Day 21

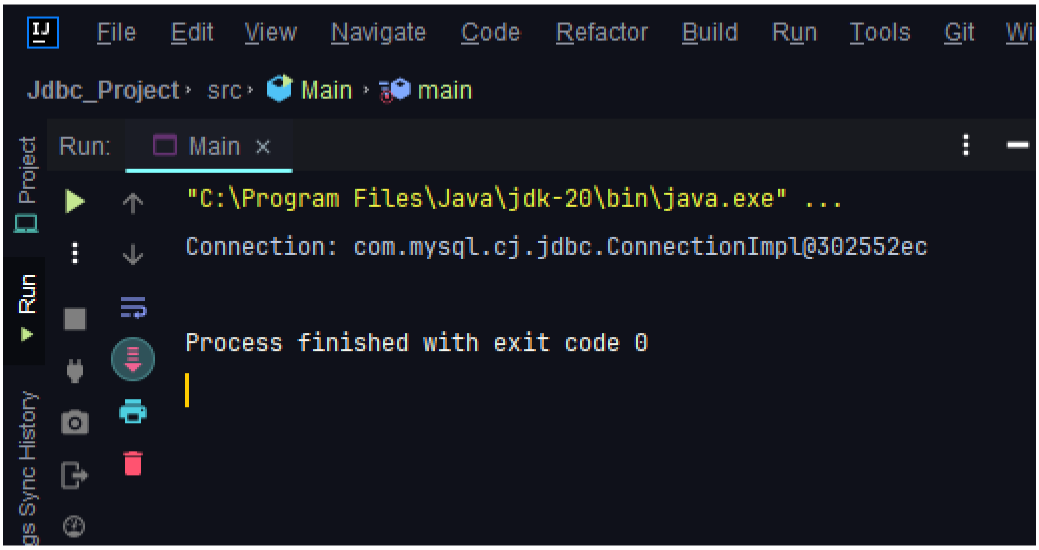
Task 1: Establishing Database Connections

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

Program:

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

public class Task1 {
    public static void main(String[] args) {
        try {
            DriverManager.registerDriver(new com.mysql.cj.jdbc.Driver());

            Connection connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb","root","5539");

