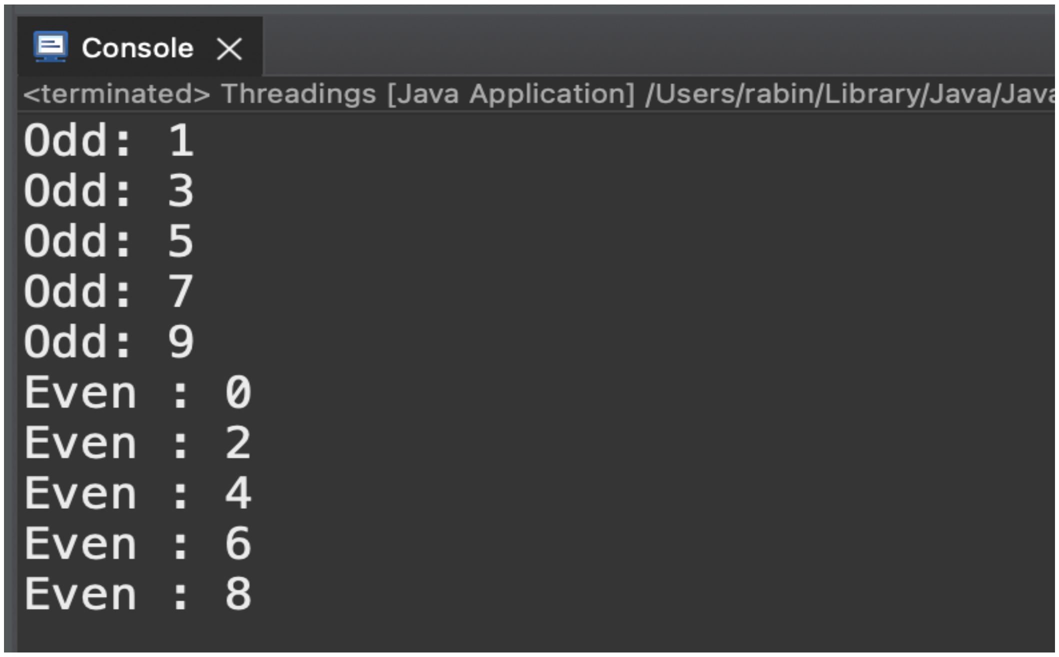
Implement a Java program that creates two threads. One thread should print even numbers, and the other should print odd numbers from 1 to 10.

```
package thread;
public class Threadings {

	class A extends Thread {
		public void run() {
			for (int i = 0; i < 10; i++) {
				if (i % 2 == 0) {
					System.out.println("Even : " + i);
				}
			}
		}
	}

	class B extends Thread {
		public void run() {
			for (int i = 0; i < 10; i++) {
				if (i % 2 != 0) {
					System.out.println("Odd: " + i);
				}
			}
		}
	}

	public static void main(String[] args) {
		Threadings threadings = new Threadings(); // create an instance of the outer class
		A a = threadings.new A(); // create an instance of class A
		B b = threadings.new B(); // create an instance of class B
		try {
			a.sleep(100);
		} catch (InterruptedException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}

		a.start(); // start the thread for class A
		b.start(); // start the thread for class B
	}
}
```

OUTPUT:

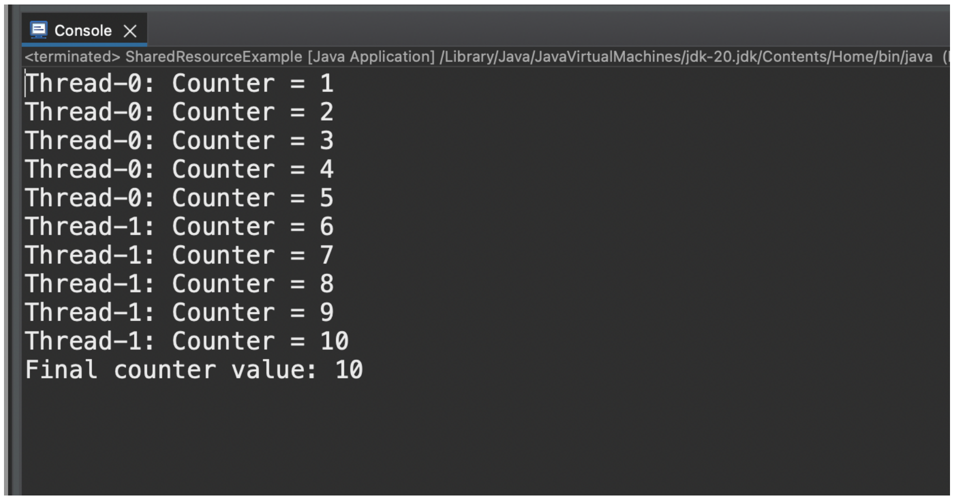

Console

```
<terminated> Threadings [Java Application] /Users/rabin/Library/Java/Java
Odd: 1
Odd: 3
Odd: 5
Odd: 7
Odd: 9
Even : 0
Even : 2
Even : 4
Even : 6
Even : 8
```

Create a Java program with two threads sharing a common resource (e.g., a counter). Implement synchronization to ensure that the threads alternate incrementing the counter.

```
package thread;

public class SharedResourceExample {
    private static final int MAX_COUNT = 5;
    private static int counter = 0;

    public static void main(String[] args) {
        // Create two threads
        Thread thread1 = new Thread(new IncrementTask());
        Thread thread2 = new Thread(new IncrementTask());

        // Start the threads
        thread1.start();
        thread2.start();

        try {
            // Wait for both threads to finish
            thread1.join();
            thread2.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println("Final counter value: " + counter);
    }

    static class IncrementTask implements Runnable {
        @Override
        public void run() {
            for (int i = 0; i < MAX_COUNT; i++) {
                synchronized (SharedResourceExample.class) {
                    // Increment the counter
                    counter++;
                    System.out.println(Thread.currentThread().getName() + ": Counter = " + counter);
                }
            }
        }
    }
}
```

