**实 验 报 告**

|  |  |
| --- | --- |
| **课程名称：** | **Java语言程序设计** |
| **学生姓名：** | **张逸松** |
| **学生学号：** | **201836664088** |
| **学生专业：** | **软件工程** |
| **授课教师：** | **汤峰** |
| **开课学期：** | **2019-2020-01** |

**华南理工大学软件学院**

**2019年10月**

**目 录**

[实验一 熟悉Java运行环境 1](#_Toc449456884)

[实验二 面向对象的编程技术 2](#_Toc449456885)

[实验三 接口实现与异常处理 4](#_Toc449456886)

[实验四 GUI程序设计与I/O文件处理 6](#_Toc449456887)

# 熟悉Java 运行环境

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7后座 楼 | 138房 | | |
| **实验日期与时间：** | 2019年11月17日，1-4节 | | | |
| **实验教师：** | 汤峰 | | **评 分：** |  |
| **批改意见：** |  | | | |

**实验1-熟悉Java运行环境**

1. **实验目的**

1.安装与配置JDK环境变量；

2.掌握Java程序的编辑、编译和运行过程，熟悉Javac和Java命令

3.掌握Java的基本语法；

4.使用类来封装对象的属性和方法；熟悉使用package和import语句。

1. **实验内容**

1、安装jdk-8u20-windows-i586.exe，并设置环境变量（若安装目录为自定义，需做相应修改）

1）JAVA\_HOME 变量值为“C:\Program Files\Java\jdk1.8.0\_20”；

ClassPath 变量值为“.;%JAVA\_HOME%\lib\dt.jar;%JAVA\_HOME%\lib\tools.jar;”

Path 变量值添加 “%JAVA\_HOME%\bin；”

2）运行cmd，输入 java –version 查看当前jdk版本

2、解压lab1\_test\_banking Project\_step1\_4.zip，分四个步骤编写Java程序，并在命令行方式下测试是否正确，将输出结果截图放入实验报告。

1）Java源代码可在任何文本编辑器中输入，建议使用Editplus（可自行安装）或记事本，暂时不使用IDE平台；

2）在包含主类的文件中，文件名应与主类的名称相同，并注意有大小写之分；

3）javac 编译器：将.java类和接口定义编译成Bytecode类文件。

熟悉javac选项 -d 目录、-sourcepath 源路径、-classpath 类路径（或-cp）

4）java解释器： 启动JVM将Bytecode文件转换为本地机器码文件。 熟悉选项 -classpath 类路径（或-cp）

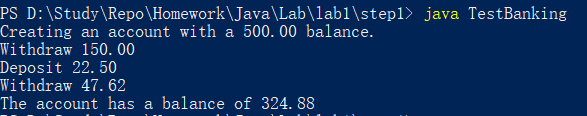
5）Java提供一种有效管理类的机制package，包是类、接口或子包的集合，package 语句指明该源文件定义的类所在的包，import语句引入所需要的类。

3、运行以上程序后，思考问题

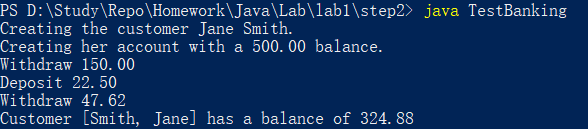
1. 假设在TestBanking类中引用了Account类和Customer类，在编辑了三个类的源文件后，仅编译TestBanking.java 源文件同时也生成了其他类的类文件，为什么？进一步归纳，当编译器需要某个类的信息时，如何搜索定义该类的源文件或类文件？
2. 假设已定义了Banking.Customer类和Banking.Account类，并且Customer类中引用了Account类，若直接在该类所在目录（/banking子目录下）运行 Javac customer.java 会出错，请分析出错原因。
3. 描述 -classpath 类路径的作用，思考JDK的编译器和虚拟机是如何定位到所需的类文件的？归纳编译器搜索类的顺序以及JVM搜索类的顺序。
4. **实验结果**

**（一）实验题（实验步骤+实验结果截图）**

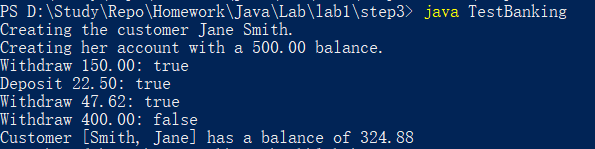
Step 1:



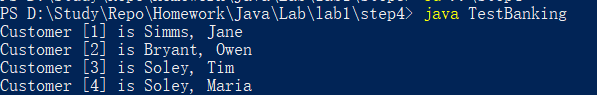
Step 2:



Step 3:



Step 4:



**（二）思考题**

1)

TestBanking类通过banking包调用了Account类和Customer类，所以在编译时同时也会联编这两个类。当编译时，编译器会先查找自举类和扩展类，然后再检索用户类。自举类通过-bootclasspath，扩展类通过-extdirs,用户类通过-classpath和-sourcepath定义。在搜索用户类时，会先调用当前路径的缺省值，若有classpath，则会覆盖。

2)

在 /banking下键入javac Customer.java后，编译器会先编译Customer.java文件，在文件里遇到Account类的语句时，会在当前目录和子目录下查找Account.class文件，但此时Account.java尚未编译，还没有生成class文件。

3)

-classpath定义了用户类的位置。

当编译时，编译器会先查找自举类和扩展类，然后再检索用户类。自举类通过-bootclasspath，扩展类通过-extdirs,用户类通过-classpath和-sourcepath定义。在搜索用户类时，会先调用当前路径的缺省值，若有classpath，则会覆盖。

# 面向对象的编程技术

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7后座 楼 | 138房 | | |
| **实验日期与时间：** | 2019年11月17日，5-8节 | | | |
| **实验教师：** | 汤峰 | | **评 分：** |  |
| **批改意见：** |  | | | |

### 实验二 面向对象的编程技术

#### 一、实验目的

1. 掌握面向对象的继承与多态以及类型转换；
2. 掌握ArrayList类创建动态数组；
3. 掌握Singleton模式的使用

#### 二、实验内容

1. 掌握查阅JDK帮助文档，获得系统预定义的类和方法。
2. chm文件见实验目录
3. 也可在线查阅：<http://docs.oracle.com/javase/8/docs/api/index.html>
4. 解压lab2.zip，分三个步骤编写Java程序，并在命令行方式下测试是否正确。
   1. **Step 5：**

创建Account类的两个子类SavingsAccount 和CheckingAccount类，掌握方法的覆盖，以及使用super()调用父类的构造函数。

* 1. **Step 6\_pre1:**

修改Customer类，支持一个客户拥有多个账户，使用java.util.ArrayList类创建客户拥有的不同类型的账户集合。

1. 当直接使用ArrayList类时编译出现警告：“Customer类使用了未经检查或不安全的操作”，此为JDK1.5以后版本的泛型警告（泛型类似C++的模板），可忽略不计，或是使用ArrayList的泛型定义ArrayList<Account>。
2. java.text.NumberFormat表示数值的格式化类，可以按照本地风格习惯进行数值的显示。
   1. **Step 6\_pre2:**

修改Bank类为单例类，实现Singleton单例设计模式。

1）设计模式用于复用好的设计方法，模式的定义包括模式名称、模式应用的问题背景及解决方案，常用的设计模式是23种GOF模式，如Singleton模式、Adaptor模式、Abstract Factory模式等。

2）单例模式：用于确保一个类仅有一个实例，并提供一个访问它的全局访问点。

3）单例类（ServicesFctory类）结构示例：

3. 思考并回答问题

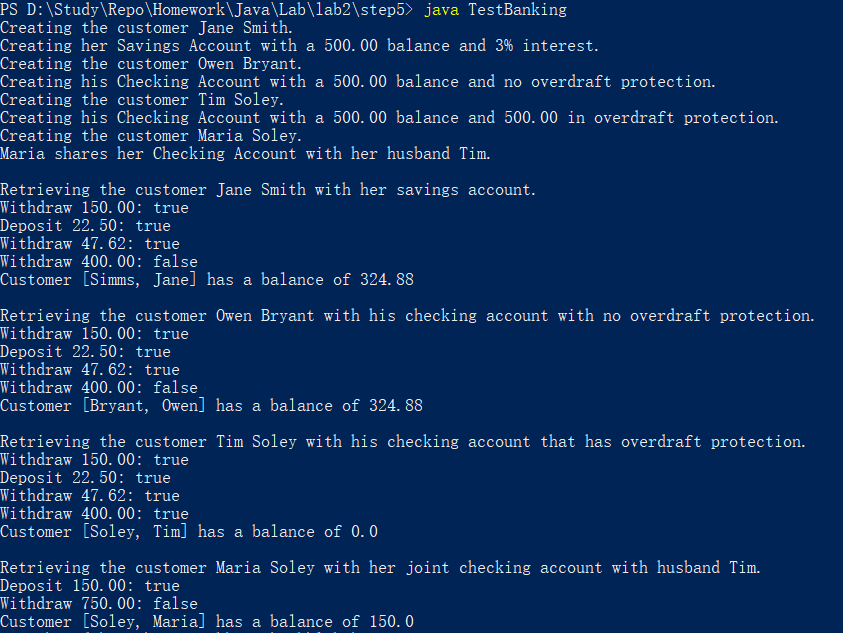
1）思考如何对ArrayList定义的Account集合排序，按照账户的余额由小到大排列？请描述实现思想或关键语句。

