

# 第七章作业

(共 100 分)

提交方式：

1. 上传到[坚果云的此链接](#)中 (上传时请勿挂 vpn 或其他梯子, 否则可能上传失败)
2. 文件名严格按照: **学号-姓名-第几次作业.docx**(或 pdf)的方式命名
3. 每次作业在截止日期之前, 若发现提交内容有误, 可以再提交一次。但每次作业最多只能提交两次。第二次提交的文件后面加上“(2)”的字样。
4. 作业提交形式: 单一 word 或 pdf 文档, 最好用 word, 方便助教留下扣分记录。**不要上传 Eclipse、VScode、IDEA 等项目文件**, 不要压缩打包, 不要以文件夹形式上传。
5. 截止时间: 10 月 25 日晚 23:59

一、单选题, 并说明理由 (对的为什么对, 错的为什么错, 每个选项都要评论) (共 10 分, 每题 5 分; 其中选对 2 分, 解释 3 分)

1. 要实现线程, 可以从以下 ( ) 类继承。  
A. Runnable                  B. Thread                  C. Throwable                  D. Stream
2. 关于线程设计, 下列描述正确的是 ( )  
A、线程对象必须实现 Runnable 接口  
B、启动一个线程直接调用线程对象的 run() 方法  
C、Java 提供对多线程同步提供语言级的支持  
D、一个线程可以包含多个进程

二、程序阅读题，写出程序的输出结果，并进行解释（每题 40 分，共 40 分，其中运行结果 20 分，解释 20 分，解释写在输出结果后，不要写在代码中）

1、

```
public class MulTr1 implements Runnable {
    public void run() {
        synchronized (this) {
            try {
                for (int i = 0; i < 3; i++) {
                    System.out.println(i);
                    Thread.sleep(100);
                    if (i == 1) {
                        throw new InterruptedException("打断了");
                    }
                }
            }
            catch (InterruptedException e) {
                System.out.println(e.getMessage());
            }
            finally {
                System.out.println("in finally");
            }
        }
    }
    public static void main(String[] args) {
        MulTr1 m1 = new MulTr1();
        Thread t1 = new Thread(m1);
        Thread t2 = new Thread(m1);
        t1.start();
        t2.start();
    }
}
```

三、阅读和编写程序题（50 分，共 1 题）

1、阅读下面的程序，写出运行结果（如果不止一种，写出一种即可）；并按要求改写程序。

（其中阅读程序部分 20 分，改写程序部分 30 分）。提示：本程序模拟两个 Dota 小兵(Creep) 攻击一个玩家的场景。



```

class Player {
    private int hp = 100; //体力
    public int getHP() {
        return hp;
    }

    public void setHP(int hp) {
        this.hp = hp;
    }
}

public class Creep implements Runnable {
    private Player player = new Player();

    @Override
    public void run() {
        synchronized (this) {
            for (int i = 0; i < 3; i++) {
                System.out.println(Thread.currentThread().getName() + "
attack...");
                this.attack(20);
                System.out.println(Thread.currentThread().getName() + ": 当前player的hp值= " + player.getHP());
                if (player.getHP() <= 0) {
                    System.out.println(Thread.currentThread().getName() + ": player is dead.");
                    break;
                }
                try {
                    Thread.sleep(100);
                }
                catch (InterruptedException e) {
                }
            } /* for-loop */
            System.out.println(Thread.currentThread().getName() + "
end.");
        } /* synchronized (this) */
    } /* run( ) */

    public void attack(int y) { //攻击掉血
        player.setHP(player.getHP() - y);
    }

    public static void main(String[] args) {
        Creep r = new Creep();
        Thread ca = new Thread(r, "Creep-A");
        Thread cb = new Thread(r, "\t\t\tCreep-B");
        ca.start();
        cb.start();
    }
}

```

}

(1) 写出运行结果，并进行解释，如果不止一种结果，写出一种即可（这部分 20 分，其中结果 10 分，解释 10 分）

(2) 改写程序。上述程序采用的是同步块的方式，现改为采用同步方法的方式。要求达到和上述程序同样的输出结果（不考虑结果随机出现）。要求（这部分 30 分）：

(a) 移除 Creep 类里的 attack 方法；移除 run 方法中的同步块。

(b) 在 Player 类里新建一个同步方法: `synchronized void beAttacked(int y)`, `y` 代表每次体力减少值。模拟 creep 三次攻击的行为，并按照同样的内容输出。如果玩家血降到 0 后，该线程提前终止。

(c) main 方法同之前的设置，创建两个线程。