**实验四**

2023 – 2024 学年第 1 学期 任课老师：余兆钗

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **课程名称** | **面向对象程序设计（Java）** | **班级** | **2022软件工程1班** | **座号** |  | **姓名** | **苏国煜** |
| **实验名称** | 类的继承 | | | **实验时间** | 2023-9-25 、27 | | |
| **实验目的** | | | | | | | |
| 学会“抽象类实现不同图形的打印程序设计”的实现思路。  独立完成“抽象类实现不同图形的打印程序设计”的源代码编写、编译及运行。  掌握抽象类的应用。  掌握方法的重写。  掌握继承的概念及其应用。 | | | | | | | |
| **实验环境** | | | | | | | |
| Win7或win10、JDK1.8 | | | | | | | |
| **实验内容与要求** | | | | | | | |
| 【案例1】打印不同的图形  **【案例介绍】**   1. **案例描述**   本案例要求编写一个程序，可以根据用户要求在控制台打印出不同的图形。例如，用户自定义半径的圆形和用户自定义边长的正方形。   1. **运行结果**     **【案例目标】**   * 学会“抽象类实现不同图形的打印程序设计”的实现思路。 * 独立完成“抽象类实现不同图形的打印程序设计”的源代码编写、编译及运行。 * 掌握抽象类的应用。 * 掌握方法的重写。 * 掌握继承的概念及其应用。   **【案例分析】**  （1）创建父类MyPrint类，包含show()方法，用于输出图形的形状。  （2）创建子类MyPrintSquare类，重写show ()方法，用“\*”打印出边长为5的正方形。  （3）创建子类MyPrintCircle类，重写show ()方法, 用“\*”打印出半径为5的圆。  （4）创建测试类，主函数中创建MyPrintSquare、MyPrintCircle的对象，分别调用show方法，输出边长为5的正方形，输出半径为5的圆，检查输出结果。    以下是部分参考源代码  public void show() {//打印圆形  for (int y = 0; y <= 2 \* 5; y += 2) {  int x = (int)Math.round(5 - Math.sqrt(2 \* 5 \* y - y \* y));  int len = 2 \* (5 - x);  for (int i = 0; i <= x; i++) {  System.out.print(' ');  }  System.out.print('\*');  for (int j = 0; j <= len; j++) {  System.out.print(' ');  }  System.out.println('\*');  }  for(int i=0;i<5;++i){//打印正方形  for(int j=0;j<5;++j){  if(j==0 || j==4)  System.out.print('\*');  else if(i==0 || i==4)  System.out.print('\*');  else System.out.print(' ');  }  System.out.println();  }  **public class MyPointTest {**  **public static void main(String[] args){**  **MyPrintSquare mp1 = new MyPrintSquare();**  **MyPrintCircle mp2 = new MyPrintCircle();**  **mp1.show();**  **mp2.show();**  **}**  **}**  **【案例4-2】多彩的声音**  **【案例介绍】**   1. **案例描述**   设计和实现一个Soundable发声接口，该接口具有发声功能，同时还能调节声音大小（提升音量、降低音量、关掉声音）。  Soundable接口的这些功能将由有3种声音设备来实现，他们分别是收音机Radio（播放新闻）、随身听Walkman（播放音乐）、手机MobilePhone（发出来电铃声）。最后还需设计一个应用程序类来使用这些实现Soundable接口的声音设备。程序运行时，先询问用户想哪个设备，用户选择设备后，程序按照该设备的工作方式打印发出的发音，接着根据用户的选择进行调节音量的操作。  本案例要求使用接口来实现。   1. **运行结果**     **【案例目标】**   * 学会分析“发声接口程序设计”实现的逻辑思路。 * 能够独立完成“发声接口程序设计”的源代码编写、编译及运行。 * 掌握接口的实现方式。   **【案例分析】**  （1）通过任务的描述可知，此程序包含了一个发声接口Soundable，Soundable接口的有三个声音设备实现类实现了Soundable接口，分别是收音机类Radio，随身听类Walkman和手机类Mobilephon。  （2）Radio、Walkman和MobilePhon这三个声音设备实现类需要分别实现Soundable接口接口的相关功能。  （3）然后，还需设计一个应用程序类SampleDisplay来使用这些实现Soundable接口的声音设备：首先播放该设备发出的声音，然后根据用户的选择，进行音量调节  （4）最后编写测试类，在main()方法中，编写程序，使用户可以自主选择要使用的设备；创建使用设备的对象，根据用户选择的声音设备，调用相关方法，模拟使用声音设备对应的功能。  **【案例实现】**    测试类源代码  import java.util.Scanner;  public class TestDemo {  public static void main(String[] args) {  Scanner in = new Scanner(System.in);  System.out.println("你想听什么？请输入：");  System.out.println("0-收音机 1-随身听 2-手机");  int choice;  choice = in.nextInt();  SampleDisplay sampledisplay = new SampleDisplay();  if (choice == 0)  sampledisplay.display(new Radio());  else if(choice == 1)  sampledisplay.display(new Walkman());  else if(choice == 2)  sampledisplay.display(new MobilePhone());  else  System.out.println("您的输入有误，请重新输入！");  in.close();  }  }  【案例4-3】饲养员喂养动物  **【案例目标】**   1. **案例描述**   饲养员在给动物喂食时，给不同的动物喂不同的食物，而且在每次喂食时，动物都会发出欢快的叫声。例如，给小狗喂骨头，小狗会汪汪叫；给小猫喂食，小猫会喵喵叫。  本案例要求编写一个程序模拟饲养员喂食动物的过程，案例要求如下：   1. 饲养员给小狗喂骨头，小狗汪汪叫。 2. 饲养员给小猫喂小鱼，小猫喵喵叫。 3. **运行结果**     **【案例目标】**   * 学会分析“饲养员喂养动物”程序设计的实现思路。 * 根据思路独立完成“饲养员喂养动物”程序设计的源代码编写、编译及运行。 * 学会灵活运用接口解决多继承问题。   **【案例分析】**  在这个动物园里，涉及的对象有饲养员，各种不同动物以及各种不同的食物。这样很容易抽象出3个类Feeder、Animal和Food。假设只考虑猫类和狗类动物，则由Animal类派生出Cat类、Dog类、同样由Food类可以进一步派生出其子类Bone、Fish。因为他们之间存在这明显的is-a关系。  同样的，鱼是一种食物，但实际上，鱼也是一种动物，Cat类和Dog类继承了Animal的很多属性和方法，如果将Animal定义为接口，Animal中是不能定义成员变量和成员方法的，Food类中虽然也有变量但是数量比Animal少，所以我们考虑将Food定义为接口，此时可以说“鱼是一种动物，同时也是一种食物”。  **【案例实现】**  IMG_256  测试类源代码：  public class Test {  public static void main(String[] args) {  Feeder feeder = new Feeder("小华");  feeder.speak();  Dog dog = new Dog("小狗");  dog.shout();  Food food = new Bone();  feeder.feed(dog, food);  Cat cat = new Cat("小猫");  cat.shout();  food = new Fish("黄花鱼");  feeder.feed(cat, food);  }  }  【案例4】模拟物流快递系统程序设计  **【案例介绍】**   1. **案例描述**   网购已成为人们生活的重要组成部分，当人们在购物网站中下订单后，订单中的货物就会在经过一系列的流程后，送到客户的手中。而在送货期间，物流管理人员可以在系统中查看所有物品的物流信息。编写一个模拟物流快递系统的程序，模拟后台系统处理货物的过程。   1. **运行结果**   G3Y2AM$~76%159{43008S15  **【案例目标】**   * 学会分析“模拟物流快递系统程序设计”任务实现的逻辑思路。 * 能够独立完成“模拟物流快递系统程序设计”的源代码编写、编译及运行。 * 掌握面向对象封装、继承和多态的概念和使用。 * 掌握抽象类和接口的使用。 * 掌握匿名内部类的使用 * 掌握lambda表达式的使用   **【案例分析】**  （1）运输货物首先需要有交通工具，所以需要定义一个交通工具类Transportation。由于交通工具可能有很多，所以可以将该交通工具类定义成一个抽象类，类中需要包含该交通工具的编号，型号以及运货负责人等属性，还需要定义一个抽象的运输方法。  （2）交通工具可能有很多种，这里可以定义一个专用保温冷藏运输车类ZTransportation，该类需要继承交通工具类。  （3）有了运输的交通工具后，就可以开始运送货物了。货物在运输前，运输时和运输后，都需要检查和记录，并且每一个快递都有快递单号，这时可以定义一个快递任务类包含快递单号和货物重量的属性，以及送前、发送货物途中（send）和送后的方法。  （4）在货物运输过程中（send方法），需要对运输车辆定位，以便随时跟踪货物的位置信息。定位功能可以使用GPS，而考虑到能够实现定位功能的设备可能有很多（如手机、专用定位仪器等），这时可以定义一个包含定位功能的GPS接口。  （5）测试类在进行发货的时候，创建一个匿名类对象交给发货方法。  （6）测试类在进行发货的时候，创建一个lambda表达式对象交给发货方法。  （7）运行测试类，查看结果。  部分参考源代码  IMG_256  /\*  \* 快递任务类  \*/  **public** **class** SendTask {  **private** String number; // 快递单号  **private** **double** goodsWeight; // 货物重量  **public** SendTask() {  **super**(); //可省略  }  **public** SendTask(String number, **double** goodsWeight) {  **this**.number = number;  **this**.goodsWeight = goodsWeight;  }  //送前准备  **public** **void** sendBefore () {  System.***out***.println("订单开始处理，仓库验货中。。。");  System.***out***.println("货物重量："+**this**.getGoodsWeight()+"kg");  System.***out***.println("货物检验完毕!");  System.***out***.println("货物填装完毕!");  System.***out***.println("运货人已通知!");  System.***out***.println("快递单号："+**this**.getNumber());  }  //发送货物  **public** **void** send(Transportation t,GPS tool) {  System.***out***.println("运货人"+t.getAdmin()  +"正在驾驶编号为"+t.getNumber()  +"的"+t.getModel()+"发送货物！");  t.transport();  String showCoordinate = tool.showCoordinate();  System.***out***.println("货物当前的坐标为："+showCoordinate);  }  //送后操作  **public** **void** sendAfter(Transportation t) {  System.***out***.println("货物运输任务已完成！");  System.***out***.println("运货人"+t.getAdmin()  +"所驾驶的编号为"+t.getNumber()  +"的"+t.getModel()+"已归还！");  }  **public** String getNumber() {  **return** number;  }  **public** **void** setNumber(String number) {  **this**.number = number;  }  **public** **double** getGoodsWeight() {  **return** goodsWeight;  }  **public** **void** setGoodsWeight(**double** goodsWeight) {  **this**.goodsWeight = goodsWeight;  }  }  /\*  \* 定义测试类  \*/  **public** **class** Task02Test {  **public** **static** **void** main(String[] args) {  // 快递任务类对象  SendTask task = **new** SendTask("HYX600235",76.34);  // 调用送前准备方法  task.sendBefore();  System.***out***.println("======================");  // 创建交通工具对象  ZTransportation t = **new** ZTransportation("Z025","保温冷藏车","小韩");  //创建GPS工具对象  Phone p = **new** Phone();  // 将交通工具与GPS工具传入送货方法  //补充代码  System.