上机作业：多线程

1、创建多线程

创建 2 个线程，一个线程负责输出英文字母表，另一个线程负责输出希腊字母表。

要求：①通过继承 Thread 类实现创建线程。

②通过实现 Runnable 接口创建线程。

```
class Output1 extends Thread{
    public void run(){
        for (char ch='a';ch<='z';ch++){
            System.out.println(ch+" ");
        }
    }
}
class Output2 extends Thread{
    public void run(){
        for (char ch='α';ch<='ω';ch++){
            System.out.println(ch+" ");
        }
    }
}
public class Main{
    public static void main(String[] args){
```

```
        Output1 A =new Output1();
        Output2 B =new Output2();
        A.start();
        B.start();
    }
}
class Output1 implements Runnable{
    public void run(){
        for (char ch='a';ch<='z';ch++){
            System.out.println(ch+" ");
        }
    }
}
class Output2 implements Runnable{
    public void run(){
        for (char ch='α';ch<='ω';ch++){
            System.out.println(ch+" ");
        }
    }
}
public class Main{
    public static void main(String[] args){
```

```
        Thread A=new Thread(new Output1());
        Thread B=new Thread(new Output2());
        A.start();
        B.start();
    }
}
```

2、改变线程状态

创建 3 个线程：老师，李四，王五。李四准备睡 10 分钟再开始听课，王五准备睡 50 分钟再开始听课。老师在输出 3 句“上课”后，吵醒休眠的线程李四，李四被吵醒后，负责再吵醒休眠的线程王五。

```
class A implements Runnable{
    Thread student1,student2,teachar;
    A(){
        student1=new Thread(this);
        student2=new Thread(this);
        teachar=new Thread(this);
        student1.setName("李四");
        student2.setName("王五");
        teachar.setName("老师");
    }
```

```
    public void run() {
        if (Thread.currentThread().getName()=="李四"){
            try {
                System.out.println("李四正在睡觉");
                Thread.sleep(10 * 1000 * 60);
            }
            catch (Exception e){
                System.out.println("李四被老师吵醒了");
                student2.interrupt();
            }
        }else if (Thread.currentThread().getName()=="王五
"){
            try {
                System.out.println("王五正在睡觉");
                Thread.sleep(50 * 1000 * 60);
            }
            catch (Exception e){
                System.out.println("王五被李四吵醒了");
            }
        }else if (Thread.currentThread().getName()=="老师
"){
            for (int i=0;i<3;i++){
```

```
                System.out.println("上课");
            }
            student1.interrupt();
        }
    }
}
public class Main{
    public static void main(String[] args){
        A B=new A();
        B.student2.start();
        B.student1.start();
        B.teachar.start();
    }
}
```

3、线程同步

对于题目 1，要求通过线程同步，保证一个线程输出完字母表之后，另一个线程再执行。

```
class Output implements Runnable {
    public void run() {
        write();
    }
```

```
    public synchronized void write() {
        if (Thread.currentThread().getName().equals("1"))
{
            for (char ch = 'a'; ch <= 'z'; ch++) {
                System.out.println(ch + " ");
            }
        } else if
(Thread.currentThread().getName().equals("2")) {
            for (char ch = 'α'; ch <= 'ω'; ch++) {
                System.out.println(ch + " ");
            }
        }
    }
}
public class Main{
    public static void main(String[] args){
        Output C =new Output();
        Thread A=new Thread(C);
        Thread B=new Thread(C);
        A.setName("1");
        B.setName("2");
```

```
        A.start();
        B.start();
    }
}
```

4、协调同步线程

模拟 3 个人排队买票，每人买 1 张票。售票员只有 1 张 5 元的钱，电影票 5 元一张。张某拿一张 20 元的人民币排在孙某前面，孙某拿一张 10 元人民币排在赵某的前面，赵某拿一张 5 元人民币排在最后。最终的卖票次序应当是孙、赵、张。

```
class TicketHouse implements Runnable {
    int fiveAmount = 1, tenAmount = 0, twentyAmount = 0;

    public void run() {
        if (Thread.currentThread().getName().equals("张某
")) {
            saleTicket(20);
        } else if
(Thread.currentThread().getName().equals("孙某")) {
            saleTicket(10);
        } else if
(Thread.currentThread().getName().equals("赵某")) {
```

```
            saleTicket(5);
        }
    }

    public synchronized void saleTicket(int money) {
        if (money == 5) {
            fiveAmount = fiveAmount + 1;
            System.out.println("给" +
Thread.currentThread().getName() + "入场券" +
Thread.currentThread().getName() + "钱正好");
        } else if (money == 20) {
            while (fiveAmount < 3) {
                try {
                    System.out.println("\n" +
Thread.currentThread().getName() + "靠边等...");
                    wait();
                    System.out.println("\n" +
Thread.currentThread().getName() + "继续买票");
                } catch (Exception e) {
                }
            }
            fiveAmount = fiveAmount - 3;
```

```
            System.out.println("给" +
Thread.currentThread().getName() + "入场券" +
Thread.currentThread().getName() + "给 20 找 15 元");
        } else if (money == 10) {
            while (fiveAmount < 1) {
                try {
                    System.out.println("\n" +
Thread.currentThread().getName() + "靠边等...");
                    wait();
                    System.out.println("\n" +
Thread.currentThread().getName() + "继续买票");
                } catch (Exception e) {
                }
            }
            fiveAmount = fiveAmount + 1;
            System.out.println("给" +
Thread.currentThread().getName() + "入场券" +
Thread.currentThread().getName() + "给 10 找 5 元");
        }
        notify();
    }
}
```

```
public class Main {
    public static void main(String args[]) {
        TicketHouse officer = new TicketHouse();
        Thread zhang = new Thread(officer);
        Thread sun = new Thread(officer);
        Thread zhao = new Thread(officer);
        zhang.setName("张某");
        sun.setName("孙某");
        zhao.setName("赵某");
        zhang.start();
        sun.start();
        zhao.start();
    }
}
```