1. JRE指的是Java运行的解释，即：如果电脑没有JDK，但是有JRE，那么也表示可以进行Java程序的执行，只不过不能开发而已，现在的电脑里面都有JRE，再次安装属于更新JRE版本。
2. public class与class声明类的区别

public class 类名称{}：文件名称必须与类名称保持一致，一个\*.java文件里面只能有一个public class定义。

Class 类名称{}：文件名称可以与类名称不同，但是生成的\*.class文件是根据文件中的class名称一致的，在一个\*.java文件里面可以定义多个class，但是编译后会分别形成不同的class文件。

1. 解释PATH和CLASSPATH的区别？

PATH：是属于操作系统属性，定义所有可执行程序的路径。

CLASSPATH：是Java程序解释类文件时所使用的加载路径。不管定义了多少个路径。一定定义一个“.”（表示由当前所在目录进行类的加载）。

1. 在日后开发的代码的时候，尽量使用单行注释。

原因：在一些开发工具里面，多行注释格式化的效果不好。而对于文档注释，需要有开发工具的支持后才能更好地编写。

1. JDK1.7后开始增加了一些神奇的特性

public class 你好 {

public static void main(String[] args) {

int 年龄 = 20;

System.out.println("年龄： " + 年龄);

}

}

但是不推荐使用！

1. 个人选择数据类型的原则：

* 表示整数使用int，表示小数使用double
* 描述日期时间数字或者表示文件大小（或内存大小）使用long
* 实现内容传递或者编码转换使用byte
* 实现逻辑控制用boolean
* 中文可以使用char避免掉乱码

1. while循环用于不确定次数但是知道结束条件的循环，而for循环适合于知道循环结束次数的循环
2. 声明对象指的是未开辟堆内存的对象，如果直接使用此对象会出现“NullPointerException”异常。
3. 真实的对象信息都保存在了堆内存空间之中，那么如果没有栈指向的对象就称为匿名对象。匿名对象由于没有其他对象对其进行引用，所以只能够使用一次，一次之后该对象空间就将成为垃圾，等待回收。
4. 类的第一开发要求：

① 类名称必须有意义；

② 类之中所有的属性必须private封装，封装后的属性必须提供setter、getter；

③ 类之中可以提供多个构造方法，但是必须保留一个无参构造方法构造方法排列顺序请按照参数的个数进行升序或者降序排列；

④ 类之中不允许出现任何输出语句，所有信息输出必须交给被调用处输出；

⑤ 类之中需要提供一个取得对象完整信息的方法，暂定为：getInfo()，而且返回String型数据。

1. 如果要判断输入的内容是否是某一字符串，请一定要将要比较字符串写在最前面，为了防止出现NullPointerException异常。

例如：”hello”.equals(input)

1. String对象的两种实例化方式（①new(“anc”) ②”ABC”）在开发中到底使用哪一种？

使用直接赋值可以不产生垃圾空间，并且可以自动入池，不要使用构造方法！

1. 观察下面代码：

String str = ”;

for (int x=0; x<1000; x++) {

str += x;

}

System.out.println(str);

以上的代码修改了String对象的引用关系1000次，并且产生大量的垃圾空间，所以此类代码在开发中严禁使用，String的内容不要过多的频繁修改！

1. 记下String类方法的名称、方法作用、参数的类型及个数、返回值类型，否则写程序的时候会有很大的挫败感。
2. 只要访问类中的属性前面必须加上“this.”。
3. 使用“this()”调用构造方法形式的代码只能放在构造方法首行，进行构造方法互相调用的时候，一定保留调用出口。
4. 引用传递的核心意义：同一块堆内存空间可以被不同栈内存所指向，不同栈内存可以对同一堆内存进行内存的修改。
5. String类对象的内容一旦声明则不可更改，对象内容的改变依靠的是引用地址的改变。
6. Java类与数据表结构对应的步骤：

第一步：设计基本字段

第二步：设置引用关系

第三步：取得数据

1. 如果现在一个类中的属性使用了private封装了，那么在类的外部不能通过类直接调用类的属性。但是如果将一个对象传回到类的方法里面，那么就相当于取消了封装形式，可以直接通过对象访问属性，
2. 对象比较时一定要判断对象是否为null、地址是否相同、属性是否相同。
3. static属性与非static属性最大的区别：

所有的非static属性必须通过产生实例化对象之后才可以访问，但是static属性不受实例化对象控制，也就是说，在没有实例化对象产生的情况下，依然可以使用static属性。

1. static方法不能够直接访问非static属性或者方法，只能够调用static属性或者方法；非static方法可以访问static的属性或者方法，不受任何限制。
2. static的实际应用：

功能一：实现类实例化对象个数的统计

功能二：实现属性的自动设置

1. 内存区一共有四个：栈内存、堆内存、全局数据区、全局代码区。
2. static属性保存在全局数据区。
3. 代码块一共可以分四种：普通代码块、构造块、静态块、同步代码块（等待多线程的时候）。

如果将一个代码块写在了方法里面，那么就称它为普通代码块。

如果将一个代码块写在了类里面，那么就称为构造块。构造块优先于构造方法执行！

如果一个代码块使用static进行定义的话，那么就称为静态块。静态块分两种：

情况一：在非主类中使用，静态块优先于构造块执行，而且不管实例化多少个对象，静态块之执行一次。

情况二：在主类中定义，此时静态块将优先于主方法执行。

1. 内部类可以与外部类方便的进行私有属性的访问；

内部类可以使用private声明，声明之后无法在外部实例化内部类对象；

语法：外部类.内部类 内部类对象 = new 外部类().new 内部类();

使用static定义的内部类就相当于一个外部类；

语法：外部类.内部类 内部类对象 = new 外部类.内部类对象();

内部类可以在方法中定义。