例 3.1 一个单 CPU 的操作系统共有 n 个进程，不考虑进程状态过度情况：

（1）给出运行进程的个数。

（2）给出就绪进程的个数。

（3）给出等待进程的个数。

解 （1）一个运行进程 （2）m 个就绪进程 （3）n-m-1 个等待进程

### 3.3.2 进程同步概念

例 3.2 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 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）属于互斥关系，因为书的数目是有限的，一本书只能借给一个同学；

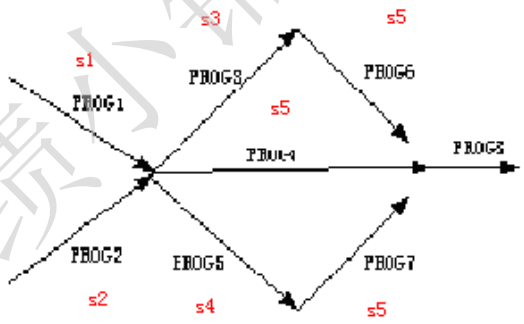
（2）属于互斥关系，因为篮球只有一个，两队都要争夺；

（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，商品没有生产出来，消费无法进行，商品未消费完，生产也无须进行。

### 3.3.3 信号量概念及应用

例 3.3 设有 8 个程序 prog1,prog2,…prog8。它们在并发系统中执行时有如图 3.4 所示的制约关系,试用 P、V 操作实现这些程序间的同步。




并发系统执行图

分析 本题是用来检查考生对使用 P、V 操作实现进程间同步的掌握情况。一般地，若要求进程 B 在进程 A 之后方可执行时，只需在进程 B 执行前进行 P 操作，而在进程 A 执行完时对同一信号量进行 V 操作即可。本题要求列出 8 个进程（程序）的控制关系，使题目显得较为复杂。但当对进程间的同步理解透彻后，应不难写出对应的程序。解这一类问题还应注意的一点是，要看清图示的制约关系，不要漏掉或多出制约条件。

解 图并发程序间的同步如下所示：

设置信号量初值：s1:=s2:=s3:=s4:= s5:=0

PARBEGIN

```
prog1:BEGIN
   do all work;
   V(s1);
   V(s1);
   V(s1);
END
prog2:BEGIN
   do all work;
   V(s2);
   V(s2);
   V(s2);
END
prog3:BEGIN
   P(s1);
   P(s2);
   do all work;
   V(s3);
END
prog4:BEGIN
   P(s1);
   P(s2);
   do all work;
   V(s5);
END
Prog5:BEGIN
   P(s1);
   P(s2);
   do all work;
   V(s4);
END
prog6:BEGIN
   P(s3);
   do all work;
   V(s5);
END
prog7:BEGIN
   P(s4);
   do all work;
   V(s5);
END
prog8:BEGIN
   P(s5);
   P(s5);
   P(s5);
```

**do all work;**
**END**
**PAREND**

**例 3.4** 有个寺庙，庙中有小和尚和老和尚若干人，有一只水缸，由小和尚提水入缸给老和尚饮用。水缸可容 10 桶水，水取自同一口水井中。水井径窄，每次仅能容一只水桶取水，水桶总数为 3 个。若每次只能入缸 1 桶水和取缸中 1 桶水，而且还不可以同时进行。试用一种同步工具写出小和尚和老和尚入水、取水的活动过程。

解　本题为两个进程共享两个缓冲区的问题。

首先考虑本题有几个进程：从井中取水后向缸中倒水此为连续动作，为一个进程；从缸中取水为另一个进程。

其次，考虑信号量。有关互斥的资源有：水井和水缸。水井依次仅能一个水桶进出，水缸一次入水为一桶。分别设互斥信号量为：mutex1 和 mutex2 控制互斥。有关同步问题为：三个水桶无论从井中取水还是入处水缸都是一次一个，应为它设信号量 count，抢不到水桶的进程只好等待。水缸满时不可入水，设信号量 empty，控制水量，水缸空时不可出水，设信号量 full，控制水量。

