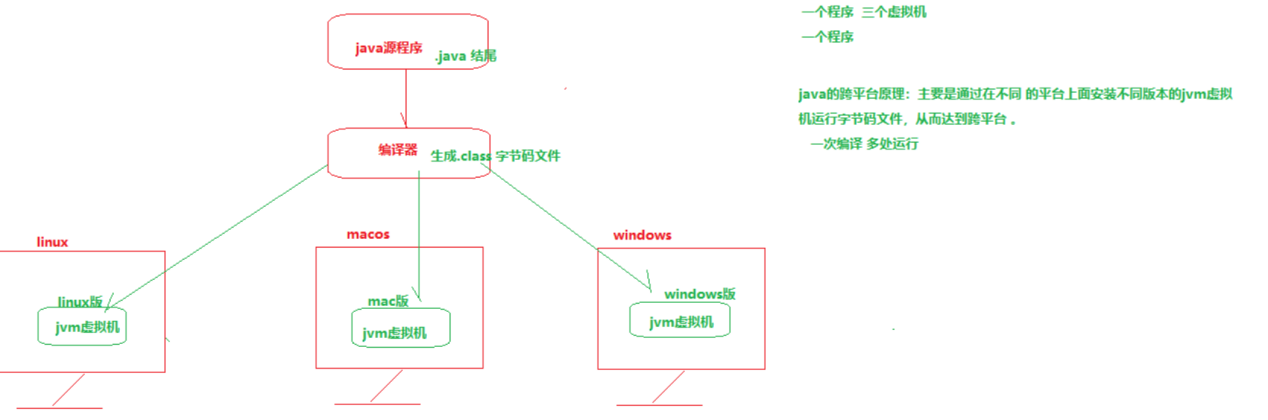
## 1.简单讲一下java的跨平台原理

由于每个操作系统（windows,liunx等）支持的指令集不是完全一致的，就会让我们的程序在不同的操作系统上要执行不同程序代码。Java开发了适用于不同操作系统及位数的java虚拟机来屏蔽各系统之间的差异，提供统一的接口。对于我们java开发者而言，你只需要在不同的系统上安装对应的不同java虚拟机、这时你的java程序只要遵循java规范，就可以在所有的操作系统上面运行java程序了。

Java通过不同的系统、不同版本、不同位数的java虚拟机(jvm),来屏蔽不同的系统指令集差异而对外体统统一的接口(java API),对于我们普通的java开发者而言，只需要按照接口开发即可。如果我系统需要部署到不同的环境时，只需在系统上面按照对应版本的虚拟机即可。



2.面向对象的特征有哪些方面

封装：

将对象封装成一个高度自治和相对封闭的个体，对象状态（属性）由这个对象自己的行为（方法）来读取和改变。

作用：保证内部数据的安全性，即外部类不能随意访问类中的成员变量（达到相应的访问权限课访问）

步骤：类要public修饰，私有化成员变量，提供公共无参构造方法，提供合理的getXxx()和setXxx(数据类型 变量名)方法

继承

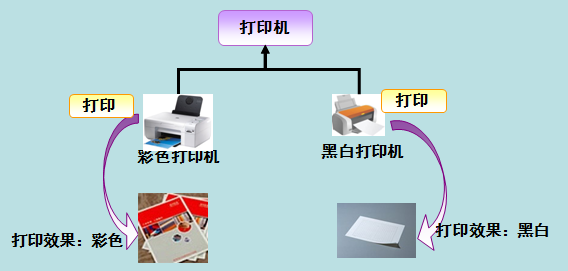
即在定义和实现一个类的时候，可以在一个已经存在的类（父类）的基础之上来进行，把这个已经存在的类所定义的内容作为自己的内容，并可以加入若干新的内容，或修改原来的方法使之更适合特殊的需要

作用：代码复用，提高开发效率和代码的拓展性

多态

多态是同一个行为具有多个不同表现形式或形态的能力。

多态就是同一个接口，使用不同的实例而执行不同操作，如图所示：



简单来说就是一个引用变量倒底会指向哪个类的实例对象，该引用变量发出的方法调用到底是哪个类中实现的方法，在编译时时不能确定的，必须在程序运行期间才能决定

抽象

找出一些事物的相似和共性之处，然后将这些事物归为一个类，这个类只考虑这些事物的相似和共性之处，并且会忽略与当前主题和目标无关的那些方面，将注意力集中在与当前目标有关的方面。 就是把现实生活中的对象，抽象为类。

