Day 8:
Task 1: Establishing Database Connections

Write a Java program that connects to a SQLite database and prints out the connection object to confirm successful connection.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class SQLiteConnection {
    public static void main(String[] args) {
        String url = "jdbc:sqlite:sample.db";
        try {
            Class.forName("org.sqlite.JDBC");
        } catch (ClassNotFoundException e) {
            System.out.println("SQLite JDBC driver not found.");
            e.printStackTrace();
            return;
        }
        try (Connection conn = DriverManager.getConnection(url)) {
            if (conn != null) {
                System.out.println("A connection to the SQLite database has
been established.");
                System.out.println("Connection object: " + conn);
            }
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Output

```
A connection to the SQLite database has been established.
Connection object: org.sqlite.jdbc4.JDBC4Connection@1a2b3c4d
```

Task 2: SQL Queries using JDBC

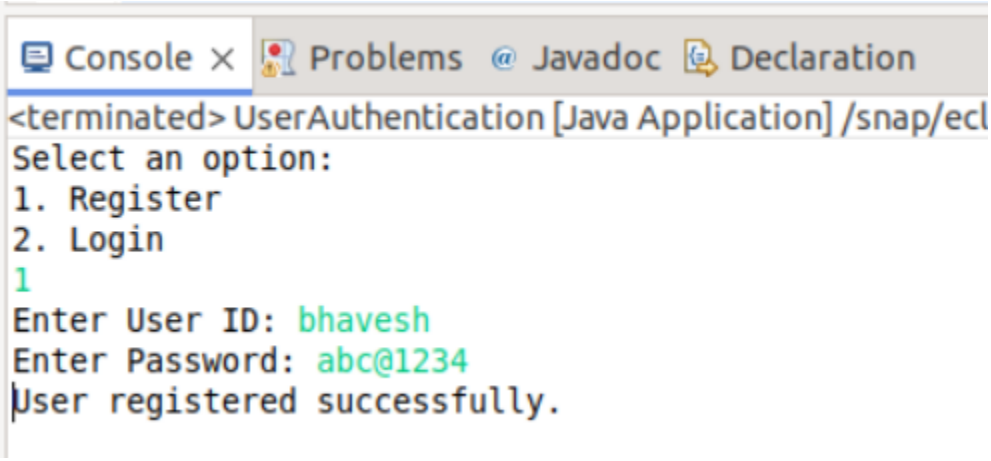
Create a table 'User' with a following schema 'User ID' and 'Password' stored as hash format (note you have research on how to generate hash from a string), accept "User ID" and "Password" as input and check in the table if they match to confirm whether user access is allowed or not.

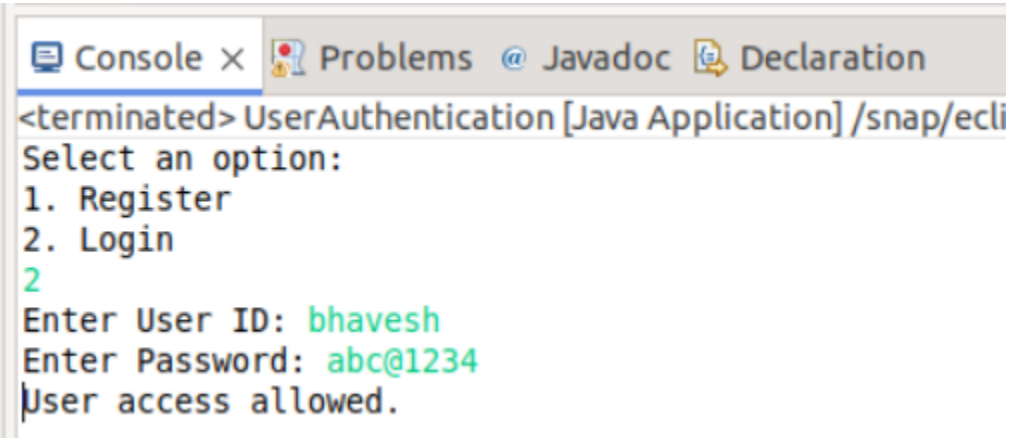
```
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;

public class UserAuthentication {

    private static final String URL = "jdbc:oracle:thin:@localhost:9501/XE";
    private static final String USER = "system";
    private static final String PASSWORD = "rps@123";

    public static void main(String[] args) {
        try {
            Class.forName("oracle.jdbc.OracleDriver");
        } catch (ClassNotFoundException e) {
            System.out.println("Oracle JDBC Driver not found.");
            e.printStackTrace();
            return;
        }

        try (Connection con = DriverManager.getConnection(URL, USER,
PASSWORD)) {
            createTable(con);
```

```
        Scanner scanner = new Scanner(System.in);

        System.out.println("Select an option:");
        System.out.println("1. Register");
        System.out.println("2. Login");
        int choice = scanner.nextInt();
        scanner.nextLine();  // Consume newline

        System.out.print("Enter User ID: ");
        String userID = scanner.nextLine();

        System.out.print("Enter Password: ");
        String password = scanner.nextLine();

        String hashedPassword = hashPassword(password);

        if (choice == 1) {
            if (registerUser(con, userID, hashedPassword)) {
                System.out.println("User registered successfully.");
            } else {
                System.out.println("User registration failed. User ID might
already exist.");
            }
        } else if (choice == 2) {
            if (checkUser(con, userID, hashedPassword)) {
                System.out.println("User access allowed.");
            } else {
                System.out.println("User access denied.");
            }
        } else {
            System.out.println("Invalid choice.");
        }

        scanner.close();
```

