|  |  |  |  |
| --- | --- | --- | --- |
| **学号** | 2113041111 | **姓名** | 乔嘉琛 |
| **专业** | 软件工程 | **日期** | 2024/9/24 |
| **实训任务：第七章 抽象、接口、内部类、枚举**   1. 请解释abstract关键字的作⽤是什么。   答：  修饰符,修饰属性为抽象类   1. 请简述抽象类的特点。   答：  只能继承,不能实现.  抽象方法和抽象类必须使用 abstract 关键字修饰  使用abstract修饰的类就是抽象类  抽象类可以包含，也可以不包含抽象方法  包含抽象方法的类，一定要声明为抽象类  3. 请回答抽象类及抽象⽅法可以被final修饰符修饰吗?  答：  不能,final是最终类和方法不能重写和继承.  4. 请阅读并分析以下程序，判断以下代码是否正确，如果正确，输出什么信息？如果不正确，请说明原因。    答：  B,多态.  5. 编写⼀个抽象类 Shape，包含以下属性和⽅法：  属性：颜⾊ color。  ⽅法：计算形状⾯积的抽象⽅法 area() 和返回颜⾊信息的普通⽅法 getColor()。  然后创建两个具体的⼦类 Rectangle（矩形） 和 Circle（圆形），实现 area() ⽅法并考虑它们各⾃的特点。  要求能够使⽤以下测试代码进⾏测试：    答：    6. 设计⼀个简单的动物园系统，包含以下⼏种动物：猫、狗、鸟。每种动物都有自己的特点和行为，但都有共同的属性和⽅法。其中，猫、狗和鸟都属于动物类，但它们的叫声和行为各不相同。  要求：  A) 使⽤抽象类来实现动物园系统，包含⼀个抽象⽅法 makeSound() ⽤于输出动物的叫声。  B) 每种动物都有⼀个属性 name 表示动物的名称，以及⼀个⽅法 move() ⽤于输出动物的⾏为。  C) 猫类具有属性 color 表示猫的颜⾊，重写 makeSound() ⽅法输出猫的叫声，并重写 move() ⽅法输出猫的⾏为。  D) 狗类具有属性 age 表示狗的年龄，重写 makeSound() ⽅法输出狗的叫声，并重写 move() ⽅法输出狗的行为。  E) 鸟类具有属性 wingSpan 表示鸟的翼展，重写 makeSound() ⽅法输出鸟的叫声，并重写 move() ⽅法输出鸟的⾏为。  要求能够使⽤以下测试代码进⾏测试：    答：      7. 请回答Java中的内部类有哪⼏种，并说明它们的特点。  答： 成员内部类（Member Inner Class）  1. **定义位置**：成员内部类定义在外围类内部，与成员变量和成员方法处于同一层次。 2. **访问权限**：可以访问外围类的所有成员（包括私有成员）。 3. **实例化**：必须通过外围类实例来创建成员内部类实例。 4. **特点**：能够实现对外围类内部数据的封装，增强封装性。  静态内部类（Static Nested Class）  1. **定义位置**：静态内部类同样定义在外围类内部，使用static关键字修饰。 2. **访问权限**：只能访问外围类的静态成员，不能直接访问非静态成员。 3. **实例化**：不需要外围类实例，可以直接通过类名进行实例化。 4. **特点**：通常用于组织类，使得类之间的关系更加清晰，不依赖于外围类的实例。  局部内部类（Local Inner Class）  1. **定义位置**：局部内部类定义在外围类的方法内部。 2. **访问权限**：可以访问外围类的所有成员，包括私有成员，以及局部final变量。 3. **实例化**：只能在定义它的方法内部实例化。 4. **特点**：通常用于处理比较复杂的方法，将类的作用域限制在方法内部。  匿名内部类（Anonymous Inner Class）  1. **定义位置**：匿名内部类没有类名，直接在创建实例的地方定义。 2. **访问权限**：可以访问外围类的所有成员，包括私有成员。 3. **实例化**：通常用来创建那些只需要一次使用的类实例，通常是接口或抽象类的子类。 4. **特点**：简化代码，不需要单独为只用一次的类创建文件。 5. 按照要求，补齐代码，要求在控制台输出 HelloWorld。     答：     1. 使⽤匿名内部类⽅式实现下⾯接⼝，并完成规定的功能。     对该接⼝进⾏实现，让其可以具有处理以下两种字符串的功能：   1. 对字符串按照 | 进⾏分割，并得到数组，例如 传⼊字符串"a|b|c"，返回["a","b","c"] 2. 对字符串按照 - 进⾏分割，并得到数组，例如 传⼊字符串"a-b-c"，返回["a","b","c"]   要求使⽤匿名内部类的⽅式来完成。  提示：分割字符串，学习String类中split⽅法实现。  答：     1. 请描述各个权限修饰符的作⽤。   