三、四章习题

2018-11-5

Chap 3

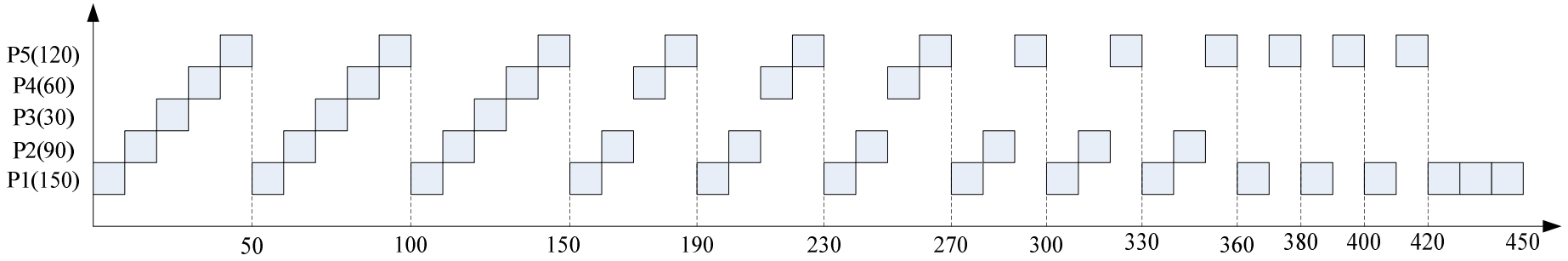
【4】为什么处理器需要区分用户态和核心态两种运行模式？操作系统的相关程序是在哪种模式下执行的？

- 用户程序通常运行在用户模式下，不能直接访问受保护的系统资源（包括：内核程序和数据、特定内存区域以及I/O设备的访问），也不能改变处理器的运行模式；而操作系统内核、异常处理程序等运行在内核模式下，可以访问系统的所有资源，而且能够自由切换系统的运行模式。
- 之所以要区分这两种模式，是为了控制用户程序对内核程序和数据、特定内存区域以及I/O设备的访问，避免越权访问，提高系统的安全性。用户程序不允许执行特权指令（如某些I/O指令），对系统所属的数据、地址空间以及硬件等访问受到严格限制，防止了恶意程序对关键资源有意或无意的使用和破坏。

【13】5个批处理作业同时到达计算中心。它们的估计运行时间分别为150,90,30,60和120秒。它们的优先级分别为6、3、7、9和4（值越小，优先级越高）。设一次作业上下文切换的平均时间为2秒。对于下面每种调度算法，计算每个作业的周转时间和所有作业的平均周转时间。

