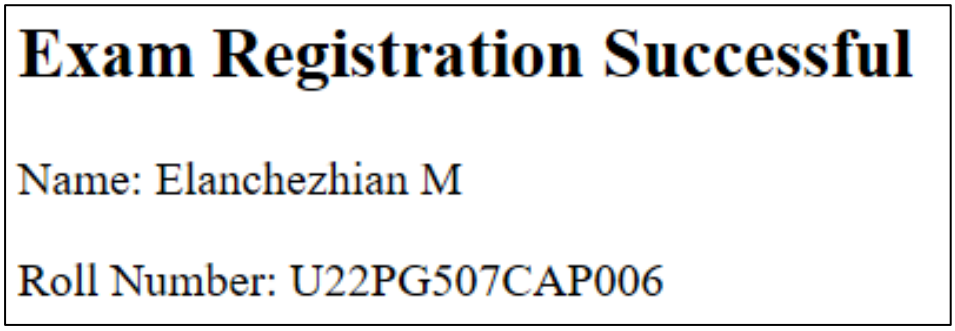
```
package jsp;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;

public class ExamRegistrationServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        String name = request.getParameter("name");
        String rollNumber = request.getParameter("rollNumber");
        response.setContentType("text/html");
        String htmlResponse = "<html><body>";
        htmlResponse += "<h2>Exam Registration Successful</h2>";
        htmlResponse += "<p>Name: " + name + "</p>";
        htmlResponse += "<p>Roll Number: " + rollNumber + "</p>";
        htmlResponse += "</body></html>";
        response.getWriter().println(htmlResponse);
    }
}
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app>
  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
  <servlet>
    <description></description>
    <servlet-name>ExamRegistrationServlet</servlet-name>
    <servlet-class>jsp.ExamRegistrationServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>ExamRegistrationServlet</servlet-name>
    <url-pattern>/register</url-pattern>
  </servlet-mapping>
</web-app>
```

OUTPUT:


Exam Registration
Name:
Elanchezhian M
Roll Number:
U22PG507CAP006
Register


Exam Registration Successful
Name: Elanchezhian M
Roll Number: U22PG507CAP006

SOURCE CODE:

MyServer.java:

```
package server;

import java.io.*;
import java.net.*;

public class MyServer {
    public static void main(String[] args) {
        try (ServerSocket serverSocket = new ServerSocket(5000)) {
            System.out.println("Server started. Listening for connections...");
            while (true) {
                Socket clientSocket = serverSocket.accept();
                System.out.println("Client connected: " +
clientSocket.getInetAddress());
                try (
                    BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
                    PrintWriter out = new
PrintWriter(clientSocket.getOutputStream(), true)
                ) {
                    String message;
                    while ((message = in.readLine()) != null) {
                        out.println("Server received: " + message);
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
                System.out.println("Client disconnected: " +
clientSocket.getInetAddress());
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

MyClient.java

```
package client;

import java.io.*;
import java.net.Socket;

public class MyClient {
   public static void main(String[] args) {
      try (Socket socket = new Socket("localhost", 5000);
          BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
          PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
          BufferedReader userInput = new BufferedReader(new
InputStreamReader(System.in)) ) {

          System.out.println("Connected to the server.");

          String message;
          while (true) {
             System.out.print("Enter a message (type 'exit' to quit): ");
             message = userInput.readLine();
             if (message.equalsIgnoreCase("exit")) break;
             out.println(message);
             System.out.println("Server response: " + in.readLine());
          }
      } catch (IOException e) {
          e.printStackTrace();
      }
   }
}
```

OUTPUT:

Starting the server:

```
Server started. Listening on port 5000
```

Connections from server to client:

```
Connected to the server.
Enter a message to send to the server (type 'exit' to quit): Hai AI_kera
Server response: Server received the message: Hai AI_kera
Enter a message to send to the server (type 'exit' to quit): exit
```

Client disconnected:

```
Server started. Listening on port 5000
Client connected: 127.0.0.1
Client connected: 127.0.0.1
Received message from client: Hai AI_kera
Client disconnected: 127.0.0.1
```

SOURCE CODE:

PersonBean.java

