**Java程序设计**

**（面向对象基础篇）**

**面向对象的三大特征：**

1）封装性：一是把对象的属性和行为看成一个密不可分的整体，封装在对象之中；

二是把不需要让外界知道的信息隐蔽起来，对外不可见。

2）继承性： 被继承的称为父类或超类，经继承产生的类称为子类或派生类。子类继承

了父类的所有成员，并可以相应的增加自己一些新的成员。若一个子类只允许继承一个父类，称为单继承；若一个子类允许继承多个父类，称为多继承。Java语言通过接口（interface）的方式来弥补Java不支持多继承的缺点。

3）多态性：多态是允许程序中出现重名现象。Java语言中含有方法重载、覆写与对象多态两种形式的多态。方法重载是指在一个类中定义了同名的方法，但方法的参数类型或参数的个数不同。方法覆写是指在子类中定义了与父类中同名的方法。对象多态是指子类对象可以与父类对象进行转换，并且根据其使用子类的不同完成的功能也不同。

**类与对象的关系：**

类是表示客观世界某类群体的一些基本特征抽象。对象是表示一个个具体的东西。要使用一个类，就必须产出对象。

**类的定义：**

1）类是由属性和方法组成的。属性就是变量，方法就是一些操作的行为。

2）类名称的首字母必须大写。

**对象的创建与访问：**

1. 创建对象并实例化： 类名称 对象名称=null； 对象名称=new 类名称( )；

类名称 对象名称=new 类名称( )；

1. 对象访问属性或方法： 对象名称 . 属性名；

对象名称 . 方法名（ ）；

**对象的内存分配过程：**

1. 创建对象开辟栈内存空间，栈内存空间存放的是对象的名字，更准确的说是存放堆内存空间的访问地址；

使用new关键字实例化对象开辟堆内存空间，堆内存空间存放对象的具体属性内容。

2）对象必须被实例化之后才能使用，没有实例化的对象是不能被使用的。

**属性的封装与访问：**

类中的所有属性可以使用private关键字进行封装，封装的属性必须通过setter和getter

方法设置和取得，并且可以在setter方法中加入验证代码对设置进去的值进行检查。

**构造方法的定义格式、调用时机与重载：**

1. 构造方法可以在实例化对象时给属性赋值，构造方法的名称必须与类名称一致，

构造方法中不允许有任何返回值的声明，也不允许有return语句。

1. 只要是类就必须存在构造方法。在Java中如果一个类中没有明确的声明一个构造方法，则在编译时会直接生成一个无参数的、什么都不做的构造方法，即类中至少会存在一个构造方法。
2. 构造方法重载时，所有的构造方法按照参数的个数由多到少，或者由少到多排列。

（编程风格）

**匿名对象的基本概念：**

匿名对象是指没有名字的对象，匿名对象只开辟堆内存空间。一般匿名对象只使用一次，使用之后就等待被垃圾收集机制回收。

**类的基本设计思路：**

1. 根据要求写出类所包含的属性
2. 对所有属性进行封装
3. 为封装的属性设置setter和getter方法
4. 如果需要可以加入若干构造方法
5. 再根据要求添加相应的方法
6. 类中所有方法不要直接输出，而是交给被调用处输出

**String类两种实例化对象方式的区别：**

1. 直接赋值：只会开辟堆内存空间，只要是以后声明的字符串内容相同，则不会开辟

新的堆内存空间。

1. 构造方法赋值：会开辟两块堆内存空间，其中有一块成为垃圾，因为一个字符串本

身就是一个匿名对象。

**String类对象内容的比较：**

1）“==”是用来进行地址值比较，适用于进行数值型相等判断

2）equals( )方法是由String类提供的一个方法，专门用来进行字符串内容的比较

**字符串内容的不可改变性：**

String类对象内容的改变实际上是通过内存地址的“断开-连接”变化完成的，而本身字符串中的内容并没有发生任何的变化。

**String类中的常用方法：**

1. 字符串与字符数组的转换 toCharArray( )
2. 从字符串中取出指定位置的字符 charAt( )
3. 字符数组与byte数组的转换 getBytes( )
4. 取一个字符串的长度 length( )
5. 查找一个指定的字符串是否存在 indexOf( )
6. 去掉字符串左右两边的空格 trim( )
7. 字符串截取 substring( )
8. 按照指定的字符串拆分字符串（将字符串转变为字符串数组） split( )
9. 字符串的大小写转换 toUpperCase( ) toLowerCase( )
10. 判断字符串是否以指定的字符串开头或结尾 startsWith( ) endsWith( )
11. 不区分大小写进行字符串的比较 equalsIgnoreCase( )

