**实验四 Linux程序库基础与系统调用和进程间通信与多线程编程**

**问题1：**

计算机程序（Computer Program）是指一组指示计算机或其他具有信息处理能力装置每一步动作的指令，通常用某种程序设计语言编写，运行于某种目标体系结构上。一个计算机程序是一系列指令的集合。

通常，计算机程序要经过编译和链接而成为一种人们不易看清而计算机可解读的格式，然后运行。未经编译就可运行的程序，通常称之为脚本程序（script）

在计算机科学中，库是用于开发软件的子程序集合。库和可执行文件的区别是，库不是独立程序，他们是向其他程序提供服务的代码。库链接是指把一个或多个库包括到程序中，有两种链接形式：静态链接和动态链接，相应的，前者链接的库叫做静态库，后者的叫做动态库，也叫共享库。

静态库指将所有相关的目标文件打包成为一个单独的文件-即静态库文件。以.a结尾。静态库可作为链接器的输入，链接器会将程序中使用的到函数的代码从库文件中拷贝到应用程序中。一旦链接完成，静态库就在程序里了。

共享库是一个目标模块。Linux系统上以.so后缀表示，Windows以.dll后缀。在运行时，可以加载到任意的存储器地址，并和一个在存储器中的程序链接起来，这个过程称为动态链接，是由一个叫做动态链接器的程序来执行的。应用程序在运行的时候需要共享库的支持。

系统调用是在内核中实现的，通过一定的方式把系统调用给用户，系统调用是用户程序和内核交互的接口

Linux下对文件操作有两种方式：系统调用(system call)和库函数调用(Library functions)。系统调用实际上就是指最底层的一个调用，在linux程序设计里面就是底层调用的意思。面向的是硬件。而库函数调用则面向的是应用开发的，相当于应用程序的api。库函数调用通常用于应用程序中对一般文件的访问。库函数调用是系统无关的，因此可移植性好。由于库函数调用是基于C库的，因此也就不可能用于内核空间的驱动程序中对设备的操作。

函数库调用在所有的ANSIC编译器版本中，C库函数是相同的，而系统调用在各个操作系统中的系统调用是不同的。函数库调用是调用函数库中的一段程序，而系统调用调用的是系统内核的服务。函数库调用它是与用户程序相联系，在用户地址空间执行，它的运行时间属于“用户时间”，而系统调用，它是操作系统的一个入口点，它是在内核空间执行，它的运行时间属于“系统时间”。另外从开销上来讲，系统调用由于是需要在用户空间和内核上下文环境间切换，开销较大，而函数库调用开销是比较小的。我们用一句话总结下就是：函数库调用是语言或应用程序的一部分，而系统调用是操作系统的一部分。

本次实验课的主要内容：

1. 程序库基础，静态库与动态库的生成，开发与使用
2. Linux系统调用

**问题1要求：**

（1）查阅相关资料，了解静态库和动态库的生成、开发、使用。并完成以下实验

**制作静态库：**

1. 分别编写add.h add.c main.c
2. add.h包含加法函数的定义 int add(int p1, int p2)
3. add.c 包含加法函数的实现
4. 对add.c进行编译生成目标文件add.o
5. 执行ar命令，生成libadd.a

⑥ 编写main.c文件，通过include add.h头文件和gcc编译实现对libadd.a中的加法函数的调用

**制作动态库：**

1. 分别编写add.h add.c main.c
2. add.h包含加法函数的定义 int add(int p1, intp2)
3. add.c 包含加法函数的实现
4. 对add.c进行编译生成共享库libadd.so
5. 编写main.c文件，通过include add.h头文件和gcc编译实现对libadd.so中的加法函数的调用

**审核要求:**

提交全部程序代码，代码思路规范清晰，命名规范。

通过带参数运行可执行文件main，如main 3 5，得出的输出是 3 + 5 = 8

（2）查阅相关资料，了解Linux系统调用函数，并完成以下实验

检测文件当前的读写权限，如果文件具有读权限，则打印可读信息，如果有可写权限，则打印可写信息。否则返回错误信息。

如果打开文件为只读文件，则输出”read only”；

如果文件是只写文件，则输出”write only”；

如果文件可读可写，则输出”read write”；否则输出”unknown mode”

**提示：**可用fcntl()函数实现。

**问题2：**

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

**问题2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。