设置信号量初值：mutex1:=mutex2:=1; count:=3; empty:=10; full:=0;

```
parbegin
{
    小和尚打水进程:
      begin
       L1: P(empty);          /*水缸满否?
           P(count);          /*取得水桶
           P(mutex1);         /*互斥从井中取水
           从井中取水;
           V(mutex1);
           P(mutex2);         /*互斥使用水缸
           倒水入缸;
           V(mutex2);
           V(count);          /*归还水桶
           V(full);           /*多了一桶水
           Goto L1;
      End
    老和尚取水进程:
      begin
       L2: P(full);           /*有水吗?
           P(count);          /*申请水桶
           P(mutex2);         /*互斥取水
           从缸中取水;
           V(mutex2);
           V(count);          /*归还水桶
           V(empty);          /*水缸中少了一桶水
          Goto L2;
       end
}
parend.
```

**例 3.5**　有三个并发进程Ｒ、Ｍ、Ｐ共享一个缓冲区Ｂ，进程Ｒ负责从输入设备读入一条记录，每读一条记录后把它存放在缓冲区Ｂ中；进程Ｍ在缓冲区Ｂ中加工进程Ｒ存入的

记录；进程Ｐ把加工后的记录打印输出。缓冲区Ｂ中每次只能存放一条记录，当记录被加工输出后，缓冲区Ｂ中才可存放另一条新记录。请用Ｐ、Ｖ操作为同步机制来描述它们并发执行时能正确工作的程序。

分析　本题是三个进程共享一个缓冲区的问题。从题中看出，R、M、P 这三个进程严格按照先做进程 R，然后做进程 M，最后做进程 P。只有进程 P 执行完后，才可以做下一次的 R，M，P。因此，要合理设计 P、V 操作的顺序与设置信号量的初值。

解

设置信号量初值：R:＝１；M:=P:=0;

parbegin

进程 R

**L1：**

**从输入设备中读取一条记录；**

**P(R)；**

**将读入记录存入缓冲区；**

**V（M）；**

**goto L1**

**进程 M**

**L2：**

**P（M）；**

**从缓冲区中取出数据信息进行加工，并将其存入缓冲区；**

**V（P）**

**goto L2**

**进程 P：**

**L3：**

**P（P）**

**输出缓冲区的信息；**

**V（R）**

**goto L3**

**parend;**

**例 3.6**　若有三个进程 **A、B 和 C** 协作解决文件打印问题；**A** 将文件记录从磁盘读入主存的缓冲区 **buffer1**，每执行一次读一个记录；**B** 将缓冲区 **buffer1** 的内容复制到缓冲区 **buffer2**，每执行一次复制一个记录；**C** 打印缓冲区 **buffer2** 的内容，每执行一次打印一个记录。缓冲区的大小和一个记录大小一样。请用 **P、V** 操作来保证文件的正确打印。

分析 本题是三个进程共享二个缓冲区的问题。它是一个典型的生产者—消费者问题，其中的难点在于进程 **B** 既是生产者又是消费者，处理不好可能造成同步错误或死锁。

解 A、B、C 三个进程协作解决文件打印问题的流程为:

设置信号量初值：

**mutex1:=mutex2:=1;**

**avail1:=avail2:=1;**

**full1:=full2:=0;**

**BEGIN**

**PARBEGIN**

**进程 A**

**L1：read from disk;**

**P(avail1);**

```
        P(mutex1);
        put to buffer 1;
        V(full1);
        V(mutex1);
        goto L1;
进程 B
  L2：P(full1);
        P(mutex1);
        get form buffer 1;
        V(avail1);
        V(mutex1);
        P(avail2);
        P(mutex2);
        put to buffer 2;
        V(full2);
        V(mutex2);
        goto L2;
进程 C
  L3：P(full2)
        P(mutex2);
        get form buffer 2;
        V(avail2);
        V(mutex2);
        print record;
        goto L3;
PAREND
END
```

