操作系统

实 验 指 导 书

目 录

实验环境介绍 3

[实验一 进程控制 4](#_Toc225750327)

[实验二 并发与调度 7](#_Toc225750328)

[实验三 存储管理 1](#_Toc225750329)0

[实验四 进程间共享内存 1](#_Toc225750330)2

**实验环境介绍**

在Visual C++6.0中创建控制台简单应用程序。针对每个实验题目，调用相应的Windows系统调用，观察输出结果，分析对应的操作系统原理。

**实验一：进程控制。**

1.实验目的：通过在Windows 任务管理器中对进程进行相应的管理操作，熟悉操作系统进程管理的概念，学习观察操作系统运行的动态性能；学习创建进程、观察进程和终止进程的程序设计方法；理解Windows进程的“一生”。

2.实验内容：

1. 使用任务管理器查看、终止进程；
2. 创建进程；
3. 获取进程运行信息，改变进程优先级；
4. 终止进程。

3.实验步骤：

程序一：创建进程

1. 父进程创建子进程，子进程创建下一个子进程；
2. 创建子进程时将子进程序号作为命令行参数传给子进程，子进程将序号加一后继续创建下一个子进程；
3. 限制创建子进程的最大个数；
4. 进程在终止之前暂停一下，输入一个字符后，进程结束，以便观察。

思考题：

1. 运行后在系统任务管理器中查看所创建进程映像名，有什么规律？
2. 父子进程使用同一程序，如何进行区别？
3. 不同进程的克隆ID是如何获取到的？
4. 如果在克隆ID为0的进程中创建克隆ID为1-8的进程，应该如何修改程序？

程序二：查看进程运行信息

1. 在main()函数中获取当前进程ID，获取当前进程版本信息；
2. 获取操作系统版本信息；
3. 获取当前进程优先级；
4. 如果操作系统版本>=Windows XP，并且当前进程优先级不是HIGH\_PRIORITY\_CLASS，将当前进程优先级改为HIGH\_PRIORITY\_CLASS；
5. 显示改变后的进程优先级。

思考题：

1. 进程的优先级存放在哪里？

程序三：终止进程

1. 在程序一的基础上，当前进程每创建一个新进程，增加一个参数，用以表示当前进程的进程ID；
2. 新进程运行时获取父进程的进程ID；
3. 在指定的进程中获取父进程的句柄；
4. 在指定的进程中终止父进程。

思考题：

1. 一个进程终止其父进程所需的句柄是如何获取的？
2. Windows下的进程ID和进程句柄有什么区别？
3. 试通过修改程序来说明对一个进程OpenProcess（）多次，返回的句柄值是否相同？

**实验二：并发与调度**

1.实验目的：掌在本实验中，通过对事件和互斥体对象的了解，来加深对Windows进程、线程同步和互斥的理解。

1. 回顾系统进程、线程的有关概念，加深对Windows进程、线程的理解。
2. 了解事件和互斥体对象。
3. 通过分析实验程序，了解管理事件对象的API。
4. 了解在进程中如何使用事件对象。
5. 了解在进程中如何使用互斥体对象。
6. 了解父进程创建子进程的程序设计方法。

2.实验内容：进程间的同步，线程间的互斥。

3.实验步骤：

程序一：进程间的同步

（1）父进程和子进程使用相同的程序，通过命令行中的”child”识别子进程；

（2）父进程创建一个人工复位事件对象，初始状态是非信号状态，然后创建子进程，利用WaitForSingleObject()无限等待事件；

（3）父进程在一个无限循环中等待事件，获取到事件后输出提示；

（4）子进程打开事件对象，利用SetEvent()发送事件，从而使父进程接收到事件。

思考题：

1. 本实验中事件是如何在父子进程间共享的？
2. Windows的哪个API对应事件的P操作？哪个API对应事件的V操作？
3. 本实验中是哪个进程向哪个进程发送事件的？

（4）获取和发送事件的调用位置有什么特点？

（5）请通过修改程序并对比运行结果来说明Windows的自动复位事件和人工复位事件的区别。

（6）Windows的事件是计数的吗？

程序二：线程间的互斥

1. 在主线程中定义线程间共享变量，创建一个互斥体，创建一个加法线程和一个减法线程；
2. 主线程等待两个线程的结束；
3. 加法线程和减法线程在互斥体的保护下对共享变量进行操作，操作期间有睡眠，并且循环操作。

