（1）请用P、V操作写出一个不会出现死锁的哲学家进餐问题的解？

筷子初值均为1，最多允许4名哲学家同时进餐

semaphore stick[5]={1,1,1,1,1};

semaphore count=4;

main(){

cobegin

philosopher(0);

philosopger(1);

philosopher(2);

philosopher(3);

philosopher(4);

coend

}

philosopher(int i){

while(true){

思考;

p(count);

p(stick[i]);

p(stick[(i+1)%5]);

进餐;

v(stick[i]);

v(stick[(i+1)%5]);

v(count);}

}

（2）设有六个进程P1、P2、P3、P4、P5、P6，它们有如图3.5所示的并发关系。试用P、V操作实现这些进程间的同步。

设五个同步信号量f1、f2、f3、f4、f5分别表示进程P1、P2、P3、P4、P5是否执行完成，其初值均为0。这六个进程的同步描述如下：

P1（）

{

执行P1的代码；

v(f1)；

v(f1)；

}

P2（）

{

p(f1)；

执行P2的代码；

v(f2)；

}

P3（）

{

p(f1)；

执行P3的代码；

v(f3)；

}

P4（）

{

p(f2)；

执行P4的代码；

v(f4)；

}

P5（）

{

p(f3)；

执行P5的代码；

v(f5)；

}

P6（）

{

p(f4)；

p(f5)；

执行P6的代码；

}

（3）今有三个并发进程R、M、P，它们共享了一个可循环使用的缓冲区B，缓冲区B共有N个单元。进程R负责从输入设备读信息，每读一个字符后，把它存入到缓冲区B的一个单元中；进程M负责处理读入的字符，若发现读入的字符中有空格符，则把它改成“，”；进程P负责把处理后的字符取出并打印输出。当缓冲区单元中的字符被进程P取出后，则又可用来存放下一次读入的字符。请用P、V操作为同步机制写出它们能正确并发执行的程序。

semaphore mutex=1;

semaphore empty=N;

semaphore full=0;

semaphore change=0;

main(){

cobegin

R();

M();

P();

coend

}

R(){

while(true){

p(empty);

p(mutex);

char c=read();

v(change);}

}

M(){

while(true){

p(change);

if(c==' ')

c='，';

v(mutex);

v(full);}

}

P(){

while(true){

p(full);

p(mutex)

print(x);

v(mutex);

v(empty);}

}

（4）设公共汽车上，司机和售票员的活动分别如下：

①司机的活动：启动车辆：正常行车；到站停车。

②售票员的活动：关车门；售票；开车门。

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？用信号量和P、V 操作实现它们的同步。

司机在售票员关车门后才可启动车辆，售票员在到站停车前需要完成售票，司机到站停车后售票员才可开车门。

设两个信号量door、 drive分别表示门是否关闭、车是否停止，P1、P2分别表示司机和售票员

semaphore door=0;

semaphore drive=1;

main(){

cobegin

P1();

P2();

coend

}

P1(){

while(true){

p(door);

启动车辆;

正常行车;

到站停车;

v(drive);

}

}

P2(){

while(true){

关车门;

v(door);

售票;

p(drive);

开车门;

}

}