程序中 **mutex1** 和 **mutex2** 是两个公用信号量，用于控制进程对缓冲区 **buffer1** 和缓冲区 **buffer2** 这两个临界资源访问的互斥。**avail1**、**full1**、**avail2** 和 **full2** 分别对应两个缓冲区，其中 **avail1**、**avail2** 初值为 **1**，表示可以利用的缓冲区数目为 **1**；**full1**、**full2** 的初值为 **0**，表示存在于缓冲区内的数据的个数为 **0**。通过对这两组私用信号量 **P**、**V** 操作，就实现了进程的同步。

例 3.7　进程 P1 使用缓冲区 buffer 向进程 P2, P3, P4 发送消息，要求每当 P1 向 buffer 中发消息时，只有当 P2, P3, P4 进程都读取这条消息后才可再向 buffer 中发送新的消息。利用 P、V 原语描述如图所示进程的动作序列。

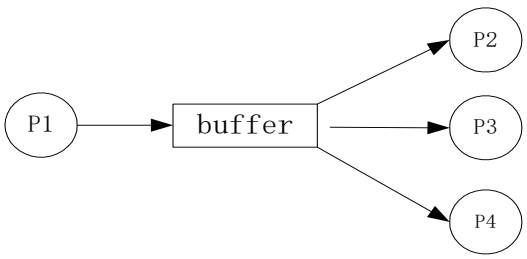
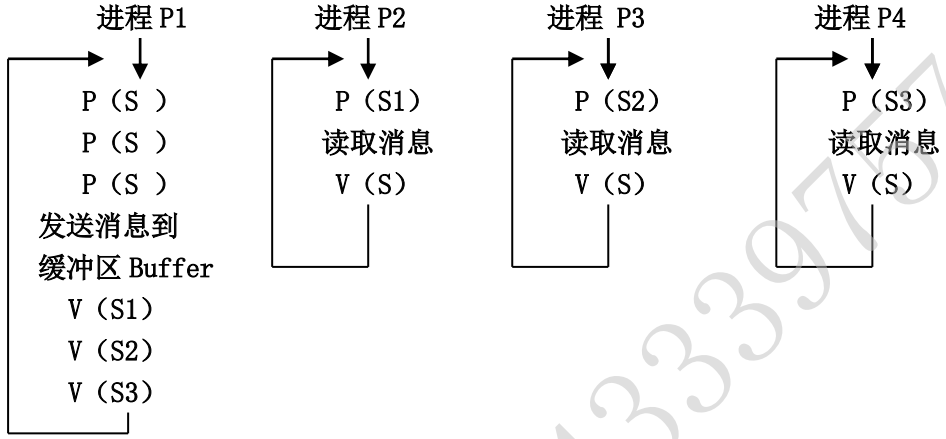



图　进程的动作序列

分析　本题是一个生产者三个消费者共享一个缓冲区问题。它是生产者和消费者问题的一个变形，在生产者和消费者问题中是共用一组缓冲区，每一个缓冲区只需要写一次，读一次。而在这个问题中，每个缓冲区只需写一次，但需读三次。本题在解题的过程中，我们可以把这一组缓冲区看做三组缓冲区。这样一来，每一个生产者需要同时写缓冲区组中相应的三个缓冲区，而每一个消费者只需读它自己所对应的那组缓冲区中对应的单元即可。生产者须在三个缓冲区都为空时才可写入。

解　设信号量初值为：S1=S2=S3=0，S =3




例 3.8 桌上有一个空盘，允许存放一个水果。爸爸可向盘中放苹果，也可向盘中放橘子，儿子专等吃盘中的橘子，女儿专等吃盘中的苹果。规定当盘空时一次放一个水果供吃者取用，请用 P，V 原语实现爸爸、儿子、女儿三个并发进程的同步。

