1. **三个进程P1、P2、P3互斥使用一个包含N（N>0）个单元的缓冲区。P1每次用produce()生成一个正整数并用put()送入缓冲区某一空单元中；P2每次用getodd()从该缓冲区中取出一个奇数并用countodd()统计奇数个数；P3每次用geteven()从该缓冲区中取出一个偶数并用counteven()统计偶数个数。请用信号量机制实现这三个进程的同步与互斥活动，并说明所定义的信号量的含义。要求用伪代码描述。**

**答：//属于一生产者，多缓冲区，二互斥消费者问题，信号量定义及含义**

**semaphore empty=N; //缓冲区空缓冲数量信号量，N代表初始N个可用**

**semaphore fullodd=0; //缓冲区中奇数个数信号量，0代表无**

**semaphore fulleven=0; //缓冲区中偶数个数信号量，0代表无**

**semaphore mutex=1; //缓冲区互斥信号量，1代表可用**

**P1(){ //进程P1，生成正整数**

**while(1){**

**X=produce(); //生成数**

**wait(empty); //等有空位**

**wait(mutex); //等缓冲区可用**

**put(X);**

**signal(mutex); //放缓冲区**

**if(X%2==0)**

**signal(fulleven); //唤p3**

**else**

**signal(fullodd); //唤p2**

**}**

**}**

**P2(){ //进程P2**

**while(1){**

**wait (fullodd); //等奇数**

**wait (mutex); //等缓冲区可用**

**X=getodd();**

**signal(mutex); //放缓冲**

**signal(empty); //唤p1**

**countodd(X);**

**}**

**}**

**P3(){ //进程P3**

**while(1){**

**wait(fulleven); //等偶数**

**wait(mutex); //等缓冲**

**X=geteven();**

**signal (mutex); //放缓冲**

**signal (empty); //唤p1**

**counteven(X);**

**}**

**}**

1. **某缴费中心提供1个服务窗口和5个供顾客等待的座位，顾客到达该中心时，若有空座位，则到取号机上领取一个号，等待叫号，无空座等待。取号机每次仅允许一位顾客使用。当收银员空闲时，通过叫号选取一位顾客，并为其服务。顾客和收银员的活动表述如下：**

**Process 顾客：**

**{**

**从取号机获取一个号码；**

**等待叫号；**

**获取服务；**

**}**

**Process 收银员**

**{**

**叫号；**

**为顾客服务；**

**}**

**请添加必要的信号量和P、V(或者wait() 和signal())操作，实现上述过程中的互斥与同步。要求写出完整的过程，说明信号量的含义并赋初值。**

**答：/\*属于多生产者，一消费者问题，信号量定义\*/**

**semaphore seats=5; //表示空余座位数量的资源信号量，初值为5。**

**semaphore customers=0; //表示等待服务顾客数量的资源信号量，初值为0。**

**semaphore mutex=1; //管理取号机的互斥信号量，初值为1，表示取号机空闲。**

**semaphore service=0; //获取服务和叫号的先后关系**

**customer(){ //顾客进程**

**wait(seats); //等空位**

**wait(mutex); //等取号机，互斥资源**

**从取号机获取一个号码**

**signal(mutex); //释放取号机**

**signal(customers); //唤收银员**

**等待叫号；**

**wait(service); //等待叫号**

**获取服务；**

**}**

**cashier(){ //收银员进程**

**while(1){**

**wait(customers); //等顾客**

**叫号;**

**signal(service); //叫号**

**signal(seats); //顾客离去，空出1个座位**

**为顾客服务;**

**}**

**}**

1. **在公共汽车不断地到站、停车、行驶过程中，司机和售票员的活动分别如下：（1）司机：启动车辆；正常行车；到站停车；（2）售票员：关车门；售票；开车门。请用记录型信号量机制描述司机和售票员的同步关系。**

