广州大学学生实验报告

**开课学院及实验室：**计算机科学与网络工程学院软件实验室 2019**年**10**月**24**日**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **学院** | 计算机科学与网络工程学院 | **年级/专业/班** | 软件171 | **姓名** | 谢金宏 | **学号** | 1706300001 |
| **实验课程名称** | 操作系统实验 | | | | | **成绩** |  |
| **实验项目名称** | 进程管理与进程通信 | | | | | **指导老师** | 陶文正 |

实验一 进程管理与进程通信

# 实验目的

1. 掌握进程的概念，明确进程的含义。
2. 认识并了解进程并发执行的实质，进程的阻塞与唤醒，终止与退出的过程。
3. 熟悉进程的睡眠、同步、撤消等进程控制方法。
4. 分析进程竞争资源的现象，学习解决进程互斥的方法。
5. 了解什么是信号，利用信号量机制熟悉系统中进程之间软中断通信的基本原理。
6. 熟悉消息传送的机理，共享存储机制。

# 实验环境

POSIX兼容的操作系统编程环境

1. Windows Subsystem Linux (WSL)
2. CentOS

# 实验内容

1. **进程创建与多进程并发** 编写一段程序，使用系统调用fork()创建两个子进程。当此程序运行时，在系统中有一个父进程和两个子进程并发执行，观察实验结果并分析原因。
2. **进程的睡眠、同步、撤销等进程控制** 用fork()创建一个进程，再调用exec()，用新的程序替换该子进程的内容，利用wait()来控制进程执行顺序，掌握进程的睡眠、同步、撤消等进程控制方法，并根据实验结果分析原因。
3. **编写一段多进程并发运行的程序**，用lockf()来给每一个进程加锁，以实现进程之间的互斥，观察并分析出现的现象及原因。
4. **软中断的处理** 编写程序：用fork()创建两个子进程，再用系统调用signal()让父进程捕捉键盘上来的中断信号（即按^c键）；捕捉到中断信号后，父进程用系统调用kill()向两个子进程发出信号，子进程捕捉到信号后分别输出下列信息后终止：

Child process1 is killed by parent!

Child process2 is killed by parent!

父进程等待两个子进程终止后，输出如下的信息后终止：

Parent process is killed!

分析利用信号量机制中的软中断通信实现进程同步的机理。

1. **消息队列** 使用系统调用msgget(),msgsnd(),msgrev(),及msgctl()编制一长度为1k的消息发送和接收的程序，并分析消息的创建、发送和接收机制及控制原理。
2. **共享存储区** 编制一长度为1k的共享存储区发送和接收的程序，并设计对该共享存储区进行互斥访问及进程同步的措施，必须保证实现正确的通信。

# 实验原理实验中用到的系统调用函数（包括实验原理中介绍的和自己采用的）

1. **fork()** **创建一个新进程。**

系统调用格式： int pid = fork()

参数定义：int fork()

返回值意义如下：

* + 0：在子进程中，pid变量保存的fork()返回值为0，表示当前进程是子进程。
  + >0：在父进程中，pid变量保存的fork()返回值为子进程的id值（进程唯一标识符）。
  + -1：创建失败。

1. **exit() 终止进程的执行。**

系统调用格式：exit(status: int): void

其中，status是返回给父进程的一个整数，以备查考。

为了及时回收进程所占用的资源并减少父进程的干预，UNIX/LINUX利用exit()来实现进程的自我终止，通常父进程在创建子进程时，应在进程的末尾安排一条exit()，使子进程自我终止。exit(0)表示进程正常终止，exit(1)表示进程运行有错，异常终止。

如果调用进程在执行exit()时，其父进程正在等待它的终止，则父进程可立即得到其返回的整数。核心须为exit()完成以下操作：

* 1. 关闭软中断
  2. 回收资源
  3. 写记帐信息
  4. 置进程为“僵死状态”