OUTPUT:

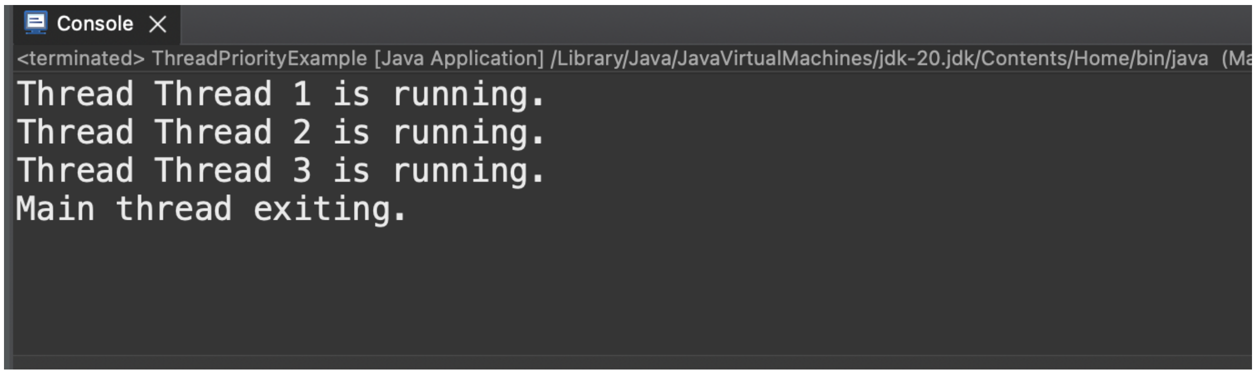


```
<terminated> SharedResourceExample [Java Application] /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java  (
Thread-0: Counter = 1
Thread-0: Counter = 2
Thread-0: Counter = 3
Thread-0: Counter = 4
Thread-0: Counter = 5
Thread-1: Counter = 6
Thread-1: Counter = 7
Thread-1: Counter = 8
Thread-1: Counter = 9
Thread-1: Counter = 10
Final counter value: 10
```

Develop a Java program that creates three threads with different priorities.

```
package thread;
public class ThreadPriorityExample {
    public static void main(String[] args) {
        PriorityThread thread1 = new PriorityThread("Thread 1");
        PriorityThread thread2 = new PriorityThread("Thread 2");
        PriorityThread thread3 = new PriorityThread("Thread 3");

        // Set thread priorities
        thread1.setPriority(Thread.MIN_PRIORITY); // Lowest priority (1)
        thread2.setPriority(Thread.NORM_PRIORITY); // Default priority (5)
        thread3.setPriority(Thread.MAX_PRIORITY); // Highest priority (10)

        // Start the threads
        thread1.start();
        thread2.start();
        thread3.start();

        // Wait for all threads to finish
        try {
            thread1.join();
            thread2.join();
            thread3.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println("Main thread exiting.");
    }
}
```

OUTPUT:

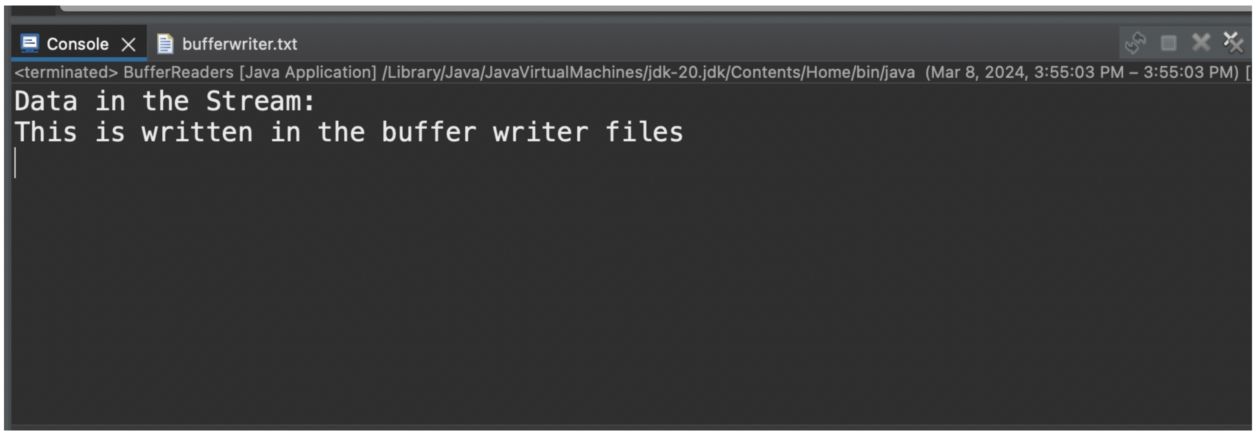


```
<terminated> ThreadPriorityExample [Java Application] /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java  (Ma
Thread Thread 1 is running.
Thread Thread 2 is running.
Thread Thread 3 is running.
Main thread exiting.
```

Create a Java program that reads data from a text file and displays it on the console. Ensure Proper exception handling.

```
package File_Handling;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

public class BufferReaders {
	public static void main(String[] args) {
		char[] array = new char[100];
		try {
			// creates a file reader
			FileReader readFile = new FileReader("../Classroom/src/bufferwriter.txt");
			// Creates a buffer reader
			BufferedReader buffers = new BufferedReader(readFile);
			// Reads characters
			try {
				System.out.println("Data in the Stream: ");
				buffers.read(array);
			} catch (IOException e) {
				e.printStackTrace();
			}
			System.out.println(array);

		} catch (FileNotFoundException e) {
			e.printStackTrace();
		}
	}
}
```

