**实验六 Linux进程间通信与多线程编程**

**问题：**

共享内存是[进程间通信](http://baike.baidu.com/view/1492468.htm)中最简单的方式之一。共享内存允许两个或更多进程访问同一块内存，就如同 malloc() 函数向不同进程返回了指向同一个[物理内存](http://baike.baidu.com/view/138684.htm)区域的[指针](http://baike.baidu.com/view/159417.htm)。当一个进程改变了这块地址中的内容的时候，其它进程都会察觉到这个更改。因为所有[进程](http://baike.baidu.com/view/19746.htm)共享同一块内存，共享内存在各种进程间通信方式中具有最高的效率。访问共享内存区域和访问进程独有的内存区域一样快，并不需要通过[系统调用](http://baike.baidu.com/view/47173.htm)或者其它需要切入[内核](http://baike.baidu.com/view/1366.htm)的过程来完成。同时它也避免了对数据的各种不必要的复制。要使用一块共享内存，进程必须首先分配它。随后需要访问这个共享内存块的每一个进程都必须将这个共享内存绑定到自己的[地址空间](http://baike.baidu.com/view/1507129.htm)中。当完成通信之后，所有进程都将脱离共享内存，并且由一个进程释放该共享内存块。

线程，有时被称为轻量级进程(Lightweight Process，LWP），是程序执行流的最小单元。一个标准的线程由线程ID，当前指令指针(PC），寄存器集合和堆栈组成。另外，线程是进程中的一个实体，是被系统独立调度和分派的基本单位，线程自己不拥有系统资源，只拥有一点儿在运行中必不可少的资源，但它可与同属一个进程的其它线程共享进程所拥有的全部资源。一个线程可以创建和撤消另一个线程，同一进程中的多个线程之间可以并发执行。由于线程之间的相互制约，致使线程在运行中呈现出间断性。线程也有就绪、阻塞和运行三种基本状态。每一个程序都至少有一个线程，若程序只有一个线程，那就是程序本身。线程是程序中一个单一的顺序控制流程。在单个程序中同时运行多个线程完成不同的工作，称为多线程。

本次实验课的主要内容：

1. 使用共享内存进行进程间通信
2. 在进程中创建新线程、在一个进程中同步线程之间的数据访问、修改线程属性、在同一个进程中，从一个线程中控制另一个线程

**实验要求：**

（1）查阅相关资料，了解Linux进程间通信。并完成以下实验

编写程序，实现父子进程通过共享内存进行数据通信。父子进程通过竞争方式来创建一个共享内存单元，然后子进程接受用户输入的信息（通过argv[1]输入），并将其写入到共享内存单元；父进程则从共享内存单元将该信息读出，并显示信息的个数。具体步骤为：

1. 创建子进程；
2. 将运行参数argv[1]字符串写入共享内存；
3. 并打印写入字符串;
4. 在父进程中读出所写入的字符串并打印。
5. 最后练习memcpy和memmove内存段数据处理函数的使用

**审核要求:**

1. 提交全部程序代码，代码思路规范清晰，命名规范。

2. 运行能直接输出所要求的结果。

3. 使用shmget、shmat、shmctl函数。

（2）查阅相关资料，了解Linux多线程编程，并完成以下实验

1、编写程序实现如下功能：编写程序，在主线程中创建一个新线程。要求在新线程中输出运行信息，在结束时返回主线程。使用函数pthread\_exit（0）退出线程。再试试exit(0)退出线程，看看是什么结果。比较一下二者的不同。

2、在两个线程之间实现交替执行输出。要求：例如主线程先输出“this is a thread0”,然后新线程输出“this is thread 2”一直交替直到结束。同时访问全局变量，修改变量的值，并打印看看输出结果。

3、编写程序实现使用信号量进行同步。编码实现输入字符串，统计每行的字符个数，以“end”结束输入。

**审核要求:**

1. 提交全部程序代码，代码思路规范清晰，命名规范。

2. 运行能直接输出所要求的结果。

**作业提交格式及要求：**

1、将实验记录**（必要步骤和截屏，尽量详细）**和分析总结整理成实验报告word文件格式提交。

2、实验报告文件按“**实验6\_班级\_学号\_姓名**”格式命名。

3、实验报告提交的最后期限为**第17周周日**。