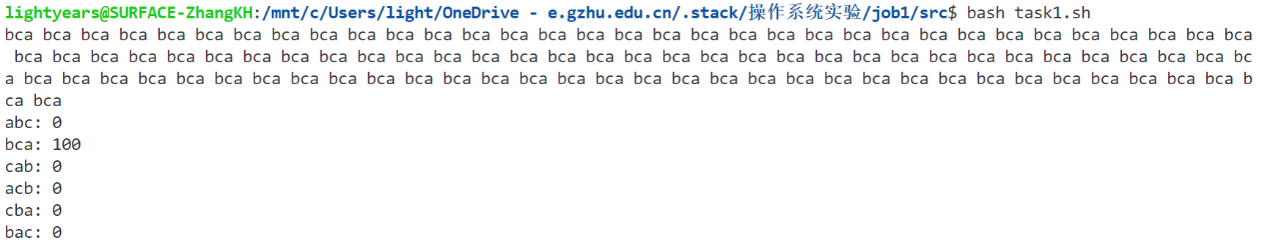
1. exec、wait、getpid、sleep、lockf、kill、signal、pipe、read、write、msgget、msgsnd、msgrcv、msgctl、shmget、shmat、shmdt、shmctl等。

# 实验结果分析（截屏的实验结果，与实验结果对应的实验分析）

1. **进程创建与多进程并发**

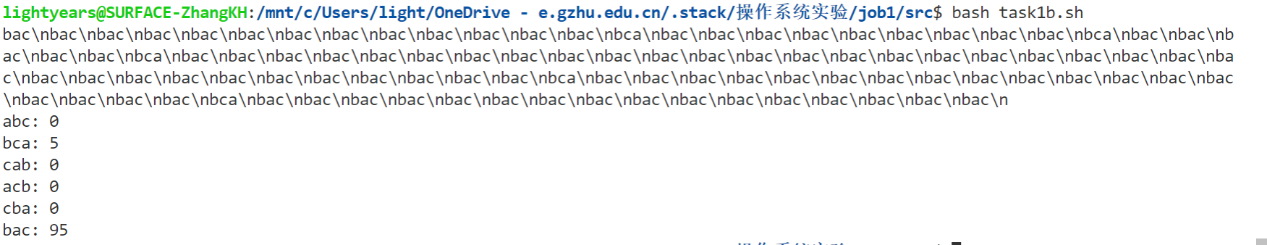
创建如下程序：

在WSL上运行上面的程序片段100次，输出的结果为：



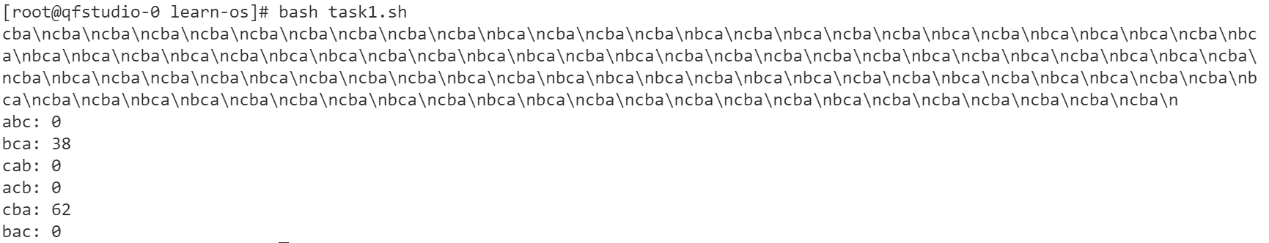
即100次“bca”，没有体现操作系统对进程的调度。

稍微改变程序（“task1b.c”）重新运行100次，输出的结果为：