OUTPUT:

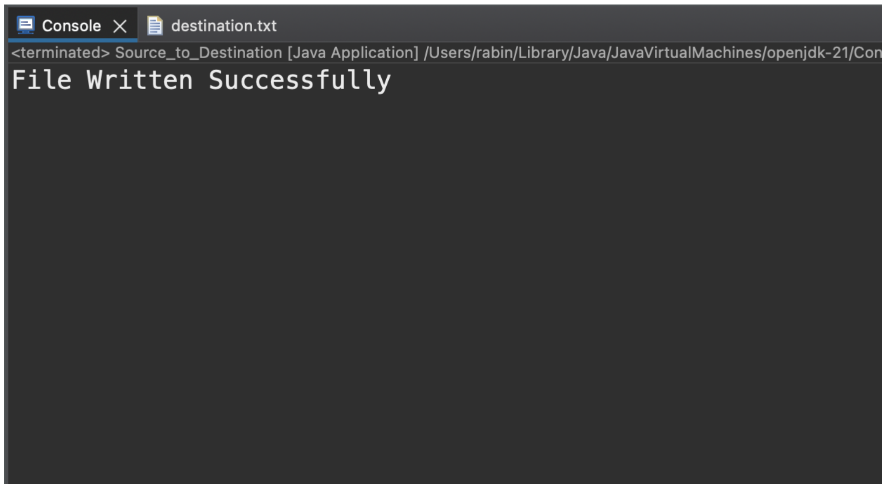


```
<terminated> BufferReaders [Java Application] /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java  (Mar 8, 2024, 3:55:03 PM – 3:55:03 PM)
Data in the Stream:
This is written in the buffer writer files
```

Write a Java program to copy the contents of one text file to another new file.

```
package File_Handling;

import java.io.BufferedReader;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.OutputStreamWriter;

public class Source_to_Destination {

	public static void main(String[] args) {

		try {
			// Read data from the source file
			BufferedReader reader = new BufferedReader(new
FileReader("../Classroom/src/bufferwriters.txt"));

			StringBuilder stringBuilder = new StringBuilder();
			String line;
			while ((line = reader.readLine()) != null) {
				stringBuilder.append(line);
				stringBuilder.append(System.lineSeparator());
			}
			reader.close();
			String data = stringBuilder.toString();

			// Creates a FileOutputStream
			FileOutputStream file = new
FileOutputStream("../Classroom/src/destination.txt");

			// Creates an OutputStreamWriter
			OutputStreamWriter output = new OutputStreamWriter(file);

			// Writes string to the file
			output.write(data);
			System.out.println("\n File Written Successfully");

			// Closes the writer
			output.close(); // close outputStream
		} catch (Exception e) {
			e.printStackTrace();
		}
	}
}
```

OUTPUT:



Console destination.txt

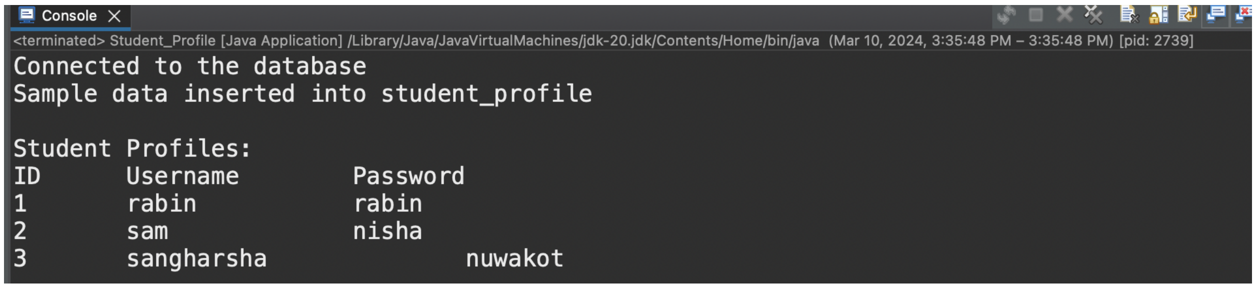
```
<terminated> Source_to_Destination [Java Application] /Users/rabin/Library/Java/JavaVirtualMachines/openjdk-21/Con
File Written Successfully
```

Database Operations in Java:
Write a Java program that connects to a MySQL or PostgreSQL database and performs operations. Create a table named student_profile in the database, including fields for username and password. Write a Java function to select and display data from the student_profile table in tabular form.

```
package mysql;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class Student_Profile {
	final String DRIVER = "com.mysql.cj.jdbc.Driver"; // Driver link provided

	// Database connection details
	final static String DBNAME = "JavaCollege"; // Database table name
	final static String HOST = "localhost"; // Database server host
	final static String DBUSER = "root"; // database Username
	final static String DBPASS = "Neupane@11"; // Database password
	final static int PORT = 3306; // Database port name
	final static String URL = "jdbc:mysql://" + HOST + ":" + PORT + "/" + DBNAME; // JDBC
connection URL

	// JDBC variables for opening and managing connection
	private static Connection connection;
	private static Statement statement;

	public static void main(String[] args) {
		try {
			// Open a connection
			connection = DriverManager.getConnection(URL, DBUSER, DBPASS);
			System.out.println("Connected to the database");

			// Insert sample data into student_profile
			insertSampleData();

			// Select and display data from student_profile table
			selectAndDisplayData();
		} catch (SQLException e) {
			e.printStackTrace();
		} finally {
			try {
				if (connection != null) {
					connection.close();
				}
			} catch (SQLException e) {
				e.printStackTrace();
			}
		}
	}

	// Insert sample data into student_profile
	private static void insertSampleData() throws SQLException {
		statement = connection.createStatement();
		String insertDataSQL = "INSERT INTO student_profile (id, username, password)
VALUES " + "(1, 'rabin', 'rabin'),"
				+ "(2, 'sam', 'nisha')," + "(3, 'sangharsha', 'nuwakot')";

		statement.executeUpdate(insertDataSQL);
		System.out.println("Sample data inserted into student_profile");
	}

	// Select and display data from student_profile table
	private static void selectAndDisplayData() throws SQLException {
```

```
                    statement = connection.createStatement();
                    String selectDataSQL = "SELECT * FROM student_profile";
                    ResultSet resultSet = statement.executeQuery(selectDataSQL);

                    System.out.println("\nStudent Profiles:");
                    System.out.println("ID\tUsername\tPassword");
                    while (resultSet.next()) {
                              int id = resultSet.getInt("id");
                              String username = resultSet.getString("username");
                              String password = resultSet.getString("password");
                              System.out.println(id + "\t" + username + "\t\t" + password);
                    }
          }
}
```

