重庆科技学院

上机实验报告(上机操作类)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 面向对象程序设计方法 | | 实验题目 | 实验2: Java类和对象的设计 | | |
| 机房编号 | I306 | | 上机时间 |  | | |
| 指导老师 | 周召敏 | | 报告序号 | 2 | 上机成绩 |  |
| 学生姓名 | 刘畅 | 学号 | 2022442728 | 教学班级 | 物联网一班 | |
| **一、上机操作目的和要求**  1. 掌握面向对象的编程思想、类与对象；  2. 掌握类的封装性、继承性和多态性的作用；  3. 掌握成员变量和成员方法的特性、构造方法、toString方法、equals方法的使用；  4. 掌握this、super、final、static等关键字的使用；  5. 掌握接口及接口对象的使用；  6. 掌握包的定义与使用；  7. 掌握四种访问控制权限private、default、protected、public。 | | | | | | |
| **二、上机操作需要的软、硬件**  处理器： 内存容量：8GB  操作系统： JDK： IDE：Eclipse | | | | | | |
| **三、上机操作内容(老师布置的具体任务)**  1.定义一个类MyProgram，包含两个属性：一个是private的整型属性data、一个是private的String类型属性str，封装这两个属性的四个方法setData( )和getData( )、setStr( )和getStr( )；将这两个属性转变为字符串的方法是toString( )。编写程序，使用MyProgram类，实现数据的访问和修改，并调用toString( )方法显示该类的属性。  [基本要求] 编写完整程序。  [思考问题] 试述程序中各个方法的作用  2.在Vehicle类的基础上创建一个Tractor(拖拉机)类，它不仅拥journey、wheelNum、LoadNum、driveSpeed四种属性和DriveAt()方法，而且还拥有自己的plough()方法，在该方法中要求能够显示journey、wheelNum、LoadNum三个属性的值。/继承  请编写测试类，运行DriveAt()、plough()方法。  3.有图形接口Shape，参照圆Circle类补充完整正方性Square和三角形Triangle类，并分析运行结果。  4.有两个类：MobileManagement和Mobile，分别描述如表2.1所示两部手机名称及价格，类MobileManagement在包cn.edu.cqust.test4中，而Mobile在包cn.edu.cqust.t  est4.data中。它们代码如下。运行MobileManagement.java，使程序能够显示两部手机的价格和数量。  手机价格表   |  |  |  | | --- | --- | --- | | 序号 | 名称 | 价格 | | 1 | 荣耀8 | 1099 | | 2 | 小米Max2 | 1699 |   Mobile [品牌：荣耀8， 价格： 1099.0元]  Mobile [品牌：小米Max2， 价格： 1699.0元]  5.USB接口程序设计  通常人们使用的计算机上都有USB接口，鼠标、键盘、麦克风等都可以连接到USB接口中使用。在计算机启动时，这些设备也随之启动；当计算机关闭时，这些设备也会随之关闭。鼠标、键盘、麦克风等USB接口设备都启动后，计算机才开机成功；当这些设备都关闭后，计算机才关机成功。  读所编写的USB接口程序，该程序模拟了计算机的开机和关机过程，写出运行结果。  6．请简述this关键字的作用，并举例说明；  7．请简述super关键字的作用，并举例说明；  8．请简述final关键字的作用，并举例说明。  9．请简述static关键字的作用，并举例说明。 | | | | | | |
| **四、上机操作的基本步骤** | | | | | | |
| 五、上机操作的结果截图及还存在的问题  **题目第一题代码：**  package test2;  import javax.xml.crypto.Data;  class MyProg<T,U>{  private T Data;  private U Str;  public MyProg(T data, U str) {  Data = data;  Str = str;  }  public T getData() {  return Data;  }  public void setData(T data) {  Data = data;  }  public U getStr() {  return Str;  }  public void setStr(U str) {  Str = str;  }  public MyProg() {  }  @Override  public String toString() {  return Data+" "+Str;  }  public boolean equals(MyProg b) {  if(this.Data==b.Data&&this.Str==b.Str)  return true;  return false;  }  }  package test2;  public class demo1 {  public static void main(String[] args) {  MyProg<Integer, String> Mp = new MyProg<>();  MyProg<Integer,String> Mp2=new MyProg<>();  Mp.setData(1);  Mp.setStr("chizi");  Mp2.setData(1);  Mp2.setStr("chizi");  MyProg<Integer, String> Mp3 = new MyProg<>(10,"chizi");  System.out.println(Mp.getData()+" "+Mp.getStr());  System.out.println(Mp.toString());  System.out.println("Mp与Mp2相等吗？"+Mp.equals(Mp2));  System.out.println("Mp与Mp3相等吗？"