**类和对象作业**

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\_\_\_\_堆内存\_\_\_\_\_\_\_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（ this ）关键字。
6. 使用（ this ）关键字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（ static ）关键字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\_\_\_\_\_\_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在的包。
11. （ long ）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。
12. **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);  }  **publicvoid** count2(){  System.*out*.println("count2="+count);  }  **publicstaticvoid** 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（ D ）。（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（ BD ）。（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（ AC ）。（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 |

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truefalse，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. 构造方法的作用是两个：一个构造类的对象，另一个作用是初始化对象的属性。（ √）
9. 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。（ √ ）
10. 构造方法的名称必须保持跟类名一致。（ √ ）
11. **简答题**
    1. 面向过程和面向对象的区别和联系。

面向对象的内部也是面向过程的设计

* 1. 类和对象的关系

类是对象的抽象

* 1. 局部变量、成员变量、静态变量分别怎么声明?

局部变量直接声明在方法中，循环中，等

成员变量声明在类中，方法外

静态变量在成员变量前加static

* 1. 构造方法的作用和特征

构造方法用来开辟堆内存空间，并为成员变量初始化

特征是，可以不写，没有返回值，名称与类名相同

* 1. this关键字的作用和用法。

This指代当前对象，用于区别同名，如set方法中赋值操作

* 1. 简述static关键字的作用。

提示：从static可以修饰变量，方法，代码块，内部类四个方面来回答。

Static声明的资源从类加载时就存在，他不能访问还没有存在的东西，就是非静态资源

* 1. 栈的特点是?存放什么内容?堆得特点是?存放什么内容?

栈的存储特点是先进后出，存取速度快，容量小

* 1. 方法区的特点是?存放什么内容? 方法区和堆有什么共同点?

方法区存储 ：

类信息（class信息）、常量、静态变量、即时编译器编译后的代码

即时编译器编译后的代码：

我们知道Java是解释型语言，速度肯定是不如C这种编译型的，那么很明显的一个可行的优化就是把部分热点字节码也直接编译成可执行的机器码（这个就是即时编译器编译后的代码）

方法区实际也是堆，只是用于存储类，常量相关的信息！

用来存放程序中“永远不变或唯一”的内容.（类信息，静态变量，字符串常量等）

GC会回收堆的内存，但是不会回收方法区

* 1. 如果同时导入：import java.util.Date; import java.sql.Date; 我们在程序中怎么区分?

一个导包，一个写全限定名，或者两个都写全限定名

1. **编码题**
2. 图形用户界面

   低可信度描述已自动生成请定义一个交通工具(Vehicle)的类其中有: 属性：速度(speed)、体积(size)等，方法：移动(move())、设置速度(setSpeed(int speed))、加速 speedUp()、减速 speedDown()等。最后在测试类 Vehicle 中的 main()中实例化一个交通工具对象并通过方法给它初始化 speed,size 的值并且通过打印出来。另外调用加速、减速的方法对速度进行改变。
3. 编写 Java 程序用于显示人的姓名和年龄。定义一个人类Person。 该类中应该有两个私有属性： 姓名 （name） 和年龄 （age） 。定义构造方法用来初始化数据成员。再定义显示（display()） 方法将姓名和年龄打印出来。在 main 方法中创建人类的实例然后将信息显示。
4. 文本

   描述已自动生成电脑萤幕截图

   描述已自动生成定义一个圆类——Circle，在类的内部提供2个属性：半径(r) 和圆心Point point，同时 提供 两个 方 法 ： 计算 面积 （ getArea() ） 和 计算 周长（getPerimeter()） 。 通过两个方法计算圆的周长和面积并且对计算结果进行输出。最后定义一个测试类对 Circle 类进行使用。
5. 为上述Cricle类添加一个方法，计算一个点（Point对象）是否在圆（Cricle对象）内，并写程序验证。

文本

描述已自动生成

1. 构造方法与重载：定义一个网络用户类，要处理的信息有用户 ID、用户密码、 email 地址。在建立类的实例时把以上三个信息都作为构造函数的参数输入，其中用户 ID 和用户密码时必须缺省时 email地址是用户 ID 加上字符串"@gameschool.com"。

文本

描述已自动生成

1. **可选题**
   1. 定义一个类 Calculaion， 其中包含四个方法： 加（add()） 、 减（sub()） 、乘（times()） 和除（div()） 。 创建一个具有 main()函数的类。 在 main()函数中创建一个 Calculation 的实例对象并对其中的方法进行调用。

文本

描述已自动生成

* 1. 定义一个类 Draw ， 在 类 中 提供 3 个 方法：输出直角三角形（drawTrian()） 、 输出矩形（drawRec()） 及平行四边形（drawPra()） 。通过方法可以输出由“\*” 组成的一个图形。 同时在类中包含两个属性：星号的个数（count） 、行数（lines） 。最后在测试类中进行调用。

电脑萤幕画面

描述已自动生成

* 1. 创建一个空调，并调用制冷、制热、通风功能。空调包含的属性有品牌、匹数、温度，功能有加热、制冷、通风等功能。

文本

描述已自动生成

* 1. 定义一个表示学生信息的类 Student，要求如下：

（1）类 Student 的属性如下：

sNO 表示学号； sName 表示姓名； sSex 表示性别； sAge 表示

年龄；sJava：表示 Java 课程成绩。

（2）类 Student 带参数的构造方法：

在构造方法中通过形参完成对成员变量的赋值操作。

（3）类 Student 的方法成员：

getNo（）：获得学号；

getName（）：获得姓名；

getSex（）：获得性别；

getAge（）获得年龄；

getJava（）：获得 Java 课程成绩

（4） 根据类 Student 的定义，创建五个该类的对象，输出每个学生的信息，计算并输出这五个学生 Java 语言成绩的平均值，以及计算并输出他们 Java 语言成绩的最大值和最小值。

