例题 1（北京大学 1999 年）

有一个仓库，可以存放 A 和 B 两种产品，仓库的存储空间足够大，但要求：
(1)一次只能存人一种产品（(A 或 B);
(2)一 N< A 产品数量一 B 产品数量＜M

其中，N 和 M 是正整数。

试用“存放 A’和‘存放 B’以及 P 操作和 V 操作描述产品 A 和产品 B 的人库过程。

解答：

应先将表达式转换成制约条件，不可在程序中直接使用该表达式

将表达式分解为：

B 产品数量—A 产品数量<N

A 产品数量—B 产品数量＜M

可这样理解：

(1) 若只放人 A 产品，而不放入 B 产品，则 A 产品最多可放 M—1 次便被阻塞，即 A 进程每操作一次就应当将计数器减 1（计数器初值为 M—1），当计数器值为 0 时，进程程 A 被阻塞；每当放入一个 B 产品，则可令 A 产品的计数器增加 1，表明 A 产品可以多一次放入产品的机会；同理，

(2) 若只放人 B 产品，而不放入 A 产品，则 B 产品最多可；放 N 一 1 次便被阻塞，即 A 进程每操作一次就应当将计数器减 1（计数器初值为 N—1）。当计数器值为 0 时，进程 B 被阻塞；每当放人一个 A 产品，则可令 B 产品的计数器增加 1，表明 B 产品可以多一次放入产品的机会。

由此可见，该问题是一个同步控制问题。又因为一次仅允许一种产品人库，设置信号量 mutex 控制粮进程互斥访问临界资源（仓库）。过程如下：

```
begin
  mutex:＝1;
  Sa := M-1;
  Sb :=   N-1;
   Parbegin
     A 产品
      begin
         repeat
         P (Sa);
         P (mutex);
         A 人库;
         V (mutex);
         V (Sb);
         Until false;
      End;
    B 产品
      begin
        repeat
           P (Sb);
           P (mutex);
           B 人库;
```

V (mutex)；
V (Sa)；
Until false;
End;
rend;

例题 2（华中理工大学 1999 年试题）

设公共汽车上，司机和售票员的活动分别是：

司机：	售票员：
启动车辆	上乘客
正常行车	关车门
到站停车	售票
	开车门
	下乘客

在汽车不断地到站，停车，行驶过程中，这两个活动有什么同步关系？并用信号灯的 P, V 操作实现它们的同步。

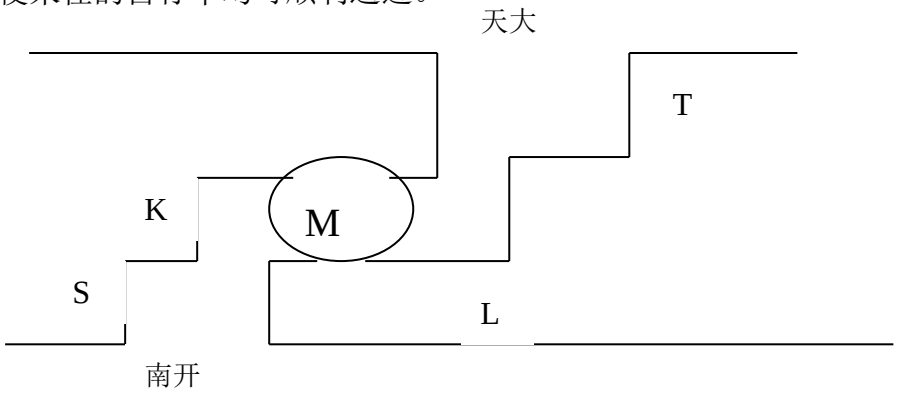
解答：

该题没有给出具体的控制关系，可根据常识自己定。如（1）售票员关车门后司机才可以启动车辆。（2）司机到站停车后，票员方可开车门。由此可见本题的控制关系较为简单。过程如下：

定义信号量 run, stop；
Stop := 0 ; run :=0；
Purbegin
司机：begin
L1：P (run);
启动车辆；
正常行车；
到站停车；
V (stop)；
Goto Ll；
End;
售票员：begin
L2：上乘客；
关车门；
V (run)；
售票；
P (stop)；
开车门；
下乘客；
gotoL2 ;
end;
parend.

