Day 26

Assignment 1: Establishing Database Connections

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

A)

Java program that connects to a MySQL database and prints out the connection object using a DataSource object:

Java code:

```
import java.sql.Connection;

import java.sql.SQLException;

import javax.sql.DataSource;

import org.apache.commons.dbcp2.BasicDataSource;

public class DatabaseConnector {

  public static void main(String[] args) {

    // Database connection details

    String url = "jdbc:mysql://localhost:3306/mydatabase";

    String username = "username"; // Replace with your MySQL username

    String password = "password"; // Replace with your MySQL password

    // Create DataSource object

    BasicDataSource dataSource = new BasicDataSource();

    dataSource.setUrl(url);

    dataSource.setUsername(username);

    dataSource.setPassword(password);

    try {

      // Establish connection

      Connection connection = dataSource.getConnection();
```

```
            // Print out the connection object
            System.out.println("Connection object: " + connection);


            // Close the connection
            connection.close();
        } catch (SQLException e) {
            System.err.println("Failed to connect to the database");
            e.printStackTrace();
        }
    }
}
```

Explanation:

Import Statements: Import necessary classes from java.sql and javax.sql packages.

Database URL, Username, and Password: Replace the placeholders url, username, and password with your MySQL database connection details.

DataSource Configuration: Create a BasicDataSource object from Apache Commons DBCP library. Set the URL, username, and password for the data source.

Connection Establishment: Use the getConnection() method of the data source object to obtain a connection to the database.

Printing Connection Object: After successfully connecting, print out the connection object, which contains information about the established connection.

Exception Handling: Catch SQLException in case of connection failure or other SQL-related errors.

Closing Connection: Finally, close the connection using the close() method to release any resources associated with it.

This approach provides a more flexible and efficient way to manage database connections, especially in enterprise applications, as it uses a connection pool provided by the DataSource.