分析　本题是一个生产者两个消费者共享一个缓冲区问题，生产者一次可生产一个不同种类的产品本题的题意是：

① 爸爸、儿子、女儿共用一个盘子，且盘中一次只能放一个水果。

② 当盘空时，爸爸可将一个水果放入果盘中。

③ 若放入盘中的是橘子，允许儿子吃，女儿必须等待。

④ 若放入盘中的是苹果，允许女儿吃，儿子必须等待。

因此，上述问题实际上是生产者消费者问题的一种变形。这里，生产者放入缓冲区的产品有两类，消费者也有两类，每类消费者只消费其中固定的一类产品。

解　设置三个信号量：

S，　初值为 1，用于爸爸、儿子、女儿三个进程间的互斥，表示盘中是否为空。

SO，　初值为 0，用于爸爸、儿子儿两个进程间的同步，表示盘中是否有橘子。

SA，　初值为 0，用于爸爸、女子两个进程间的同步，表示盘中是否有苹果。

三个进程之间的同步描述如下：

| father 进程 | son 进程 | daughter 进程 |
|---|---|---|
| L1： | L2： | L3： |
| P（S） | P（SO） | P（SA） |
| 将水果放入盘中 | 从盘中取出橘子 | 从盘中取出苹果 |
| if (放入是橘子) V（SO） | V(S) | V(S) |
| else　V(SA) | 吃橘子 | 吃苹果 |
| goto L1 | goto L2 | goto L3 |

例 3.9　某高校计算机开设一系列网络课程并安排学生上机实习，假设机房共有 2m 台机器，有 2n 名学生选该课，规定：

(1) 每两个学生组成一组，但占有一台机器，协同完成上机实习；

**(2) 只有一组的两个学生都到齐，并且此时机房有空闲机器时，该组学生才能进入机房；**

**(3) 上机实习完成后，由一名教师检查，检查完毕，这一组学生同时离开机房。**

**试用 P，V 操作模拟上机实习过程。**

**解 上机实习过程如下：**

**设置信号量：student=0; computer=2m,; enter=finish=check=0**

**设置变量：rc=0**

**parbegin**

**学生进程：BEGIN**

**V(student)　　表示有学生到达**

**If rc=0 then P(computer)**

**获取一台计算机**

**else rc=0**

**P(enter)　　等待允许进入**

**协同完成上机实习**

**V(finish)　　表示实习完成**

**P(check)　　等待教师检查**

**V(computer)　　释放计算机资源**

**END**

**教师进程:BEGIN**

**L1：P(finish)　　等待学生实习完成**

**P(finish)　　等待另一学生实习完成**

**教师检查工作**

**V(check)　　表示检查完成**

**V(check)　　表示检查完成**

**goto L1**

**END**

**管理员进程：BEGIN**

**L2：P(student) 等待学生到达**

**P(student) 等待另一学生到达**

**V(enter) 允许学生进入**

**V(enter) 允许学生进入**

**END**

**parend**

**例 3.10** 有一个仓库存放两种零件 A 和 B，它们最大库存容量各为 m 个。有一车间不断地取 A 和 B 进行装配，每次各取一个。为避免零件锈蚀，遵循先入库者先出库的原则。有两组供应商分别不断地供应 A 和 B（每次一个）。为保证齐套和合理库存，当某种零件的数量比另一种的数量超过 n(n<m)个时，暂停对数量大的零件的进货，集中补充数量少的零件。试用 P、V 操作正确地实现之。

分析 本题不是直接给出进程之间的制约关系，而是给出一个数量上的控制关系。在做这样的题目，千万不可在表达式中使用信号量的值来控制程序的流程。

解 控制关系有四个：

（A 的数量-B 的数量）≤n，

（B 的数量-A 的数量）≤n，

A 的数量≤m，

**B 的数量≤m。**

**根据这种控制关系，零件进货与使用的流程如下所示：**

**设置信号量为：**

**mutex=1** 用于互斥信号量；

**availa= m** 用于控制操作 A 的执行次数；

**availb=m** 用于控制操作 B 的执行次数；

**sa=sb=n** 用于控制操作 A、B 之间的制约关系；

**fulla=0** 用于同步控制，表示当前供应 A 的数量；

**fullb=0** 用于同步控制，表示当前供应 B 的数量；

```
PARBEGIN
供应 A：BEGIN
  L1:
      P(availa);
      P(sa);
      P(mutex);
      供应 A;
      V(fulla);
      V(sb);
      V(mutex);
      goto L1;
  END
供应 B：BEGIN
  L2:
      P(availb);
      P(sb);
      P(mutex);
      供应 B;
      V(fullb);
      V(sa);
      V(mutex);
      goto L2;
  END
取：BEGIN
  L3:
      P(fulla);
      P(fullb);
      P(mutex);
      取 A 或 B;
      V(availa);
      V(availb);
      V(mutex);
      goto L3;
  END
PAREND
```