+Mp.equals(Mp3));  }  }    **题目第二题代码：**  package test2.exer02;  import java.security.PublicKey;  public abstract class Vehicle {  private double journey;  private int wheelNum;  private int LoadNum;  private int driveSpeed;  public double getJourney() {  return journey;  }  public void setJourney(double journey) {  this.journey = journey;  }  public int getWheelNum() {  return wheelNum;  }  public void setWheelNum(int wheelNum) {  this.wheelNum = wheelNum;  }  public int getLoadNum() {  return LoadNum;  }  public void setLoadNum(int loadNum) {  LoadNum = loadNum;  }  public Vehicle() {  }  public Vehicle(double journey, int wheelNum, int loadNum, int driveSpeed) {  this.journey = journey;  this.wheelNum = wheelNum;  LoadNum = loadNum;  this.driveSpeed = driveSpeed;  }  public int getDriveSpeed() {  return driveSpeed;  }  public void setDriveSpeed(int driveSpeed) {  this.driveSpeed = driveSpeed;  }  public void DriveAt() {  if (driveSpeed > 60) {  System.out.println("行车速度过快，容易造成事故");  driveSpeed = 40;  } else {  System.out.println("你在安全速度中行驶");  driveSpeed = driveSpeed;  }  }  public abstract void plough();  }  package test2.exer02;  public class Tractor extends Vehicle{  public Tractor() {  }  public Tractor(double journey, int wheelNum, int loadNum, int driveSpeed) {  super(journey, wheelNum, loadNum, driveSpeed);  }  @Override  public void plough() {  System.out.println(this.getJourney()+" "+this.getWheelNum()+" "+this.getLoadNum());  }  }  package test2.exer02;  public class demo2 {  public static void main(String[] args) {  Tractor t=new Tractor(103.5,4,1,65);  t.DriveAt();  System.out.println(t.getDriveSpeed());  t.plough();  }  }    **题目第三题代码：**  package test2.exer03;  import javax.crypto.interfaces.PBEKey;  public interface Shape {  public abstract void draw();  public abstract void erase();  }  package test2.exer03;  public class Circle implements Shape{  @Override  public void draw() {  System.out.println("重写了接口shape中的draw函数");  }  @Override  public void erase() {  System.out.println("重写了接口shape中的erase函数");  }  }  package test2.exer03;  public class Square implements Shape{  @Override  public void draw() {  System.out.println("重写了接口shape中的draw函数");  }  @Override  public void erase() {  System.out.println("重写了接口shape中的erase函数");  }  }  package test2.exer03;  public class Triangle implements Shape{  @Override  public void draw() {  System.out.println("重写了接口shape中的draw函数");  }  @Override  public void erase() {  System.out.println("重写了接口shape中的erase函数");  }  }  package test2.exer03;  import test2.exer02.Tractor;  public class demo3 {  public static void main(String[] args) {  Circle c=new Circle();  c.draw();  c.erase();  Square s=new Square();  s.draw();  s.erase();  Triangle t=new Triangle();  t.draw();  t.erase();  }  }    **题目第四题代码**：  package cn.edu.cqust.t;  public class Mobile {  private String name;  private double price;  public Mobile() {  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public double getPrice() {  return price;  }  public void setPrice(double price) {  this.price = price;  }  public Mobile(String name, double price) {  this.name = name;  this.price = price;  }  public void show()  {  System.out.println("Mobile ["+"品牌："+name+","+"价格："+price+" ]");  }  }  package cn.edu.cqust.test4;  import cn.edu.cqust.t.Mobile;//导包，让MobileManagement可以使用Mobile类  public class MobileManagement {  public static void main(String[] args) {  Mobile mb1 = new Mobile();  mb1.setName("荣耀8");  mb1.setPrice(1099.0);  Mobile mb2=new Mobile("小米Max2",1699.0);  mb1.show();  mb2.show();  }  }    **题目第五题代码：**  package test2.exer05;  public interface USB {  void turnon();  void turnoff();  }  package test2.exer05;  public class Mouse implements USB{  @Override  public void turnon() {  System.out.println("鼠标？