**抽象类+接口+内部类作业题**

1. **选择题**

|  |  |  |
| --- | --- | --- |
| **1.** | **Person类和Test类的代码如下所示，则代码中的错误语句是（ C ）。（选择一项）** | |
|  | **public class** Person {  public String name;  **public** Person(String name) {  **this**.name = name;  }  }  **Public class** Test {  **Public static void** main(String[] args) {  **final** Person person = **new** Person("欧欧");  person.name = "美美";  person = **new** Person("亚亚");  }  } | |
|  |  |  |
|  | **A.** | final Person person = new Person("欧欧"); |
|  | **B.** | person.name = "美美"; |
|  | **C.** | person = new Person("亚亚"); |
|  | **D.** | 没有错误 |

|  |  |  |
| --- | --- | --- |
| **2.** | **如下Java代码运行的结果是（ B ）。（选择一项）** | |
|  | **Public class** Test {  **final int** age;  **public** Test(){  age=10;  }  **publicstaticvoid** main(String[ ] args) {  System.*out*.println(**new** Test().age);  }  } | |
|  |  |  |
|  | **A** | 输出：0 |
|  | **B.** | 输出：10 |
|  | **C.** | 输出：null |
|  | **D.** | 运行时出现异常 |

|  |  |  |
| --- | --- | --- |
| **3.** | **以下代码中错误的语句是（ D ）。（选择一项）** | |
|  | **Public class** Something{  **Public static void** main(String[] args){  **final** Other o=**new** Other();  **new** Something().addOne(o);//1  }  **Public void** addOne( Other o){  o.i++;//2  o = **new** Other();//3  }  }  **class** Other{  **public int** i;  } | |
|  |  |  |
|  | **A** | 1 |
|  | **B.** | 2 |
|  | **C.** | 3 |
|  | **D.** | 没有错误 |

|  |  |  |
| --- | --- | --- |
| **4.** | **下列选项中，关于Java的抽象类和抽象方法说法正确的是（ AD ）。（选择二项）** | |
|  |  |  |
|  | **A** | 抽象类中可以含有0个或多个抽象方法 |
|  | **B.** | 抽象类中不可以有构造方法 |
|  | **C.** | 一个类中若有抽象方法，则这个类必为抽象类 |
|  | **D.** | 子类必须重写父类所有的抽象方法 |

|  |  |  |
| --- | --- | --- |
| **5.** | **在Java中关于abstract关键字，以下说法正确的是（ AB ）。（选择两项）** | |
|  |  |  |
|  | **A** | abstract类中可以没有抽象方法 |
|  | **B.** | abstract类的子类也可以是抽象类 |
|  | **C.** | abstract方法可以有方法体 |
|  | **D.** | abstract类可以创建对象 |

|  |  |  |
| --- | --- | --- |
| **6.** | **以下选项可替换题目中//add code here而不产生编译错误的是（ AD ）。（选择二项）** | |
|  | **publicabstractclass** MyClass{  **public int** constInt=5;  //add code here  **publicvoid** method( ){  }  } | |
|  |  |  |
|  | **A** | **Public abstract void** method(**int** a); |
|  | **B.** | constInt =constInt+5; |
|  | **C.** | **Public int** method( ); |
|  | **D.** | **publicabstractvoid** anotherMethod( ); |

|  |  |  |
| --- | --- | --- |
| **7.** | **在Java接口中，下列选项中属于有效的方法声明是（ AD ）。（选择二项）** | |
|  |  |  |
|  | **A** | public void aMethod(); |
|  | **B.** | final void aMethod(); |
|  | **C.** | void aMethod(){ } |
|  | **D.** | private void aMethod(); |

|  |  |  |
| --- | --- | --- |
| **8.** | **以下选项中关于匿名内部类的说法正确的是（ BC ）。（选择二项）** | |
|  |  |  |
|  | **A.** | 匿名内部类可以实现多个接口，或者继承一个父类 |
|  | **B.** | 匿名内部类不能是抽象类，必须实现它的抽象父类或者接口里包含的所有抽象方法 |
|  | **C.** | 匿名内部类没有类名，所以匿名内部类不能定义构造方法 |
|  | **D.** | 匿名内部类可以直接访问外部类的所有局部变量 |

**9.** **接口定义正确的说法是（ AD ）。（选择二项）**

**A.** 接口中只能定义常量和抽象方法

**B.** 接口中可以定义普通方法和普通变量

C. 接口可以被new，抽象方法不行

D． 接口不可以被继承，只能被实现，也可以被多个类实现

1. **判断题（共20个题目，总计10分）**
2. 声明为final的类不能是超类。（ ✔ ）
3. 使用final修饰的变量将变成常量，其中不能再被改变；使用final修饰的方法将无法被子类重载；使用final修饰的类将无法再被其他类继承。（ ✖ ）
4. 抽象方法不能含有方法体，并且必须定义在抽象类中。（ ✔ ）
5. 抽象类是指在class前加使用abstract关键字修饰，且可以存在抽象方法和普通方法的类。（ ✔ ）
6. 接口中只有常量，没有变量；只有抽象方法，并且全部都是public方法。（ ✔ ）
7. 抽象类和接口都不能实例化，都没有构造方法。（ ✖ ）
8. 接口并不要求实现类和接口在概念本质上一致的，仅仅是实现了接口定义的约定或者能力而已。接口定义了“做什么”，而实现类负责完成“怎么做”，体现了功能（规范）和实现分离的原则。（ ✔ ）
9. 内部类作为外部类成员，权限修饰符和其他成员一样，可声明为private、默认、protected或public。( ✔ )
10. 匿名内部类适合创建那种只需要使用一次的类，它可以实现一个或多个接口，或者继承一个父类。（ ✖ ）
11. 对于物理连接，比如数据库连接、输入流输出流、Socket连接等，垃圾回收机制无能为力，必须手动关闭才可以。( ✔ )
12. 垃圾回收机制回收任何对象之前，总会先调用它gc( )方法，该方法是Object类提供的方法。不要主动调用某个对象的该方法，应该交给垃圾回收机制调用。（ ✔ ）
13. final修饰的方法不能被重写，但可以被重载（ ✔ ）
14. 抽象方法的类必须是抽象类，同样抽象类也必须包含抽象方法（ ✖ ）
15. 抽象类可以定义普通方法，但不能定义构造方法（ ✖ ）
16. 抽象类不能被new，抽象类的构造方法也不能被调用（ ✖ ）
17. **简答题**
    1. final修饰变量、方法、类分别表示什么？