即有5次输出abc的顺序与其余95次不同，体现了操作系统对进程的调度。

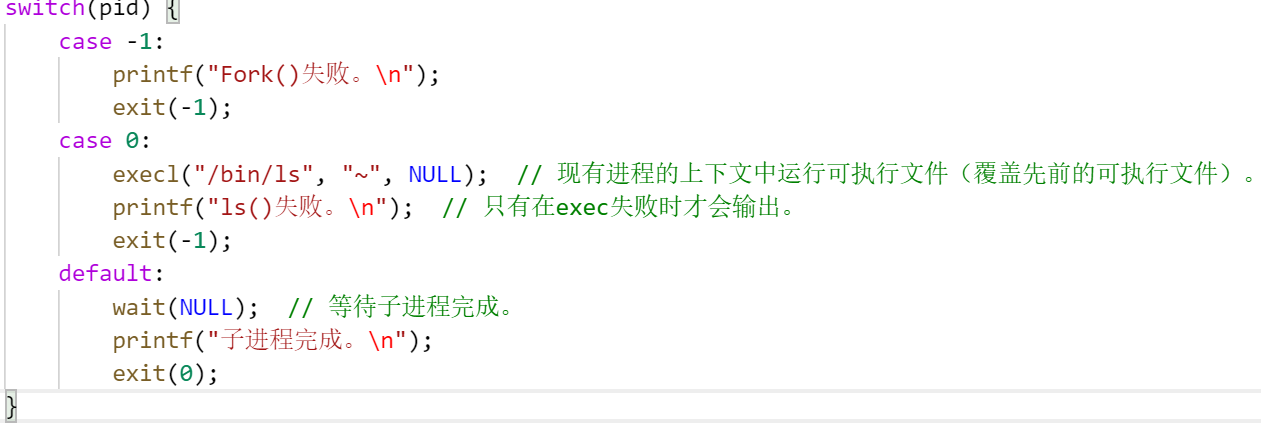
在CentOS上运行：



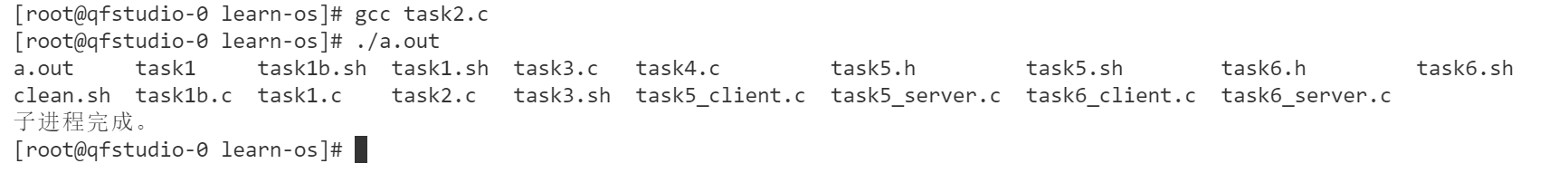
这时进程输出字符的顺序有比较明显的差异。

1. **进程的睡眠、同步、撤销等进程控制**

创建如下程序：

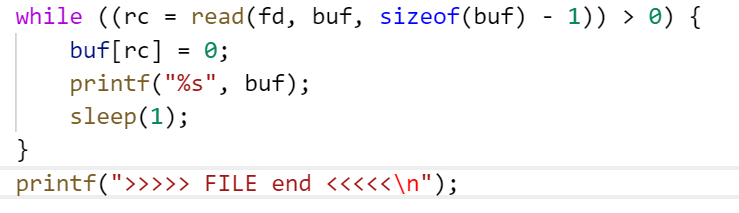


运行结果如下：

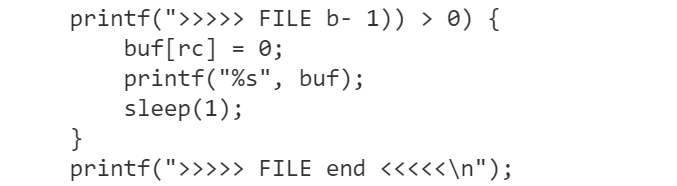


1. **多进程通过加锁互斥并发运行**

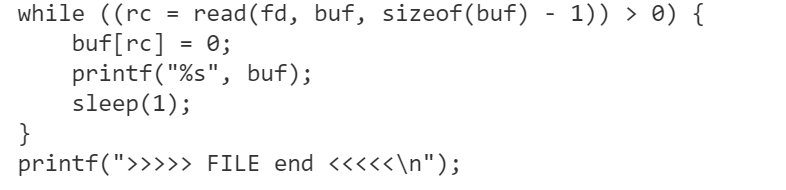
创建一个C语言的程序，该程序使用两个子进程分别读取程序的源文件并输出到标准输出。



