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)

Certainly! Below is a Java program that establishes a connection to a MySQL database using JDBC (Java Database Connectivity) and prints out the connection object to confirm successful connection:

Java code:

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

public class DatabaseConnector {
   public static void main(String[] args) {
      // JDBC URL for MySQL database
      String url = "jdbc:mysql://localhost:3306/mydatabase";
      String username = "username"; // Replace with your MySQL username
      String password = "password"; // Replace with your MySQL password

      try {
         // Load the MySQL JDBC driver
         Class.forName("com.mysql.cj.jdbc.Driver");

         // Establish the connection
         Connection connection = DriverManager.getConnection(url, username, password);

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

         // Close the connection
         connection.close();
      } catch (ClassNotFoundException e) {
         System.err.println("MySQL JDBC driver not found");
         e.printStackTrace();
      } catch (SQLException e) {
         System.err.println("Failed to connect to the database");
         e.printStackTrace();
      }
   }
}
```

Explanation:

Import Statements: The import statements import necessary classes from java.sql and java.sql.SQLException packages.

Database URL, Username, and Password: You need to replace the placeholders url, username, and password with your MySQL database connection details. The URL typically follows the format jdbc:mysql://host:port/database.

Driver Registration: The Class.forName() method is used to dynamically load the MySQL JDBC driver class com.mysql.cj.jdbc.Driver.

Connection Establishment: The DriverManager.getConnection() method establishes a connection to the MySQL database using the provided URL, username, and password.

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

Exception Handling: ClassNotFoundException is caught if the JDBC driver class is not found. SQLException is caught if there is an error while attempting to connect to the database.

Closing Connection: Finally, the close() method is called on the connection object to release any resources associated with it.

Make sure to have the MySQL JDBC driver (typically a JAR file) in your classpath for this code to work.