15）将一个指定的字符替换成其他的字符串 replaceAll( )

**引用传递：**

引用传递就是将堆内存空间的使用权交给多个栈内存空间。每个栈内存空间都可以对堆内存中的内容进行修改。

一个栈内存空间只能指向一个堆内存空间。但一个堆内存空间可以被多个栈内存空间指向。

**String类的引用传递：**

由于字符串的内容一旦声明便不可更改，所以字符串的引用传递并不会发生改变，但如果String类是作为一个类中的属性存在的话，则引用传递后会发生改变。

**接受本类的引用：**

在对象的引用传递上也可以在一个类中接收自己本类对象的实例。而且接收完之后，可以方便地通过此对象直接进行本类中封装属性的访问。

**引用传递下的一对一关系：**

使用引用传递还可以表示出生活中的一对一关系。例如一个人有一本书，一本书属于一个人。

**this关键字的作用：**

1. 表示本类中的属性或方法
2. 调用本类中的构造方法。使用this调用本类中的构造方法时只能放在构造方法的第一行，且类中至少存在一个构造方法是不使用this调用其他构造方法的。
3. 表示当前对象。当前对象就是正在调用方法的对象

**对象的比较：**

只有是同一个类的对象才能进行比较。比较两个对象时，需要对对象中的每个属性进行比较。首先进行地址的比较，如果地址一样，则对象肯定相等，再判断所有的属性的内容是否相等，如果所有属性都相等，就证明了这两个对象相等。

**static关键字：**

1. 使用static声明的属性称为全局属性或静态属性，有时也被称为类属性；

使用static声明的方法有时也被称为类方法；

使用static声明的属性或方法可以由类名称直接调用。

1. 非static声明的方法可以调用static声明的属性或方法，但static声明的方法不可以调用非static声明的属性或方法。即static方法只能调用static属性或static方法

**Java中常用的内存区域：**

1. 栈内存空间：保存所有对象的名称（更准确的说是保存了堆内存空间的访问地址）
2. 堆内存空间：保存每个对象的具体属性内容
3. 全局数据区：保存static类型的属性
4. 全局代码区：保存所有的方法定义

**理解main方法：**

1. public: 表示此方法可以由外部类调用
2. static：表示此方法可以由类名称直接调用
3. void：主方法是程序的起点，所以不需要任何的返回值
4. main：系统规定的默认方法名称，执行时默认找到main（）方法
5. String args[]：对象数组，表示运行时的参数，参数传递形式为：java 类名称 参数

如果在输入的时候希望参数中有空格，则在输入参数时直接加上” ”即可。System.exit( )中，只要exit（）方法中设置一个非零的数，则系统执行到此语句之后退出程序。

**代码块：**

代码块是指使用“{ }”括起来的一段代码，代码块可以分为4种：普通代码块、构造代码块、静态代码块、同步代码块。

1. 普通代码块是指直接在方法或语句中定义的代码块，主要作用是进行程序分割使用
2. 构造代码块是指直接在类中定义的代码块。构造代码块优先于构造方法执行，并且每次实例化对象时都会执行构造代码块
3. 静态代码块是使用static关键字声明的代码块，静态代码块优先于主方法执行，在类中定义的静态代码块会优先于构造代码块执行。但静态代码块只执行一次。
4. 同步代码块将在多线程部分讲解

**构造方法私有化的访问：**

1. 构造方法私有化后，在外部创建的对象无法实例化，这时只需通过在类的内部创建本类的对象并实例化，并使用static关键字进行声明，在外部就可以通过类名称调用此对象为在外部创建的本类的对象进行实例化操作。但如果将本类中声明的对象进行了封装，就需要定义一个static关键字修饰的getter方法，在外部通过类名称调用此方法返回对象为外部创建的本类的对象进行实例化。
2. 只要将构造方法私有化了，就可以控制实例化对象的产生。

