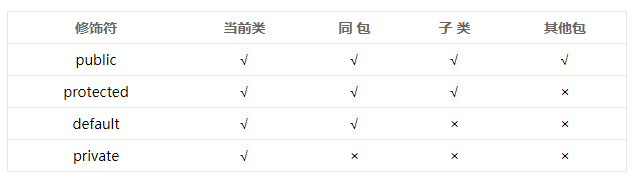
面试题目：

<http://www.importnew.com/22083.html> 

基本类型（primitive type）和枚举类型（enumeration type），剩下的都是引用类型（reference type）

1. JAVA语言特点：纯面向对象，平台无关、很多内置库、垃圾回收器、安全健壮、简单、单继承、解释型
2. java显示强制类型转换，
3. 只有静态方法才是类成员46
4. 静态块先于主方法执行
5. 静态初始化，然后非静态初始化
6. 父类静态变量，父类静态代码块，子类静态变量，子类静态代码块
7. 父类非静态变量，父类非静态代码块，父类构造函数 子类……
8. 变量类型：成员变量，静态变量，局部变量
9. 只有public abstract 和final可以修饰类
10. 普通方法可以与构造函数方法名相同
11. 标识接口，用来标识对象属于某一类
12. 值传递和引用传递，clone()复制对象，深复制和浅复制
13. 反射：得到对象所在的类，获取一个类的成员，运行时创建对象，运行时调用对象方法
14. 函数指针的实现：接口，实现接口，接口作为参数
15. 面向对象特点：抽象 继承 封装 多态（方法的重载（编译时多态）与覆盖（运行时多态），允许不同类的对象对同一消息做出相应）
16. 组合和继承：能用组合就不用继承
17. 接口的实现类和抽象类的子类（方法实现有无，实现——继承，强度功能，成员变量，作用，抽象类也可以有静态main方法）
18. 内部类包括：静态，成员，局部（静态，成员），匿名四钟。非静态内部类不能有静态成员
19. Object的方法包括（getClass:返回运行时类；Finalize:垃圾回收时调用；equals；hashCode; toString; clone notify(), notifyAll(), wait(),）
20. Break跳出双重循环out,
21. Final属性不可变（引用不可变，对象不可变），方法不可覆盖，类不可被继承：不可继承的类：String StringBuffer
22. Finally一般用来释放资源
23. Assert代码中正确性检查机制而if是控制程序流程的
24. Static:成员变量，成员方法，代码块，内部类
25. 实例变量，类变量，局部变量，final变量
26. Switch：枚举常量或整数表达或字符串或强制转换为int
27. 尽量不要volatile
28. a instanceof A
29. strictfp
30. Boolean byte char short int long float double
31. 原始类型合引用类型
32. 不可变类（简单，节省内存，线程安全）：不允许改变成员值包括基本类型的包装类，String类，
33. Math.round ceil floor
34. 移位操作前先转换为int类型32位整数
35. 每个中文字符占两个字节，英文字符占1个字节
36. String对象可以被共享，字符串池在常量区，对象在堆内存，对象的引用在栈内存，
37. String（效率最大低，数据量小优先） StringBuffer（可变类，线程安全） StringBuilder（非线程安全，单线程效率高） StringTokenizer（分割字符串的）
38. 数组是对象
39. Finally先执行，接着return，如果finally中有return语句，则覆盖别处的return语句，finally块中的改变return的值对返回值没有任何影响，而对引用类型的数据会有影响。