- 时隙为10秒的轮转法。
- 基于优先级的不可抢占调度。
- FCFS（按150,90,30,60和120的顺序执行）。
- 最短作业优先。

（1）对于轮转法，假定按照P1~P5的顺序轮转。调度如下图所示：



根据上图，

P1的周转时间：450+42*2=534秒

P2的周转时间：350+35*2=420秒

P3的周转时间：130+13*2=156秒

P4的周转时间：260+26*2=312秒

P5的周转时间：420+42*2=504秒

平均周转时间为（534+420+156+312+504）/5=385秒。

（2）采用基于优先级的不可抢占调度策略，调度顺序为：P2，P5，P1，P3，P4。

则P2的周转时间为：90秒

P5的周转时间为：90+120+2=212秒

P1的周转时间为：210+150+2*2=364秒

P3的周转时间为：360+30+2*3=396秒

P4的周转时间为：390+60+2*4=458秒

平均周转时间为：（90+212+364+396+458）/5=304秒

（3）FCFS策略采用非抢占决策模式，因此后来的进程必须等待前面的进程执行完毕才能执行。

P1的周转时间为：150秒

P2的周转时间为：150+90+2=242秒

P3的周转时间为：240+30+2*2=274秒

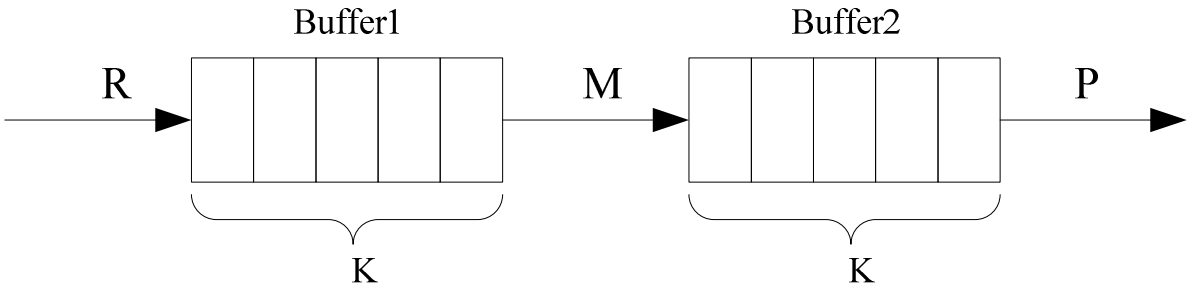
P4的周转时间为：270+60+2*3=336秒

P5的周转时间为：330+120+2*4=458秒

平均周转时间为：（150+242+274+336+458）/5=292秒

Ch4 习题

【12】今有三个并发进程，R负责从输入设备读入信息并把信息放入缓冲区Buffer1。M从Buffer1中取出信息并加工，并把加工的信息放入缓冲区Buffer2。P把Buffer2中的信息取出并打印输出。两个缓冲区的容量均为K。试用**P**、**V**操作写出三个进程能正确工作的程序。



解：首先分析并发需求。

- 互斥需求：R和M不能同时访问Buffer1中的同一单元；同样，M和P不能同时访问Buffer2中的同一单元；
- 同步需求：

  当Buffer1非空时，M才能从Buffer1中取信息；当Buffer1不满时，R才能向Buffer1存信息；

  当Buffer2非空时，P才能从Buffer2中取信息；当Buffer2不满时，M才能向Buffer2存信息。

R	M	P

R(){
while(1){
info:=produce();
Buffer1[i]:=info;
i:=(i+1) mod K;
}
}
	M(){
	while(1){
	info:= Buffer1[j];
	j:=(j+1) mod K;
	newInfo:=transform(info)
	;
	Buffer2[k]:=newInfo;
	}
	}
		P(){
		while(1){
		info:=Buffer2[l];
		l:=(l+1) mod K;
		consume(info);
		}
		}

为了实现上述并发需求，设计如下信号量：

Semaphore Full1:=0;　　/*表示Buffer1中占位的个数*/

Semaphore Empty1:=K;　　/*表示Buffer1中空位的个数*/

Semaphore Full2:=0;　　/*表示Buffer1中占位的个数*/

Semaphore Empty2:=K;　　/*表示Buffer1中空位的个数*/

Sys= R || M || P

R	M	P

R(){
int i:=0;
while(1){
info:=produce();
P(Empty1);
Buffer1[i]:=info;
i:=(i+1) mod K;
V(Full1);
}
}
	M(){
	int j:=0;
	int k:=0;
	while(1){
	P(Full1);
	info:= Buffer1[j];
	j:=(j+1) mod K;
	V(Empty1);
	newInfo:=transform(info);
	P(Empty2);
	Buffer2[k]:=newInfo;
	k:=(k+1) mod K;
	V(Full2);
	}
	}
		P(){
		int l:=0;
		while(1){
		P(Full2);
		info:=Buffer2[l];
		l:=(l+1) mod K;
		V(Empty2);
		consume(info);
		}
		}

典型错误！！！！

Semaphore Full1:=0;　　/*表示Buffer1中占位的个数*/
Semaphore Empty1:=K;　　/*表示Buffer1中空位的个数*/
Semaphore Full2:=0;　　/*表示Buffer1中占位的个数*/
Semaphore Empty2:=K;　　/*表示Buffer1中空位的个数*/
Semaphore Mutex1:=1, Mutext2:=1;