未加锁时输出如下：



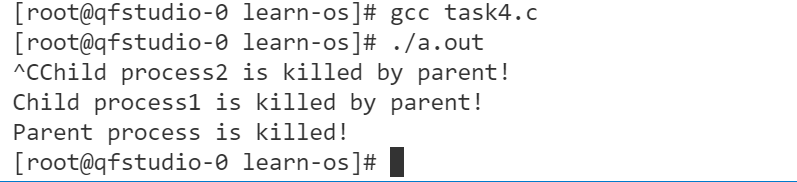
加锁时输出如下：



1. **使用软中断信号通信实现父子进程同步**

按照实验要求编写程序。

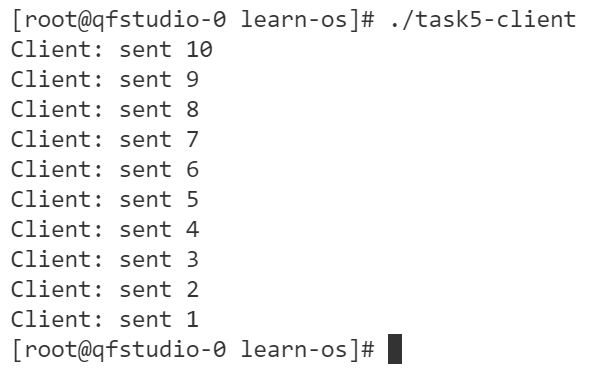
运行结果如下：

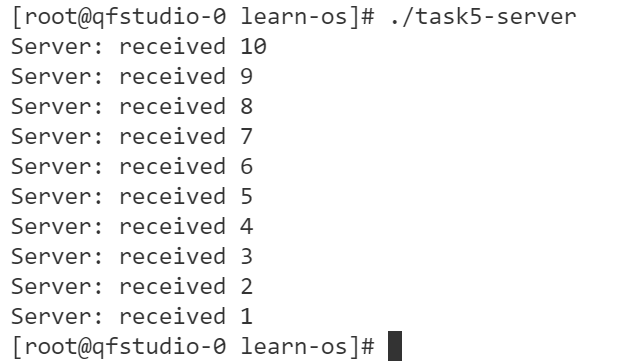


1. **消息的创建、发送和接收机制**

按照实验要求编写程序。

运行结果如下：

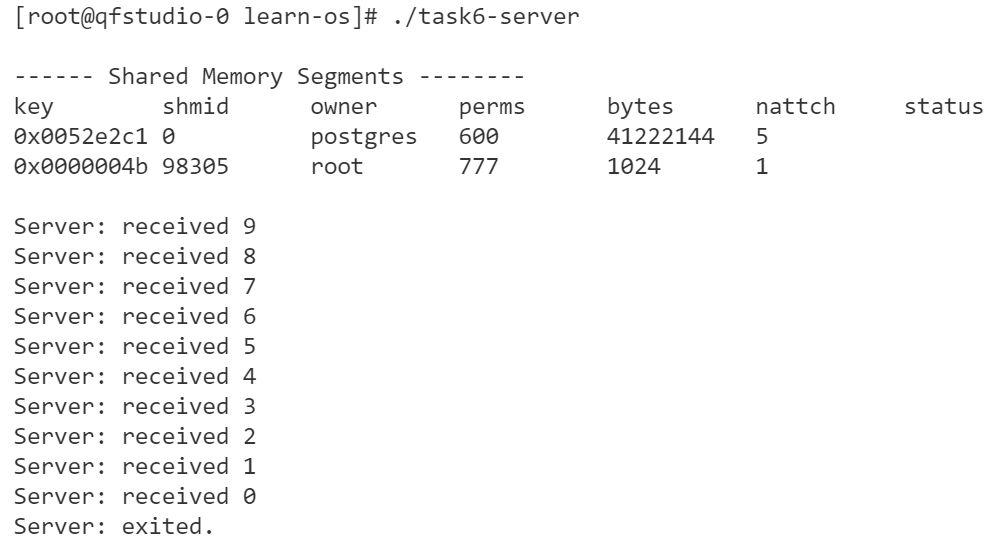


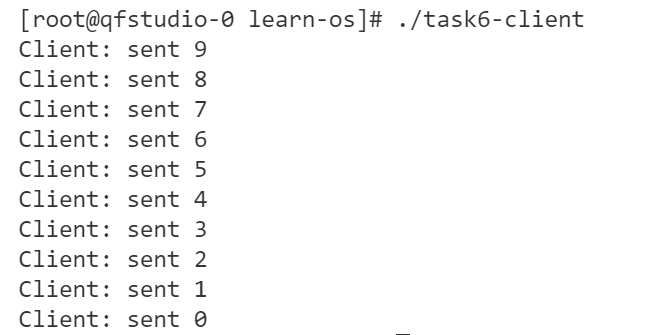


1. **进程间共享存储区通信**

按照实验要求编写程序。

运行结果如下：





# 实验总结

1. **进程创建与多进程并发**
   1. 使用fork创建进程时，操作系统核心为新进程分配进程表项和一个唯一的进程标识符，并将父进程的当前目录和所有已经打开的数据复制到子进程表项中。此时子进程就处在就绪状态了。
   2. 新创建的进程运行时，并不是从main()函数开始，而是从创建子进程的fork语句处开始。对于父进程，fork返回大于零的整数；子进程返回0。因此可使用分支语句使父子进程执行不同的代码。
   3. 使用ltrace -f -i -S <filename>查看进程执行过程，

画出进程家族树如下：

1. **进程的睡眠、同步、撤销等进程控制**
   1. 可执行文件在加载时，除去在SHELL中对指令的处理外，还需要根据可执行文件的类型搜索合适的可执行文件装载处理程序。以Executable and Linkable Format (ELF)格式文件为例，此时ELF加载程序负责创建新的虚拟控件（页表），建立从虚拟空间到可执行文件的映射，并置可执行文件的入口。
   2. 进程同步指多个进程以一定的次序相互协作。调用wait会阻塞当前进程，直到当前的进程的一个子进程结束。调用waitpid会阻塞当前进程，直到指定的pid的子进程结束。