注意：原则：回答比较抽象问题的时候，要举例说明

## 有了基本数据类型，为什么还需要包装类型

因为java是一个面向对象的语言，然而基本数据类型不具备面向对象的特性，当我们把基本数据类型包装成包装类型之后，它就具有了面向对象的特性。而且，在往ArrayList、HashMap这些容器传数据的时候，基本类型int和double是传输不进去的，因为容器都是装object类型的，所以需要转为包装类型进行传输。  
 每一个基本数据类型都有对应的包装类型，这里要解释下装箱和拆箱方法。  
 装箱：把基本数据类型包装成对应的包装类型，如：Integer .valueOf(1)

自动装箱：在编译的时候会调用相应的方法来进行装箱：Integer i = 1;会调用Integer .valueOf方法来装箱  
 拆箱：将包装类型拆成对应的基本数据类型，如：Integer i = 1;

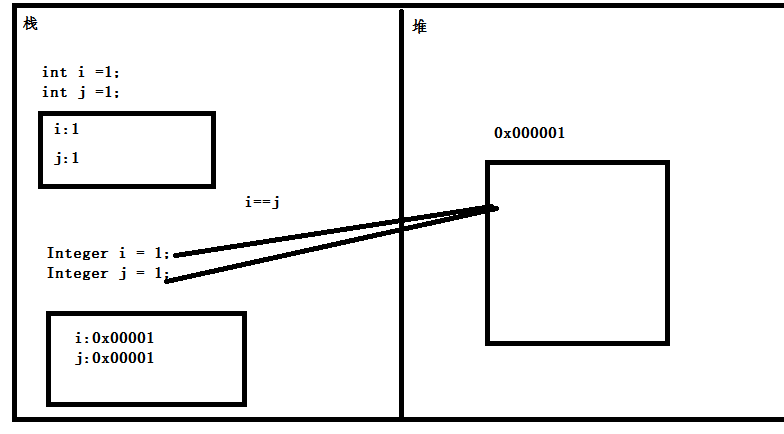
自动拆箱：在编译的时候自动调用相应的方法进行拆箱，如：int j = i;//自动拆箱//int j = i=intValue();手动拆箱

## 4.说一下"=="和equals方法究竟有什么区别

==用来判断两个变量之间的的值是否相等。变量就可以分为基本数据类型变量，引用类型。

如果是基本数据类型的变量直接比较值，引用类型要比较对应的引用的内存的首地址。

equals 用来比较两个对象长得是否一样。判断两个对象的某些特征是否一样。实际上就是调用对象的equals方法进行比较。



## 5.讲一下String和StringBuilder的区别(final)？

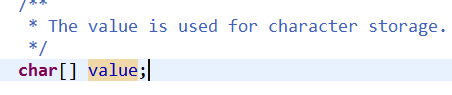
1.在java中提供三个类String ，StringBuillder， StringBuffer来表示和操作字符串。字符串就是多个字符的集合。

String是内容不可变的字符串。String底层使用了一个不可变的字符数组(final char[])



String str =new String(“bbbb”);

StringBuillder StringBuffer,是内容可以改变的字符串。StringBuillder ，StringBuffer底层使用的 可变的字符数组（没有使用final来修饰）



2.最经典就是拼接字符串。

String进行拼接.String c = “a”+”b”;

StringBuilder或者StringBuffer

StringBuilder sb = new StringBuilder(); sb.apend(“a”).apend(“b”)

拼接字符串不能使用String进行拼接，要使用StringBuilder或者StringBuffer

3.StringBuilder是线程不安全的，效率较高.而StringBuffer是线程安全的，效率较低。

## 6.StringBuffer和StringBuilder的区别?

StringBuffer：字符串变量，线程安全，效率低。适用多线程下在字符缓冲区进行大量操作的情况。

StringBulider：字符串变量，线程不安全，效率高。适用于单线程下在字符缓冲区进行大量操作

## 7.讲一下java中的集合?

Java中的集合分为value，key--vale(Conllection Map)两种。

存储值有分为List 和Set.

List：是有序的，可以重复的。

Set：是无序的，不可以重复的。根据equals和hashcode判断，也就是如果

一个对象要存储在Set中，必须重写equals和hashCode方法。

存储key-value的为map.