2）归纳单例模式的设计原理，思考为什么属性和方法为static ？构造函数为private？

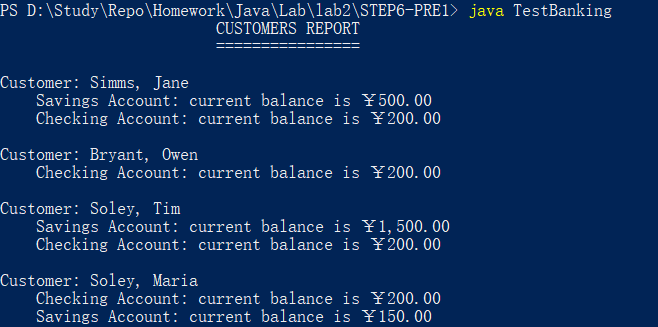
**三、实验结果**

**（一）实验题（实验步骤+实验结果截图）**

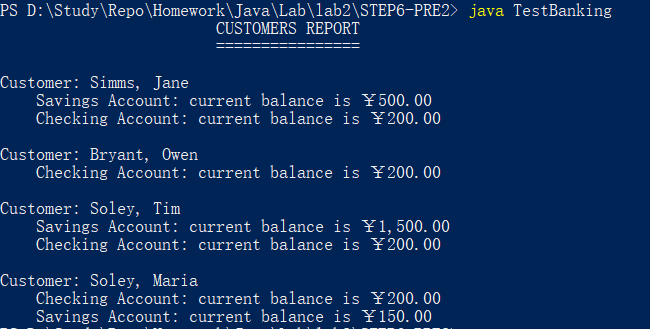
Step 5:



Step 6 Pre 1:



Step 6 Pre 2:



**（二）思考题**

1)考虑到只有新增（即插入）操作，可以考虑使用插入排序，时空间都比较优秀。

2)单例设计模型用于确保一个类仅有一个实例，并提供一个访问它的全局访问点。为了达到这个目的，通过对构造函数限定(private)，使得其他程序无法生成该类的对象。静态属性或者说类属性是为了能让静态方法调用，方法静态则是为了防止被其他程序调用。

# 接口实现与异常处理

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7后座 楼 | 138房 | | |
| **实验日期与时间：** | 2019年11月24日，1-4节 | | | |
| **实验教师：** | 汤峰 | | **评 分：** |  |
| **批改意见：** |  | | | |

#### 一、实验目的

1. 熟悉在Eclipse IDE集成开发环境中开发与调试Java项目；
2. 掌握接口的定义与接口的实现，区分接口与抽象类；
3. 掌握异常的概念及异常处理的机制；
4. 掌握try-catch-finally异常处理语句的使用；
5. 熟悉用户自定义异常及处理用户自定义异常的方法。

#### 实验内容

1. **安装Eclipse IDE集成开发环境，建立Java项目，开发与调试Java程序**
2. 从http://www.eclipse.org/downloads下载 "Eclipse IDE for Java Developers"，解压到所需目录，不需其他的安装程序即可使用，直接运行eclipse.exe启动Eclipse；（注：实验室中直接运行Myeclipse，Myeclipse为在Eclipse基础上扩展的插件集）

2）新建Java项目，名称为“BankProject”，新建后在Package Explorer 可见以项目名称命名的文件夹及src子目录；

3） 选定src，执行Import命令导入上次实验Java源文件所在的工作目录；

4）进一步按以下要求修改程序，熟悉Eclipse的编辑器和运行调试过程。

1. **解压lab3.zip，分两个步骤编写Java程序，并在Eclipse平台下测试是否正确。**
2. Step6： 在系统中添加新的功能，支持银行客户按姓名排序和搜索；

首先修改Customer类实现Comparable 接口；然后修改Bank类，增加sortCustomers 和searchCustomers 方法；最后修改CustomerReport 类，调用sortCustomers进行输出。

