1. 请默写出上课讲解的两种实现单例的方式

|  |
| --- |
| 饿汉模式  Class Sing{  Private Sing(){}  Priavte static Sing s = new Sing(); 在类中就创建好对象  Public static Sing getSing(){  Return s;  }  } |

|  |
| --- |
| 懒汉模式  Class Sing{  Private Sing(){}  Priavte static Sing s; 在类中只定义但不创建对象  } |

1. 在网上查找学习其他的单例实现方式

|  |
| --- |
| 线程安全的“懒汉模式”  2 public class SingleDemo {  3 //懒汉式，线程安全的，双重判断加同步线程  4 private static SingleDemo singleDemo=null;  5 private SingleDemo() {}  6 public static SingleDemo getInstance() {  7 if (singleDemo==null) {  8 synchronized(SingleDemo.class) {  9 if (singleDemo==null) {  10 singleDemo = new SingleDemo();  11 }  12 }  13 }  14 return singleDemo;  15 }  16 } |

|  |
| --- |
| DCL双检查锁机制（DCL：double checked locking）    Synchronized加锁以解决线程问题  线程A访问到getInstance方法获取锁，然后斤进入if判断然后创建对象，然后释放锁  线程B获得锁，然后进入if判断然后直接返回instance对象  但是加锁很产生性能开销，为了解决加锁产生的性能开销，然后想到延迟加锁，采用块级锁来提高性能降低开销（synchronized（类名.Class）） |

这又回到了两线程同时访问会实例化两个对象的问题，所以有得在其中加一个if判断