Finally块不一定执行，可能在try中就结束执行了。

1. 方法内部变量存在栈中，
2. 异常是程序运行时发生非正常情况或错误：Throwable(Error, Exception(检查异常和运行时异常))
3. 要处理异常，不要一直抛出
4. 输入输出流：字符长度==2个字节==16位，字符流一次可以读多个字节，用到缓存（Reader，Writer）（InputStream OutputStream）流的作用是为了改善程序性能和使用方便
5. 管理文件和目录的类是File
6. 对象持久化方式：序列化和外部序列化。序列化将对象以一连串的字节描述的过程，用于解决在对对象进行读写操作时所引发的问题。序列化在需要网络来发送对象或实现深复制时需要用到。
7. 非阻塞的IO，NIO
8. Java socket分为两种类型（TCP UDP）
9. JVM加载class文件的原理机制：ClassLoader，动态加载，（系统类，扩展类，自定义类）三步（装载，链接，初始化）
10. GC分配内存，确保被引用的内存不被错误地回收，回收不再被引用的对象的内存空间
11. 内存泄漏：一个不再被程序使用的对象或变量还在内存中占有存储空间。垃圾回收标准，内存泄漏两种情况，
12. 栈：基本类型，对象的引用变量，主要用来执行程序的，存取速度快，栈的大小和生存期确定；堆：引用类型变量，每个java程序都运行在一个单独的jvm实例上，每个实例对应一个堆，各个线程共享堆内存。因此多线程访问数据时需要对数据进行同步。主要用来存放对象
13. 设计模式：

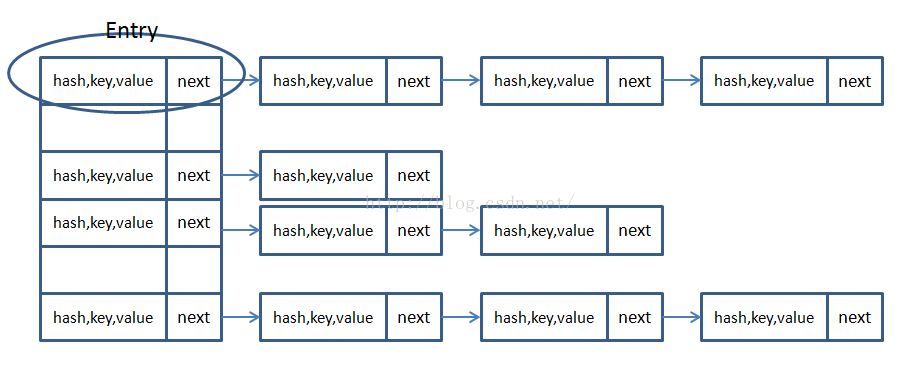
|  |  |  |  |
| --- | --- | --- | --- |
|  | 创建型 | 结构型 | 行为型 |
| 类 | 工厂 | 适配器 |  |
| 对象 | 单例 | 享元  装饰 | 观察者 |

单例：任何时刻单例类的实例只有一个，

工厂：简单工厂，工厂方法，抽象工厂

适配器：

观察者：

1. 多线程：线程是能够执行程序代码的一个执行单元。状态：运行，就绪，挂起，结束。进程是一段正在执行的程序。多线程（减少程序相应时间，线程的创建和切换开销小，提高CPU的利用率，简化程序结构）
2. 同步机制确保资源在某一时刻只能被一个线程使用，必须获得每一个线程对象的锁，利用同步代码块实现同步或利用同步方法。异步机制能提高程序的效率，不必关心其它线程的状态或行为
3. 实现多线程：继承Thread类重写run方法，实现Runnable接口实现该接口的run方法，实现callable接口重写call方法
4. Hashtable不允许 null 值(key 和 value 都不可以)，线程安全，效率低，初始数组大小为11，HashMap允许 null 值(key和value都可以)，初始数组为16。 WeakHashMap弱引用
5. ArrayList基于数组实现，随机访问更快，每次扩充1.5倍（而vector扩充2倍，且是同步的，线程安全，但性能更次），初始容量为10，LinkedList基于链表实现，添加和删除更快，非线程安全。索引或只在末端增加或删除元素时，使用ArrayList或Vector;指定位置插入或删除时，用LinkedList效率更高，当在多线程中时使用Vector安全
6. HashMap使用时，自定义类作为key值，一定要重新equals和hashcode方法，设计成不可变类，如果两个对象相等，那么这两个对象有相同的hashcode
7. HashMap采用位桶+链表+红黑树实现，当链表长度超过阈值（8）时，将链表转换为红黑树，这样大大减少了查找时间。当链表数组的容量超过初始容量的0.75时，再散列将链表数组扩大2倍，把原链表数组的搬移到新的数组中
8. Start()方法异步调用run()，而run()同步的。只有调用start方法才能实现多线程
9. 多线程同步的实现机制有：synchronized关键字，wait和notify，lock