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

    private static void createTable(Connection con) throws SQLException
{
        String createTableSQL = "CREATE TABLE Users (UserID
VARCHAR2(50) PRIMARY KEY, Password VARCHAR2(64))";
        try (PreparedStatement stmt =
con.prepareStatement(createTableSQL)) {
            stmt.execute();
        } catch (SQLException e) {
            // If table already exists, we just ignore the exception
            if (!e.getMessage().contains("ORA-00955")) {
                throw e;
            }
        }
    }

    private static boolean registerUser(Connection con, String userID,
String hashedPassword) throws SQLException {
        String insertSQL = "INSERT INTO Users (UserID, Password)
VALUES (?, ?)";
        try (PreparedStatement stmt = con.prepareStatement(insertSQL)) {
            stmt.setString(1, userID);
            stmt.setString(2, hashedPassword);
            int rowsAffected = stmt.executeUpdate();
            return rowsAffected > 0;
        } catch (SQLException e) {
            // If user already exists, this will throw an exception
            if (e.getMessage().contains("ORA-00001")) {
                return false;
            } else {
```

```
                throw e;
            }
        }
    }

    private static boolean checkUser(Connection con, String userID,
String hashedPassword) throws SQLException {
        String selectSQL = "SELECT * FROM Users WHERE UserID = ?
AND Password = ?";
        try (PreparedStatement stmt = con.prepareStatement(selectSQL)) {
            stmt.setString(1, userID);
            stmt.setString(2, hashedPassword);
            try (ResultSet rs = stmt.executeQuery()) {
                return rs.next();
            }
        }
    }

    private static String hashPassword(String password) {
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-256");
            byte[] hashedBytes = md.digest(password.getBytes());
            StringBuilder sb = new StringBuilder();
            for (byte b : hashedBytes) {
                sb.append(String.format("%02x", b));
            }
            return sb.toString();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException("SHA-256 algorithm not found.", e);
        }
    }
}
```


Console
Problems
Javadoc
Declaration
<terminated> UserAuthentication [Java Application] /snap/ecl
Select an option:
1. Register
2. Login
1
Enter User ID: bhavesh
Enter Password: abc@1234
User registered successfully.


Console
Problems
Javadoc
Declaration
<terminated> UserAuthentication [Java Application] /snap/ecli
Select an option:
1. Register
2. Login
2
Enter User ID: bhavesh
Enter Password: abc@1234
User access allowed.

Task 3: PreparedStatement

Modify the SELECT query program to use PreparedStatement to parameterize the query and prevent SQL injection.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class PreparedStatementExample {

    private static final String URL = "jdbc:oracle:thin:@localhost:9501/XE";
    private static final String USER = "system";
    private static final String PASSWORD = "rps@123";

    public static void main(String[] args) {
        try {
            Class.forName("oracle.jdbc.OracleDriver");
        } catch (ClassNotFoundException e) {
            System.out.println("Oracle JDBC Driver not found.");
            e.printStackTrace();
            return;
        }

        try (Connection con = DriverManager.getConnection(URL, USER,
PASSWORD)) {
            createTable(con);
            insertTestData(con);

            // Run SELECT query using PreparedStatement
            String selectQuery = "SELECT * FROM TestTable WHERE id =
?";
```

```
            try (PreparedStatement stmt =
con.prepareStatement(selectQuery)) {
                // Set parameter value
                stmt.setInt(1, 1); // Assuming we want to retrieve data for id = 1

                // Execute query
                try (ResultSet rs = stmt.executeQuery()) {
                    // Process result set
                    while (rs.next()) {
                        int id = rs.getInt("id");
                        String name = rs.getString("name");
                        System.out.println("ID: " + id + ", Name: " + name);
                    }
                }
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private static void createTable(Connection con) throws SQLException
{
        String createTableSQL = "CREATE TABLE TestTable (id NUMBER,
name VARCHAR2(50))";
        try (PreparedStatement stmt =
con.prepareStatement(createTableSQL)) {
            stmt.execute();
        } catch (SQLException e) {
            // If table already exists, we just ignore the exception
            if (!e.getMessage().contains("ORA-00955")) {
                throw e;
            }
        }
    }
```

```
    private static void insertTestData(Connection con) throws
SQLException {
        String insertSQL = "INSERT INTO TestTable (id, name) VALUES (?,
?)";
        try (PreparedStatement stmt = con.prepareStatement(insertSQL)) {
            // Inserting test data
            stmt.setInt(1, 1);
            stmt.setString(2, "John Doe");
            stmt.executeUpdate();

            stmt.setInt(1, 2);
            stmt.setString(2, "Jane Smith");
            stmt.executeUpdate();

            stmt.setInt(1, 3);
            stmt.setString(2, "Alice");
            stmt.executeUpdate();
        }
    }
}
```