例题 3（南开大学 1997 年试题）

在南开大学和天津大学之间有一条弯曲的小路，其中从 S 到 T 一段路每次只允许一辆自行车通过，但中间有一个小的“安全岛”M（同时允许两辆自行车停留），可供两辆自行车已从两端进人小路情况下错车使用，如图 4.3 所示。试设计一个算法使来往的自行车均可顺利通过。



解答:

本题临界资源较多，需仔细考虑。首先中间的安全岛 M 仅允许两辆自行车通过，应作为临界资源设置信号量。但仔细分析发现，在任何时刻进人小路的自行车最多不会超过两辆（南开和天大方向各一辆），因此不需为安全岛 M 设置信号量。在路口 S 处，南开出发的若干辆自行车应进行路口资源的争夺，以决定谁先进人小路 SK 段，为此设置信号量 S，用以控制路口资源的争夺；同理，设置信号量 T，控制天大方向自行车对路口 T 的争夺。又小路 SK 段仅允许一辆车通过，设置信号量 SK 初值为 1，同理设置小路 LT 段信号量 LT 初值为 1。

程序如下：

```
S ：= l;   T：=1;  SK ：＝1;    LT：=1;
Parbegin
  进程 P:（南开方向自行车）
   begin
   P(S) ;  {与其它同方向的自行车争夺路口 S}
   P(SK);  {同对面自行车争夺路段 SK}
    通过 SK;
   进人 M；  **
    V (SK); {一旦进人 M，便可释放路段 SK}
   P (LT) ; {同对面的自行车争夺路段 LT}
   通过 LT;
   V (LT)；{将路段 LT 释放}
    V(S);   {将路口 S 释放给同方向的正在路口 S 处等待的自行车}
    end，
  进程 Q:（天大方向自行车 ‘）
    begin
    P(T)；
    P(LT)；
```

通过 LT;
进人 M;
V(LT)；
P(SK)；
通过 SK;
V(SK)；
V(T)；
End；
Parend。

说明＊＊：

P 进程进人安全岛 M 后，释放了路段 SK，但没有释放路口 S，原因在于它是向对面的 4 进程释放路段资源 SK，而在 P 进程离开小路 LT 后，才会将路口 S 释放给其他 P 进程，如不这样，就会死锁。请考虑如下情况：两个方向各有一辆车前进，若在 P 进程到达安全岛 M 后，执行 V (S)及 V (SK)操作，则有可能使得同方向的其它 P 进程得到路段 SK 的使用权，而进人小路；同理，Q 进程到达安全岛后执行 V (LT)及 V (T)操作，有可能使得同方向的其它 Q 进程得到路段 LT 而进人小路。此时共有四辆车在整个路径中，最终出现死锁状态。

例题 4（北京理工大学 1996 年试题）

进程之间存在哪几种相互制约关系？各是什么原因引起的？下列活动分别属于哪种
制约关系？

(1)若干同学去图书馆借书；
(2)两队举行篮球比赛；
(3)流水线生产的各道工序；
(4)商品生产和社会消费。

解答：

有直接制约关系（即同步问题）和间接制约关系（即互斥问题）；同步问题是存在逻辑关系的进程之间相互等待所产生的制约关系，互斥问题是相互无逻辑关系的进程间竞争使用资源所发生的制约关系。

（1） 约属于互斥关系，因为书的个数是有限的，一本书只能借给一个同学；
（2） 属于互斥关系，篮球只有一个，两队都要争夺；
（3） 属于同步关系，各道工序的开始都依赖前道工序的完成；
（4） 属于同步关系，商品没生产出来，消费无法进行，商品未消费完，生产也无须
进行。

例题 5。（北京邮电大学 1999 年试题）

某招待所有 100 个床位，住宿者住人要先登记（在登记表上填写姓名及床位号），离去时要撤消登记（在登记表上删去姓名和床位号）。请给出住宿登记及撤消登记(退宿)过程的算法描述。