**答：/\*定义信号量及初值\*/**

**semaphore S1=0; //表示是否允许司机启动车辆，其初值为0；**

**semaphore S2=0; //表示是否允许售票员开车门，其初值为0。**

**driver(){ /\*司机描述\*/**

**while(1){**

**wait(S1);**

**启动车辆;**

**正常行车;**

**到站停车;**

**signal(S2);**

**}**

**}**

**Seller(){ /\*售票员描述\*/**

**while(1){**

**关车门;**

**signal(S1);**

**售票；**

**wait(S2);**

**开车门;**

**}**

**}**

**main(){**

**cobegin**

**Seller(); driver();**

**coend**

**}**

1. **假定系统有三个并发进程read, move和print共享缓冲器B1和B2.进程read负责从输入设备上读信息,每读出一个记录后把它存放到缓冲器B1中.进程move从缓冲器B1中取出一记录,加工后存入缓冲器B2.进程print将B2中的记录取出打印输出.缓冲器B1和B2每次只能存放一个记录.要求三个进程协调完成任务,使打印出来的与读入的记录的个数,次序完全一样.请用PV操作,写出它们的并发程序.**

**答：/\*定义信号量及初值\*/**

**record B1,B2; //B1第一个缓冲区，B2第二个缓冲区**

**semaphore empty1=1; //缓冲区b1空缓冲区数量信号量，1代表可用。**

**semaphore full1=0; //缓冲区b1满缓冲区数量信号量，0代表无资源。**

**semaphore empty2=1; //缓冲区b2空缓冲区数量信号量，1代表可用。**

**semaphore full2=0; //缓冲区b2满缓冲区数量信号量，0代表无资源。**

**read(){**

**record X;**

**while(1){**

**X=接收来自输入设备上一个记录;**

**wait(empty1); //等b1空**

**B1=X;**

**signal(full1); //置b1满，唤move**

**}**

**}**

**move(){**

**record Y;**

**while(1){**

**wait(full1); //等b1满**

**Y=B1;**

**signal(empty1); //置b1空，唤read**

**加工Y**

**wait(empty2); //等b2空**

**B2=Y;**

**signal(full2); //置b2满，唤print**

**}**

**}**

**print(){**

**record Z;**

**while(1){**

**wait(full2); //等b2满**

**Z=B2;**

**signal(empty2); //置b2空，唤move**

**打印Z;**

**}**

**}**

**main(){**

**cobegin**

**read(); move(); print();**

**coend**

**}**

1. **有一阅览室，读者进入时必须先在一张登记表上登记，该表为每一座位列出一个表目，包括座号、姓名，读者离开时要注销登记信息；假如阅览室共有100 个座位。用信号量和P 、V 操作，来实现用户进程的同步算法。试阅读下列程序段，在空白处填写正确答案：**

type Tab=record

number: integer;

name: string;

end

var A:array[1..100] of Tab;

for i:=1 to 100 do

begin

A[i].number=i;

A[i].name=null;

end

mutex,seatcount: semaphore;

i: integer;

**; mutex:=1;**

**; seatcount:=100;**

cobegin

process readeri ( var readename: string ) (i=1 , 2 …)

begin

**; P(seatcount);**

**; P(mutex);**

for i : = 1 to 100 do i++

if A[i].name=null then

A[i].name:=readername;

reader get the seat number=i;

**; V(mutex);**

进入阅览室，座位号i ，座下读书；

**; P(mutex);**

A[i]name=null;

**; V(mutex);**

**; V(seatcount);**

离开阅览室；

end

Coend

**写成书上的类C伪代码：**

/\*定义信号量及初值\*/

string A[100]; //登记表，座位号0-99，存储座位上读者的名字

semaphore **mutexA=l;** //mutexA座位数组A互斥信号量，初始值1代表可用

semaphore **seatcount=100;**  //seatcount空余的座位数，初始值100

reader(string readername){ //读者进程，传递读者名

int readernumber; //读者座位号

wait(seatcount); //等有空座位，座位-1

wait(mutexA); //等座位数组A可用

for(i=0;i<100;i++) //找空座，并占位

if(A[i].name==NULL)

