试题

1. **填空题**
2. 类是组成Java程序的基本要素，类体有两部分构成：一部分是变量的定义，另一部分是 方法 的定义。
3. 执行Person p = new Person();语句后，将在\_堆\_中给Person对象分配空间，并在栈内存中给引用变量p分配空间，存放Person对象的引用。
4. 构造方法 是一种特殊方法，它的名字必须与它所在的类的名字完全相同，并且不书写返回值类型，在创建对象实例时由new运算符自动调用。
5. 局部变量的名字与成员变量的名字相同，若想在该方法内使用成员变量，必须使用关键字 this 。
6. 使用关键字 this(形参列表) 来调用同类的其它构造方法，优点同样是以最大限度地代码的利用程度 ，减少程序的维护工作量。
7. 用关键字 static 修饰的成员变量是类变量，类变量是指不管类创建了多少对象，系统仅在第一次调用类的时候为类变量分配内存，所有对象共享该类的类变量。
8. 使用static修饰的变量称为静态变量，静态变量可以有两种访问方式，分别是类名.静态变量名和\_\_\_实例对象.变量名\_\_\_。
9. 在一个类文件中的关键字package，import，class出现的可能顺序是\_\_\_package import class\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
10. package 关键字作为Java源文件的第一条非注释性语句，指明该源文件定义的类所有的包。

lang 包是Java语言的核心类库，它包含了运行Java程序必不可少的系统类，使用该包下的类和接口不需要使用import导入。

1. **选择题**

|  |  |  |
| --- | --- | --- |
| **1.** | **在Java中，以下程序编译运行后的输出结果为（ D ）。（选择一项）** | |
|  | **public** **class** Test {  **int** x, y;  Test(**int** x, **int** y) {  **this**.x = x;  **this**.y = y;  }  **public** **static** **void** main(String[] args) {  Test pt1, pt2;  pt1 = **new** Test(3, 3);  pt2 = **new** Test(4, 4);  System.***out***.print(pt1.x + pt2.x);  }  } | |
|  |  |  |
|  | **A** | 6 |
|  | **B.** | 34 |
|  | **C.** | 8 |
|  | **D.** | 7 |

|  |  |  |
| --- | --- | --- |
| **2.** | **分析如下Java程序的代码所示，则编译运行后的输出结果是（ C ）。（选择一项）** | |
|  | **public** **class** Test {  **int** count=9;  **public** **void** count1(){  count=10;  System.*out*.println("count1="+count);  }  **public** **void** count2(){  System.*out*.println("count2="+count);  }  **public** **static** **void** main(String[ ] args) {  Test t=**new** Test();  t.count1();  t.count2();  }  } | |
|  |  |  |
|  | **A** | count1=9;  count2=9; |
|  | **B.** | count1=10;  count2=9; |
|  | **C.** | count1=10;  count2=10; |
|  | **D.** | count1=9;  count2=10; |

|  |  |  |
| --- | --- | --- |
| **3.** | **以下语句中关于Java构造方法的说法错误的是（ B ）。（选择一项）** | |
|  |  |  |
|  | **A.** | 构造方法的作用是为创建对象进行初始化工作，比如给成员变量赋值 |
|  | **B.** | 一个Java类可以没有构造方法，也可以提供1个或多个构造方法 |
|  | **C.** | 构造方法与类同名，不能书写返回值类型 |
|  | **D.** | 构造方法的第一条语句如果是super()，则可以省略，该语句作用是调用父类无参数的构造方法 |

|  |  |  |
| --- | --- | --- |
| **4.** | **在Java中关于静态方法，以下说法中正确的是（ AD ）。（选择两项）** | |
|  |  |  |
|  | **A** | 静态方法中不能直接调用非静态方法 |
|  | **B.** | 非静态方法中不能直接调用静态方法 |
|  | **C.** | 静态方法可以用类名直接调用 |
|  | **D.** | 静态方法里可以使用this |

|  |  |  |
| --- | --- | --- |
| **5.** | **下列选项中关于Java中类方法的说法错误的是（ B C ）。（选择二项）** | |
|  |  |  |
|  | **A** | 在类方法中可用this来调用本类的类方法 |
|  | **B.** | 在类方法中调用本类的类方法时可直接调用 |
|  | **C.** | 在类方法中只能调用本类中的类方法 |
|  | **D.** | 在类方法中调用实例方法需要先创建对象 |

|  |  |  |
| --- | --- | --- |
| **6.** | **使用权限修饰符（ B ）修饰的类的成员变量和成员方法，可以被当前包中所有类访问，也可以被它的子类（同一个包以及不同包中的子类）访问。（选择一项）** | |
|  |  |  |
|  | **A** | public |
|  | **B.** | protected |
|  | **C.** | 默认 |
|  | **D.** | private |

