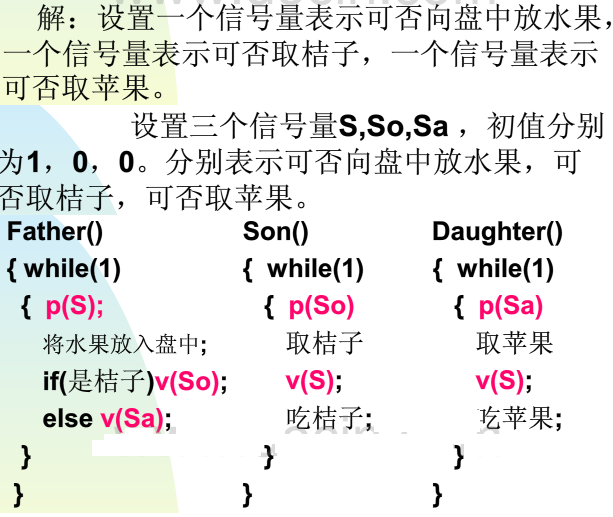
1. **桌上有1空盘，允许存放1个水果。爸爸向盘中放苹果，也可以向盘中放桔子。儿子专等吃盘中的桔子，女儿专等吃盘中的苹果。规定当盘空时一次只能放1个水果供吃者取用。请用wait( )、signal( )原语实现爸爸、儿子、女儿三个并发进程的同步。**



2、**如图7所示，有多个PUT操作同时向Buff1放数据，有一个MOVE操作不断地将Buff1的数据移到Buff2，有多个GET操作不断地从Buff2中将数据取走。Buff1的容量为m，Buff2的容量是n，PUT、MOVE、GET每次操作一个数据，在操作的过程中要保证数据不丢失。试用P、V原语协调PUT、MOVE的操作，并说明每个信号量的含义和初值。**

答案：设置6个信号量full1,empty1,mutex1,full2,empty2,mutex2,他们的含义和初值如下：

1. Full1表示buff1是否有数据，初值为0；
2. Empty表示buff1有空间，初值为m；
3. Mutex1表示buff1是否可操作，初值为1；
4. Full2表示buff2是否有数据，初值为0；
5. Empty2表示buff2有空间，初值为n；
6. Mutex2表示buff2是否可操作，初值为1；

Put(){

Repeat

P(empty1);

P(mutex1);

Put();

V(mutex1);

V(full1);

Until false

}

Move(){

Repeat

P(full1);

P(empty2);

P(mutex2);

Move();

V(mutex2);

V(full2);

V(empty1);

Until false

}

Get(){

Repeat

P(full2);

P(mutex3);

取走;

V(mutex3);

V(empty2);

Until false

}

1. **某银行提供1个服务窗口和10个供顾客等待的座位。顾客到达银行时，若有空座位，则到取号机上领取一个号，等待叫号。取号机每次仅允许一位顾客使用，当营业员空闲时，通过叫号选取1位顾客，并为其服务。**

答案：(1)互斥资源：取号机(一次只允许一位顾客领号)，因此设一个互斥信号量mutex；  
(2)同步问题：顾客需要获得空座位等待叫号，当营业员空闲时，将选取一位顾客并为其服务。空座位的有、无影响等待顾客数量，顾客的有、无决定了营业员是否能开始服务，故分别设置信号量empty和full来实现这一同步关系。另外，顾客获得空座位后，需要等待叫号和被服务。这样，顾客与营业员就服务何时开始又构成了一个同步关系，定义信号量service来完成这一同步过程。  
semaphore emptySeats:=10 //空闲座位数

semaphore fullSeats:=0 //已占座位数

semaphore mutex:=1 //顾客使用取号机互斥信号量

cobegin

{

process 顾客i

{

wait(emptySeats); //获取一个座位

wait(mutex); //占用取号机取号

从取号机获得一个号码；

signal(mutex) //释放取号机

signal(fullSeats) //座位上增加一个顾客

等待叫号；

…………

获得服务；

signal(emptySeats); //释放一个座位

}

process 营业员

{

while （TRUE）

{

wait(fullSeats); //座位上减少一个顾客

wait(mutex2);

叫号；

Signal(mutex2);

为顾客服务；

}

}

} coend