### 3.3.4 经典进程同步问题

**例 3.11** 设有图所示工作模型。有四个进程 P0、P1、P2 和 P3 和四个信箱 M0、M1、M2 和 M3。进程间借助相邻的信箱传送消息：Pi 每次从 Mi 中取一条消息，经加工送入 M（i+1）mod 4 中。M0，M1，M2，M3 分别设有 3，3，2，2 个格子，每个格子放一条消息。初始时 M0 装了 3 条消息，其余为空。写出使用信号量实现进程同步与互斥的流程。

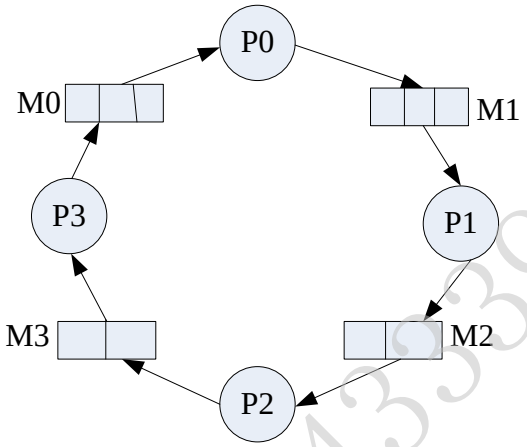



图　四个进程共享四个信箱工作模型

**解** 本题可看成四对生产者和消费者问题。用 **mutexi**（i=0，1，2，3）控制四个信箱的互斥使用，初值均为 1；emptyi（i=0，1，2，3）表示各信箱中空单元的个数，初值分别为 0，3，2，2；fulli（i=0，1，2，3）表示各信箱中信息的个数，初值分别为 3，0，0，0。

```
parbegin
{
    P0 进程：
      begin
       L1：P(full0);
           P(mutex0);
           从 M0 中取一个消息;
           V(mutex0);
           V(empty0);
           加工消息;
           P(empty1);
           P(mutex1);
           送消息进入 M1;
           V(mutex1);
           V(full1);
           Goto L1;
       End;
    P1 进程：
      begin
       L2：P(full1);
           P(mutex1);
           从 M1 中取一个消息;
           V(mutex1);
           V(empty1);
           加工消息;
```

```
    P2 进程：
    begin
      L3：P(full2);
          P(mutex2);
          从 M3 中取一个消息;
          V(mutex2);
          V(empty2);
          加工消息;
          P(empty3);
          P(mutex3);
          送消息进入 M3;
          V(mutex3);
          V(full3);
          Goto L3;
          End
    P3 进程：
    begin
      L4：P(full3);
          P(mutex3);
          从 M3 中取一个消息;
          V(mutex3);
          V(empty3);
          加工消息;
```

```
        P(empty2);                          P(empty0);
        P(mutex2);                          P(mutex0);
        送消息进入 M2;                      送消息进入 M0;
        V(mutex2);                          V(mutex0);
        V(full2);                           V(full0);
        Goto L2;                            Goto L4;
    End;                                End
```

例 3.12　多个进程共享一个文件，其中读文件的进程称之为读者，写文件的进程称为写者。读者可以同时读，但是写者只能独立地写。请：

(1) 说明进程间的相互制约关系。

(2) 用 P，V 操作写出读者优先的同步算法。

(3) 用 P，V 操作写出写者优先的同步算法，即一旦有写者到达，后续的读者都必须等待，而不管是否有读者在读文件。

解：　（1）进程之间相互制约关系有三种：一是读者之间允许同时读；二是读者与写者之间须互斥进行；三是写者之间须互斥写；

（2）　读者优先

```
读者 i:
        {
        P(mutex);
          readcount++;
          if (readcount==1)
                P (w);
        V(mutex);
              读
        P(mutex);
          readcount--;
          if (readcount==0)
             V(w);
        V(mutex);
        };

写者 j:
        {
        P(w);
        写
        V(w);
        };
```

