【案例8-2】 Svip优先办理服务

### 【案例介绍】

**1.任务描述**

在日常工作生活中，无论哪个行业都会设置一些Svip用户，Svip用户具有超级优先权，在办理业务时，Svip用户具有最大的优先级。

本案例要求编写一个模拟Svip优先办理业务的程序，在正常的业务办理中，插入一个Svip用户，优先为Svip用户办理业务。本案例在实现时，可以通过多线程实现。

**2.运行结果**

运行结果如图8-1所示。

手机屏幕截图

描述已自动生成

图8-1 运行结果

### 【案例目标】

* 学会分析”Svip优先办理服务”任务实现的逻辑思路。
* 能够独立完成”Svip优先办理服务”程序的源代码编写、编译以及运行。
* 能够在程序中使用多线程的”插队”完成逻辑思路。

### 【案例思路】

1. 查看运行结果分析后，创建一个special()方法模拟Svip办理业务。
2. 查看运行结果分析后，首先创建一个normal()方法模拟正常的窗口排队，当有Svip客户是使用join线程让步，调用special()优先让Svip办理业务。
3. 最后在main方法中调用normal()方法。

### 【案例代码】

Svip优先办理服务程序的代码实现如文件8-1所示。

文件8-1 svip.java

package chapter0402;

public class svip {

public static void main(String[]args) throws InterruptedException

{

new Thread(new normal()).start();

}

}

class special extends Thread{

public void run()

{

System.out.println("svip客户开始办理业务");

System.out.println("svip客户办理业务的倒计时");

for(int i=10;i>=0;i--)

{

System.out.println(i+"秒");

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

System.out.println("svip客户办理完毕");

}

}

class normal extends Thread{

public void run(){

System.out.println("业务办理窗口在正常排队中");

System.out.println("此时来了一位svip客户");

Thread t=new Thread(new special());

//各走各的逻辑错误，再加入join先执行完special，再执行normal剩下的

t.start();

try {

t.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println("业务办理窗口又恢复正常排队");

}

}

文件8-1中第8~24行代码是svip客户办理业务，使用for循环10次模拟办理业务第25~39行代码是模拟窗口正常排队，第32~35行代码是normal线程让步special线程。