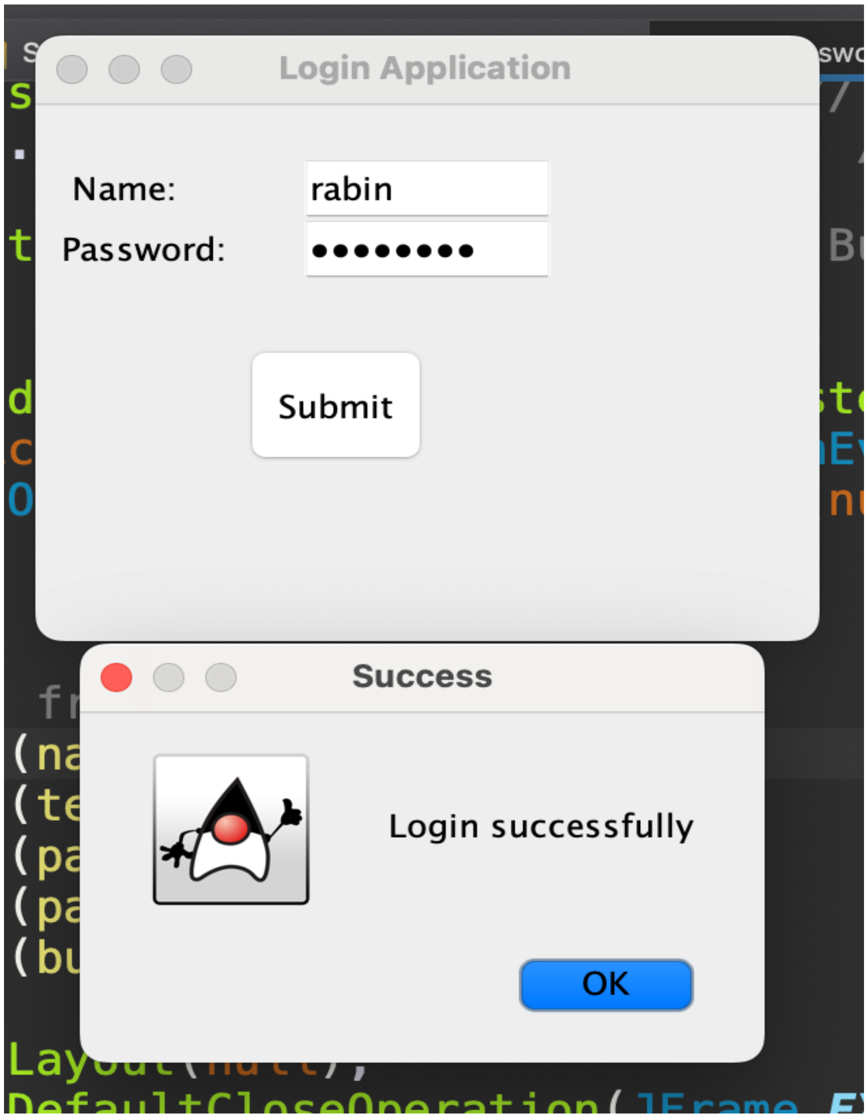
OUTPUT:



```
<terminated> Student_Profile [Java Application] /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java  (Mar 10, 2024, 3:35:48 PM – 3:35:48 PM) [pid: 2739]
Connected to the database
Sample data inserted into student_profile

Student Profiles:
ID      Username        Password
1       rabin           rabin
2       sam             nisha
3       sangharsha              nuwakot
```