R	M	P

R(){
int i:=0;
while(1){
info:=produce();
P(Empty1);
P(Mutext1);
Buffer1[i]:=info;
i:=(i+1) mod K;
V(Mutex1);
V(Full1);
}
}
	M(){
	int j:=0;
	int k:=0;
	while(1){
	P(Full1);
	P(Mutext1);
	info:= Buffer1[j];
	j:=(j+1) mod K;
	V(Mutext1);
	V(Empty1);
	newInfo:=transform(info);
	P(Empty2);
	P(Mutext2);
	Buffer2[k]:=newInfo;
	k:=(k+1) mod K;
	V(Mutext2);
	V(Full2);
	}
	}
		P(){
		int l:=0;
		while(1){
		P(Full2);
		P(Mutext2);
		info:=Buffer2[l];
		l:=(l+1) mod K;
		V(Mutext2);
		V(Empty2);
		consume(info);
		}
		}

【15】桌子上有一只盘子，最多可以容纳1个水果。爸爸一次只能向盘子中放入一个苹果，妈妈一次只能向盘子中放入一个桔子。女儿专等吃盘中的苹果，儿子专等吃盘子里的桔子。（用PA，PO，TA，TO分别表示向盘中“放苹果”，“放桔子”，从盘中“取苹果”和“取桔子”的操作）。试用P、V操作写出能同步的程序。

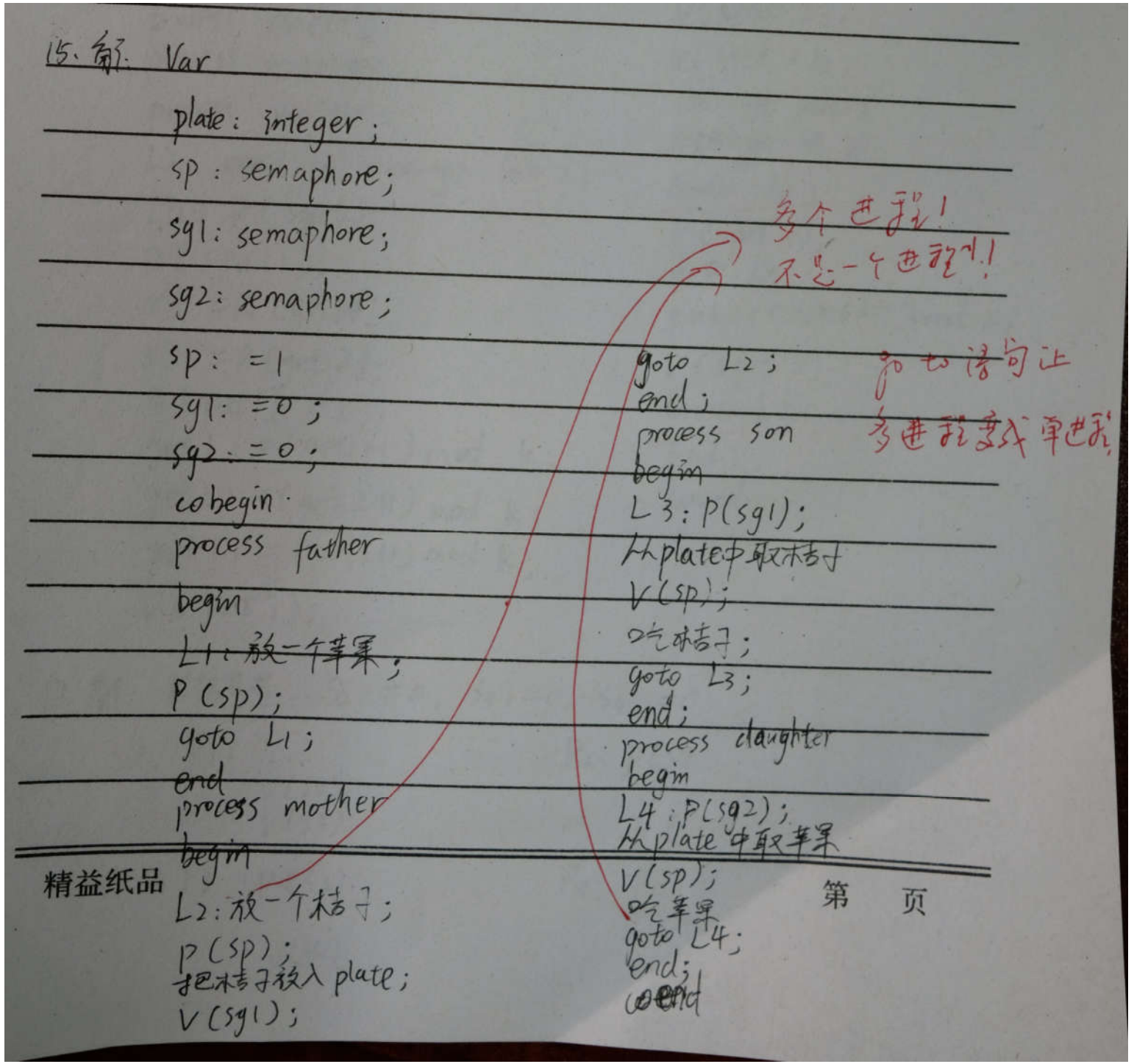
Sem SA:=0, SO:=0;

