第二章

3.生产者-消费者问题的解决方案如下：

#define N 100 //缓冲区内槽数

Typedef int semaphore;

semaphore mutex=1;

semaphore full=0;

semaphore empty=N;

Void producer(void)

{

int item;

while(TRUE) {

produce\_item ( &item ) ;

down( &empty ) ;

down ( &mutex ) ;

enter\_item ( item ) ;

up ( &mutex ) ;

up( &full ) ;

}

}

Void consumer(void)

{

int item;

while(TRUE) {

down( &full ) ;

down ( &mutex ) ;

remove\_item ( item ) ;

up ( &mutex ) ;

up( &empty ) ;

consumer\_item ( item ) ;

}

}

试问：如果将down( &empty )和 down ( &mutex ) 两条语句的顺序对换，会出现什么情况？

**答:**

**会发生死锁。  
假设先down (&mutex)，如果当前当前不为空，即empty为0，full为1，生产者和消费者有且只有一个会down成功，假如生产者down (&mutex)成功，则它会阻塞在down (&empty)，等待消费者消费。而另一端的消费者又在等待down (&mutex)，这样就会发生死锁。  
如果生产者先down (&empty)，就会阻塞在这里，消费者down (&full)和down (&mutex)都会成功，然后up (&empty)，然后生产者就能往下执行了。**