思考题：

1. 如果main()中没有语句WaitForSingleObject()会出现什么现象？为什么？
2. 为什么两个线程使用同一个句柄来引用互斥体？
3. Windows的哪个API对应互斥体的P操作？哪个API对应互斥体的V操作？
4. 获取和释放互斥体的调用位置有什么特点？
5. 请指出本实验中的共享资源和临界区。
6. 两个线程hThreadInc和hThreadDec，如果一个线程运行到:: Sleep(1000)处，另一线程很可能运行到哪个地方？处于什么状态？

**实验三：存储管理**

1.实验目的：

（1）通过实验了解windows内存的使用，学习如何在应用程序中管理内存、体会Windows应用程序内存的简单性和自我防护能力；

（2）了解windows的内存结构和虚拟内存的管理，进而了解进程堆和windows为使用内存而提供的一些扩展功能。

2.实验内容

（1）Windows提供了一个API即GetSystemInfo() ，以便用户能检查系统中虚拟内存的一些特性；

（2）使用VirtualQueryEX()函数来检查虚拟内存空间；

（3）能正确使用系统函数GetMeoryStatus()和数据结构MEMORY\_STATUS了解系统内存和虚拟存储空间使用情况，会使用VirsualAlloc（）函数和VirsualFree（）函数分配和释放虚拟内存空间。

3.实验步骤：

程序一：虚拟内存信息检测

1. 利用GetSystemInfo（）获取系统信息，得到进程地址空间的下界和上界；
2. 循环进程的地址空间，利用VirtualQueryEx（）得到每个虚拟内存区域的信息，包括虚拟内存区域的块大小、块的状态、保护方式、显示类型以及可执行映像名；

思考题：

1. 进程的虚拟地址空间可以映射到哪些文件？

程序二：分配虚拟内存

1. 在main函数中利用malloc()分配1G内存，然后将前1M字节填充为0，测试是否能够成功填充；
2. 利用VirtualAlloc（）分配1G内存，使用MEM\_COMMIT标志，然后将前1M字节填充为0，测试是否能够成功填充；
3. 利用VirtualAlloc（）分配1G内存，使用MEM\_RESERVE标志，然后将前1M字节填充为0，测试是否能够成功填充；
4. 利用VirtualAlloc（）分配1G内存，使用MEM\_RESERVE标志，然后将前1M字节提交并填充为0，测试是否能够成功填充.

思考题：

1. 访问没有提交的进程空间能成功吗？
2. 使用MEM\_COMMIT标志调用VirtualAlloc（）成功后物理内存已经分配了吗？
3. 利用“虚拟内存的检测”程序检测上述虚拟内存分配方式4所分配虚拟地址块的信息。

**实验四：进程间共享内存**

1.实验目的：

通过实验了解windows如何通过内存映射文件机制来实现进程间共享内存；

2.实验内容

（1）创建一个写进程，创建一个命名的内存映射文件，将一个文件映射对象映射到当前应用程序的虚拟地址空间，在虚拟地址空间中写入数据；

（2）创建一个读进程，打开命名的内存映射文件，将文件映射对象映射到当前应用程序的虚拟地址空间，在虚拟地址空间中读出数据。

3.实验步骤：

程序一：写进程

（1） 利用CreateFileMapping（）创建一个命名的内存映射文件对象；

（2） 利用MapViewOfFile（）将文件映射到当前应用程序的虚拟地址空间；

（3）在虚拟地址空间中写入数据。

程序二：读进程

（1） 利用OpenFileMapping（）打开已创建的命名内存映射文件对象；

（2） 利用MapViewOfFile（）将文件映射到当前应用程序的虚拟地址空间；

（3）从虚拟地址空间中读出数据。

思考题：

1. Windows下的进程间共享内存是利用什么机制实现的？
2. 对于读写进程，物理内存是什么时候分配的？
3. 读写进程之间的同步和互斥在共享内存机制中已经存在了，还是需要用户自己来实现？
4. 利用“虚拟内存的检测”程序检测读进程所映射的虚拟地址空间块的信息。