**单例设计模式（Singleton）：**

将构造方法私有化后，通过在本类中定义的一个对象为外部中创建的多个对象实例化时，

虽然在外部定义了多个对象，但都只使用了一个本类对象对它们进行实例化。在设计模式中将这样的设计称为单例设计模式（Singleton）

**对象数组：**

1. 对象数组就是指一组相关对象的集合。
2. 对象数组中的每一个对象必须分别进行实例化操作。
3. 对象数组也分静态初始化和动态初始化。

**内部类：**

1）内部类是在类的内部定义的类。内部类的存在会破坏程序的基本结构，因为类是由属性和方法组成的。但内部类的唯一好处就是可以方便的访问外部类中的私有属性。

2）使用static声明的内部类则变成外部类，但是使用static声明的内部类不能访问外部类中的非static属性

3）在外部访问内部类，首先要找到外部类的实例化对象，通过外部类的实例化的对象去实例化内部类对象，再通过内部类对象去访问内部类中的属性或方法。

**外部类 . 内部类 内部类对象=外部类实例化对象 . new 内部类( );**

1. 在方法中定义的内部类不能直接访问方法中的参数，如果方法中参数想要被内部类

访问，则参数前面必须加上final关键字，但在JDK1.8之后此规则被打破

**系统登录实例讲解：**

使用初始化参数的方式编写一个简单的系统登录。首先需要判断输入的参数个数是否合法，如果不合法，则提示提示错误并退出。如果正确输入参数则可以进行验证。主方法中的代码尽可能减少。

**单向链表实现：**

建立一个单向链表，节点操作主要有增加数据、打印数据、查找数据、删除数据操作。如果要删除节点的话，则直接修改上一个节点的引用即可。

1. 编写并测试一个代表地址的Address类，地址信息由国家、省份、城市、街道、邮政编码组成，编写测试类并输出信息。
2. 编写程序，统计出字符串“want you to know one thing”中字母n和字母o的出现次数。
3. 设计一个表示用户User类，类中的变量有用户名、口令和记录用户个数的变量，定义类的三个构造方法（无参、为用户名赋值、为用户名和口令赋值）、获取和设置口令的方法和返回类信息的方法。

4. 编写一个公司员工类。

（1）数据成员：员工号、姓名、薪水、部门

（2）方法：

利用构造方法完成设置信息。

* 无参，则均为空值。
* 单参，只传递员工号，则员工姓名：无名氏，薪水：0，部门：未定。
* 双参，传递员工号，姓名，则员工薪水为1000，部门：后勤。
* 4参，传递员工号、姓名、部门、薪水。

显示信息。

5. 构造一个银行账户类，类的构成包括如下内容：

（1）数据成员用户的账户名称、用户的账户余额（private数据类型）。

（2）方法包括开户（设置账户名称及余额），利用构造方法完成。

（3）取款和存款

（4）查询余额

6. 声明一个图书类，其数据成员为书名、编号（利用静态变量实现自动编号）、书价，并拥

有静态数据成员册数，记录图书的总册数，在构造方法中利用此静态变量为对象的编号赋值，在主方法中定义对象数组，并求出总册数。

7. 字符串操作：

1. 将字符串“helloworld”变为字符数组，并输出；

将字符数组“helloworld”变为字符串，并输出；

将字符数组“helloworld”前五个字符（部分字符）变为字符串，并输出。

2）从字符串“Java技术学习班20070326”中提取字符‘学’，并输出。

3）求字符串“MLDNJAVA”的长度，并输出。

4）从字符串“java语言程序设计”中查找字符串“va”是否存在；

从字符串“java语言程序设计”第五个位置开始查找字符串“va”是否存在。

5）去掉字符串“ hello ”中所有的空格后输出字符串。

6）截取字符串“helloworld”中的字符串“world”并输出；

截取字符串“helloworld”中字符串“owo”并输出。

1. 按“o”拆分字符串“helloworld”，并输出；

将字符串“helloworld”按每一个字符拆分，并输出。

1. 将字符串“hello”与字符串“Hello”进行不区分大小写比较与区分大小写比较，输出布尔值。
2. 将字符串“hello”装换为大写字母并输出；

将字符串“HELLO”装换为小写字母并输出。

1. 判断字符串“hello”是否以字符串“he”开头和结尾；
2. 将字符串“JAVA语言程序设计”中的字符串“JAVA”替换为字符串“java”。

8. 编写Person类，包括姓名和年龄属性，创建两个该类对象并判断两对象是否相等。

9. 编写一个一对一关系，一个人有一本书，一本书属于一个人，从书可以找到人，从人也

可以找到书。

10. 编写一个系统登录的实例。

11. 建立一个单向链表，主要操作有增加节点、打印节点、查询节点和删除节点。