Sem Mutex:=1;

Parent	Mother	Son	Daught

while(1){
P(Mutex);
PA;
P(SA);
}
	while(1){
	P(Mutex);
	PO;
	P(SO);
	}
		while(1){
		V(SA);
		TA;
		V(Mutex);
		}
			while(1){
			V(SO);
			TO;
			V(Mutex);
			}

典型错误1



15. 解: Var

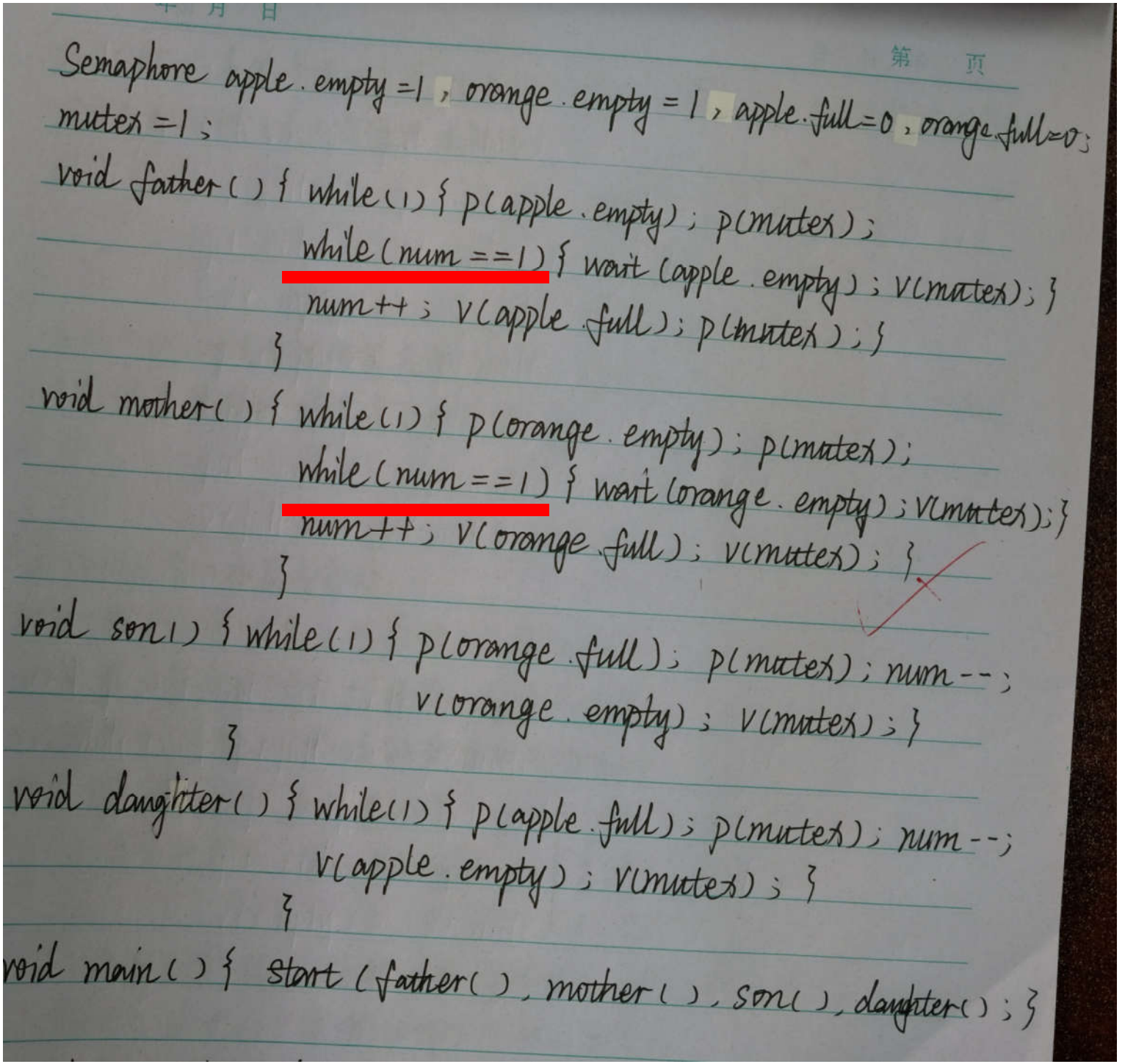
plate: integer;
sp: semaphore;
sg1: semaphore;
sg2: semaphore;
sp:=1
sg1:=0;
sg2:=0;
cobegin
process father
begin
L1: 放一个苹果;
P(sp);
goto L1;
end
process mother
begin
L2: 放一个桔子;
P(sp);
把桔子放入plate;
V(sg1);
goto L2;
end;
process son
begin
L3: P(sg1);
从plate中取桔子
V(sp);
吃桔子;
goto L3;
end;
process daughter
begin
L4: P(sg2);
从plate中取苹果
V(sp);
吃苹果
goto L4;
end;
coend