***out***.println("======================");  // 调用送后操作方法  task.sendAfter(t);    }  }  【案例5】异常捕获处理  以下程序代码段中，当n取值为0或取值>5时，语句 x = a[n] / n 将出现异常。编写程序对以下语句进行异常处理。输入n为0、2、5，s等不同的值验证异常处理结果。程序保存为E4\_5.java。  程序代码段：  Scanner sc=new Scanner(System.in);  int x, n, a[] = {1, 2, 3, 4, 5};  n =sc.nextInt();  x = a[n] / n;  System.out.println("x=" + x);  提示：需要处理多个异常。其中算术运算异常类是：ArithmeticException；数组下标越界运行异常类是：ArrayIndexOutOfBoundsException。  【案例6】自定义异常练习  1.在员工类Employee中,定义double属性salary。  2.添加方法setSalary0 ,无返回值，参数double ,采用throws声明SalaryException异常。  3.在方法中判断salary,当salary< 1500通过throw抛出SalaryException异常,在异常类构造函数中传入"工资低于1500"。  4、在该类的main方法中实例化ep ,在异常处理中调用setSalary(1200)方法,在异常处理中通过异常的getMessage方法获得异常信息并输出。  代码结构：  Employee.java  //错误处理类  class SalaryException extends Exception{  //无参构造方法  //有参构造方法  }  //员工类  public class Employee {  //成员变量工资salary  //成员方法setSalary()  }  Test.java  public static void main(String[] args) {    } | | | | | | | |
| **学生实验步骤记录** | | | | | | | |
| 【案例1】打印不同的图形    **【案例4-2】多彩的声音**  【案例4-3】饲养员喂养动物    【案例4】模拟物流快递系统程序设计    【案例5】异常捕获处理    【案例6】自定义异常练习 | | | | | | | |
| **实验分析与小结** | | | | | | | |
| 在进行前述六个实验案例时，我们深入探索了众多Java编程的核心思想和手段，如异常管理、抽象类定义、接口设计、继承关系、多态性特点以及Lambda语法的应用。以下是各案例的简要回顾与总结：  案例4-2：声音的变化  发现与心得：该示例进一步明确了如何运用多态性和接口，使得不同的类能够实现同一接口，并在测试环境中统一地调用它们。  案例4-3：饲养人员的动物喂食行为  遇到的问题与对策：该实验涉及接口多重继承的难题，由于 Fish 同时被分类为动物和食物。这一问题得以解决，原因在于将 Food 构建为接口，这样就实现了多重继承。  发现与心得：此实验示范了接口如何解决多继承难题，并进一步加深了我们对面向对象编程中的抽象和继承机制的理解。  案例4：仿真的快递配送系统设计  遇到的问题与对策：此实验融合了多个面向对象编程的精髓，例如抽象类、接口、多态和封装。需要建立一个名为 Transportation 的抽象类及其派生类，同时设计 GPS 接口。在测试类里，采纳了匿名内部类和 lambda 表达式技巧。  发现与心得：此示例巩固了多种面向对象编程原则，如继承、多态、接口及异常管理。此外，我们还学习了如何使用匿名内部类和 lambda 语法简化代码逻辑。  案例4-5：异常情境的捕获与处理  遇到的问题与对策：这一实验主要关注异常处理，包含了数学运算错误和数组边界溢出异常。我们采用 try-catch 结构来捕捉并展示异常详情。  发现与心得：此案例加强了对异常处理手段及其重要性的认知，同时明确了如何针对不同情境选用合适的异常类。  案例4-6：个性化异常的设计与应用  遇到的问题与对策：在此实验中，要求设计一个名为 SalaryException 的特定异常，并在 setSalary 功能中触发这一异常。接着，在主功能中，我们采用 try-catch 结构来捕捉并展示异常信息。  发现与心得：这一实验强调了如何定义并应用个性化异常，并重申了处理异常的关键性，特别是在涉及多样业务逻辑的场合。  结论：  在完成上述实验后，我们对Java编程的核心理念和方法有了更全面的洞察。我们学会了如何有效地管理异常、如何利用抽象类与接口实现多态性，以及如何构建和应用个性化异常。这些技巧对于编写高效稳定的Java程序至关重要。实验过程还让我们实际体验到了，如何根据面向对象的编程哲学，来高效地处理真实问题。 | | | | | | | |
| **指导老师评阅意见** | | | | | | | |
| 点评意见：（实验态度、实验结果、实验记录完整性、实验报告表达能力、分析问题与解决问题能力） | | | | | | | |
| 指导老师： | | | | | | | |

填写内容时，可把表格扩大。