文本

描述已自动生成

* 1. 定义自己的类：手机类（没有标准答案，按照自己的想法定义，一定要包含属性和方法）。

文本

描述已自动生成

* 1. 简答题
     1. 使用面向对象的方式分析一个事例(模仿老师的解放战争分析)

数据就像是菜

封装就像是去餐厅，不是在自己家，触碰不到菜

你去餐厅是不能自己去后厨端菜的

get就像是服务员上菜，他帮你去后厨拿'数据' // 你桌上的菜=餐厅.服务员上菜( ) ;

set就像是服务员点菜，他帮你改变后厨的'数据' // 餐厅.服务员点菜( ) ;

类中的方法就像是餐厅提供的一些服务，比如说show方法

就像是回转寿司，餐厅是可以直接处理你的菜

他选择把菜放到履带上 // 餐厅.服务员把菜放到转盘上( ) ;

也可以达成让你看到你点的菜的目的

构造方法就像是我没看菜单，但我知道我想吃什么

比如每次去肯德基只吃全家桶

就可以// new 肯德基（点菜全家桶）;

* + 1. 局部变量使用之前，必须要手动初始化吗?

是的

* + 1. 如果不手动指定成员变量的值，系统将会自动初始化。那么初始化的规则是?

有零给零，有空给空，boolean给false

* + 1. 成员变量从属于谁? 静态变量又叫什么以及从属于谁? 局部变量从属于谁?

属于对象，静态变量属于类，局部变量属于他的外层（方法等）

* + 1. 构造方法有没有返回值?详细用文字描述返回值问题。

没有，因为它只是完成初始化的工作，或者说它返回了这个对象实例？

* + 1. 构造方法如何被调用?

New的时候被调用

* + 1. 构造方法中能不能有return语句?

不能，因为它没有返回值

* + 1. 系统一定会给我们添加无参数的构造方法吗? 请详细解释。

如果没有，就会。不给构造方法的话，我们又没写的话，那永远创造不出对象，要么写个抽象类，或者接口，定义为class了，就要有构造方法

* + 1. 下面的代码有什么问题:

|  |
| --- |
| **class** Computer {  **int**price;  **int**type;  String brand;  **publicvoid** start() {  System.***out***.println("启动中....");  }  Computer(**int**\_price, **int**type, String \_brand) {  // this.price = price;  price = \_price;  type = type;  brand = \_brand;  }  } |

1.没加空格

2.没加this关键字以示区分

* + 1. 构造方法能不能重载?

当然能

* + 1. 一个构造方法调用另一个构造方法怎么调用? this()，这样的调用方式必须位于第一句吗?

是的

* + 1. static变量在内存中放置在哪个区? static变量和方法为什么被称为类变量和类方法?可以被该类所有对象共享吗?

在静态存储区，因为它属于类，当然可以

* + 1. 静态初始化块和main方法哪个先被执行?

静态初始化会先执行

* + 1. 画出如下程序的内存结构：

|  |
| --- |
| **class** Engine {  **int** speed;  **int** weight;  }  **class** Car {  **staticint** tyreNum = 4;  Engine engine;  String color; // char sequence :字符序列  **void** changeColor(String c) {  color = c;  }  **void** showColor() {  System.out.println("我的颜色是：" + color);  }  }  //测试类和对象  **publicclass** TestObject {  **publicstaticvoid** main(String[] args) {  Car c1 = **new** Car();  c1.changeColor("红色");  c1.showColor();  System.out.println(Car.tyreNum);  System.out.println(c1.tyreNum);  Car c2 = **new** Car();  Engine e = **new** Engine();  e.speed = 1000;  e.weight = 10;  c2.engine = e;  c2.color = "黑色";  c2.tyreNum = 10;  System.out.println(c1.tyreNum);  }  } |

图示

描述已自动生成

* + 1. package的两个作用是什么?

打包，避免重名

保护隐私，有些东西只能同包访问

* + 1. 增加package以后，我们在DOS下编译怎么做?

多加一条-d命令用于编译

运行多加-cp命令

* + 1. import是用于导入包还是导入类?

包

* + 1. import java.util.\*; 会不会降低程序运行速度?为什么?

会，因为导的多

* + 1. import static 静态导入的作用是导入类还是导入类的静态属性和静态方法?

应该是属性和方法，因为静态导入不需要写类名

* + 1. javadoc注释怎么写?

/\*\*

\*摘要，描述功能

\*详细描述

\*描述作者，参数等

\*/

* + 1. java项目的API文档如何生成?请将步骤写出。

写好描述，点击导出javadoc

* + 1. eclipse可以开发java，可以开发c++吗？

可以

* + 1. eclilpse中的debug模式有什么好处?

便于我们一步步排查错误

* + 1. eclipse中运行一个java程序，如何操作?

点击上面的小三角，或者选择类名右键运行Junit等单元测试

* + 1. eclipse需要安装吗? 使用eclispe需要安装JDK吗？

当然需要，但是有些eclipse版本包含JDK

* + 1. 假如我找不到package explore视图了，怎么样才能让它出现?

Window——show view