多个进程!
不是一个进程!!

加上语句让多进程变单进程

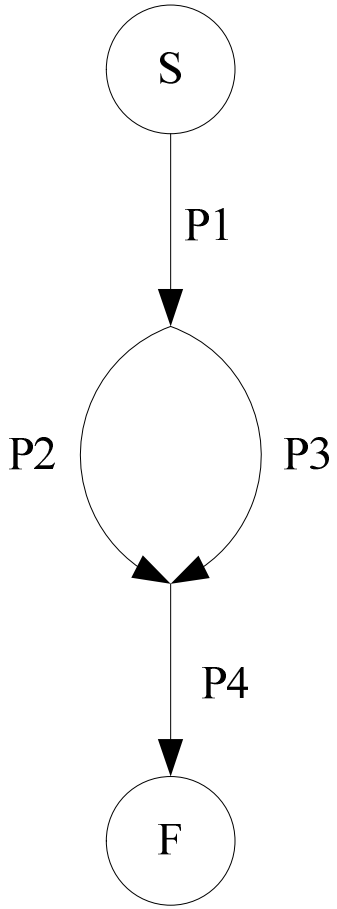
精益纸品 第 页

典型错误2

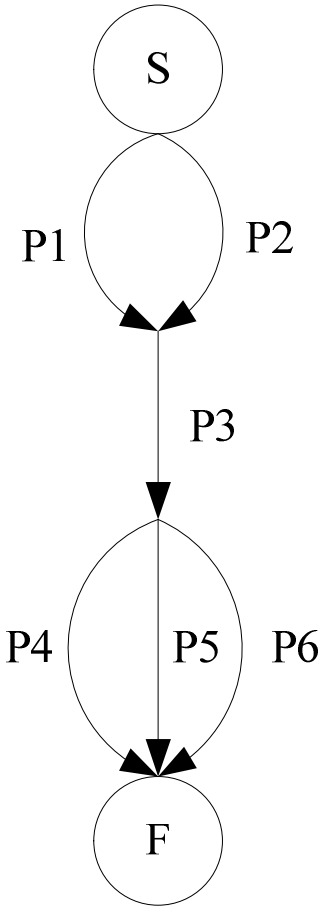


```
Semaphore apple.empty=1, orange.empty=1, apple.full=0, orange.full=0;
mutex=1;
void father(){ while(1){ p(apple.empty); p(mutex);
                while(num==1){ wait(apple.empty); v(mutex);}
                num++; v(apple.full); p(mutex);}
              }
void mother(){ while(1){ p(orange.empty); p(mutex);
                while(num==1){ wait(orange.empty); v(mutex);}
                num++; v(orange.full); v(mutex);}
              }
void son(){ while(1){ p(orange.full); p(mutex); num--;
                v(orange.empty); v(mutex);}
          }
void daughter(){ while(1){ p(apple.full); p(mutex); num--;
                v(apple.empty); v(mutex);}
               }
void main(){ start(father(), mother(), son(), daughter());}
```