Java Swing GUI Program:
Write a Java Swing program with a GUI containing username and password fields, and a submit button. Connect the program to the student_profile database created in the previous program. If the provided credentials exist in the table, move to a success window; otherwise, display the error message Credentials not matched.

```
package mysql_Database;

// imports
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
import javax.swing.JButton;
import javax.swing.JPasswordField;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class Id_Password_Database {

	public static void main(String[] args) {
		final String DRIVER = "com.mysql.cj.jdbc.Driver"; // JDBC Driver class

		// Database connection details
		final String DBNAME = "JavaCollege"; // Database table name
		final String HOST = "localhost"; // Database server host
		final String DBUSER = "root"; // database Username
		final String DBPASS = "Neupane@11"; // Database password
		final int PORT = 3306; // Database port name
		final String URL = "jdbc:mysql://" + HOST + ":" + PORT + "/" + DBNAME; // JDBC
connection URL

		// JFrame and UI objects
		JFrame frame = new JFrame("Login Application");
		JLabel name = new JLabel(" Name:");
		JLabel password = new JLabel("Password: ");
		JButton button = new JButton("Submit");

		JTextField textField = new JTextField();
		JPasswordField passfield = new JPasswordField(); // create for passwordField

		// position set Display
		name.setBounds(10, 20, 150, 30); // name label position
		textField.setBounds(100, 20, 100, 30); // name text field position

		password.setBounds(10, 45, 150, 30); // Password name position
		passfield.setBounds(100, 45, 100, 30); // password text field Position

		button.setBounds(80, 100, 70, 50); // Submit Button position

		// Action Listener for Submit
		button.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				try {
					// Retrieve username and password from UI components

					String username = textField.getText();
					char[] passwordChars = passfield.getPassword();
					String password = new String(passwordChars);

					// JDBC connection SETUP
					Class.forName(DRIVER); // Loading Driver
					Connection conn = DriverManager.getConnection(URL,
DBUSER, DBPASS); // Establish the connection

					// Insert Records
					Statement state = conn.createStatement(); // object create
for connection
					String sql = "SELECT * FROM student_profile where
username='" + username + "' and password='"+ password + "'";
					ResultSet rs = state.executeQuery(sql); // Get all records
from table

					// Create an instance
					Display_From_Database tableData = new
Display_From_Database();

					if (rs.next()) {
						System.out.println(rs.getInt("id") + "\t \t" +
rs.getString("username") + "\t\t"+ rs.getString("password"));
						JOptionPane.showMessageDialog(null, "Login
successfully", "Success",
          JOptionPane.INFORMATION_MESSAGE);
                                        tableData.show(); // Display data from database
                                    } else {
                                        System.out.println("Invalid login credentials");
                                        JOptionPane.showMessageDialog(null, "Invalid
login credentials", "Error",
                                                JOptionPane.ERROR_MESSAGE);
                                    }
                                    rs.close();
                                    state.close();
                                    conn.close();

                                } catch (SQLException ex) {
                                    System.out.println(ex);

                                } catch (ClassNotFoundException e1) {
                                    // TODO Auto-generated catch block
                                    e1.printStackTrace();
                                }

                        }
                });
                // add components to frames
                frame.add(name);
                frame.add(textField);
                frame.add(password);
                frame.add(passfield);
                frame.add(button);

                // Set layout and display settings for the frame
                frame.setLayout(null);
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.setSize(300, 250);
                frame.setVisible(true);

        }
}
```
OUTPUT:

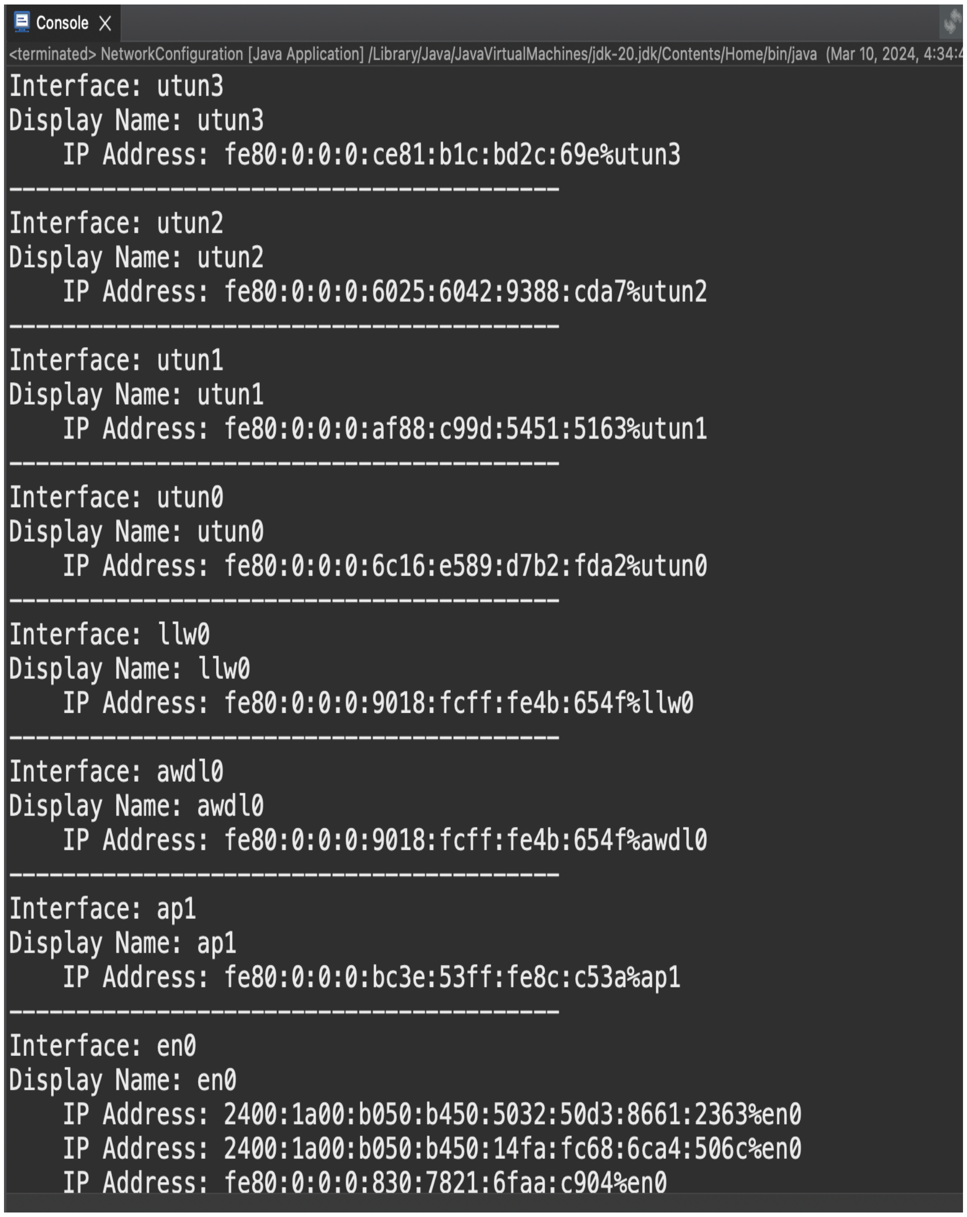


Login successfully

Java Program for Network Configuration:
Write a simple Java program that displays the network configuration of your computer.

```
package network_Configuration;

import java.net.InetAddress;
import java.net.NetworkInterface;
import java.net.SocketException;
import java.util.Enumeration;
public class NetworkConfiguration {
   public static void main(String[] args) {
      try {
         // Get all network interfaces
```

```
        Enumeration<NetworkInterface>                    networkInterfaces                          =
NetworkInterface.getNetworkInterfaces();
        while (networkInterfaces.hasMoreElements()) {
          NetworkInterface networkInterface = networkInterfaces.nextElement();
          System.out.println("Interface: " + networkInterface.getName());
          System.out.println("Display Name: " + networkInterface.getDisplayName());

          // Get all IP addresses for the network interface
          Enumeration<InetAddress> inetAddresses = networkInterface.getInetAddresses();
          while (inetAddresses.hasMoreElements()) {
            InetAddress inetAddress = inetAddresses.nextElement();
            System.out.println("    IP Address: " + inetAddress.getHostAddress());
          }
          System.out.println("----------------------------------------");
        }
      } catch (SocketException e) {
        e.printStackTrace();
      }
    }
}
```
OUTPUT:



```
<terminated> NetworkConfiguration [Java Application] /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java  (Mar 10, 2024, 4:34:4
Interface: utun3
Display Name: utun3
    IP Address: fe80:0:0:0:ce81:b1c:bd2c:69e%utun3
----------------------------------------
Interface: utun2
Display Name: utun2
    IP Address: fe80:0:0:0:6025:6042:9388:cda7%utun2
----------------------------------------
Interface: utun1
Display Name: utun1
    IP Address: fe80:0:0:0:af88:c99d:5451:5163%utun1
----------------------------------------
Interface: utun0
Display Name: utun0
    IP Address: fe80:0:0:0:6c16:e589:d7b2:fda2%utun0
----------------------------------------
Interface: llw0
Display Name: llw0
    IP Address: fe80:0:0:0:9018:fcff:fe4b:654f%llw0
----------------------------------------
Interface: awdl0
Display Name: awdl0
    IP Address: fe80:0:0:0:9018:fcff:fe4b:654f%awdl0
----------------------------------------
Interface: ap1
Display Name: ap1
    IP Address: fe80:0:0:0:bc3e:53ff:fe8c:c53a%ap1
----------------------------------------
Interface: en0
Display Name: en0
    IP Address: 2400:1a00:b050:b450:5032:50d3:8661:2363%en0
    IP Address: 2400:1a00:b050:b450:14fa:fc68:6ca4:506c%en0
    IP Address: fe80:0:0:0:830:7821:6faa:c904%en0
```

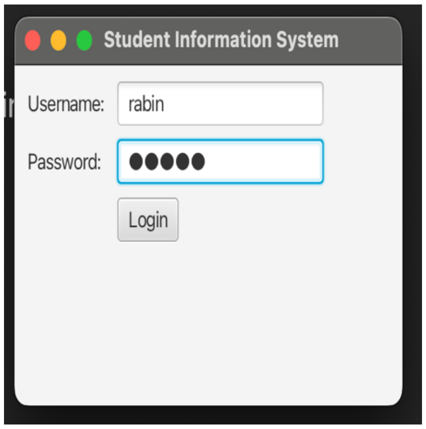
Create a JavaFX Program for a Student Information System:

Develop a JavaFX program to achieve the following tasks:
Login Page:
Implement a login page using JavaFX. Allow users to input their credentials (e.g., username and password).
Display Student Information:
Connect the program to the student_profile table in the database. After successful login, create a display page to showcase information from the student_profile table, such as student details.
Ensure that the program provides a seamless transition from the login page to the

student information display page in the JavaFX application.

