Java 程序设计 LAB11

实验目的

- 理解程序、进程和多线程概念及特点
- 理解线程的状态、生命周期及调度策略
- 理解并掌握线程类 `Thread` 和 `Runnable` 接口，并能够进行相关应用的程序设计，实现多线程编程

实验题目

Question1 简述程序，进程，线程的概念 简答

1.程序：是静态的代码，指一组指示计算机或其他具有信息处理能力装置执行动作或做出判断的指令，通常用某种程序设计语言编写，运行于某种目标计算机体系结构上。含有指令和数据的文件。
2.进程：是计算机中的软件程序关于某数据集合上的一次运行活动，是系统进行资源分配和调度的基本单位，是操作系统结构的基础，是系统分配资源和调度的基本单位，也就是说进程可以单独运行一段程序。
3.线程：是程中的一个执行流程，运行中的应用程序，进程中的一个实体，是被系统独立调度和分派的基本单位，是CPU调度和分派的最小基本单位，线程自己不拥有操作系统资源，但是该线程可与同属进程的其他线程共享该进程所拥有的全部资源。

Question2 产生死锁的四个条件是什么？ 简答

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个进程所占有。
2.请求和保持条件。当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，不能被剥夺，只能在使用完后由自己释放。
4.环路等待条件。当发生死锁时，必然存在一个进程—资源的环形链。

Question3 创建线程的两种方式分别是什么？各有什么优缺点 简答

方法一：`java.lang.Thread` 类
优点：编写简单。并且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：不能再继承其他类。

方法二：实现 `Runnable`
优点：线程类只是实现了该接口，还可以继承其他类。在这种创建方式下，多个线程共享同一个目标对象，适合多个线程处理同一份资源的情况。
缺点：编程较为复杂。

Question4 判断题

(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。
错
(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。
错
(3) 线程可以用 `yield` 使低优先级的线程运行。
错
(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。
错
(5) `notify` 是唤醒所在对象 `wait pool` 中的第一个线程。
错

Question5 程序输出简答

```
public class Main {
    public static void main(String[] args) {
        SyncThread syncThread = new SyncThread();
        Thread thread1 = new Thread(syncThread, "SyncThread1");
        Thread thread2 = new Thread(syncThread, "SyncThread2");
        thread1.start();
        thread2.start();
    }
}
class SyncThread implements Runnable {
    private static int count;
    public SyncThread() {
        count = 0;
    }
    public synchronized void run() {
        for (int i = 0; i < 5; i++) {
            try {
                System.out.println(Thread.currentThread().getName() + ":" +
(count++));
                Thread.sleep(100);//【1】
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
    public int getCount() {
        return count;
    }
}
```

(1) 请写出上述程序的输出
SyncThread1:0

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

(2) 用 `synchronized` 修饰 `run()` 的作用是什么？

保证两个线程能正常执行原子操作

(3) 标号【1】处 `sleep` 的作用是什么？如果改为 `wait(100);` 输出会发生改变吗，为什么？

作用是导致此线程暂停执行指定时间，试图给其他线程机会.

改后结果：

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

因为执行了wait()方法，这个线程会释放锁，进入的对象的等待中，还有另一个线程就会获得锁，执行同步代码块。

Question6 程序补全题

```
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--;
                    【1】
                }
                try{
                    if(count==0)
                        【2】
                    else
                        【3】
                }catch (InterruptedException e){
                    e.printStackTrace();
                }
            }
        }
    }
}
```

(1) 补全标号处的代码
【1】.self.notifyAll();
【2】.break;
【3】.prev.wait();
(2) 详细说明上述程序的功能
主要的目的是ThreadA->ThreadB->ThreadC->ThreadA 循环执行三个线程
(3) 主函数 `main` 中的 `Thread.sleep(100)` 语句不能省略，请简述原因。
保证三个线程 threadA, threadB, threadC 的执行顺序，从而保证 ABC 的顺序。
(4) 主函数 `main` 中的 `Thread.sleep(100)` 语句全部去掉后程序可能出现死锁吗？试举例说明。
可能现死锁。按照线程竞争的顺序，可能会出现在 synchronized (prev)处，线程threadA 等待 c,threadB 等待 a,threadC 等待 b 的死锁情况。

Question7 创建两个线程，其中一个输出1-52，另外一个输出A-Z。输出格式要求：编程

```
12A 34B 56C 78D 910E 1112F 1314G 1516H 1718I 1920J 2122K 2324L 2526M 2728N 2930O
3132P 3334Q 3536R 3738S 3940T 4142U 4344V 4546W 4748X 4950Y 5152Z
```

注意：

> (1) 可以参考T6
>
> (2) 12A 34B..... 看清楚A和3之间是有一个空格的，且输出以Z结尾，即结尾不能有空格
>
> (3) 请确保你的程序能够正常结束，不要在输出Z之后一直阻塞下去