(3) 写者优先的同步算法

写者优先

条件：

1）多个读者可以同时进行读

2）写者必须互斥（只允许一个写者写，也不能读者写者同时进行）

3）写者优先于读者（一旦有写者，则后续读者必须等待，唤醒时优先考虑写者）

解　写者优先的同步算法为：为了提高写者的优先级，我们增加了一个信号量 W，用

以在写进程到达时封锁其后续的读者进程。相应控制算法如下：

```
设置信号量初值为：mutex1:=1; mutex2:=1;
设置变量初值为：rc:=0; w:=1
PARBEGIN
Reader:BEGIN
    P(w);                    一旦有写者，后续的读者就不能进来
    P(mutex1);
    rc:=rc+1;
    IF rc=1 THEN P(mutex2);
    V(mutex1);
    V(w);
    Reading the file;
    P(mutex1);
    rc:=rc-1;
    IF rc=0 THEN V(mutex2);
    V(mutex1);
END
Writer:BEGIN
    P(w);
    P(mutex2);
    Writeing the file;
    V(mutex2);
    V(w);
END
PAREND
```

例 3.13　有一个阅览室，共有 100 个座位，读者进入时必须先在一张登记表上登记，该表为每一座位列一目录，包括座位号和读者姓名等，读者离开时要消掉登记的信息，试问：

（1）　为描述读者的动作，应编写几个程序，设置几个进程？

（2）　试用 PV 操作描述读者进程之间的同步关系。

解 读者的动作有两个，一是填表进入阅览室，这时要考虑阅览室是否有座位；一时读者阅读完毕，离开阅览室，这时的操作要考虑阅览室里是否有读者。读者在阅览室读书时，由于没有引起资源的变动，不算动作变化。

信号量有三个：

```
seates     表示阅览室是否有座位（初值为 100，代表阅览室的空座位数）
readers    表示阅览室的读者数，初值为 0
mutex      用于互斥，初值为 1
读者进入阅览室进程：
    P(seates)
    P(mutex)
        填写登记表
    V(mutex)
    V(readers)
```

**读者进入阅览室进程：**

**P(readers)**
**P(mutex)**
**消掉登记**
**离开阅览室**
**V(mutex)**
**V(seates)**

**例 3.14 哲学家 A、B、C 和 D 在一起讨论问题，讨论间隙这四位哲学家进餐，每人进餐时都使用刀叉各一把，但餐桌上只在两个人之间放一套餐具。请用信号量及 P、V 操作说明这四位哲学家的同步、互斥过程。**

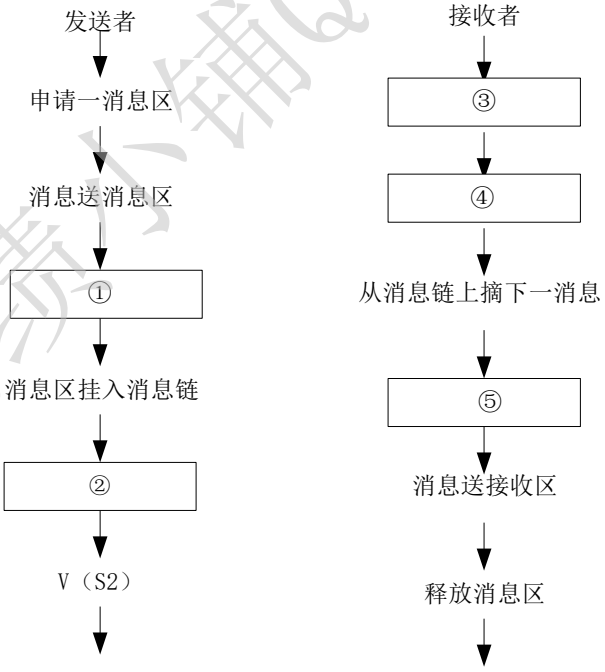
**解　为了避免死锁发生，可作如下规定：设置信号量 S1、S2、S3、S4，其初值全为 1。各位哲学家进程就餐情况如下：**

