Lab 11 Assignment

> 班级：212113
>
> 学号：21371220
>
> 姓名：杨硕

Question1

- 程序：是为了完成某个特定的任务，而用某种语言编写的一组指令的集合，指的是一段静态的代码
- 进程：是程序的一次执行过程，或是一个正在运行的程序。是一个动态的过程
- 线程：进程可进一步细化为线程，是一个程序内部的执行的路径。一个进程在执行过程中可以有多个线程。线程作为调度和执行的单位，每个线程拥有独立的运行栈和程序计数器（PC），一个进程中的多个线程共享相同的内存单元/内存地址空间，可以访问相同的变量和对象

Question2

1. 互斥条件：进程要求对所分配的资源进行排它性控制，即在一段时间内某资源仅为一进程所占用
2. 请求和保持条件：一个进程因请求资源而阻塞时，对已获得的资源保持不放
3. 不剥夺条件：进程已获得的资源在未使用完之前，不能剥夺，只能在使用完时由自己释放
4. 环路等待条件：在发生死锁时，必然存在一个进程-资源的环形链

Question3

1. 采用继承`Thread`类的方式：
   优点：编写简单，如果需要访问当前线程，无需使用`Thread.currentThread()`方法，直接使用this，即可获得当前线程
   缺点：因为线程类已经继承了Thread类，所以不能再继承其他的父类
2. 采用实现`Runnable`接口方式：
   （1）优点：线程类只是实现了`Runable`接口，还可以继承其他的类。在这种方式下，可以多个线程共享同一个目标对象，所以非常适合多个相同线程来处理同一份资源的情况
   （2）缺点：编程稍微复杂，如果需要访问当前线程，必须使用`Thread.currentThread()`方法。

Question4

(1) 进程是线程 `Thread` 内部的一个执行单元，它是程序中一个单一顺序控制流程。 False

(2) 一个进程可以包括多个线程。两者的一个主要区别是：线程是资源分配的单位，而进程是CPU调度和执行的单位。False

(3) 线程可以用 `yield` 使低优先级的线程运行。False

(4) 当一个线程进入一个对象的一个 `synchronized` 方法后，其它线程可以再进入该对象的其它同步方法执行。True

(5) `notify` 是唤醒所在对象 `wait pool` 中的第一个线程。False

Question5

1. 程序的输出为

```
SyncThread1:0
SyncThread1:1
SyncThread1:2
SyncThread1:3
SyncThread1:4
SyncThread2:5
SyncThread2:6
SyncThread2:7
SyncThread2:8
SyncThread2:9
```

2. `synchronized` 修饰 `run()` 方法，这时 `run()` 方法为同步方法，作用为：

   给调用该方法的对象加上“对象锁”

   即当两个并发线程访问同一个对象中的 `run()` 方法时，同一时间内只能有一个线程得到执行。另一个线程必须等待当前线程执行完以后才能执行；
   题目中，即 `thread1` 和 `thread2` 都访问的 `syncThread` 的 `run()` 方法，只有 `thread1` 执行完，`thread2` 才执行

3. `sleep` 的作用是暂停当前线程，暂停时间为方法参数，当线程停止时，状态会被设置为wait，暂停时间结束，恢复为Runnable状态；

   改成wait，输出会变为：

```
SyncThread1:0
SyncThread2:1
SyncThread1:2
SyncThread2:3
SyncThread1:4
SyncThread2:5
SyncThread1:6
SyncThread2:7
SyncThread1:8
SyncThread2:9
```

   原因为sleep不会释放对象锁，而wait会释放对象锁

Question6

1. 代码补全如下

```
public class ThreadPrint {
    public static void main(String[] args) throws InterruptedException{
        Object a=new Object();
        Object b=new Object();
        Object c=new Object();
        Thread8 threadA=new Thread8("A",c,a);
        Thread8 threadB=new Thread8("B",a,b);
        Thread8 threadC=new Thread8("C",b,c);
        new Thread(threadA).start();
        Thread.sleep(100);
        new Thread(threadB).start();
        Thread.sleep(100);
        new Thread(threadC).start();
```

```
            Thread.sleep(100);
        }
    }
    class Thread8 implements Runnable{
        private String name;
        private Object prev;
        private Object self;
        public Thread8(String name,Object prev,Object self){
            this.name=name;
            this.prev=prev;
            this.self=self;
        }
        @Override
        public void run(){
            int count=10;
            while(count>0){
                synchronized (prev){
                    synchronized (self){
                        System.out.print(name);
                        count--;
                        self.notify();
                    }
                    try{
                        if(count==0){
                            prev.notify();
                        }
                        else{
                            prev.wait();
                        }
                    }catch (InterruptedException e){
                        e.printStackTrace();
                    }
                }
            }
        }
    }
```

2. 进行三个线程，顺序输出ABC10个
3. 不能省略，如果省略，三个进程会同时抢占cpu，输出顺序会颠倒
4. 有可能，例如线程`threadA`在`count`=1时`wait`，线程`threadB`、`threadC`抢占cpu，count进行两次减1变为-1，线程始终被堵塞造成死锁

Question7

代码入口：solution->Question7->Test.java