{A[i].name=readername; break;}

readernumber=i;

signal(mutexA); //置座位数组A可用

读者进入阅览室，做在readernumber号座位读书;

读者准备离开阅览室

wait(mutexA); //等座位数组A可用

A[readernumber]=NULL; //离开座位，空出座位

signal(mutexA); //置座位数组A可用

signal(seatcount); //座位+1，唤读者

读者离开阅览室；

}

main(){

for(i=0;i<100;i++) A[i].name=null;

cobegin

while(1){

readername=新来的读者名;

reader(readername);

}

coend

}

**一、生产者-消费者问题及扩展**

1. **桌上有一空盘，允许存放一只水果。妈妈可向盘中放白梨，也可向盘中放橙子，儿子专等吃盘中的白梨，女儿专等吃盘中的橙子。规定当盘空时一次只放一只水果供吃者取用，请用P,V原语实现妈妈，儿子，女儿三个并发进程的同步。**

**答：/\*属于一生产者，单缓冲区，二互斥消费者问题，记录型信号量定义 \*/**

**item plate; //盘子临界资源**

**semaphore empty=1; //盘子中可放水果数信号量，互斥，1代表可放水果**

**semaphore fullp=0; //盘子中白梨数信号量，0代表无**

**semaphore fullo=0; //盘子中橙子数信号量，0代表无**

**/\*程序代码，pv原语\*/**

**mother(){ //母亲进程**

**while(1){**

**wait(empty); //等待盘子空**

**plate=随机水果(pear or orange);**

**if(plate==pear)**

**signal(fullp); //置白梨可用，通知儿子  
 else**

**signal(fullo); //置橙子可用，通知女儿  
 }**

**}**

**son(){ //儿子进程**

**item x;**

**while(1){**

**wait(fullp); //等白梨**

**x=plate;**

**signal(empty); //置盘子可用，通知妈妈**

**吃白梨x;**

**}**

**}**

**daughter(){ //女儿进程**

**item x;**

**while(1){**

**wait(fullo); //等橙子**

**x=plate;**

**signal(empty); //置盘子可用，通知妈妈**

**吃橙子x;**

**}**

**}**

**main(){**

**cobegin**

**mother(); son(); daughter();**

**coend**

**}**

1. **三个进程P1、P2、P3互斥使用一个包含N（N>0）个单元的缓冲区。P1每次用produce()生成一个正整数并用put()送入缓冲区某一空单元中；P2每次用getodd()从该缓冲区中取出一个奇数并用countodd()统计奇数个数；P3每次用geteven()从该缓冲区中取出一个偶数并用counteven()统计偶数个数。请用信号量机制实现这三个进程的同步与互斥活动，并说明所定义的信号量的含义。要求用伪代码描述。**

**答：//属于一生产者，多缓冲区，二互斥消费者问题，信号量定义及含义**

**semaphore empty=N; //缓冲区空缓冲数量信号量，N代表初始N个可用**

**semaphore fullodd=0; //缓冲区中奇数个数信号量，0代表无**

**semaphore fulleven=0; //缓冲区中偶数个数信号量，0代表无**

**semaphore mutex=1; //缓冲区互斥信号量，1代表可用**

**P1(){ //进程P1，生成正整数**

**while(1){**

**X=produce(); //生成数**

**wait(empty); //等有空位**

**wait(mutex); //等缓冲区可用**

**put(X);**

**signal(mutex); //放缓冲区**

**if(X%2==0)**

**signal(fulleven); //唤p3**

**else**

**signal(fullodd); //唤p2**

**}**

**}**

**P2(){ //进程P2**

**while(1){**

**wait (fullodd); //等奇数**

**wait (mutex); //等缓冲区可用**

**X=getodd();**

**signal(mutex); //放缓冲**

**signal(empty); //唤p1**

**countodd();**

**}**

**}**

**P3(){ //进程P3**

**while(1){**

**wait(fulleven); //等偶数**

**wait(mutex); //等缓冲**

**X=geteven();**

**signal (mutex); //放缓冲**

**signal (empty); //唤p1**

**counteven();**

**}**

**}**

**main(){**

**cobegin**

**P1(); P2(); P3();**

**coend**

**}**

1. **计算进程PC和打印进程PO1、PO2共享一个单缓冲区。计算进程负责计算，并把计算结果放入单缓冲区中。打印进程PO1、PO2负责从单缓冲区中取出计算结果进行打印，而且对于每一个计算结果，PO1和PO2都需分别打印一次。请用记录型信号量机制描述上述三个进程之间的同步关系。**