**Pi (i=1,2,3) ：**
**P（Si）;**
**P（S（i+1））;**
**eating;**
**V(Si);**
**V（S（i+1））;**

**P4：**
**P（S4）;**
**P（S1）;**
**eating;**
**V(S4);**
**V（S1）;**

### 3.3.5 消息传递通信

**例 3.15**　如图所示的是高级通讯原语 SEND 和 RECEIVE 不完整的框图。请填充适当的 P、V 操作，并说明所用信号量的意义和初值。




**分析**　从图中看出，发送者进程和接收者进程之间的同步关系是：发送者进程生成的信息送入消息链中，接收者进程从消息链中接收信息。由于发送者进程产生一个消息并链入消息链后用 V 操作增加消息计数并唤醒接收者进程，表示发送者进程和接收者进程是通过信

号量 S 实现同步的，因此接收者进程应该在取信息之前先使用一个 P 操作来查看消息链上是否有消息，若无消息，则阻塞自己。另外，发送者和接收者对消息链的访问应使用信号量进行互斥，即在访问前使用 P 操作，在访问后使用 V 操作。

**解** 由上述分析可知：

① P(S1) ②V(S1) ③P(S2) ④P(S1) ⑤V(S1)

其中 S1 是用于控制互斥访问消息链的互斥信号量，其初值为 1；S2 是用于记录消息个数的同步信号量，其初值为 0。

### 3.3.6 处理机调度

**例 3.16 有五个批处理的作业 A，B，C，D，E 几乎同时到达一个计算中心，估计的运行时间分别为 2,4,6,8,10min，它们的优先权分别为 1,2,3,4,5(5 为最高优先级)。请用下面的调度算法，分别计算作业的平均周转时间(忽略作业的切换开销)：**

**(1) 优先级调度；**

**(2) 时间片轮转(时间片为 2min)；**

**分析：本题是对给出的一个作业序列，使用四种不同的调度算法模拟作业的调度与执行过程，并给出对于这个作业序列的各种作业调度算法的平均周转时间，从而可以比较不同调度算法的性能。**

**题中所指的五个作业几乎同时到达，其含义是任何调度算法都可以认为这五个作业是同时到达的，因此，在调度过程中不需要考虑作业到达的顺序。**

**(1) 最高优先级优先**

**解(1) 最高优先级优先算法**

| | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 | 30 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | | | | | | | | | | | | | | A | A |
| B | | | | | | | | | | | | | B | | B | |
| C | | | | | | | | | | C | | | C | | | |
| D | | | | | | D | | | | D | | | | | | |
| E | E | | | | | E | | | | | | | | | | |

注:红颜色字母表示该作业完成.

**(1)最高优先级优先算法**

**各作业的执行结束时间分别为：30，28，24，18，10。**

**作业的平均周转时间为：**

$$T=\sum Ti/\ n\ =(30+28+24+18+10)/\ 5=22(min)$$

**(2) 时间片轮转(时间片为 2min)算法**

| | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 | 26 | 28 | 30 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | A | A | | | | | | | | | | | | | | |
| B | | B | | | | B | B | | | | | | | | | |
| C | | | C | | | | C | | | C | C | | | | | |
| D | | | | D | | | | D | | | D | | D | D | | |
| E | | | | | E | | | | E | | | E | | E | E | E |

注:红颜色字母表示该作业完成.