解答：

该题同有限缓冲区的生产者／消费者问题一致，缓冲单元为 100 个，顾客人住时执行生产者操作，登记过程对应于放产品过程，故对登记表的修改须互斥执

行，当床位客满时便不可人住（即放产品）. 顾客离开时执行消费者操作, 退宿过程对应于取产品过程，包括修改登记表，删去姓名（即取产品），并将床位归还（空出一个单元）。

过程: 略

例题 6。（南京大学 2000 年试题）

桌子上有一只盘子，最多可容纳两个水果，每次只能放人或取出一个水果。爸爸专向盘子中放苹果（apple），妈妈专向盘子中放橘子（orange），两个儿子专等吃盘子中的橘子，两个女儿专等吃盘子中的苹果。请用 P, V 操作来实现爸爸、妈妈、儿子、女儿之间的同步与互斥关系。

解答：

盘子为互斥资源，因可以放两个水果，empty 初值为 2; father 放苹果前先看看有无空间，若有则抢盘子，放人 apple。后向女儿发信号（V (apple)）; mother 放橘子前先看看有无空间，若有则抢盘子，放人橘子后向儿子发信号（V (orange)）；女儿先看有无苹果，若有则抢盘子，取走苹果后将盘子置空（V (empty)）；儿子先看有无橘子，若有则抢盘子，取走橘子后将盘子置空。

该题是生产者 / 消费者问题的变形，有两对生产者和消费者。生产者需指明是给哪个消费者的产品，但消费者取走产品后无须特别通知某个生产者，因为空出的缓冲区（盘子）可由两个生产者随意争夺。

设信号量 mutex 初值为 1，控制对盘子的互斥访问；apple 表示盘中苹果个数，orange 表示盘中橘子个数，初值均为 0.

```
parbegin
    father:
                begin
          Ll：  P( empty)；
                P(mmex )；
                放苹果；。
                V (mutex)；
                V(apple)；
                Goto Ll ;
                End;
    mother:
                begin
                L2: P(empty) ;
                P(mutex)；
                放橘子；
                V (murex )；
                V(orange)；
                Coto L2;
                End;
  daughter:
              begin
              L3: P(apple)；
              P(mutex)；
```

```
            取苹果
            V (murex)；
            V(empty)；
            Goto L3 ;
            End;
    son:
            begin
      L4:   P(orange);
            P(mutex)；
            取橘子
            V (mutex)；
            V (empty) ;
            GotoM；
            End;
  Parend
```

例题 7。

某工厂有两个生产车间和一个装配车间，两个生产车间分别生产 A, B 两种零件，装配车间的任务是把 A, B 两种零件组装成产品。两个生产车间每生产一个零件后都要分别把它们送到装配车向的货架 F1, F2 上，Fl 存放零件 A, F2 存放零件 B, Fl 和 F2 的容量均为可以存放 10 个零件。装配工人每次从货架上取一个 A 零件和一个 B 零件然后组装成产品。请用 PV 操作进行正确管理。

解答：

该题是生产者 / 消费者问题的变形，可认为一个消费者（装配工人）同两个生产者(A, B 车间)互斥使用两个缓冲区（Fl, F2），可设 mutexl，mutex2（初值为 1）控制进程对 Fl, F2 的互斥操作，另设一 emptyl, empty2（初值均为 10）, fulll, full2（初值均为 0）。

过程如下：

```
parbegin
A 车间：begin
      Ll：生产一个产品；
          P (emptyl)；
          P (mutexl);
          放人 Fl;
          V (mutexl)；
          V (full));
          Goto Ll;
          End;
 B 车间：begin
      L2：生产一个产品；
          P (empty2)；
          P (mutex2)；
          放人 F2;
```

V (mutex2);
V (full2)；
Goto L2；
装配工人：begin
L3：P（full 1）；
P (full2)；
P (mutexl);
P (mutex2);
取 A 和 B；
V (mutex1)；
V (mutex2)；
V (emptyl)；
V (empty2)；
Goto L3;
End;
Parend.