|  |  |  |  |
| --- | --- | --- | --- |
| **7.** | **给出如下代码，如何使成员变量m被方法fun()直接访问（ C ）。（选择一项）** | | |
|  | **class** Test {  **private** **int** m;  **public** **static** **void** fun() {  }  } | | |
|  |  |  | |
|  | **A** | 将private int m 改为protected int m | |
|  | **B.** | 将private int m 改为public int m | |
|  | **C.** | 将private int m 改为static int m | |
|  | **D.** | 将private int m 改为int m | |
| **8.** | **以下关于继承条件下构造方法执行过程的代码的执行结果是（A ）。（选择一项）** | | |
|  | **class** Person {  **public** Person() {  System.*out*.println("execute Person()");  }  }  **class** Student **extends** Person {  **public** Student() {  System.*out*.println("execute Student() ");  }  }  **class** PostGraduate **extends** Student {  **public** PostGraduate() {  System.*out*.println("execute PostGraduate()");  }  }  **public** **class** TestInherit {  **public** **static** **void** main(String[] args) {  **new** PostGraduate();  }  } | | |
|  |  | |  |
|  | **A.** | | execute Person()  execute Student()  execute PostGraduate() |
|  | **B.** | | execute PostGraduate() |
|  | **C.** | | execute PostGraduate()  execute Student()  execute Person() |
|  | **D.** | | 没有结果输出 |

|  |  |  |
| --- | --- | --- |
| **9.** | **以下关于this和super关键字的说法错误的是（ BD ）。（选择二项）** | |
|  |  |  |
|  | **A.** | this关键字指向当前对象自身，super关键字指向当前对象的直接父类 |
|  | **B.** | 在main方法中可以存在this或super关键字，但不能同时存在。 |
|  | **C.** | this和super关键字都可以访问成员属性，成员方法和构造方法 |
|  | **D.** | 在一个类的构造方法中可以同时使用this和super来调用其他构造方法 |

|  |  |  |
| --- | --- | --- |
| **10.** | **给出下面的代码段，在代码说明//assignment x=a,y=b处写入如下（ D ）个代码是正确的。（选择一项）** | |
|  | **public** **class** Base {  **int** w, x, y, z;  **public** Base(**int** a, **int** b) {  x = a;  y = b;  }  **public** Base(**int** a, **int** b, **int** c, **int** d) {  // assignment x=a,y=b  w = d;  z = c;  }  } | |
|  |  |  |
|  | **A** | Base(a,b); |
|  | **B.** | x=a,y=b; |
|  | **C.** | this(a),this(b); |
|  | **D.** | this(a,b) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **11.** | **以下Java程序编译运行后的输出结果是（ A ）。（选择一项）** | | | |
|  | **class** HelloA {  **public** HelloA( ) {  System.*out*.println("HelloA");  }  { System.*out*.println("I'm A class");  }  **static** {  System.*out*.println("static A");  }  }  **public** **class** HelloB **extends** HelloA {  **public** HelloB( ) {  System.*out*.println("HelloB");  }  { System.*out*.println("I'm B class");  }  **static** {  System.*out*.println("static B");  }  **public** **static** **void** main(String[] args) {  **new** HelloB( );  }  } | | | |
|  |  |  | | |
|  | **A** | static A  static B  I'm A class  HelloA  I'm B class  HelloB | C. | static A  I'm A class  HelloA  static B  I'm B class  HelloB |
|  | **B.** | static A  static B  I'm A class  I'm B class  HelloA  HelloB | D | static A  static B  HelloA  HelloB  I'm A class  I'm B class |

|  |  |  |  |
| --- | --- | --- | --- |
| **12.** | | **下列选项中关于Java中super关键字的说法正确的是（ AD ）。（选择二项）** | |
|  | |  |  |
|  | | **A** | super关键字是在子类对象内部指代其父类对象的引用 |
|  | | **B.** | super关键字不仅可以指代子类的直接父类，还可以指代父类的父类 |
|  | | **C.** | 子类通过super关键字只能调用父类的方法，而不能调用父类的属性 |
|  | | **D.** | 子类通过super关键字可以调用父类的构造方法 |
| **13.** | **关于Java中的多态，以下说法不正确的为（ B ）。（选择一项）** | |
|  |  |  |
|  | **A** | 多态不仅可以减少代码量，还可以提高代码的可扩展性和可维护性 |
|  | **B.** | 把子类转换为父类，称为向下转型，自动进行类型转换 |
|  | **C.** | 多态是指同一个实现接口，使用不同的实例而执行不同的操作 |
|  | **D.** | 继承是多态的基础，没有继承就没有多态 |

