**哈尔滨工业大学（威海）**

Java语言实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | Java程序设计 | 课程编号 | SE33601 |
| 实验名称 | 实验2：利用抽象类及抽象方法进行重构 | | |
| 实验类型 | 验证与编程 | 实验学时 | 2学时 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 班号 | 1911103 | 学号 | 2191110315 | 姓名 | 刘梓铭 |
| 开始 | 2020.10.16 | 截止 | 2020.10.21 | 成绩 |  |
| 批语 |  | | | | |

目录

[1 实验目的、内容和要求 1](#_Toc53755575)

[1.1 实验目的 1](#_Toc53755576)

[1.2 实验内容 1](#_Toc53755577)

[2 实验内容 1](#_Toc53755578)

[2.1 需求描述 1](#_Toc53755579)

[2.2 重构2：子类中相似方法的统一处理 1](#_Toc53755580)

[2.3 重构3：防止流程篡改 4](#_Toc53755581)

[2.4 判断是否满足OCP 5](#_Toc53755582)

# 1 实验目的、内容和要求

## 1.1 实验目的

1. 掌握抽象类抽象方法的应用
2. 掌握多态的使用

## 1.2 实验内容

1. 利用StarUML对抽象类进行设计，并实现代码；
2. 利用StarUML绘制扩展的类图，分析是否符合OCP，并实现代码；

# 2 实验内容

## 2.1 需求描述

为了满足不同用户数据量的要求，某软件产品拟采用多种数据库管理系统对数据进行管理，目前采用MySQL，Oracle两种数据库。

数据库的连接操作严格按照如下顺序进行：

* MySQL的数据库连接操作
  1. 注册MySQL驱动（Registration MySQL Driver）
  2. 建立MySQL连接（Establish MySQL Connection）
  3. 创建执行语句（Create Execute Statements）
  4. 执行语句（Execute Statement）
  5. 处理结果（Processing Results）
  6. 释放资源 （Free Resource）
* Oracle的数据库连接操作

1. 注册Oracle驱动（Registration Oracle Driver）
2. 建立Oracle连接（Establish Oracle Connection）
3. 创建执行语句（Create Execute Statements）
4. 执行语句（Execute Statement）
5. 处理结果（Processing Results）
6. 释放资源 （Free Resource）

要求：要求在“实验1：类的定义及继承的应用”的基础上，完成本次实验内容。

## 2.2 重构2：子类中相似方法的统一处理

数据库连接操作中的第1）步和第2）步的代码是不同的，它们需要加载不同的驱动程序，连接的URL和端口也不同。

* **实验内容1：统一步骤1)和2)的命名以后，重构类的设计：首先，将流程控制方法放到超类，以便子类继承；其次，根据《咖啡冲泡机流程控制案例》，对子类中都有，但表现行为不同的方法在超类中设计抽象方法。**

实验内容1的类图设计

|  |
| --- |
|  |

**对以上设计的类图进行实现，各类代码如下：**

超类的代码实现

|  |
| --- |
| public abstract class SQL {  public void CreateExecuteStatements(){  System.out.println("Create Execute Statements");  }  public void ExecuteStatement(){  System.out.println("Execute Statement");  }  public void ProcessingResults(){  System.out.println("Processing Results");  }  public void FreeResource(){  System.out.println("Free Resource");  }  public void useSQL(){  RegistrationDriver();  EstablishConnection();  CreateExecuteStatements();  ExecuteStatement();  ProcessingResults();  FreeResource();  }  public abstract void RegistrationDriver()  public abstract void EstablishConnection()  } |

MySQL的数据库连接操作类代码：

|  |
| --- |
| public class MySQL extends SQL{  public void RegistrationDriver(){  System.out.println("Registration MySQL Driver");  }  public void EstablishConnection() {  System.out.println("Establish MySQL Connection"); }  } |

Oracle的数据库连接操作类代码：

|  |
| --- |
| public class Oracle extends SQL{  public void RegistrationDriver(){  System.out.println("Registration Oracle Driver");  }  public void EstablishConnection(){  System.out.println("Establish Oracle Connection");  }  } |