不可更改，不可复写，不可被继承

* 1. final和abstract关键字的作用。

Final是最终的，不可更改

Abstract是抽象的，没有实现方法，或者不能被创建对象

* 1. 接口和抽象类的联系和区别。

接口可以实现多个，类只能继承一个

* 1. java中有多继承吗?如果没有，java为什么取消了多继承？

C++中有多继承吗? 单继承相比多继承优势在什么地方?

没有多继承，C++可以

**A有一个方法，B和C都继承了它，并且复写了方法。D同时继承了B和C，方法该是哪个的**

* 1. 内部类的类型及其特点。

匿名内部类和成员内部类

匿名内部类没有名字，只用一次

成员内部类可以使用多次

* 1. 介绍Java垃圾回收机制。

没有被引用的内存空间会被jvm杀掉

1. **编码题**
2. 编写程序描述兔子和青蛙

需求说明：使用面向对象的思想，设计自定义类描述兔子和青蛙。

实现思路及关键代码

1. 分析兔子和青蛙的共性
2. 根据共性，定义抽象的动物类

属性：名字、颜色、类别（哺乳类、非哺乳类）

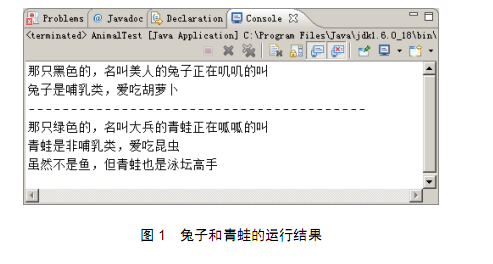
方法：吃饭，发出叫声

1. 根据青蛙会游泳 抽象游泳的接口

方法：游泳

1. 定义兔子继承动物类，青蛙继承动物同时实现游泳接口

程序运行结果如图所示。



电脑萤幕的截图

描述已自动生成

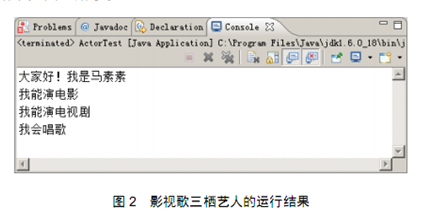
1. 编写程序描述影视歌三栖艺人

需求说明：请使用面向对象的思想，设计自定义类，描述影视歌三梄艺人。

实现思路及关键代码

1. 分析影视歌三栖艺人的特性
   1. 可以演电影
   2. 可以演电视剧
   3. 可以唱歌
2. 定义多个接口描述特性
   1. 演电影的接口-----方法：演电影
   2. 演电视剧的接口-----方法：演电视剧
   3. 唱歌的接口-----方法：唱歌
3. 定义艺人类实现多个接口

程序运行结果如图2所示。



文本

描述已自动生成

1. 写一个方法对任意引用数据类型数组进行排序。具体要求如下:
   1. 方法声明为public void sortArr(Object arr[]){ }
   2. 方法中首先输出排序前数组内容，然后进行排序，最后输出排序后数组内容。
   3. 可以是冒泡排序或其他算法实现，不直接调用Java提供的方法实现排序。

思路：任意类实现Comparable接口来实现该引用数据类型的元素排序，在sort()方法中将Object强转成Comparable实现两个对象的比较。

电脑萤幕画面

描述已自动生成

1. **可选题**
   1. 实现不同符合PCI规范的适配器

需求说明：PCI是一种规范，所有实现了该规范的适配器，必如显卡、声卡、网卡都可以安装到PCI插槽上并工作。模拟实现该功能。



实现思路及关键代码

1. 定义PCI接口，具有传送数据send()方法
2. 定义显卡VideaCard类，实现该接口
3. 定义声卡AudioCard类，实现PCI接口
4. 定义网卡NetCard类，实现PCI接口
5. 定义测试类，让显卡、声卡、网卡发送数据

电脑屏幕的截图

描述已自动生成

* 1. 实现不同引用类型对象的大小比较

需求说明：学生类，新闻类，商品类虽然是完全不同的类，但是都具有比较的能力，比如可以比较两个学生的大小，但需要指定比较的依据是学号、姓名还是成绩等。

实现思路及关键代码：

将比较的能力定义为接口，让学生、新闻、商品类都实现该接口。

1. 定义接口Comparable，其中包含唯一的方法int compareTo(Object obj)；返回值>0，表示大于，返回值=0，表示等于，返回值<0，表示小于。
2. 定义学生类，包括学号、姓名、年龄和分数，实现Comparable接口，按照分数倒序排列；
3. 定义新闻类，包括编号（int类型）、标题、内容和点击数，实现Comparable接口，按照编号正序排列；
4. 定义测试类，分别创建两个学生对象、新闻对象，进行比较并输出结果。

文本

描述已自动生成