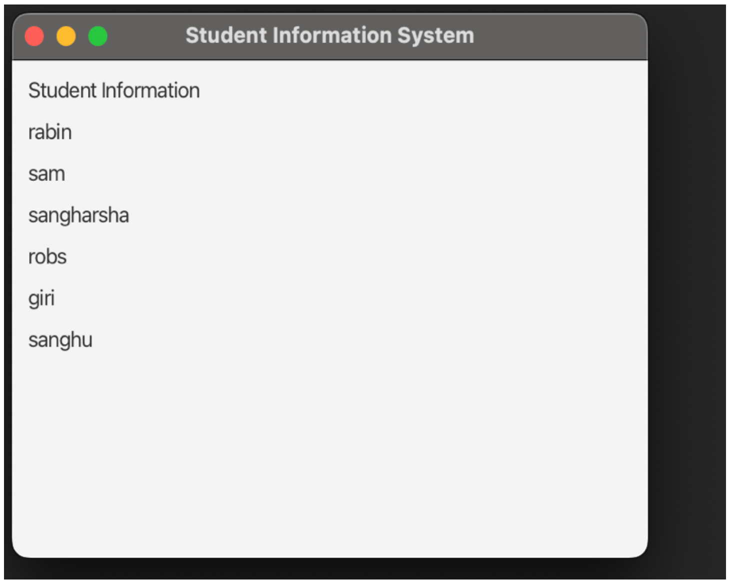
```
package com.example.javafxdemo;

import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.stage.Stage;
import java.sql.*;

public class StudentInformationSystem extends Application {

  @Override
  public void start(Stage primaryStage) {
    primaryStage.setTitle("Student Information System");

    // Login Page
    GridPane loginGrid = new GridPane();
    loginGrid.setPadding(new Insets(10, 10, 10, 10));
    loginGrid.setVgap(8);
    loginGrid.setHgap(10);

    // Username Label
    Label usernameLabel = new Label("Username:");
    GridPane.setConstraints(usernameLabel, 0, 0);

    // Username Input
    TextField usernameInput = new TextField();
    GridPane.setConstraints(usernameInput, 1, 0);

    // Password Label
    Label passwordLabel = new Label("Password:");
    GridPane.setConstraints(passwordLabel, 0, 1);

    // Password Input
    PasswordField passwordInput = new PasswordField();
    GridPane.setConstraints(passwordInput, 1, 1);

    // Login Button
    Button loginButton = new Button("Login");
    GridPane.setConstraints(loginButton, 1, 2);
    loginButton.setOnAction(e -> {
            // Authenticate user here (e.g., check credentials against database)
            // If authenticated, show student information page
            primaryStage.setScene(createStudentInfoScene());
        });

        loginGrid.getChildren().addAll(usernameLabel,     usernameInput,     passwordLabel,
passwordInput, loginButton);

        Scene loginScene = new Scene(loginGrid, 300, 200);

        primaryStage.setScene(loginScene);
        primaryStage.show();
    }

    // Method to create the student information display page
    private Scene createStudentInfoScene() {
        GridPane studentInfoGrid = new GridPane();
        studentInfoGrid.setPadding(new Insets(10, 10, 10, 10));
        studentInfoGrid.setVgap(8);
        studentInfoGrid.setHgap(10);

        // Placeholder student information display
        Label studentLabel = new Label("Student Information");
        GridPane.setConstraints(studentLabel, 0, 0);

        // Display student information fetched from the database
        try {
            final String DRIVER = "com.mysql.cj.jdbc.Driver";
            final String DBNAME = "JavaCollege";
            final String HOST = "localhost";
            final String DBUSER = "root";
            final String DBPASS = "Neupane@11";
            final int PORT = 3306;
            final String URL = "jdbc:mysql://" + HOST + ":" + PORT + "/" + DBNAME;

            Class.forName(DRIVER);
            try (Connection conn = DriverManager.getConnection(URL, DBUSER, DBPASS)) {
                String sql = "SELECT username FROM student_profile";
                try (PreparedStatement statement = conn.prepareStatement(sql);
                     ResultSet resultSet = statement.executeQuery()) {
                    int row = 1;
                    while (resultSet.next()) {
                        String data = resultSet.getString("username");
                        Label usernameLabel = new Label(data);
                    GridPane.setConstraints(usernameLabel, 0, row++);
                    studentInfoGrid.getChildren().add(usernameLabel);
                }
            }
        }
    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setTitle("Error");
        alert.setHeaderText("Database Error");
        alert.setContentText("An error occurred while accessing the database.");
        alert.showAndWait();
    }

    studentInfoGrid.getChildren().add(studentLabel);

    return new Scene(studentInfoGrid, 400, 300);
  }

  public static void main(String[] args) {
    launch(args);
  }
}
```

OUTPUT



Username: rabin

Password: ●●●●●



```
Student Information
rabin
sam
sangharsha
robs
giri
sanghu
```

