### JAVA第五阶段—DAY03-JAVA作业答案

1. 简答题：描述一下 volatile 关键字的特性与使用方法

答案：

volatile 变量是一种轻量级同步机制，用来确保将变量的更新操作通知到其他线程。

被volatile关键字修饰的变量，编译器与运行时都会注意到这个变量是共享的，因此不会将该变量上的操作与其他内存操作一起重排序。volatile变量不会被缓存在寄存器或者对其他处理器不可见的地方，因此在读取volatile类型的变量时总会返回最新写入的值。

volatile在Java并发编程中常用于保持内存可见性和防止指令重排序。内存可见性（Memory Visibility）：所有线程都能看到共享内存的最新状态；防止指令重排：在基于偏序关系的Happens-Before内存模型中，指令重排技术大大提高了程序执行效率，但同时也引入了一些问题。

1. 简答题：简述CAS算法的概念。

答案：

CAS (Compare-And-Swap 比较与交换) 是一种硬件对并发的支持，针对多处理器操作而设计的处理器中的一种特殊指令，用于管理对共享数据的并发访问。

CAS 是一种无锁的非阻塞算法的实现。

CAS 包含了 3 个操作数：需要读写的内存值 V、进行比较的值 A、拟写入的新值 B

当且仅当 V 的值等于 A 时，CAS 通过原子方式用新值 B 来更新 V 的值，否则不会执行任何操作。

1. 简答题：简述ConCurrentHashMap与HashMap、HashTable的区别

答案：

HashMap是一个线程不安全的传统Java容器类，HashTable可以视为一个提供了线程安全的HashMap，以避免多线程并发出现错误。

ConcurrentHashMap 同步容器类是 Java5 增加的一个线程安全的哈希表；性能介于 HashMap 与 Hashtable 之间；内部采用"锁分段"机制替代Hashtable的独占锁,进而提高性能;

此包还提供了设计用于多线程上下文中的 Collection、Map 实现：ConcurrentHashMap、ConcurrentSkipListMap、ConcurrentSkipListSet、CopyOnWriteArrayList 和CopyOnWriteArraySet。

Hashtable 之所以效率低下，主要是因为其实现使用了synchronized 关键字对 put 等操作进行加锁，而 synchronized 关键字加锁是对整个对象加锁，也就是说在进行put等修改Hashtable的操作时，锁住了整个Hash表，从而使得其效率低下。因此在JDK1.5~1.7版本，Java使用分段锁机制实现ConcurrentHashMap，从而提升性能。

1. 简答题：简述Callable接口的使用方法与特点。

答案：

Callable接口是Java 5.0 在 java.util.concurrent 提供的一个新的创建执行线程的方式。

Callable 接口类似于 Runnable，两者都是为那些其实例可能被另一个线程执行的类设计的。但是 Runnable 不会返回结果，并且无法抛出经过检查的异常。Callable可以返回结果，并允许抛出异常。

Callable 需要依赖FutureTask ，FutureTask 也可以用作闭锁。FutureTask.get方法可以获取Callable对象的结果，默认情况下，获取结果时会造成线程阻塞。

使用方法是

1. 先创建一个实现 Callable 接口的类的实例。
2. 将此实例作为 FutureTask 的构造参数，创建 FutureTask 实例。
3. 将 FutureTask 实例作为 Thread 对象的构造参数，创建线程实例并运行。
4. 简答题：synchronized同步锁与Lock锁有什么区别。

答案：

1.synchronized是java内置关键字，在jvm层面，Lock是java类；  
2.synchronized无法判断是否获取锁的状态，Lock可以判断是否获取到锁；  
3.synchronized会自动释放锁(a 线程执行完同步代码会释放锁 ；b 线程执行过程中发生异常会释放锁)，Lock需在finally中手工释放锁（unlock()方法释放锁），否则容易造成线程死锁；  
4.用synchronized关键字的两个线程1和线程2，如果当前线程1获得锁，线程2线程等待。如果线程1阻塞，线程2则会一直等待下去，而Lock锁就不一定会等待下去，如果尝试获取不到锁，线程可以不用一直等待就结束了；  
5.synchronized的锁可重入、不可中断、非公平，而Lock锁可重入、可判断、可公平  
6.Lock锁适合大量同步的代码的同步问题，synchronized锁适合代码少量的同步问题。  
7.Lock是一个接口，synchronized是一个关键字，synchronized放弃锁只有两种情况：线程执行完了同步代码块的内容时发生异常；而lock不同，它可以设定超时时间，也就是说他可以在获取锁时便设定超时时间，如果在你设定的时间内它还没有获取到锁，那么它会放弃获取锁然后响应放弃操作。