   3. 调用exit使当前进程结束并返回指定值。调用wait的父进程会阻塞，直到某个子进程退出，并返回这个子进程的pid。父进程可以使用WEXITSTATUS宏获取子进程exit时的返回值。
2. **多进程通过加锁互斥并发运行**
   1. 进程加锁和未加锁运行的情况不同。当加锁时，只有取得锁的进程对被加锁的资源具有使用权，其他进程只能等待释放锁。这样避免了多个进程同时使用互斥资源时的混乱。
3. **进程间通过信号机制实现软中断通信**
   1. Kill的作用是向子进程发送信号。
   2. Signal的作用是注册信号处理程序，当特定的信号到来时，调用对应的信号处理程序。
4. **消息的发送与接收**
   1. 为便于操作和观察结果，编写task5\_client.c和task5\_server.c两个程序，编译程序并先后运行task5-server和task5-client。
   2. 服务器进程先通过msgget创建消息队列，并通过msgrcv获取消息队列中的消息。客户端进程通过msgget获取消息队列，并通过msgsnd向消息队列发送消息。
   3. 相比于进程共享存储区，消息队列可以方便的发送和接收固定格式的消息。
5. **进程间的共享存储区**
   1. 为便于操作和观察结果，编写task6\_client.c和task6\_server.c两个程序，编译程序并先后运行task6-server和task6-client。
   2. 服务器进程通过shmget创建共享储存区，并使用shmat获取共享储存区的地址空间。客户端进程通过shmget获取共享储存区，并通过shmat获取共享储存区的地址空间。
   3. 相比于消息队列，共享存储区可以方便地在进程之间共享大量的数据，但具体的数据结构需要应用程序自行设计。

# 实验数据及源代码（学生必须提交自己设计的程序源代码，并有注释，源代码电子版也一并提交），包括思考题的程序。

见“软件171-1706300001-谢金宏-实验1.zip”。