## 8.讲一下HashMap哈HashTable的区别?HashTable和ConcurrentHashMap的区别?

相同点：

HashMap和HasheTalbe都可以使用来存储key--value的数据。

区别：

1.HashMap是可以把null作为key或者value的，而HashTable是不可以的。

2.HashMap是线程不安全的，效率较高。而HashTalbe是线程安全的，效率较低。

？我想线程安全但是我又想效率高？

通过把整个Map分为N个Segment（类似HashTable），可以提供相同的线程安全，但是效率提升N倍，默认提升16倍。

## 9.实现一个拷贝文件的工具类使用字节流还是字符流?

我们拷贝的文件不确定是只包含字符流，有可以能有字节流(图片、声音、图像等)，为考虑到通用性，要使用字节流。

## 10.讲一下线程的几种实现方式?启动方式？区分方式？

①实现方式

1)通过继承Thread类实现一个线程

2)通过实现Runnable接口实现一个线程

继承扩展性不强，java总只支持单继承，如果一个类继承Thread就不能继承其他的类了。

3)通过Callable和FutureTask创建线程   
 a:创建Callable接口的实现类 ，并实现Call方法   
 b:创建Callable实现类的实现，使用FutureTask类包装Callable对象，该FutureTask 对象封装了Callable对象的Call方法的返回值   
 c:使用FutureTask对象作为Thread对象的target创建并启动线程   
 d:调用FutureTask对象的get()来获取子线程执行结束的返回值

4)通过线程池创建

②怎么启动？

Thread thread = new Thread(继承了Thread的对象/实现了Runnable的对象)

thread.setName(“设置一个线程名称”);

thread.start();

启动线程使用start方法，而启动了以后执行的是run方法。

③怎么区分线程？

在一个系统中有很多线程，每个线程都会打印日志，我想区分是哪个线程打印的怎么办？

thread.setName(“设置一个线程名称”); 这是一种规范，在创建线程完成后，都需要设置名称。

**11.讲一下什么是设计模式？常用的设计模式有哪些？**

设计模式就是经过前人无数次的实践总结出的，设计过程中可以反复使用的、可以解决特定问题的设计方法。

①单例(饱汉模式、饥汉模式)

1).构造方法私有化，作用是除了自己类中能创建对象外其他地方都不能创建对象

2).在自己的类中创建一个单实例（饱汉模式是一出来就创建创建单实例，而饥汉模式需要 的时候才创建）

3).提供一个方法获取该实例对象(创建时需要进行方法同步)

②工厂模式:Spring IOC就是使用了工厂模式.

对象的创建交给一个工厂去创建。

③代理模式:Spring AOP就是使用的动态代理。

# **12.final finally finalize区别**

Final:可以修饰类、变量、方法，修饰类表示该类不能被继承、修饰方法表示该方法不能被重写、修饰变量表示该变量是一个常量不能被重新赋值。

Finally:一般作用在try-catch代码块中，在处理异常的时候，通常我们将一定要执行的代码方法finally代码块中，表示不管是否出现异常，该代码块都会执行，一般用来存放一些关闭资源的代码。

Finalize:是一个方法，属于Object类的一个方法，而Object类是所有类的父类，该方法一般由垃圾回收器来调用，当我们调用System.gc() 方法的时候，由垃圾回收器调用finalize()，回收垃圾，一个对象是否可回收的最后判断。

# **13.静态变量与实例变量、普通变量的区别**

静态变量： 静态变量由于不属于任何实例对象，属于类的，所以在内存中只会有一份，在类的加载过程中，JVM只为静态变量分配一次内存空间。

实例变量： 每次创建对象，都会为每个对象分配成员变量内存空间，实例变量是属于实例对象的，在内存中，创建几次对象，就有几份成员变量

# **14.重载和重写的区别?重载的方法能否根据返回类型进行区分？**

方法的重载和重写都是实现多态的方式，区别在于重载实现的是编译时的多态性，而重写实现的是运行时的多态性。

重载：发生在同一个类中，方法名相同参数列表不同（参数类型不同、个数不同、顺序不同），与方法返回值和访问修饰符无关，即重载的方法不能根据返回类型进行区分

重写：发生在父子类中，方法名、参数列表必须相同，返回值小于等于父类，抛出的异常小于等于父类，访问修饰符大于等于父类（里氏代换原则）；如果父类方法访问修饰符为private则子类中就不是重写。

# **15.抽象类和接口的区别？**

抽象类是用来捕捉子类的通用特性的。接口是抽象方法的集合。

从设计层面来说，抽象类是对类的抽象，是一种模板设计，接口是行为的抽象，是一种行为的规范。

相同点:

①接口和抽象类都不能实例化

②都位于继承的顶端，用于被其他实现或继承

③都包含抽象方法，其子类都必须覆写这些抽象方法

不同点

|  | 抽象类 | 接口 |
| --- | --- | --- |
| 声明 | 抽象类使用abstract关键字声明 | 接口使用interface关键字声明 |
| 实现 | 子类使用extends关键字来继承抽象类。如果子类不是抽象类的话，它需要提供抽象类中所有声明的方法的实现 | 子类使用implements关键字来实现接口。它需要提供接口中所有声明的方法的实现 |
| 构造器 | 抽象类可以有构造器 | 接口不能有构造器 |
| 访问修饰符 | 抽象类中的方法可以是任意访问修饰符 | 接口方法默认修饰符是public。并且不允许定义为 private 或者 protected |
| 多继承 | 一个类最多只能继承一个抽象类 | 一个类可以实现多个接口 |
| 字段声明 | 抽象类的字段声明可以是任意的 | 接口的字段默认都是 static 和 final 的 |