**答：/\*属于一生产者，单缓冲区，二共用消费者问题，定义信号量\*/**

**semaphore empty1=1; //表示计算结果是否已被PO1取走，其初值为1；**

**semaphore full1=0; //表示缓冲区中是否有可供PO1打印的计算结果，其初值为0；**

**semaphore empty2=1; //表示计算结果是否已被PO2取走，其初值为1。**

**semaphore full2=0; //表示缓冲区中是否有可供PO2打印的计算结果，其初值为0；**

**PC(){ /\*PC描述\*/**

**while(1){**

**计算;**

**wait(empty1); //等po1取**

**wait(empty2); //等po2取**

**计算结果放入缓冲;**

**signal(full1); //唤po1**

**signal(full2); //唤po2**

**}**

**}**

**PO1(){ /\*PO1描述\*/**

**while(1){**

**wait(full1); //等结果，po1可用**

**取出计算结果;**

**signal(empty1); //唤pc，po1用完**

**打印结果;**

**}**

**}**

**PO2(){ /\*PO2描述\*/**

**while(1){**

**wait(full2); //等结果，po2可用**

**取出计算结果;**

**signal(empty2); //唤pc，po2用完**

**打印结果;**

**}**

**}**

**main(){**

**cobegin**

**PC(); PO1(); PO2();**

**coend**

**}**

1. **某理发店理有一位理发师、一把理发椅和N把供等候理发的顾客坐的椅子。如果没有顾客，理发师便在理发椅上睡觉。一个顾客到来时，它必须叫醒理发师。如果理发师正在理发时又有顾客来到，则如果有空椅子可坐，就坐下来等待，否则就离开。**

**答：/\*属于多生产者，一消费者问题，信号量定义\*/**

**#define N=5; //顾客椅总数。**

**int nc=0; //等待顾客数。**

**semaphore customers=0; //表示等待服务顾客数量的资源信号量，初值为0。**

**semaphore mutex=1; //nc的互斥信号量，初值为1，表示初始空闲。**

**semaphore service=0; //叫顾客和获取服务的先后关系**

**customer(){ //顾客进程**

**wait(mutex); //等互斥资源**

**if(nc<N){ //有空位，等待**

**nc++;**

**signal(mutex); //释放互斥资源**

**signal(customers); //顾客来了**

**等待理发；**

**wait(service); //等待理发**

**获取服务，离开。**

**}else{ //无空座，不等**

**signal(mutex); //释放互斥资源**

**无空座，离开。**

**}**

**}**

**barber(){ //理发师进程**

**while(1){**

**wait(customers); //等顾客**

**wait(mutex); //等互斥资源**

**nc--; //叫顾客**

**signal(mutex); //释放互斥资源**

**叫顾客**

**signal(service); //唤醒一个等待服务的顾客**

**为顾客服务;**

**}**

**}**

1. **某缴费中心提供1个服务窗口和5个供顾客等待的座位，顾客到达该中心时，若有空座位，则到取号机上领取一个号，等待叫号，无空座等待。取号机每次仅允许一位顾客使用。当收银员空闲时，通过叫号选取一位顾客，并为其服务。顾客和收银员的活动表述如下：**

