|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | 闫鹏宇 | **学号** | 4062017034 |
| **实验题目** | **实验3：进程控制和进程通信** | | |
| **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!**  代码：    编程完后保存到虚拟机，在虚拟机执行gcc命令生成a.out文件  输入./a.out执行源文件  运行界面结果如图：未输出child process1 is killed、child process2 is killed，因为输入中断指令后，由于信号会并发传到每个进程中，进程受到该信号会立刻终止，不会等父进程传来的软中断信号    **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。**  在（1）的程序中增加signal(SIGNAL,SIG-IGN)和signal(SIGQUIT,SIG-IGN)  如图：    加入上述语句后，运行结果如下：    **（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。**  代码：          在虚拟机输入gcc的lpthread命令并执行该可执行文件： | | |
| **总结** | （1）掌握了Linux操作系统的进程创建和终止；  （2）了解了利用Linux操作系统提供的“软中断”机制实现进程间的同步。  （3）了解了几种典型的进程通信机制（例如：管道通信，消息队列，共享内存、套接字等）以及使用信号量进行进程通信。 | | |
| **日期** | 2020.06.06 | **成绩** |  |