```
package javaBean;

public class PersonBean {
   private String name;
   private int age;
   private String address;

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }

   public int getAge() {
      return age;
   }

   public void setAge(int age) {
      this.age = age;
   }

   public String getAddress() {
      return address;
   }

   public void setAddress(String address) {
      this.address = address;
   }
}
```

JavaBean.java

```
package javaBean;

public class JavaBean {
   public static void main(String[] args) {

      PersonBean person = new PersonBean();

      person.setName("Elanchezhian M");
      person.setAge(22);
      person.setAddress("E/15, Annathanapatty Police Quarters, Salem-02");

      System.out.println("Name: " + person.getName());
      System.out.println("Age: " + person.getAge());
      System.out.println("Address: " + person.getAddress());
   }
}
```

OUTPUT:

```
Name: Elanchezhian M
Age: 22
Address: E/15, Annathanapatty Police Quarters, Salem-02
```

SOURCE CODE:

```
package accMod;

class Employee {
    public String name;
    protected int employeeId;
    private double salary;

    public Employee(String name, int employeeId, double salary) {
        this.name = name;
        this.employeeId = employeeId;
        this.salary = salary;
    }

    public void displayDetails() {
        System.out.println("Name: " + name);
        System.out.println("Employee ID: " + employeeId);
        System.out.println("Salary: $" + salary);
    }
}

class Manager extends Employee {
    private String department;

    public Manager(String name, int employeeId, double salary, String
department) {
        super(name, employeeId, salary);
        this.department = department;
    }

    public void displayDetails() {
        super.displayDetails();
        System.out.println("Department: " + department);
    }
}
```

```
class Staff extends Employee {
    private int workingHours;

    public Staff(String name, int employeeId, double salary, int
workingHours) {
        super(name, employeeId, salary);

        this.workingHours = workingHours;
    }

    public void displayDetails() {
        super.displayDetails();
        System.out.println("Working Hours: " + workingHours);
    }
}

public class AccessModifier {
    public static void main(String[] args) {
        Manager manager = new Manager("Elanchezhian M", 101, 60000.0,
"Sales");
        System.out.println("Manager Details:");
        manager.displayDetails();

        System.out.println();

        Staff staff = new Staff("Dhanush B", 202, 40000.0, 40);
        System.out.println("Staff Details:");
        staff.displayDetails();
    }
}
```

OUTPUT:

```
Manager Details:
Name: Elanchezhian M
Employee ID: 101
Salary: $60000.0
Department: Sales

Staff Details:
Name: Dhanush B
Employee ID: 202
Salary: $40000.0
Working Hours: 40
```

SOURCE CODE:

```
package javaInterface;

import java.util.Scanner;

interface Shape {
   void getArea();
   void getPerimeter();
}

class Rectangle implements Shape {
   private int length, width;

   Rectangle(int length, int width) {
      this.length = length;
      this.width = width;
   }
     public void getArea() {
      System.out.println("Rectangle Area: " + (length * width));
   }
      public void getPerimeter() {
      System.out.println("Rectangle Perimeter: " + 2 * (length + width));
   }
}

class Circle implements Shape {
   private double radius;

   Circle(double radius) {
      this.radius = radius;
   }
    public void getArea() {
      System.out.println("Circle Area: " + (Math.PI * radius * radius));
   }
    public void getPerimeter() {
      System.out.println("Circle Perimeter: " + (2 * Math.PI * radius));
   }
}
```

```
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the length of the rectangle: ");
        int length = scanner.nextInt();
        System.out.print("Enter the width of the rectangle: ");
        int width = scanner.nextInt();

        Shape rectangle = new Rectangle(length, width);
        rectangle.getArea();
        rectangle.getPerimeter();

        System.out.print("Enter the radius of the circle: ");
        double radius = scanner.nextDouble();

        Shape circle = new Circle(radius);
        circle.getArea();
        circle.getPerimeter();

        scanner.close();
    }
}
```

OUTPUT:

```
Enter the length of the rectangle: 3
Enter the width of the rectangle: 5
Rectangle Area: 15
Rectangle Perimeter: 16
Enter the radius of the circle: 3.5
Circle Area: 38.48451000647496
Circle Perimeter: 21.991148575128552
```

SOURCE CODE:

RemoteCalculator.java(interface)

```
package rmi_squ;
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface RemoteCalculator extends Remote {
   int add(int a, int b) throws RemoteException;
}
```

CalculatorImp.java

```
package rmi_squ;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class CalculatorImpl extends UnicastRemoteObject
implements RemoteCalculator {
    protected CalculatorImpl() throws RemoteException {
       super();
    }
    @Override
    public int add(int a, int b) throws RemoteException {
       return a + b;
    }
}
```

Server.java

```
package rmi_squ;

import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Server {
      public static void main(String[] args) {
            try {
             RemoteCalculator calculator = new CalculatorImpl();
```

```
            Registry registry = LocateRegistry.createRegistry(1099);
            registry.rebind("CalculatorService", calculator);
            System.out.println("Server is running...");
        }
        catch (Exception e) {
            System.err.println("Server exception: " + e.toString());
            e.printStackTrace();
        }
    }
}
```

Client.java

```
package rmi_squ;

import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Client {
    public static void main(String[] args) {
        try {
                Registry registry = LocateRegistry.getRegistry("localhost",
1099);
            RemoteCalculator calculator = (RemoteCalculator)
registry.lookup("CalculatorService");
             int result = calculator.add(10, 20);
            System.out.println("Result: " + result);
        } catch (Exception e) {
            System.err.println("Client exception: " + e.toString());
            e.printStackTrace();
        }
    }
}
```

OUTPUT:

Server started:

```
Server is running...
```

When Remote Method Invocation:

```
Result: 30
```

SOURCE CODE:

```
import java.sql.*;

public class MyJdbc {
   public static void main(String args[]) {
      String jdbcUrl = "jdbc:mysql://localhost:3306/elan";
      String username = "root";
      String password = "Elan@27";
       try (Connection connection = DriverManager.getConnection(jdbcUrl,
username, password)) {
         System.out.println("Connected to the database successfully!");
          String insertData1 = "INSERT INTO emp (id, name, age) VALUES
(1, 'Elanchezhian M', 21)";
         String insertData2 = "INSERT INTO emp (id, name, age) VALUES
(2, 'Dhanush B', 30)";
         String insertData3 = "INSERT INTO emp (id, name, age) VALUES
(3, 'Hariharan M', 28)";
         try (Statement statement = connection.createStatement()) {
            statement.executeUpdate(insertData1);
            statement.executeUpdate(insertData2);
            statement.executeUpdate(insertData3);
            System.out.println("Data inserted successfully!");
         }
      } catch (SQLException e) {
         System.err.println("Database connection or insertion error: " +
e.getMessage());
         e.printStackTrace();
      }
   }
}
```

Mysql creation in MySQL Command Line Client:

mysql> show databases;

```
+--------------------+
| Database           |
+--------------------+
| information_schema |
| elan               |
| mysql              |
| performance_schema |
| test               |
+--------------------+
```

5 rows in set (0.00 sec)

mysql> create database elan;

mysql> use elan;

Database changed

mysql> create table emp(id int(10),name varchar(40),age int(3));

Query OK, 0 rows affected (0.00 sec)

mysql> desc emp;

```
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| id    | int(10)    | YES  |     | NULL    |       |
| name  | varchar(40) | YES  |     | NULL    |       |
| age   | int(3)     | YES  |     | NULL    |       |
+-------+------------+------+-----+---------+-------+
```

3 rows in set (0.01 sec)

OUTPUT:

```
Connected to the database successfully!
Data inserted successfully!
```

mysql> select * from emp;

```
+------+---------------+------+
| id   | name          | age |
+------+---------------+------+
|    1 | Elanchezhian M |   21 |
|    2 | Dhanush B      |   30 |
|    3 | Hariharan M    |   28 |
+------+---------------+------+
3 rows in set (0.00 sec)
```