(2) 时间片轮转(时间片为2min)算法

根据表3.5的分析，各作业的执行结束时间分别为：2，12，20，26，30。作业的平均周转时间为：

$T=\sum Ti/n=(2+12+20+26+30)/5=18(min)$

例 3.17　一个具有两道作业的批处理系统，作业调度采用短作业优先的调度算法，进程调度采用基于优先级的强占式调度算法，如下表的作业序列（表中给出的作业优先数即为进程优先数，优先数越小优先级越高）。要求：

（1）　列出所有作业进入内存时间及结束时间。

（2）　计算平均周转时间。

| 作业名 | 到达时间 | 估计运算时间 | 优先数 | 装入时间 | 运行时间 | 结束时间 | 周转时间 |
|---|---|---|---|---|---|---|---|
| A | 8：00 | 50 分 | 4 | 8：00 | 8：00 | | |
| B | 8：20 | 40 分 | 2 | | | | |
| C | 8：30 | 60 分 | 3 | | | | |
| D | 8：50 | 30 分 | 6 | | | | |

例 3.18 一个实时系统使用了 4 个周期时间，其周期分别为 50ms，100ms，200ms 和 250ms。假设这 4 个周期事件分别需要 35ms，20ms，10ms 和 x ms 的 CPU 时间。保持系统可调度的最大 x 值是多少？

解　该实时系统是以周期进行调度，即 m 个周期事件，事件 i 以周期 $P_i$ 发生，需要 $C_i$ 秒钟 CPU 时间处理一个事件，那么可处理负载的条件是：$\sum_{i=1}^{m}\frac{Ci}{Pi}\le 1$。因此，使用 $\frac{35}{50}+\frac{20}{100}+\frac{10}{200}+\frac{x}{250}$ 为调度，他必须小于 1，于是 x 必须小于 12.5ms。

**死锁**

例 3.19 假设现在有同类资源 m 个供 n 个进程共享，如果每个进程最多需要 x 个资源(1≤x≤m)，并且各进程的最大需求量之和小于（m+n），证明该系统不会发生死锁。

解　某系统有同类资源 m 个，可并发执行且共享该资源的进程最多 n 个，而每个进程申请该类资源的最大量 x 个(1≤x≤m)，只要 n(x-1)+1≤m 成立，则系统一定不会发生死锁。

nx≤m+n+1

例 3.20 一个系统有 4 个进程和 5 个可分配资源，当前分配和最大需求如下：

已分配资源　　最大需求量　　可用资源

| | | | |
|---|---|---|---|
| 进程 A | 1 0 2 1 1 | 1 1 2 1 3 | 0 0 x 1 1 |
| 进程 B | 2 0 1 1 0 | 2 2 2 1 0 | |
| 进程 C | 1 1 0 1 0 | 2 1 3 1 0 | |
| 进程 D | 1 1 1 1 0 | 1 1 2 2 1 | |

若保持该状态是安全状态，x 的值是多少？

解　需要的矩阵如下所示：

0 1 0 0 2
0 2 1 0 0
1 0 3 0 0
0 0 1 1 1

如果 x 是 0，会立即出现死锁。如果 x 是 1，进程 D 能够运行结束。当它结束时，可用向量是 1 1 2 2 1。不幸的是此时也会出现死锁。如果 x 是 2，D 进程运行后，可用向量是 1 1 3 2 1，C 进程也可以运行，C 进程结束后释放自己的资源，于是可用的向量是 2 2 3 3 1，那么 B 进程就会运行并结束，然后是 A 进程。因此，避免死锁发生的 x 的最小值是 2。

例 3．21 在一个分时系统中，为每个进程分配时间片可以随进程的执行特点和执行情况而变化。现在有两类进程，一类进程经常产生中断，另一类进程中断次数很少，请问这两类进程哪类应该分得长的时间片，哪类应该分得短的时间片？为什么？为哪类进程指定高优先级？为哪类进程指定低优先级？

解　经常产生中断的进程应该分配较短的时间片，很少产生中断的进程分配较长的时间片。经常产生中断的进程运行的时间相对较短，即使给它长的时间片，也可能在时间片未使用完前因中断而让出处理器，所以只需分配较短的时间片。中断次数很少的进程，它连续运行的时间较长，为减少调度次数应给较大的时间片，尽可能让它较长时间占有处理器运行，可减少系统在调度上的花费。

让经常产生中断的进程优先级高于中断次数少的进程。因为这类进程经常产生中断，当他占用 CPU 后因某事件主动让出 CPU 时，就可以让其他进程使用 CPU，使 CPU 与其他部件并行工作。