Chp12 线程

参考答案

1. 具体答案略。重点是“宏观上串行，微观上并行”的概念
2. C
   run 方法可以直接调用，但是直接调用 run 方法并不产生新线程。

3. C
   Runnable 接口中的 run 方法没有参数。

4. A
   Thread.sleep()方法会抛出已检查异常 java.lang.InterruptedException，必须要处理。

5. 能打印 In xxx，在主线程中利用 start 启动了一个新线程，并不会影响主线程继续执行。

6. 略

7. B。编译正常。而在运行时，由于 start 方法把处于初始状态的线程转为可运行状态，在第一次调用 t.start()方法时正确，而第二次调用时线程 t 并不是处于初始状态，因此会产生状态错误，抛出一个异常。

8. 加上 synchronized 表示对 lock 对象加锁,这样能保证输出 10 个$$$之后再输出 10 个###。如果不加上 synchronized 的话，则程序运行时输出的顺序每次执行都不同。

9. C
   两个线程都是对属性 data 加锁，然而，t1 对象的 data 属性是字符串对象“hello”，t2 对象的 data 属性是字符串对象“world”，因为不是同一个对象，因此 t1 线程和 t2 线程之间没有交互和加锁、解锁的操作，因此不会有同步的效果。

10. C
    A 错误，因为对 run 方法表示对当前对象加锁，具体到这个例子，分别表示对 t1 对象和 t2 对象加锁。由于加锁的不是同一个对象，因此不会产生同步效果。
    B 错误，原因如上一题所说。

11. 输出 2 2 2
    原因：由于产生了三个对象，三个线程分别操作一个各不相同的对象，因此不会产生数据不一致的同步问题。

12. 1） 输出结果为
    2 2

2 4
2 6
原因：因为 result 是局部变量，并每次运行的时候赋值为 0，因此三个线程输出的 result 结果都为 2；
因为多个线程共同使用同一个 MyValue 属性，因此 data 属性共享，每次运行新线程时都会把 data 属性加上 2。
2） 应当把 m 方法加上 synchronized 修饰符。这样，每次调用 mv.m()方法时，都需要获得 mv 对象的锁标记。从而，让三个线程对象的 mv 属性都指向同一个对象，这样三个线程都需要对同一个对象加锁，从而能使得三个线程同步的执行。

13. 参考答案

```
class MyThread1 implements Runnable{
    public void run() {
        for(int i = 0; i<100; i++){
            this.sleep((int)(Math.random()*1000));
            System.out.println("hello");
        }
    }
}

class MyThread2 extends Thread{
    // run 方法不能抛出任何异常
    public void run() throws Exception {
        for(int i = 0; i<100; i++){
            this.sleep((int)(Math.random()*1000));
            System.out.println("world");
        }
    }
}

public class TestMyThread{
    public static void main(String args[]){
        Runnable t1 = new MyThread1();
        Thread t2 = new MyThread2();
        //不能对 t1 调用 start 方法，应该利用 t1 创建一个 Thread 对象
        t1.start();
        t2.start();
    }
}
```

14. 在主线程中可以修改 data，如果要保持同步，需要把 setData 方法也作为 synchronized 的。

15. 参考 TeacherThread.java

16. 输出结果为： 2 2 2。

    由于 countprint 中修改的 result 是 countprint 方法的参数，相当于局部变量。而局部变量是每个线程都独立的，没有同步问题。