**Process 顾客：**

**{**

**从取号机获取一个号码；**

**等待叫号；**

**获取服务；**

**}**

**Process 收银员**

**{**

**叫号；**

**为顾客服务；**

**}**

**请添加必要的信号量和P、V(或者wait() 和signal())操作，实现上述过程中的互斥与同步。要求写出完整的过程，说明信号量的含义并赋初值。**

**答：/\*属于多生产者，一消费者问题，信号量定义\*/**

**semaphore seats=5; //表示空余座位数量的资源信号量，初值为5。**

**semaphore customers=0; //表示等待服务顾客数量的资源信号量，初值为0。**

**semaphore mutex=1; //管理取号机的互斥信号量，初值为1，表示取号机空闲。**

**semaphore service=0; //获取服务和叫号的先后关系**

**customer(){ //顾客进程**

**wait(seats); //等空位**

**wait(mutex); //等取号机，互斥资源**

**从取号机获取一个号码**

**signal(mutex); //释放取号机**

**signal(customers); //唤收银员**

**等待叫号；**

**wait(service); //等待叫号**

**获取服务；**

**}**

**cashier(){ //收银员进程**

**while(1){**

**wait(customers); //等顾客**

**叫号;**

**signal(service); //叫号**

**signal(seats); //顾客离去，空出1个座位**

**为顾客服务;**

**}**

**}**

**main(){**

**cobegin**

**cashier(); while(1){ customer();}**

**coend**

**}**

**二、哲学家进餐问题及扩展**

1. **哲学家进餐问题。若至多只允许有四位哲学家同时去拿左边的筷子，最终能保证至少有一位哲学家能够进餐，并在用毕时能释放出他用过的两只筷子，从而使更多的哲学家能够进餐。试用信号量与P 、V 操作解决。**

**答：**

**semaphore seat=4; //吃饭人数信号量，代表最多允许4人拿筷子**

**semaphore chopstick[5]={1,1,1,1,1}; //5个筷子的信号量数组**

**process(int i)**

**{ while(ture)**

**{ think();**

**wait(seat); //等人数资源**

**wait(chopstick[i]); //拿左边筷子**

**wait(chopstick[i+1]%5);} //拿右边筷子**

**eat();**

**signal(chopstick[i]); //放左边筷子**

**signal(chopstick[i+1]%5); //放右边筷子**

**signal(seat); //吃完，释放人数资源**

**}**

**}**

1. **哲学家进餐问题。规定奇数号哲学家先拿他左边的筷子，然后再去拿右边的筷子；而偶数号哲学家则相反。按此规定，将是1、 2号哲学家竞争1号筷子；3、4号哲学家竞争3号筷子。即五位哲学家都先竞争奇数号筷子，获得后，再去竞争偶数号筷子，最后总会有一位哲学家能获得两只筷子而进餐。试用用信号量与P 、V 操作完成上述解决办法。**

**答：**

**semaphore chopstick[5]={1,1,1,1,1}; //5个筷子的信号量数组**

**process(int i)**

**{ while(ture)**

**{ think();**

**if(i%2!=0){ //奇数哲学家**

**wait(chopstick[i]); //拿左筷子**

**wait(chopstick[i+1]%5); //拿右筷子**

**}**

**else{ //偶数哲学家**

**wait(chopstick[i+1]%5); //拿右筷子**

**wait(chopstick[i]); //拿左筷子**

**}**

**eat();**

**signal(chopstick[i]); //放左筷子**

**signal(chopstick[i+1]%5); //放右筷子**

**}**

1. **“过独木桥”问题：同一方向的行人可连续过桥，当某一方向有人过桥时，另一方向的行人必须等待；当某一方向无人过桥时，另一方向的行人可以过桥。请用记录型信号量机制描述两个方向行人的同步关系。**