            System.out.println("Connection: "+ connection);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }
    }
}
```

Output:

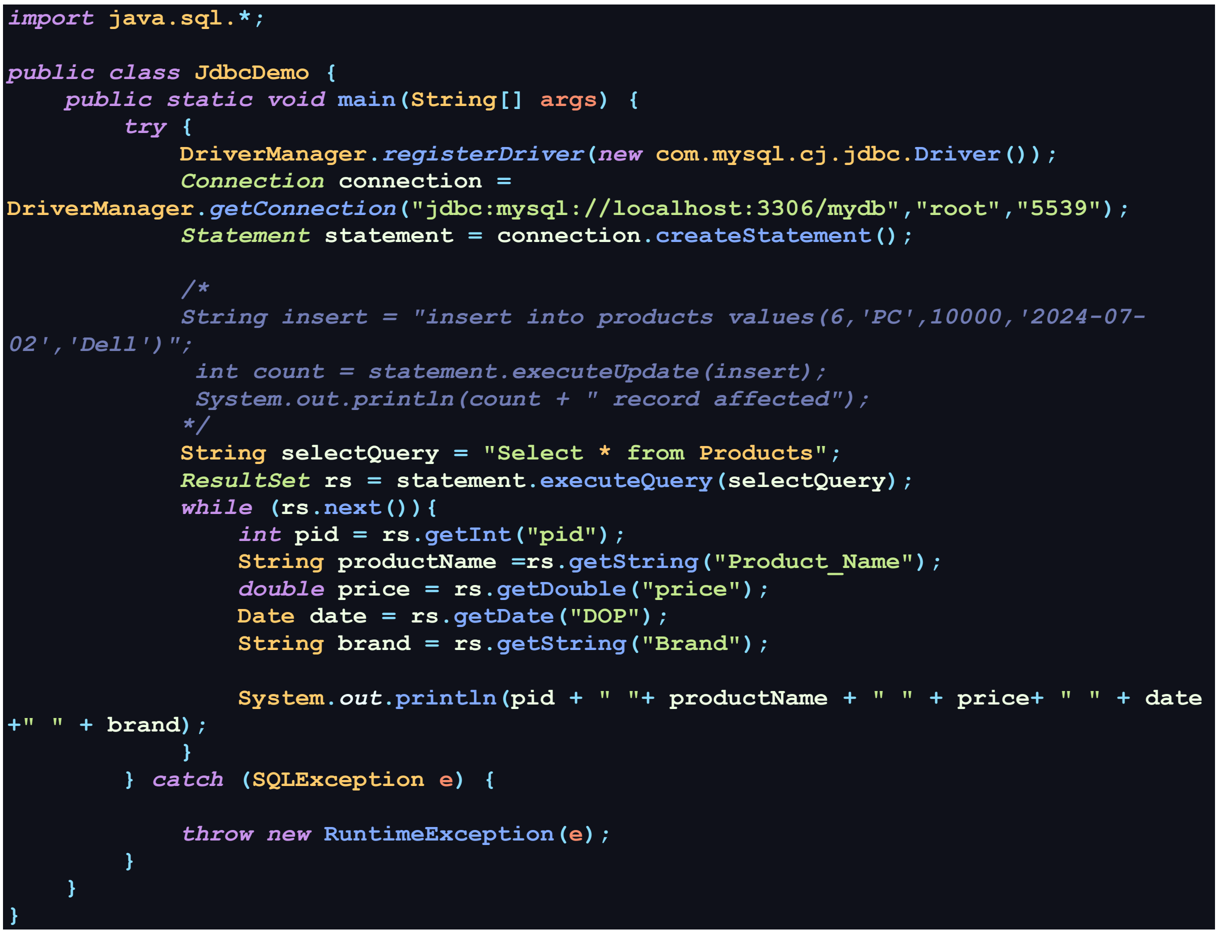


```
"C:\Program Files\Java\jdk-20\bin\java.exe" ...
Connection: com.mysql.cj.jdbc.ConnectionImpl@302552ec

Process finished with exit code 0
```

Task 2: SQL Queries using JDBC

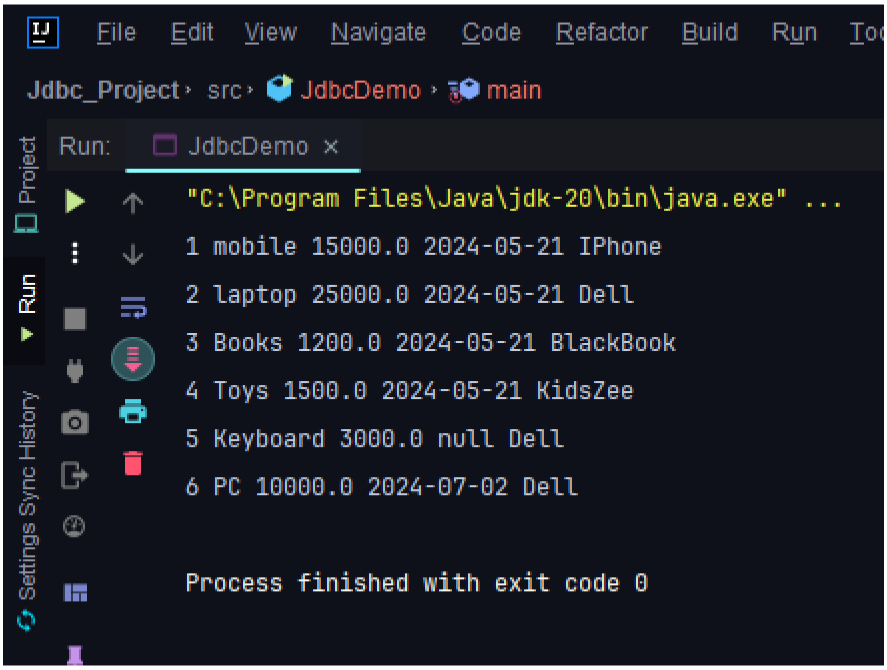
Create a table 'User' with a following schema 'User ID' and 'Password' stored as hash format, accept "User ID" and "Password" as input and check in the table if they match to confirm whether user access is allowed or not.

Program:



```
import java.sql.*;

public class JdbcDemo {
    public static void main(String[] args) {
        try {
            DriverManager.registerDriver(new com.mysql.cj.jdbc.Driver());
            Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb","root","5539");
            Statement statement = connection.createStatement();

            /*
            String insert = "insert into products values(6,'PC',10000,'2024-07-
02','Dell')";
             int count = statement.executeUpdate(insert);
             System.out.println(count + " record affected");
            */
            String selectQuery = "Select * from Products";
            ResultSet rs = statement.executeQuery(selectQuery);
            while (rs.next()){
                int pid = rs.getInt("pid");
                String productName =rs.getString("Product_Name");
                double price = rs.getDouble("price");
                Date date = rs.getDate("DOP");
                String brand = rs.getString("Brand");

                System.out.println(pid + " "+ productName + " " + price+ " " + date
+" " + brand);
            }
        } catch (SQLException e) {

            throw new RuntimeException(e);
        }
    }
}
```

Output:

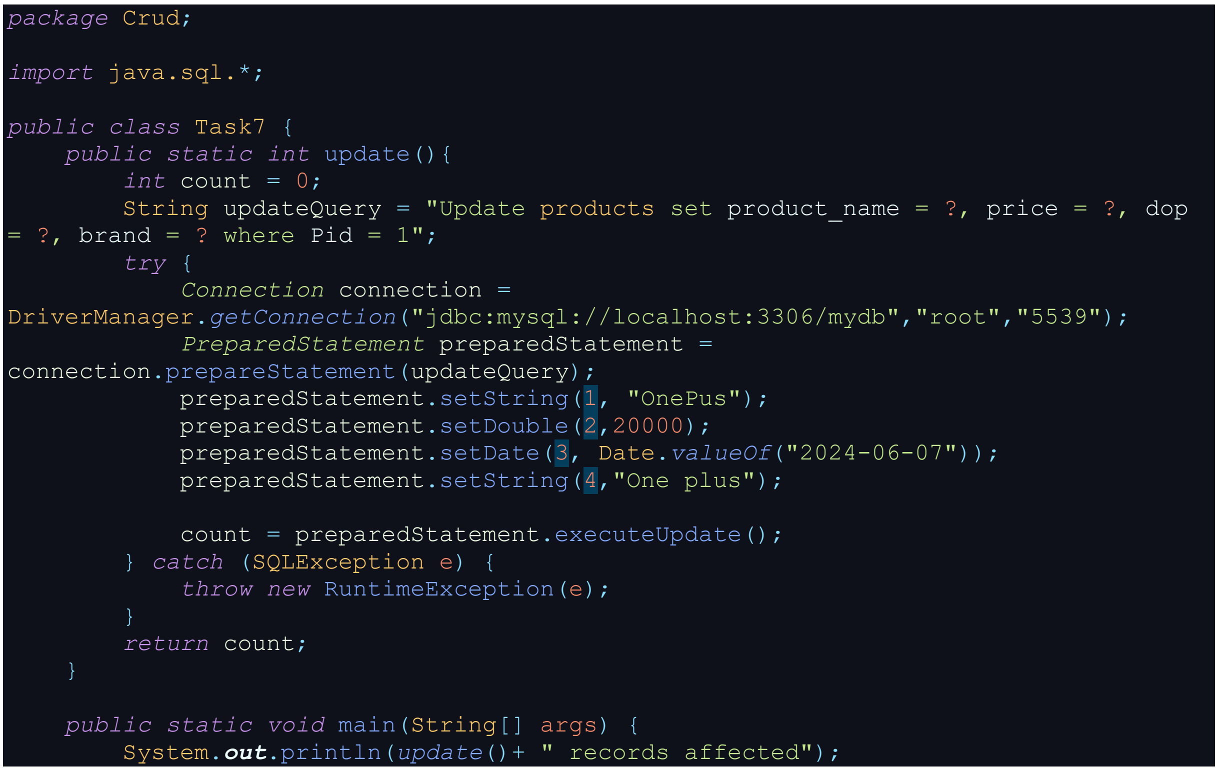


