实验二 生产者与消费者

1. 实验目的

利用Windows提供的API函数，编写程序，解决生产者与消费者问题，实现进程的互斥与同步。

1. 实验内容
2. 进程的互斥与同步。编写一段程序，模拟生产者和消费者线程，实现进程的互斥与同步。
3. 利用VC++6.0实现上述程序设计和调试操作，对于生产者和消费者线程操作的成功与否提供一定的提示框。
4. 通过阅读和分析实验程序，熟悉进程的互斥与同步的概念。
5. 实验步骤
6. 程序设计

本实验要求设计并实现一个进程，该进程拥有一个生产者线程和一个消费者线程，它们使用N个不同的缓冲区（N为一个确定的数值，例如N=32）。

需要使用如下信号量：

1. l一个互斥信号量，用以阻止生产者线程和消费者线程同时操作缓冲区列表；
2. l一个信号量，当生产者线程生产出一个物品时可以用它向消费者线程发出信号；
3. l一个信号量，消费者线程释放出一个空缓冲区时可以用它向生产者线程发出信号.

信号量(s)的使用规则： (当前资源计数-n,最大资源计数-N)

1. n>0，s-触发状态；
2. n=0，s-未触发状态；调用线程->等待状态:

CreateSemaphore(NULL,0,1,NULL); 当第二个参数为0时，调用线程就会进入等待状态.

1. 0<=n<=N.

通过main()函数调用函数CreateThread创建生产者和消费者线程并对3个信号量进行初始化。

生产者线程生产物品，然后将物品放置在一个空缓冲区中供消费者线程消费。消费者线程从缓冲区中获得物品，然后释放缓冲区。

生产者线程生产物品时，若无空缓冲区可用，生产者线程必须等待消费者线程释放出一个空缓冲区；消费者线程消费物品时，若缓冲区为空，消费者线程将被阻塞，直到新的物品被生产出来。

1. 具体实现
2. 创建线程

使用windows提供的CreateThread函数创建一个在调用进程的地址空间中执行的线程。

函数原型：

HANDLE **CreateThread** (

LPSECURITY\_ATTRIBUTES lpThreadAttributes,//其值决定返回的句柄是否可被子进程继承.若为NULL，则句柄不能被继承

DWORD dwStackSize,//定义原始堆栈提交时的大小(按字节计).若值为0，视为缺省情况，使用与调用线程同样的大小

LPTHREAD\_START\_ROUTINE lpStartAddress,//该指针表示远程进程中线程的起始地址.该函数必须存在于远程进程中

LPVOID lpParameter,//定义一个传递给该进程的32位值

DWORD dwCreationFlags,//定义控制进程创建的附加标志.若定义了CREATE\_SUSPENDED标志，线程创建时处于挂起状态，并且直到ResumeThread函数调用时才能运行.若该值为0，则该线程在创建后立即执行

LPDWORD lpThreadld//指向一个32位值，它接受该线程的标识符.若函数调用成功，返回值为新线程的句柄；若函数调用失败,返回值为NULL

);

1. 终止线程

函数原型：

VOID **ExitThread**( DWORD dwExitCode );//dwExitCode：定义调用线程的退出代码。使用GetExitCodeThread函数来检取一个线程的退出代码

1. 创建有名或无名的互斥对象

函数原型：

HANDLE **CreateMutex**(

LPSECURITY\_ATTRIBUTES lpMutexAttributes，// 指向安全属性的指针

BOOL bInitialOwner, // 初始化互斥对象的所有者

LPCTSTR lpName // 指向互斥对象名的指针

);

1. 释放指定互斥对象

函数原型：

**ReleaseMutex**(HANDLE hMutex)；//hMutex:互斥对象句柄.为CreateMutex或OpenMutex函数的返回值.若函数调用成功，则返回值是非零值；如果函数调用失败，则返回值为0

1. 测试指定对象是否有信号或超时(获得信号量的一个占有权)

函数原型：

DWORD **WaitForSingleObject**(

HANDLE hHandle，//等待对象句柄

DWORD dwMilliseconds //指定以毫秒为单位的超时间隔

)；//若函数调用成功，返回值表明引起函数返回的事件.若函数调用失败，返回值是WAIT\_FAILED

1. 创建一个有名或者无名信号对象

函数原型：

HANDLE **CreateSemaphore**(

LPSECURITY\_ATTRIBUTES lpAttributes,//安全属性，如果为NULL则是默认安全属性

LONG lInitialCount,

LONG lMaximumCount,

LPCTSTR lpName);//如果成功就传回一个handle，否则传回NULL

1. 指定信号对象的计数增加一个指定的数量(释放信号量的占有权)

