|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | 史新宽 | **学号** | 29 |
| **实验题目** | Lab3 | | |
| **实验内容** | **（1）编制一段程序，使用系统调用 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【Del】键），当捕捉到中断信号后，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（pid=XX ,ppid=XX） is killed by parent!**  **child process2 (pid=XX ,ppid=XX) is killed by parent!**  **父进程等待两个子进程终止后，输出以下信息后终止：**  **Parent(pid=XX) process is killed!**  编译执行      （2）在上面的程序中增加语句signal(SIGINT,SIG-IGN)和signal(SIGQUIT,SIG-IGN)，观察执行结果，并分析原因。  加入signal(SIGINT,SIG-IGN)后 | | |
|  | 加入signal(SIGQUIT,SIG-IGN)后      分折：signal(SIGINT,SlG\_SGN）和signal(SIGQUlT,SIG\_IGN)  分别为忽略键入信号以及忽略中断信号。在子进程中开始添加  signal(SIGINT,SlG\_SGN） 则子程忽略键盘中断信号， 此时按ctrl-c  不会将子进程终止， 程序可以正常运行， 而在子进程中开始添加  signal(SlGQUIT,SIG\_IGN), 则相应的中断信号被忽略，只有父进程中断。  **（3）使用多线程和信号量解决生产者/消费者问题：有一个长度为N的缓冲池被生产者和消费者共同使用。只要缓冲池未满，生产者就可以将消息送入缓冲池；只要缓冲池不空，消费者便可以从缓冲池中取走一个消息。生产者向缓冲池放入消息的同时，消费者不能操作缓冲池，反之亦然。**  加上-lpthread是在链接阶段，链接指定库    初始状态缓冲区为空，此时只允许生产者生产资源放入缓冲区，而消费者试图取缓冲区产品时，将进入等待队列，等待资源释放，在等待队列中的第一个进程将得到资源的控制权。当缓冲区产品满了，将拒绝生产者向缓冲区输送资源。生产者每隔三秒向缓冲区输送产品，缓冲区满了，则等待消费者消费产品后再将产品输送进去；消费者每隔六秒向缓冲区取得产品，缓冲区为空，则等待生产者输送产品后再消费产品 | | |
| **总结** | * 1. 掌握了Linux操作系统的进程创建和终止。 * 2. 弄清了fork创建子进程时返回值取不同值时分别代表的含义。 * 3.了解到wait()和sleep（）的用法。   4.弄明白了忽略键盘中断信号和忽略中断信号的结果和成因。  5.对解决生产者消费者问题有了更深的理解也弄懂了原来不太懂得一些细节。  6.提升了编程能力尤其是接口的调用。 | | |
| **日期** | 2020.06.8 | **成绩** |  |