|  |  |  |
| --- | --- | --- |
| **14.** | **编译运行如下Java代码，输出结果是（ D ）。（选择一项）** | |
|  | **class** Base {  **public** **void** method(){  System.***out***.print ("Base method");  }  }  **class** Child **extends** Base{  **public** **void** methodB(){  System.***out***.print ("Child methodB");  }  }  **class** Sample {  **public** **static** **void** main(String[] args) {  Base base= **new** Child();  base.methodB();  }  } | |
|  |  |  |
|  | **A** | Base method |
|  | **B.** | Child methodB |
|  | **C.** | Base method  Child methodB |
|  | **D.** | 编译错误 |

|  |  |  |
| --- | --- | --- |
| **15.** | **在Java中，关于引用数据类型的类型转换说法正确的是（ A B ）。（选择二项）** | |
|  |  |  |
|  | **A** | 引用数据类型的类型转换有向上转型和向下转型 |
|  | **B.** | 向下转型，必须转换成其真实子类型，而不能随意转换 |
|  | **C.** | 向下转型是自动进行的，也称隐式转换 |
|  | **D.** | 向上转型可以使用instanceof操作符来判断转型的合法性 |

|  |  |  |
| --- | --- | --- |
| **16.** | **给定如下Java程序，Test类中的四个输出语句输出结果依次是（C ）。（选择一项）** | |
|  | **class** Person {  String name="person";  **public** **void** shout(){  System.*out*.println(name);  }  }  **class** Student **extends** Person{  String name="student";  String school="school";  }  **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  Person p=**new** Student();  System.*out*.println(p **instanceof** Student);  System.*out*.println(p **instanceof** Person);  System.*out*.println(p **instanceof** Object);;  System.*out*.println(p **instanceof** System);  }  } | |
|  |  |  |
|  | **A** | true,false,true,false |
|  | **B.** | false,true,false,false |
|  | **C.** | true,true,true,编译错误 |
|  | **D.** | true,true,false,编译错误 |

1. **判断题**
2. 类可以看成一类对象的模板，对象可以看成该类的一个具体实例。( √ )
3. 如果没有为类中的某些成员赋初始值，Java会为类成员赋予固定的初始值，如数值变量的值为0，布尔变量的值为true，未初始化的引用为null。（ × ）
4. Java中所有的变量，不管是成员变量还是局部变量，在使用前都必须进行初始化。（ ×）
5. 在Java中对象可以赋值，只要使用赋值运算符即可，相当于生成了一个各属性与赋值对象相同的新对象。（ × ）
6. System.out.println(“Hello java!”)中out是System类的一个静态成员变量。（ √ ）
7. 构造方法用于创建对象，一般不允许有任何返回值，因此需要在构造方法返回类型处标注为void。（ × ）
8. Java语言为所有的Java程序自动导入包“java.lang”，因此Java程序可以直接用“java.lang”中的类和接口。（ √ ）
9. 使用public修饰的成员属性和方法可以被当前项目中所有包的所有类访问。( √ )
10. 类的方法通常设为public，而类的实例变量一般也设为public。（ × ）
11. 与未加访问控制符的缺省情况相比，public和protected修饰符扩大了类及其属性和方法的被访问范围，private修饰符则缩小了这种范围。（ √ ）
12. 访问权限是private的变量，只能在本类和与本类同一个包中的其他类使用。（ × ）
13. 如果定义一个类时没有使用extends，则它的父类是java.lang.Object。（ √ ）
14. 对子类实例化对象需要调用超类的构造函数进行初始化工作，完成对子类中的超类实例变量的初始化，则子类可以继承超类的构造函数。（ × ）
15. 在Java中任何一个子类的构造方法都必须调用其父类的构造方法（包括隐式调用），并且调用父类的构造方法的语句必须是子类构造方法的第一条语句。（ √ ）
16. Java中的类不允许多重继承，一个类只能有有一个直接父类。 （√）
17. Object类有一个public方法是toString( )，一个对象通过调用该方法可以获得该对象的字符串表示。（ √ ）
18. 父类Person中定义了一个private void show()的方法，那么子类要重写这个方法时，方法的访问修饰符可以是默认的，protected或public。（ × ）
19. 运算符“==”用于比较引用时，如果两个引用指向内存同一个对象，则返回true。（ √ ）
20. 将子类对象赋给父类引用变量，称为向下转型，将无法访问子类特有的方法。( × )

继承是多态的基础，没有继承就没有多态。( √ )