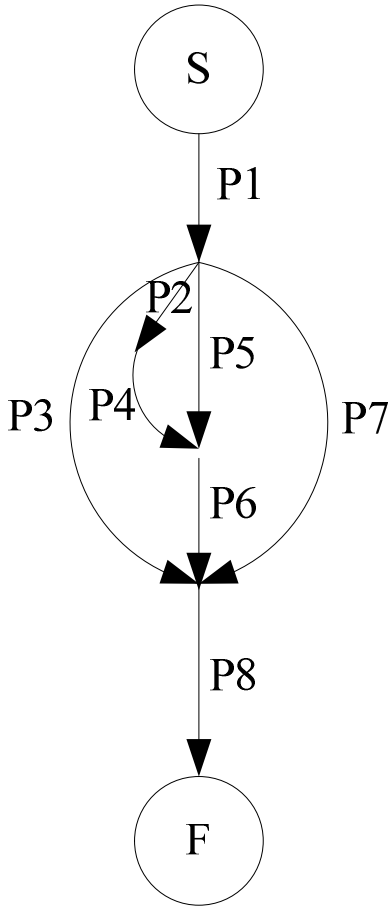
【17】利用**P**、**V**操作，怎样才能保证进程Pi能按下图的次序正确执行，其中S表示开始，F表示结束。



a)



b)



c)

(a) 的同步需求为：

- P2和P3要等待P1结束后才能开始执行；
- P4要等待P2和P3都结束之后才能开始执行。

设计信号量：

Semaphore endP1:= 0;　//表示P1执行结束

Semaphore endP2:= 0;　//表示P2执行结束

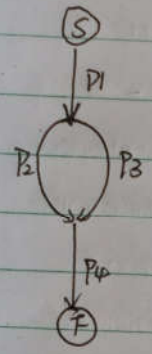
Semaphore endP3:= 0;　//表示P3执行结束

P1	P2	P3	P4

P1( ){
process P1;
V(endP1);
V(endP1);
}
	P2( ){
	P(endP1);
	process P2;
	V(endP2);
	}
		P3( ){
		P(endP1);
		process P3;
		V(endP3);
		}
			P4( ){
			P(endP2);
			P(endP3);
			process P4;
			}

典型错误

门.利用 P.V 操作，怎样才能保证进程 P_3 按题门图的次序正确地行，其中S表示开始，F表示结束.




```
int P1done=0, P3.done1); P4done=0;
Semaphone mutex=1;
void P1(){
P(mutex);
V(mutex);
}
void P2(){
P(mutex);
while(P1done==0){
wait();
V(mutex);}
if(P3_down==0){
P3.down=1;
V(mutex);}
else V(mutex);
}
void P4(){
P(mutex);
while(P1_down==0 && P3_down==0){
wait();
V(mutex);}
P4_down=1;
V(mutex);
}
void main(){
start(P1(),P2(),P3(),P4());
}
```

练习题：

1、下列中断中属于软中断的是：（ ）

（A）缺页中断　（B）除零中断　（C）系统调用　（D）I/O中断

2、临界区是：（ ）

（A）专用缓冲区　（B）缓冲池

（C）临时存放数据的内存区　（D）互斥执行的程序代码段

3、在一个多用户系统中，有3个并发进程，都需要同类资源4个，

试问系统不会发生死锁的最少资源数量是：（ ）

（A）9　（B）10　（C）11　（D）12

4、动态地址再定位（或重定位），是指地址的定位发生在：（ ）

（A）编译时　（B）链接时　（C）装载时　（D）运行时

5、Linux虚拟内存采用4KB为分页大小，代码段是从虚拟地址0x08048000开始编址，代码段中有一条指令的地址是0x08049010，请问该指令所在页面的起始地址和末地址分别是什么？