|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | 倪毅男 | **学号** | 25 |
| **实验题目** | Lab3 | | |
| **实验内容** | **lab3\_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，输出结果信息。  源代码如下：    Ubuntu中Gcc运行：  2b0d47023dc718a32d85b462c026283结果分析：Pid为2967的父进程创建了两个子进程，在接收到中断信号后父进程被终止，同时由于子进程没有忽略该中断信号，也被终止，没能打印出子进程信息。  **Lab3\_2**：在上面的程序中增加语句signal(SIGNAL,SIG-IGN)和signal(SIGQUIT,SIG-IGN)，观察执行结果，并分析原因。    执行后：  a7d0d5f94ff44b9b6486830e629a4aa  结果分析：Pid为3008的父进程创建了两个pid分别为1974和1975的子进程，signal(SIGINT, SIG-IGN)和signal(SIGQUIT, SIG-IGN)的作用是屏蔽从键盘上传来的中断信号，所以子进程忽略了中断信号，所有能打印出信息。  **lab3\_3**：使用多线程和信号量解决生产者/消费者问题  源代码如下：      执行之后：  e16154bfee31410851fd194732c3f2d  C:\Users\Lenovo\AppData\Local\Temp\WeChat Files\79ba79c8dc3988f62aa2fd4fd256b260_.png | | |
| **总结** | 1.了解了Linux操作系统的进程创建和终止，以及fork（）创建进程的  过程以及Linux操作系统提供的“软中断”机制实现进程间的同步   1. 了解了管道通信，消息队列，共享内存、套接字等几种典型的进程通信机制以及使用信号量进行进程通信。 | | |
| **日期** | 2020.06.08 | **成绩** |  |