**答：/\*定义信号量及初值\*/**

**//属于读者-写者问题扩展，多读，多写，读写互斥，读读不互斥，写写不互斥，这里将独木桥的两个方向分别标记为A和B；**

**int countA=0,countB=0; //并用整形变量countA和countB分别表示A和B方向上已在独木桥上的行人数，它们的初值为0；**

**semaphore SA=1; //用信号量SA来实现对countA的互斥访问，其初值为1；**

**semaphore SB=1; //用信号量SB来实现对countB的互斥访问，其初值为1；**

**semaphore mutex=1; //用信号量mutex来实现两个方向的行人对独木桥的互斥使用。**

**A(){ /\*A方向描述\*/**

**while(1){**

**wait(SA); //等互斥资源countA可用**

**if(countA==0) wait(mutex); //如A方向无人过桥，则竞争桥方向**

**countA++; //人数+1**

**signal(SA); //释放互斥资源**

**通过独木桥;**

**wait(SA); //等互斥资源countA可用**

**countA--; //过完桥了，人数-1**

**if(countA==0) signal(mutex); //没人了，释放桥资源**

**signal(SA); //释放**

**}**

**}**

**B(){ /\*B方向描述\*/**

**while(1){**

**wait(SB);**

**if(countB==0) wait(mutex);**

**countB++;**

**signal(SB);**

**通过独木桥;**

**wait(SB);**

**countB--;**

**if(countB==0) signal(mutex);**

**signal(SB);**

**}**

**}**

**main(){**

**cobegin**

**A(); B();**

**coend**

**}**

1. 学校羽毛球馆，馆内提供羽毛球拍和羽毛球若干。有A、B两组学生，A组学生每人都备有羽毛球拍，B组学生每人都备有羽毛球。任意一组学生只要能得到其他一种材料就可以打球。有一个可以存放一个球拍或一个羽毛球的筐子，当筐子中无物品时，管理员就随机的放一个球拍或一个羽毛球供学生取用，每次允许一个学生从中取出自己所需的材料，当学生从筐子中取走材料后允许管理员再放一件材料，请用信号量与P、V操作。

semaphore s, sa , sb, mutexa , mutexb ;   
s =1;mutexa =1;mutexb=1; sa =0; sb= 0 ;   
box (Racket , badminton ) ;

cobegin   
{   
process 保管员  
begin   
repeat   
P ( s ) ;   
take a material intobox ;   
if ( box ) = badminton then V ( sa ) ; //放球通知A组  
else V( sb ) ; //放球拍通知B组  
untile false ;   
end

Process A组学生  
begin   
repeat   
P ( sa ) ;   
P ( mutexa ) ;   
take the badminton from box ;   
V ( mutexa ) ;   
V ( s ) ;   
write a letter;  
untile false ;   
end

Process B组学生  
begin   
repeat   
P ( sb ) ;   
P ( mutexb ) ;   
take the Racket from box ;  
V ( mutexb ) ;   
V ( s ) ;   
write a letter ;   
untile false ;   
end   
}   
Coend

1. 有一体育器材保管员，他网球拍和网球若干。有A、B两组学生，A组学生每人都备有网球拍，B组学生每人都备有网球。任意一组学生只要能得到其他一种材料就可以打网球。有一个可以放一个球拍或一个网球的筐子，当筐子中无物品时，保管员就可任意放一个球拍或一个网球供学生取用，每次允许一个学生从中取出自己所需的材料，当学生从筐子中取走材料后允许保管员再放一件材料，请用信号量与P、V操作实现。

semaphore s, sa , sb, mutexa , mutexb ;   
s =1;mutexa =1;mutexb=1; sa =0; sb= 0 ;   
box (Racket , Tennis ) ;

cobegin   
{   
process 保管员  
begin   
repeat   
P ( s ) ;   
take a material intobox ;   
if ( box ) = Tennis then V ( sa ) ; //放球通知A组  
else V( sb ) ; //放球拍通知B组  
untile false ;   
end

Process A组学生  
begin   
repeat   
P ( sa ) ;   
P ( mutexa ) ;   
take the Tennis from box ;   
V ( mutexa ) ;   
V ( s ) ;   
write a letter;  
untile false ;   
end

Process B组学生  
begin   
repeat   
P ( sb ) ;   
P ( mutexb ) ;   
take the Racket from box ;  
V ( mutexb ) ;   
V ( s ) ;   
write a letter ;   
untile false ;   
end   
}   
Coend