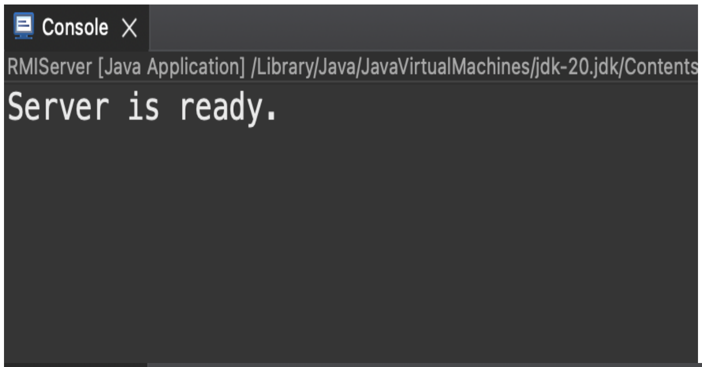
```
}
```

Write a program to illustrate the architecture of JAVA RMI

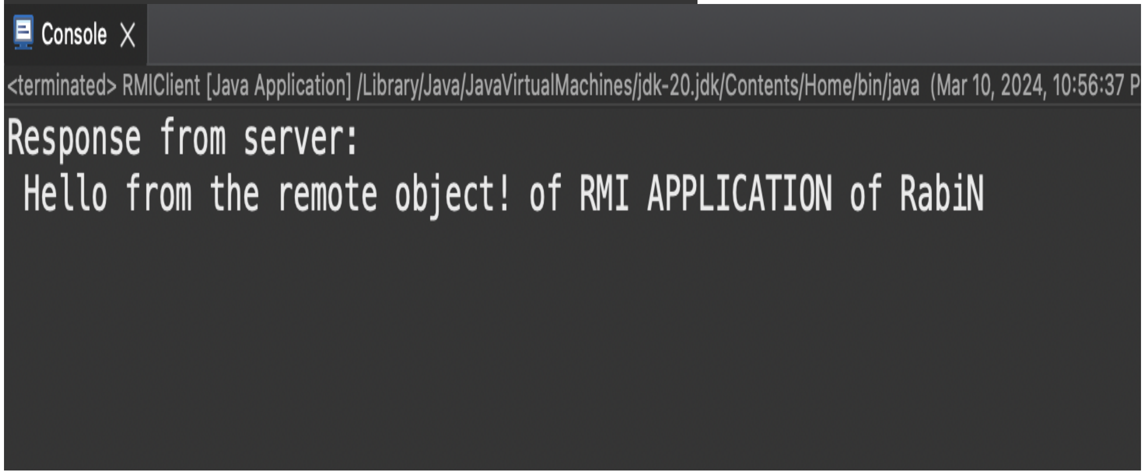
```
package rmi;
import java.rmi.Remote;
import java.rmi.RemoteException;

interface MyRemoteInterface extends Remote {
	String sayHello() throws RemoteException;
}



package rmi;
import java.rmi.RemoteException;
class MyRemoteObject implements MyRemoteInterface {
		@Override
		public String sayHello() throws RemoteException {
			return "\n Hello from the remote object! of RMI APPLICATION of RabiN";
		}
}

package rmi;
//Server program
import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;

public class RMIServer {
 public static void main(String[] args) {
     try {
         MyRemoteObject remoteObject = new MyRemoteObject();
         MyRemoteInterface stub = (MyRemoteInterface)
UnicastRemoteObject.exportObject(remoteObject, 0);

         Registry registry = LocateRegistry.createRegistry(1099);
         registry.rebind("MyRemoteObject", stub);

         System.out.println("Server is ready.");
     } catch (RemoteException e) {
         System.err.println("Server exception: " + e.toString());
         e.printStackTrace();
     }
 }
}
```

```
package rmi;
//Client program
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class RMIClient {

	public static void main(String[] args) {
		try {
			Registry registry = LocateRegistry.getRegistry("localhost", 1099);
			MyRemoteInterface remoteObject = (MyRemoteInterface)
registry.lookup("MyRemoteObject");

			String response = remoteObject.sayHello();
			System.out.println("Response from server: " + response);
		} catch (Exception e) {
			System.err.println("Client exception: " + e.toString());
			e.printStackTrace();
		}
	}
}
```

OUTPUT:



```
RMIServer [Java Application] /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents
Server is ready.
```



```
<terminated> RMIClient [Java Application] /Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/bin/java (Mar 10, 2024, 10:56:37 P
Response from server:
 Hello from the remote object! of RMI APPLICATION of RabiN
```

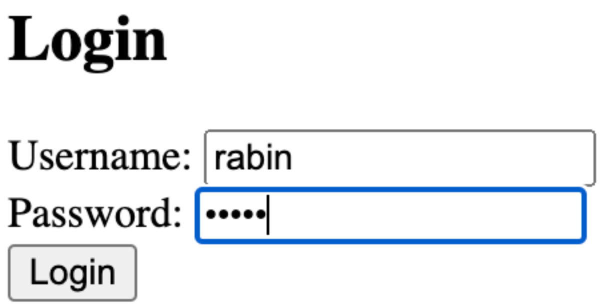
JSP Handling HTML Form Data:
Write a JSP program that handles HTML form data. Create a login page in JSP, check if the credentials match the ones stored in the database table, and forward to a success page or show an
error. Additionally, create a separate page to display a list of users.

```
// LOGIN .JSP
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<%@ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Login Page</title>
</head>
<body>
   <h2>Login</h2>
   <form method="post" action="loginController.jsp">
      Username: <input type="text" name="username" required><br>
      Password: <input type="password" name="password" required><br>
      <input type="submit" value="Login">
   </form>
</body>
</html>

//loginController
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<%@ page import="java.sql.*" %>
<%@ page import="java.io.*" %>

<%
   // Establish database connection
   Connection conn = null;
   String url = "jdbc:mysql://localhost:3306/JavaCollege";
   String user = "root";
   String password = "Neupane@11";

   try {
      Class.forName("com.mysql.jdbc.Driver");
      conn = DriverManager.getConnection(url, user, password);
      Statement statement = conn.createStatement();
```

```
        // Retrieve form data
        String username = request.getParameter("username");
        String passwordInput = request.getParameter("password");

        // Query database for user
        String query = "SELECT * FROM Login_Check WHERE username='" + username + "' AND
password='" + passwordInput + "'";
        ResultSet rs = statement.executeQuery(query);

        if (rs.next()) {
            // If user exists, forward to success page
            response.sendRedirect("success.jsp");
        } else {
            // If user does not exist, show error
            out.println("Invalid credentials. Please try again.");
        }

        rs.close();
        statement.close();
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        if (conn != null) {
            try {
                conn.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
%>

// success.jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Success</title>
</head>
<body>
    <h2>Login Successful</h2>
    <p>Welcome, you have successfully logged in.</p>
```

```
</body>
</html>
```

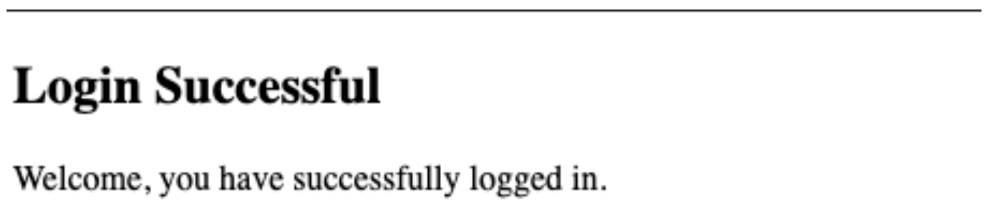
OUTPUT:



Login

Username: rabin

Password: •••••



Login Successful

Welcome, you have successfully logged in.