测试类的代码：

|  |
| --- |
| public class Test1 {  public static void main(String args[]){  System.out.println("use MySQL:");  MySQL sql1 = new MySQL();  sql1.useSQL();  System.out.println("================");  System.out.println("use Oracle:");  Oracle sql2 = new Oracle();  sql2.useSQL();  }  } |

运行结果截图：

|  |
| --- |
|  |

## 2.3 重构3：防止流程篡改

* **实验内容2：根据《咖啡冲泡机流程控制案例》，防止流程控制方法被子类篡改。请修改超类代码，并测试，证明可以实现超类中的流程控制方法不能被子类覆盖。**

超类的代码实现：

|  |
| --- |
| public abstract class SQL {  public void CreateExecuteStatements(){  System.out.println("Create Execute Statements");  }  public void ExecuteStatement(){  System.out.println("Execute Statement");  }  public void ProcessingResults(){  System.out.println("Processing Results");  }  public void FreeResource(){  System.out.println("Free Resource");  }  public final void useSQL(){  RegistrationDriver();  EstablishConnection();  CreateExecuteStatements();  ExecuteStatement();  ProcessingResults();  FreeResource();  }  public abstract void RegistrationDriver()  public abstract void EstablishConnection()  } |

测试类的代码：

|  |
| --- |
| public class Test1 {  public static void main(String args[]){  System.out.println("use MySQL:");  MySQL sql1 = new MySQL();  sql1.useSQL();  System.out.println("================");  System.out.println("use Oracle:");  Oracle sql2 = new Oracle();  sql2.useSQL();  }  } |

运行结果截图：

|  |
| --- |
|  |

## 2.4 重构后的系统扩展

* **实验内容3：请添加一个新的数据库Access的连接类，绘制类图，编写Access类，在测试类中构建Access对象，并测试该对象是否能够满足数据库连接的6个步骤。**

绘制新加入Access数据库后全体类的类图：

|  |
| --- |
|  |

Access的数据库连接操作类代码：

|  |
| --- |
| public class Access extends SQL{  public void RegistrationDriver(){  System.out.println("Registration Access Driver");  }  public void EstablishConnection() {  System.out.println("Establish Access Connection"); }  } |

测试类的代码：

|  |
| --- |
| public class Test1 {  public static void main(String[] args){  System.out.println("use MySQL:");  MySQL sql1 = new MySQL();  sql1.useSQL();  System.out.println("================");  System.out.println("use Oracle:");  Oracle sql2 = new Oracle();  sql2.useSQL();  System.out.println("================");  System.out.println("use Access:");  Access sql3 = new Access();  sql3.useSQL();  }  } |

运行结果截图：

|  |
| --- |
|  |

“实验1 类的定义及继承的应用”和“实验2 利用抽象类及抽象方法进行重构”两次实验构成了一个完整的设计优化过程，请对两次实验的设计、优化过程进行总结：

|  |
| --- |
| 在实验1中，我们利用类的继承，将MySQL和Oracle中共有的方法写到父类SQL中，实现了代码的复用。  在实验2中，我们利用多态的思想，将实验1的父类改为抽象类，并在其中加入抽象函数，通过在子类中根据不同数据库的不同需求覆写父类中的抽象函数，使子类中操作原理一致但又有细微差别的函数得以必须实现（父类中的抽象函数必须被子类覆写）。  此外，我们又尝试将父类中“useSQL”函数声明为“final”，又故意在子类中写了一个改变了的“useSQL”函数，结果报错“final声明的函数不能再被覆写”，这样就避免了子类擅自篡改父类中不希望被改变的方法。  最后，我们又设计了一个“Access”数据库验证程序的开闭性，经检验程序开闭性良好，可复用性强。  通过这次实验，我加深了对面向对象基本概念继承，多态的理解，并熟悉了一个好的程序不断优化迭代的过程。 |