1. **编程题**
   1. 某公司要开发名为”我爱购物狂”的购物网站，请使用面向对象的思想设计描述商品信息(封装)

要求：

1. 分析商品类别和商品详细信息属性和方法，设计商品类别类和商品详细信息类
2. 在商品详细信息类中通过属性描述该商品所属类别
3. 设置属性的私有访问权限，通过公有的get,set方法实现对属性的访问
4. 编写测试类，测试商品类别类和商品详细信息类的对象及相关方法（测试数据信息自定）
5. 创建包info—存放商品类别类和商品详细信息类，创建包test—存放测试类

参考分析思路:

商品类别类：

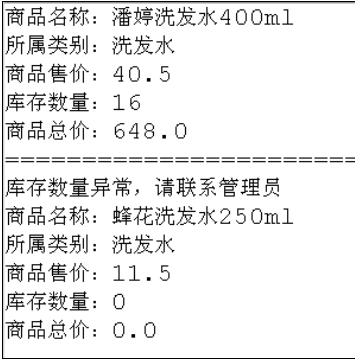
属性：类别编号，类别名称

商品详细信息类：

属性：商品编号，商品名称，所属类别，商品数量（大于0），商品价格（大于0），

方法：盘点的方法，描述商品信息。内容包括商品名称，商品数量，商品价格，现在商品总价以及所属类别信息

运行效果图:



* 1. 请用面向对象的思想，设计自定义类描述演员和运动员的信息(继承)

设定

1. 演员类:

属性包括：姓名，年龄，性别，毕业院校，代表作

方法包括：自我介绍

1. 运动员类：

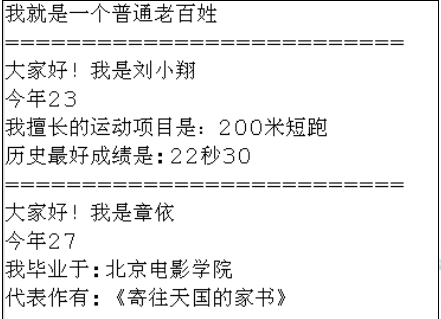
属性包括：姓名，年龄，性别，运动项目，历史最好成绩

方法包括：自我介始

要求

1. 分析演员和运动员的公共成员，提取出父类—人类
2. 利用继承机制，实现演员类和运动员类
3. 编写测试类，分别测试人类，演员类和运动员类对象及相关方法
4. 定义名为act的包存人类，演员类，运动员类和测试类

运行效果

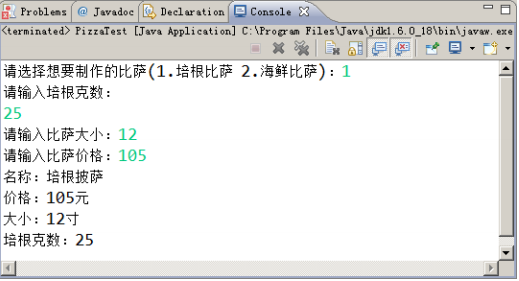


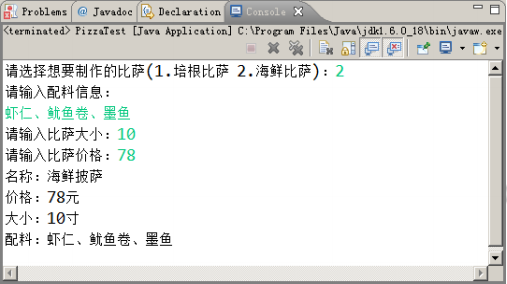
* 1. 编写程序实现比萨制作。需求说明编写程序，接收用户输入的信息，选择需要制作的比萨。可供选择的比萨有：培根比萨和海鲜比萨。(多态)

实现思路及关键代码

1. 分析培根比萨和海鲜比萨
2. 定义比萨类
3. 属性：名称、价格、大小
4. 方法：展示
5. 定义培根比萨和海鲜比萨继承自比萨类
6. 定义比萨工厂类，根据输入信息产生具体的比萨对象

程序运行结果如图所示





4.编写程序描述兔子和青蛙（抽象+接口）

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）

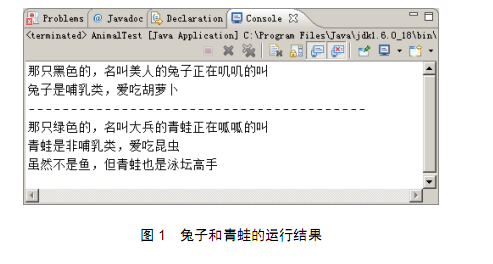
方法：吃饭，发出叫声

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。



5.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。