1. Step 7： 添加发生账户透支错误时的异常处理功能；

首先自定义新的异常类OverdraftException，然后修改Account类和CheckingAccount 类声明和抛出OverdraftException异常。

1. **新建一个新的Java项目，完成以下练习并回答问题：**

创建接口Speakable和Runner，然后创建两个类Dog和Person实现该接口。

**程序代码：**

|  |
| --- |
| interface Speakable{  public void speak();  }  interface Runner{  public void run();  }  class Dog implements Speakable,Runner{  public void speak(){  System.out.println("狗的声音:汪、汪！");  }  public void run(){  System.out.println("狗用四肢跑步");  }  }  class Person implements Speakable,Runner{  public void speak(){  System.out.println("人们见面时经常说:您好！");  }  public void run(){  System.out.println("人用两腿跑步");  }  }  public class TestInterface{  public static void main(String[] args) {  Dog d=new Dog();  d.speak(); d.run();  Person p=new Person();  p.speak(); p.run();  }  } |

运行上面的程序，思考如下的问题：

1. 该程序编译后生成几个class文件？
2. 创建一个类Bird（鸟），给出其声音特征，并在主类中创建一个Bird类的实例，输出其特征。
3. 如何编写抽象类代替程序中的接口，实现程序同样的功能。试比较它们的不同。

**三、实验结果**

**（一）实验题（实验步骤+实验结果截图）**

**（二）思考题**

# 接口实现与异常处理

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **地 点：** | B7后座 楼 | 138房 | | |
| **实验日期与时间：** | 2019年11月24日，4-8节 | | | |
| **实验教师：** | 汤峰 | | **评 分：** |  |
| **批改意见：** |  | | | |

#### 一、实验目的

1. 理解Java系统图形用户界面的工作原理和界面设计步骤；
2. 掌握图形界面容器（Frame、Panel）的使用；
3. 掌握主要布局管理器（FlowLayout、GridLayout、BorderLayout）的使用
4. 掌握图形用户界面的常用组件（JLabel、JTextField、JButton、JList）的使用方法；
5. 理解图形用户界面的事件响应机制，掌握ActionEvent事件编程方法；
6. 理解java IO的基本原理，掌握RandomAccessFile类使用方法。

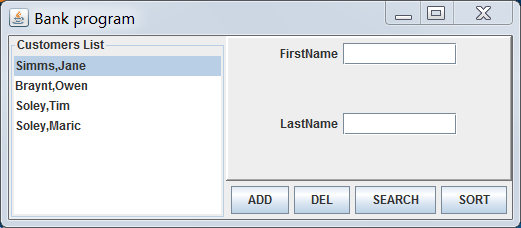
#### 二、实验内容

1. **解压lab4.zip，分两个步骤编写Java程序，并在Eclipse平台下测试。**
2. 创建一个java工程，导入AddressBook.java 和FixedLengthStringIO.java文件，编译运行，观察源代码与运行结果；学习 GUI界面程序结构、常用界面组件的使用、布局管理器的使用、按钮时间以及RandomAccessFile类的使用。其中FixedLengthStringIO是自定义的用于在记录文件中存取固定长度字符串的IO辅助类，该类可复用。
3. 创建一个Java工程，导入之前STEP7所有文件，按以下方式修改代码。

**STEP 8:** 创建简单的GUI界面管理银行客户，包括增加客户、删除客户、搜索客户和对客户列表排序，客户的数据持久保存在当前目录的customers.dat文件中。注意，为简化程序编写，暂不考虑客户的账户信息的显示与维护。

**界面与功能要求如下：**

（包括Frame容器、Panel容器及JList、JLabel、JTextField、JButton组件，具体方法可查看）



ADD： 增加客户，根据文本框中输入信息创建新的客户，并保存至文件；

DEL：将列表中选定的客户删除，并从文件中删除；

Search：在文本框中输入客户信息，进行查询，查询结果在列表中显示；

Sort： 对列表中所有的客户进行排序。

**推荐完成步骤（也可自行决定源文件结构）：**

1. 新建banking.DAO 包，在该包中导入FixedLengthStringIO.java、并且新建CustomerDAO.java。CustomerDAO类封装了Customer文件访问的所有方法；
2. 修改Bank类，调用CustomerDAO类的方法，使其支持从文件读取客户或保存客户至文件。
3. 新建banking.UI 包，新建MainWindow.java, 调用Bank类的方法，处理界面组件的显示及相应按钮事件。

**三、实验结果**

**（一）实验题（实验步骤+实验结果截图）**

**（二）思考题**