答：  public：  当成员（类、方法、变量）被声明为public时，它可以从任何其他类中访问，无论这些类是否在同一个包中，还是在不同的包中。  public修饰符提供了最高的访问级别。  protected：  当成员被声明为protected时，它可以被同一个包中的任何类访问，以及不同包中的子类（即使子类不是直接的，也可以是间接的）。  protected提供了比默认更高的访问级别，但比public低。  default（无修饰符）：  当成员没有指定访问修饰符时，它具有默认访问级别，也称为包私有（package-private）。  默认成员只能被同一个包中的类访问，不能被不同包中的类访问。  private：  当成员被声明为private时，它只能在其所属的类内部访问，不能被任何外部类访问，即使是同一个包中的类或子类。  private提供了最低的访问级别。   1. 请简述什么情况下该⽤抽象类、什么情况下该⽤接⼝。   答：  抽象类是包含未实现方法的类，用于定义共同特性，需子类继承并实现这些方法。接口是方法的集合，实现类必须实现所有方法。抽象类适合部分实现共享，接口适合定义不同类的共同行为。   1. 根据以下描述编写程序。   鸟类（Bird）可以飞翔，飞机类（Plane）也可以飞翔，那么可以把飞翔（flyAction）这个动作抽象出来，抽象到⼀个抽象类或者接⼝中，然后让鸟类和飞机类去继承或实现，那么这⾥是把⽅法抽象到⽗类型中合适，还是抽象到接⼝中合适？编写代码完成本题。  要求能够使⽤以下测试代码进⾏测试：    答：     1. 按照以下要求完成编程题。     答：     1. 请写出基本数据类型对应的的包装类名。   答：   1. 请阅读并分析以下程序案例，写出程序运行输出的结果。     答：new是创建对象,==是直接比较地址值,直接赋值可以==比较,new对象则地址值不一样,则不能用==比较.equals是比较对象值,包装类因为重写过equals方法;   1. 请简述==和equals()的区别。   答：  == 是一个比较运算符，用于比较两个变量的值是否相等。  当用于基本数据类型（如 int、float、double 等）时，== 比较的是它们的值是否相等。  当用于引用数据类型（如对象、数组等）时，== 比较的是两个引用是否指向同一个对象，即它们是否引用内存中的同一个位置。  equals 方法  equals 是 Object 类中的一个方法，用于比较两个对象的内容是否相等。  对于基本数据类型的包装类（如 Integer、Double 等），equals 方法已经被重写，用于比较包装对象内部的值是否相等。  对于自定义类，如果不重写 equals 方法，则默认使用 Object 类中的实现，它实际上等同于 ==，比较的是两个对象的引用是否相同。但是，通常建议对于自定义类重写 equals 方法，以比较对象的内容。   1. 请阅读并分析下⾯程序的运行结果。     答：  都返回b的类名.   1. 阅读并分析以下两个案例中代码的运⾏结果。   案例⼀：    案例二：    答：  例：  String str = "Hello";如果"Hello"之前还没有被存入 字符串常量池中，那么JVM会先将这个字符串添加到字符串常 量池中，再返回该字符串的引用。如果"Hello"已经被放入，则 直接返回引用。   1. 使用new关键字创建对象 例： String str = new String("World");如果字符串常 量池中不存在"World"对象，则先将其添加到字符串常量池中， 然后根据该对象在堆空间中去创建对象 2. 另外两个对象相加也跟new一个新对象一致.但但是类似stringbuild一样拼接字符,只会创建一个类,不和new一样在常量区创建 3. 请回答String类型与基本数据类型与包装类如何转换？   答：  // 使用包装类的parseXxx静态方法  int i = Integer.parseInt("123");  long l = Long.parseLong("1234567890");  float f = Float.parseFloat("3.14");  double d = Double.parseDouble("3.141592653589793");  boolean b = Boolean.parseBoolean("true"); // 注意：除了"true"，其他字符串都会解析为false  // char类型稍有不同，因为String实际上是由char数组构成的  char c = "A".charAt(0);  // 使用包装类的valueOf方法，然后通过xxxValue方法转换为基本数据类型  int i = Integer.valueOf("123").intValue();  long l = Long.valueOf("1234567890").longValue();  // ... 其他基本类型转换类似 | | | |