例题 8（北京邮电大学 1998 年试题）

某寺庙，有小、老和尚若干，有一水缸，由小和尚提水人缸（向缸中倒水）供老和尚饮用。水缸可容 10 桶水，水取自同一井中。水井径窄，每次只能容一个捅取水。水桶总数为 3 个。每次人、取缸水仅为 1 桶，且不可同时进行。试给出有关从缸中取水和向缸中倒水的算法描述。

解答：

应首先考虑清楚本题需要几个进程。从井中取水后向缸中倒水为连续动作，可算同一进程，从缸中取水为另一进程。

再考虑信号量．有关互斥的资源有水井（一次仅一个水桶进出）和水缸（一次入、取水为一桶），分别为之设信号量 mutexl , mutex2 控制互斥；

另有同步问题存在：三个水桶无论从井中取水还是人出水缸都是一次一个，应为之设信号量 count，抢不到水桶的进程只好等待；还有水缸满时，不可人水，设信号量 empty 控制人水量．水缸空时不可出水，设信号量 full，控制出水量。

mutexl：＝1；mutex 2 :＝1; empty:＝10; full：＝0 ;　　count：＝3;

parbegin

入水：　begin

Ll: P(empty)；

P(count) ;

P (mutexl) ;

从井中取水；

V(mutext1);

P(mutex2);

送人水缸；

V(mutex2);

V(count);

V(full)；

GotoLl；
End;
取水： begin
L2: P(full)；
P(count);
P(mutex2) ;
从缸中取水；
V (mutex2) ;
V(empty);
V(count)；
Goto L2；
End;
Parend.

例题 9（北方交通大学 1999 年试题）

有一阅览室，读者进人阅览室必须先在一张登记表（(TB)上登记，该表为每一座位设一个表目，读者离开时要消掉其登记信息，阅览室共有 t00 个座位，为了描述读者的动作，请用类 Pascal 语言和 P, V 操作写出进程间的同步算法。

约定：

1) flag 的值：0 座位空闲，1 座位被占用。

2)用语句 I=getflag (0)可搜索到一个空座位 i，用语句 i.falg =0 或 1，可给标志位赋值。

3）用 i = getname (readername)可搜索到某读者所登记的座位号 i;

用 i..name=0 或 i.name = readrname。可给姓名字段赋值，0 表示清除读者姓名。

4）计数信号量用 count，互斥信号量用 mutex .

解答：

该题所提供的已知条件和语句过多，容易给考生造成混乱，因此看到此题应先理清思路，先不要考虑约定中提供的各种语句。首先，按照惯常的同步互斥问题思路来思考。

读者进程：

讲入阅览室首先应检查是否有空位子。若没有则等待，若有则登记，可设信号量 count，控制各进程对座位的争夺；读者离去时，便可释放该座位。登记时要对表中各量进行修改，该过程相对于各进程而言为互斥操作，因此设信号量 mutex，控制读者进人和离去时对表的互斥修改，题目所提供的各种语句只是用在修改登记表时所进行的具体操作，对本题的同步互斥问题无任何影响，照抄即可。

count：=100; mutex : = 1；
读者进程：begin
进人阅览室；
P(count)；
P(mutex) ;
i=getflag(0)；
i..flag=1；

i. name=rademame;
V (mutex)；
坐下看书；
P(mutex) ;
i = getname(mademame)；
i. name=0; i.flag=0;
V (mutex)；
V (count)；
离开；
end.

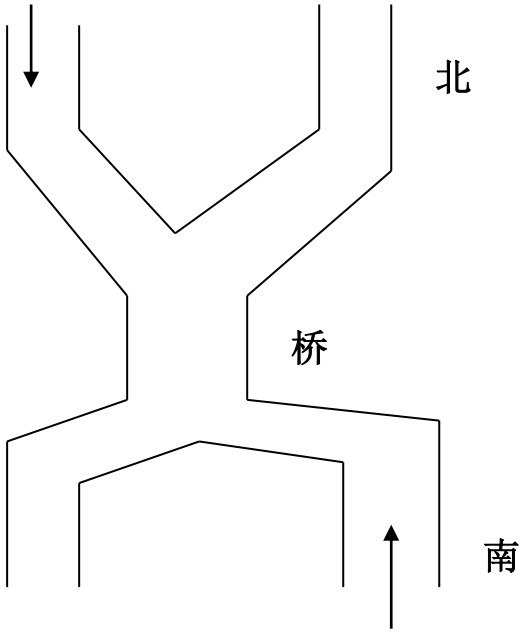
例题 10。用信号量描述 4 乘 100 米接力过程

解答：

P1:	P2: P(Sl);	P3: P(S2);	P4: P(S3);
起跑，前进 l00m;	起跑，前进 l00m;	起跑，前进 l00m;	起跑，前进 l00m;
V(S1);	V(S2);	V(S3)；	到达终点。

