6一条河上架设了由若干个桥墩组成的一座桥。若一个桥墩只能站一个人，过河的人只能沿着桥向前走而不能向后退。过河时，只要对岸无人过，就可以过；但不允许河对岸的两个人同时过，以防止出现死锁。请给出两个方向的人顺利过河的同步算法。

解：假设一座桥由N个桥墩，也即最多允许有N个人同向过河，用一个计数器R记录同时过河的人数(2’)。用S1信号量保护计数器，其初值为1，R的初值为0；互斥使用桥的信号量用S表示，其初值为1。(2’)

同步算法描述如下：

procedure goriver()

begin

L:P(S1); //为同时过河,申请对计数器计数

If R>N begin V(S1); goto L; end //同方向过河的人站满桥墩时,重新申请计数

R=R+1;

If R==1 P(S); //申请过河

V(S1); //释放计数器的使用权 (3’)

占有一个桥墩,并顺序过河到对岸;

P(S1);

R=R-1;

If R==0 V(S); //如果已经无同向的人过河,释放占用权

V(S1); (3’)

end.

使用信号量解决: 桌子上有一个篮子，最多放两个水果，每次只能放or 拿一个水果，爸爸只放orange，妈妈只放apple，儿子只拿orange、女儿只拿apple

1. Semaphore mutex=1;
2. Semaphore empty=2;
3. Semaphore apple=0;
4. Semaphore orange=0;
5. main()
6. {   Cobegin
7. {   进程father                    //父亲进程
8. {   while (true)
9. {   P(empty);           //减少盘中可放入的水果数
10. P(mutex);           //申请向盘中取、放水果
11. 向盘中放苹果;
12. V(mutex);           //允许向盘中取、放水果
13. V(apple);           //递增盘中的苹果数
14. }
15. }
16. 进程mother                    //母亲进程
17. {   while (true)
18. {   P(empty);           //减少盘中可放入的水果数
19. P(mutex);           //申请向盘中取、放水果
20. 向盘中放桔子;
21. V(mutex);           //允许向盘中取、放水果
22. V(orange);          //递增盘中的桔子数
23. }
24. }
25. 进程daughteri（i=1,2）      //两女儿进程
26. {   while (true)
27. {   P(apple);           //减少盘中苹果数
28. P(mutex);           //申请向盘中取、放水果
29. 取盘中苹果;
30. V(mutex);           //允许向盘中取、放水果
31. V(empty);           //递增盘中可放入的水果数
32. }
33. }
34. 进程sonj（j=1,2）           //两儿子进程
35. {   while (true)
36. {   P(orange);          //减少盘中桔子数
37. P(mutex);           //申请向盘中取、放水果
38. 取盘中桔子;
39. V(mutex);           ／/允许向盘中取、放水果
40. V(empty);           //递增盘中可放入的水果数
41. }
42. }
43. }
44. Coend
45. }

有一个仓库，可存放X 、Y 两种产品，仓库的存储空间足够大，但要求：( l ）每次只能存入一种产品X或Y , ( 2 ）满足-N＜X 产品数量-Y 产品数量＜M 。其中，N 和M 是正整数，试用信号量实现产品X 与Y 的入库过程

Semaphore mutex = 1;

Semaphore sx = M-1; // X产品还能入库的数量

Semaphore sy = N-1; // Y产品还能入库的数量

Void X() {

While(true) {

Wait(sx);

Wait(mutex);

将X产品入库

Signal(mutex);

Signal(sy);

}   
}

Void Y() {

While(true) {

Wait(sy);

Wait(mutex);

将Y产品入库

Signal(mutex);

Signal(sx);

}   
}

有一个仓库可存放A 、B 两种零件，最大库容量各为m 个。生产车间不断地取A 和B 进行装配，每次各取一个．为避免零件锈蚀，按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。为保证配套和合理库存，当某种零件比另一种零件超过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．试用信号量正确地实现它们之间的同步关系。

Semaphore mutex = 1;

Semaphore sa = n; // A产品还能入库的数量

Semaphore sb = n; // B产品还能入库的数量

Semaphore emptyA = m;

Semaphore emptyB = m;

Semaphore fullA = 0;

Semaphore fullB = 0;

int bufferA[m];

int bufferB[m];

int inA = 0, inB = 0, outA = 0, outB = 0;

Void A(int materialA) {

While(true) {

Wait(emptyA);

Wait(sx);

Wait(mutex);

bufferA[inA] = materialA;

inA = (inA + 1) % m;

Signal(mutex);

Signal(sy);

Signal(fullA);

}   
}

Void B(int materialB) {

While(true) {

Wait(emptyB);

Wait(sy);

Wait(mutex);

bufferB[inB] = materialB;

inB = (inB + 1) % m;

Signal(mutex);

Signal(sx);

Signal(fullB);

}   
}

Void Take(int& materialA, int& c) {

While(true){

Wait(fullA);

Wait(fullB);

Wait(mutex);

materialA = bufferA[outA];

materialB = bufferB[outB];

outA = (outA + 1) % m;

outB = (outB + 1) % m;

Signal(mutex);

Signal(emptyA);

Signal(emptyB);

组装  
}

}