备注：Java8中接口中引入默认方法和静态方法，以此来减少抽象类和接口之间的差异。

现在，我们可以为接口提供默认实现的方法了，并且不用强制子类来实现它。

接口和抽象类各有优缺点，在接口和抽象类的选择上，必须遵守这样一个原则：

行为模型应该总是通过接口而不是抽象类定义，所以通常是优先选用接口，尽量少用抽象类。

选择抽象类的时候通常是如下情况：需要定义子类的行为，又要为子类提供通用的功能。

# **16.两个对象的 hashCode() 为true，则 equals() 也一定为 true，对吗？**

不对，两个对象的 hashCode() 相同，equals() 不一定 true。

代码示例：

String str1 = "通话";String str2 = "重地";

System. out. println(String. format("str1：%d | str2：%d", str1. hashCode(),str2. hashCode()));

System. out. println(str1. equals(str2));

执行的结果：

str1：1179395 | str2：1179395

false

代码解读：

很显然“通话”和“重地”的 hashCode() 相同，然而 equals() 则为 false，因为在散列表中，hashCode() 相等即两个键值对的哈希值相等，然而哈希值相等，并不一定能得出键值对相等。

# **17.Java 中的 Math. round(-1. 5) 等于多少？**

等于 -1，因为在数轴上取值时，中间值（0.5）向右取整，所以正 0.5 是往上取整，负 0.5 是直接舍弃。

# **18.如何将字符串反转？**

使用 StringBuilder 或者 stringBuffer 的 reverse() 方法。

// StringBuffer reverse

StringBuffer stringBuffer = new StringBuffer();

stringBuffer. append("abcdefg");

System. out. println(stringBuffer. reverse()); // gfedcba// StringBuilder reverse

StringBuilder stringBuilder = new StringBuilder();

stringBuilder. append("abcdefg");

System. out. println(stringBuilder. reverse()); // gfedcba

# **19.String常用的方法有哪些？**

* indexOf()：返回指定字符的索引。
* charAt()：返回指定索引处的字符。
* replace()：字符串替换。
* trim()：去除字符串两端空白。
* split()：分割字符串，返回一个分割后的字符串数组。
* getBytes()：返回字符串的 byte 类型数组。
* length()：返回字符串长度。
* toLowerCase()：将字符串转成小写字母。
* toUpperCase()：将字符串转成大写字符。
* substring()：截取字符串。
* equals()：字符串比较。

# **20.Java中的IO流分为几种？**

按功能来分：

输入流（input）

输出流（output）。

按类型来分：

字节流

字符流

字节流和字符流的区别是：

字节流按 8 位传输以字节为单位输入输出数据，

字符流按 16 位传输以字符为单位输入输出数据。

# **21.BIO和NIO和AIO有什么区别？**

BIO：Block IO 同步阻塞式 IO，就是我们平常使用的传统 IO，它的特点是模式简单使用方便，并发处理能力低。

NIO：Non IO 同步非阻塞 IO，是传统 IO 的升级，客户端和服务器端通过 Channel（通道）通讯，实现了多路复用。

AIO：Asynchronous IO 是 NIO 的升级，也叫 NIO2，实现了异步非堵塞 IO ，异步 IO 的操作基于事件和回调机制。

# **22.Files常用的方法有哪些？**

* Files. exists()：检测文件路径是否存在。
* Files. createFile()：创建文件。
* Files. createDirectory()：创建文件夹。
* Files. delete()：删除一个文件或目录。
* Files. copy()：复制文件。
* Files. move()：移动文件。
* Files. size()：查看文件个数。
* Files. read()：读取文件。
* Files. write()：写入文件。

# **23.String str="i"与 String str=new String("i")一样吗？**

不一样，因为内存的分配方式不一样。String str="i"的方式，Java 虚拟机会将其分配到常量池中；而 String str=new String("i") 则会被分到堆内存中。

# **24.JDK 和 JRE 有什么区别？**

JDK：Java Development Kit 的简称，Java 开发工具包，提供了 Java 的开发环境和运行环境。

JRE：Java Runtime Environment 的简称，Java 运行环境，为 Java 的运行提供了所需 环境。

具体来说 JDK 其实包含了 JRE，同时还包含了编译 Java 源码的编译器 Javac，还包 含了很多 Java 程序调试和分析的工具。简单来说：如果你需要运行 Java 程序，只需安装 JRE 就可以了，如果你需要编写 Java 程序，需要安装 JDK。