启动！！！");  }  @Override  public void turnoff() {  System.out.println("鼠标关闭了");  }  }  package test2.exer05;  public class Mic implements USB{  @Override  public void turnon() {  System.out.println("麦克风？启动！！！");  }  @Override  public void turnoff() {  System.out.println("麦克风关闭了");  }  }  package test2.exer05;  public class KeyBoard implements USB{  @Override  public void turnon() {  System.out.println("键盘?启动！！！");  }  @Override  public void turnoff() {  System.out.println("键盘关闭了");  }  }  package test2.exer05;  public class Computer {  private USB[] usbArr=new USB[4];  public void add(USB usb){  for(int i=0;i<usbArr.length;i++){  if(usbArr[i]==null){  usbArr[i]=usb;  break;  }  }  }  //计算的开机功能  public void powerOn(){  //遍历所有的插槽  for(int i=0;i<usbArr.length;i++){  //如果发现有设备  if(usbArr[i]!=null){  //将USB设备启动  usbArr[i].turnon();  }  }  System.out.println("计算机开机成功！！！");  }  //计算的关机功能  public void powerOff(){  //遍历所有的插槽  for(int i=0;i<usbArr.length;i++){  //如果发现有设备  if(usbArr[i]!=null){  //将USB设备关闭  usbArr[i].turnoff();  }  }  System.out.println("计算机关机成功！！！");  }  }  package test2.exer05;  public class test {  public static void main(String[] args) {  Computer computer=new Computer();  //向计算机中添加鼠标、麦克风、键盘  computer.add(new Mouse());  computer.add(new Mic());  computer.add(new KeyBoard());  //启动计算机  computer.powerOn();  //关闭计算机  computer.powerOff();  }  }    **第六题答案：**  当局部变量和成员变量同名时，可以使用 this 关键字来引用成员变量，以区分局部变量和成员变量  public class Person {  private String name; // 成员变量  public void setName(String name) { // 局部变量  this.name = name; // this.name 表示成员变量，name 表示局部变量  }  }  **第七题答案：**  当子类中的成员变量和父类中的成员变量同名时，可以使用 super 关键字来引用父类中的成员变量，以区分子类和父类中的同名变量  public class Animal {  protected String name; // 父类中的成员变量  }  public class Dog extends Animal {  private String name; // 子类中的成员变量  public void showName() {  System.out.println("子类中的 name: " + this.name); // this.name 表示子类中的成员变量  System.out.println("父类中的 name: " + super.name); // super.name 表示父类中的成员变量  }  }  **第八题答案：**  修饰类：表示该类不能被继承，也就是说，没有其他类可以作为它的子类。这样可以保证类的完整性和安全性，避免被其他类修改或扩展。例如，String 类就是一个 final 类，它不能被继承  // 修饰类  final class Animal {    }  class Dog extends Animal { // 编译错误：不能继承 final 类    }  **第九题答案：**  修饰成员变量：表示该变量属于类，而不是对象，也就是说，所有该类的对象共享同一个 static 变量。static 变量只在类加载时初始化一次，可以通过类名直接访问，不需要创建对象。  public class Student {  private static int count = 0; // static 变量，记录学生的数量  private String name;  public Student(String name) {  this.name = name;  count++; // 每创建一个学生对象，count 加一  }  public static int getCount() { // static 方法，返回学生的数量  return count;  }  }  public class Test {  public static void main(String[] args) {  System.out.println("学生数量：" + Student.getCount()); // 通过类名访问 static 方法  Student s1 = new Student("张三");  Student s2 = new Student("李四");  System.out.println("学生数量：" + Student.getCount()); // 通过类名访问 static 方法  System.out.println("学生数量：" + s1.getCount()); // 也可以通过对象访问 static 方法  System.out.println("学生数量：" + s2.getCount()); // 也可以通过对象访问 static 方法  }  } | | | | | | |
| 六、上机操作的收获及心得  1.学会运用基本的继承，接口，导包。，对this，super，final，static有了基本的了解。  2.学会定义抽象类，重写抽象方法。  3.掌握了java语言中继承，接口，类的语法。 | | | | | | |

建议：正文采用五宋体，代码用小五号Times new Roman字