实验 2.1 进程的软中断通信

```
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
#include<stdlib.h>
#include<sys/wait.h>
#include <sys/types.h>
int wait_flag=0;
void stop( int signum);
int main( ) {
    int pid1, pid2;                          // 定义两个进程号变量
    while((pid1 = fork( )) == -1);       // 若创建子进程 1 不成功,则空循环
    if(pid1 > 0) {                    // 子进程创建成功,pid1 为进程号
        while((pid2 = fork( )) == -1);  // 创建子进程 2
        if(pid2 > 0) {
wait_flag = 1;
        signal(3,stop);
        signal(2,stop);
        signal(14,stop);

            alarm(5);
    while(wait_flag ==1);
            kill(pid1,16);                      // 杀死进程 1 发中断号 16
        kill(pid2,17);                       // 杀死进程 2
            wait(0);                        // 等待第 1 个子进程 1 结束的信号
            wait(0);                        // 等待第 2 个子进程 2 结束的信号
            printf("\n Parent process is killed !!\n");
            exit(0);                                    // 父进程结束
            }
```

```
        else {
        wait_flag = 1;
        signal(17,stop);        // 等待进程 2 被杀死的中断号 17
        signal(2,SIG_IGN);
        signal(3,SIG_IGN);
        while(wait_flag);
        printf("\n Child process 2 is killed by parent !!\n");

        exit(0);
            }
    }
    else {
   wait_flag = 1;
   signal(16,stop);                  // 等待进程 1 被杀死的中断号 16
    signal(2,SIG_IGN);
    signal(3,SIG_IGN);
    while(wait_flag);
    lockf(1,1,0);
    printf("\n Child process 1 is killed by parent !!\n");
    lockf(1,0,0);
    exit(0);
}
}
void stop( int signum) {
    wait_flag=0;
    printf("\n %d stop test \n",signum);
}
```