|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | 查显中 | **学号** | 4062017001 |
| **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ctrl c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：    在Ubuntu中使用Gcc运行会看到生成a.out文件，在终端中运行，结果如下：  结果分析：Pid为3638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。    执行后同样产生a.out文件，在终端执行运行结果  结果分析：Pid为3853的父进程创建了两个pid分别为3854和3855的两个子进程，因为signal(SIGINT, SIG-IGN)和signal(SIGQUIT, SIG-IGN)屏蔽了从键盘上传来的中断信号，所以子进程忽略了中断信号，所以能打印出信息。  **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：    在终端中执行高gcc task3.c -o task3 -之后：  e16154bfee31410851fd194732c3f2d | | |
| **总结** | 1. 了解了Linux操作系统fork（）创建进程的过程 2. 学习了Linux操作系统提供的“软中断”机制实现进程间的同步。   3.了解了使用多线程和信号量解决生产者/消费者问题。 | | |
| **日期** | 2020.6.9 | **成绩** |  |