```
"C:\Program Files\Java\jdk-20\bin\java.exe" ...
1 mobile 15000.0 2024-05-21 IPhone
2 laptop 25000.0 2024-05-21 Dell
3 Books 1200.0 2024-05-21 BlackBook
4 Toys 1500.0 2024-05-21 KidsZee
5 Keyboard 3000.0 null Dell
6 PC 10000.0 2024-07-02 Dell

Process finished with exit code 0
```

Task 3: PreparedStatement

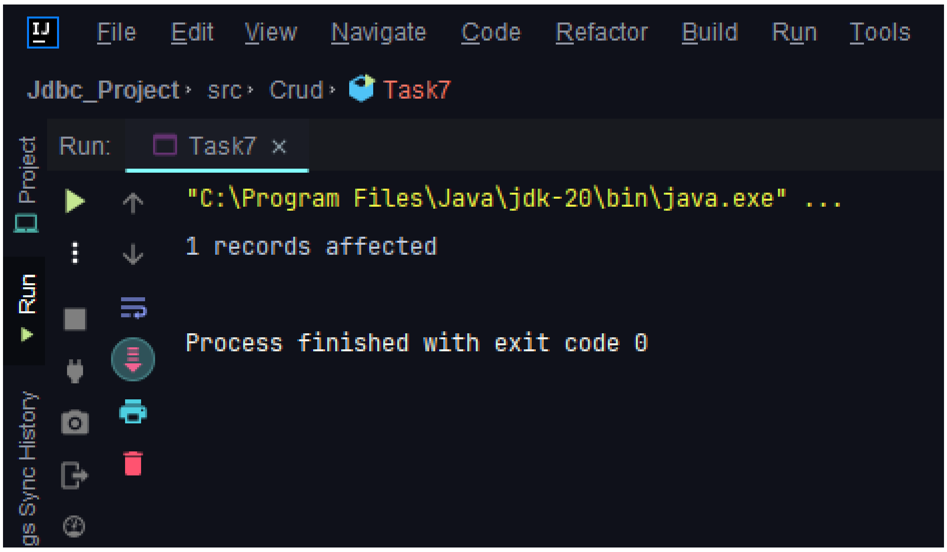
Modify the SELECT query program to use PreparedStatement to parameterize the query and prevent SQL injection.

Program:



```
package Crud;

import java.sql.*;

public class Task7 {
    public static int update(){
        int count = 0;
        String updateQuery = "Update products set product_name = ?, price = ?, dop
= ?, brand = ? where Pid = 1";
        try {
            Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb","root","5539");
            PreparedStatement preparedStatement =
connection.prepareStatement(updateQuery);
            preparedStatement.setString(1, "OnePus");
            preparedStatement.setDouble(2,20000);
            preparedStatement.setDate(3, Date.valueOf("2024-06-07"));
            preparedStatement.setString(4,"One plus");

            count = preparedStatement.executeUpdate();
        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
        return count;
    }

    public static void main(String[] args) {
        System.out.println(update()+ " records affected");
    }
}
```

Output:



```
"C:\Program Files\Java\jdk-20\bin\java.exe" ...
1 records affected

Process finished with exit code 0
```