函数原型：

**ReleaseSemaphore**(

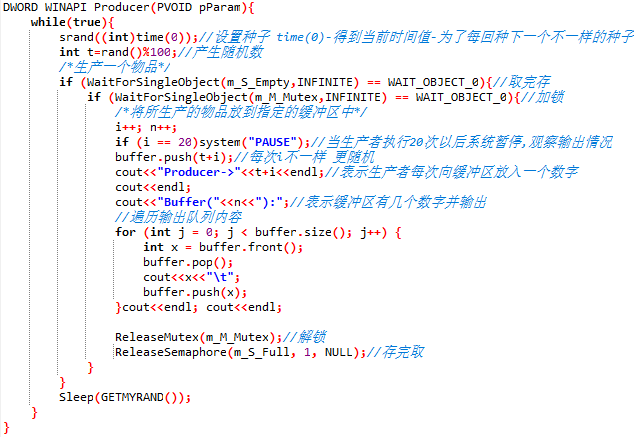
HANDLE hSemaphore,

LONG lReleaseCount,

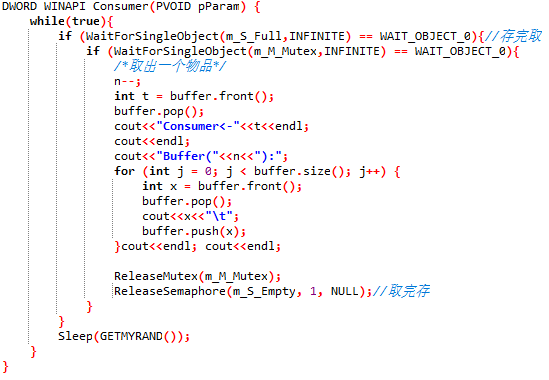
LPLONG lpPreviousCount);//成功则传回TRUE,否则传回FALSE

1. 主要程序结构（附注释）

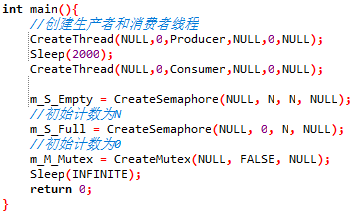
生产者函数



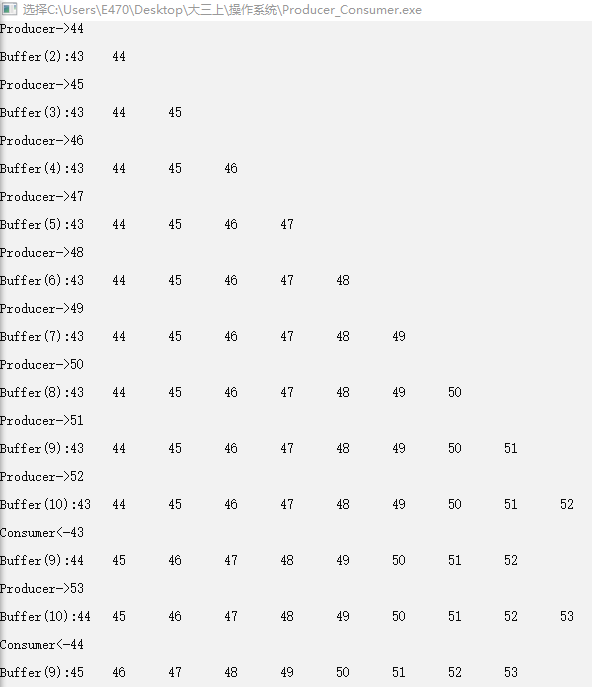
消费者函数



main函数



1. 运行结果



1. 问题及解决方法

问题：原本想通过判断变量i的大小来模拟消费者不会从空共享区取数和生产者不会向已满共享区放数据的情况，但是发现没必要(而且这样跑出来的结果到后面共享区的大小都是5或6并没有占满过)，因为信号量那里已经限制了最大资源数。





解决方法：用Sleep(GETMYRAND());代替，此前要有定义：#define GETMYRAND() (int)(((double)rand()/(double)RAND\_MAX)\*300)