例题 11（北京大学 1992 年试题）

有桥如图 4.15 所示。车流如箭头所示。桥上不允许两车交会，但允许同方向多辆车依次通行（即桥上可以有多个同方向的车）。用 P, V 操作实现交通管理以防桥上堵塞。



解答：

设置: 信号量 mutex 1, murex 2，初值为 1; bridge 初值为 1; 计数器 count 1, count 2，初值为 0。

```
北方:                              南方：
P(mutex 1)；                        P(mutex 2) ;
count 1: =count 1 + 1;              count-2: = count 2＋1;
if count 1=1 then P(bridge) ;       if count 2=1 then P(bridge) ;
V(mutexl);                          V(mutex2);
过桥；                               过桥；
P(mutexl);                          P(mutex2);
count 1: = count 1 一 1;             count 2: = count 2 一 1;
if count 1=0 then V(bridge) ;       if count 2=0 then V(bridge) ;
V(mutex1);                          V(mutex2);
```

例题 12。“吸烟者”问题：假设一个系统有三个吸烟者（Smoker）进程和一个供货商（(Agent)进程。每个吸烟者连续不断地制造香烟并吸掉它。但是，制造一支香烟需要三种材料；烟丝、纸、火柴。第一个吸烟者进程有纸，第二个有烟丝，第三个有火柴。供货商进程可以无限地提供这三种材料。供货商将两种材料一起放在桌上，持有另一种材料的吸烟者即可以制造一支香烟并吸掉它。**供货商一次只能处理一个吸烟者的请求**。当此吸烟者抽香烟时，它发出一个信号通知供货商进程，供货商进程马上给出另外两种材料，如此循环往复。试编写一个程序使供货商与吸烟者同步执行。

解答：

```
var smoker1, smoker2, smoker3 :   Boolean; /*表明 smoker i  是否需要抽烟*/
semaphore    t, w, m;     /* t  一烟丝信号量，  w一纸信号量，  m一火柴信号量*/
binary semaphore    mutex1, mutex2, mutex3,
                          /*控制 agent 每次只能为一个吸烟者按顺序服务*/
binary semaphore     smoker_mutex;
                          /*控制三个吸烟者每次只能有一个提出申请并从桌子上
                            取材料  */


begin
smoker1:= false ; smoker2:= false; smoker3:= false;
t:=0; w:=0; m:=0;
mutex1 :=1;    mutex2 :=1;    mutex3 :=1;    smoker_mutex :=1;

Agent:
Loop
   P(mutex1);
   If somker1 =true         /*第一个拥有纸的吸烟者想吸烟*/
    then    V(t);           /*提供烟丝*/
            V(m);           /*提供火柴*/
            somker1 =false;
   V(mutex1);
   P(mutex2);
   If somker2 =true         /*第二个拥有烟丝的吸烟者想吸烟*/
    then    V(w);           /*提供纸*/
```

```
            V(m);            /*提供火柴*/
            Somker2 =false;
    V(mutex2);
    P(mutex3);
    If somker3 =true         /*第三个拥有火柴的吸烟者想吸烟*/
     then   V(w);             /*提供纸*/
            V(t);            /*提供烟丝*/
            Somker3 =false;
    V(mutex3);
Endloop

Smoker1:
    Loop   P(smoker_mutex)     /*判断是否有其它吸烟者正在向供货商提出申请*/
            P(mutex1)          /*判断所需的烟丝和火柴是否已经放在桌子上*/
            Smoker1 := true;
            V(mutex1);
            P(m);
            P(t);
            V(smoker_mutex)
     Endloop
```

Smoker2 和 Smoker3 与 Smoker1 类似。