ICS EXE 7

April 22, 2020

1 Signal

```c
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

static int beeps = 0;
char beep[] = "BEEP!\n";
char boom[] = "BOOM!\n";

void handler(int sig) {
    write(STDOUT_FILENO, beep, strlen(beep));
    if (beeps < 5) {
        beeps += 1;
        fork();
        alarm(1);
    } else{
        write(STDOUT_FILENO, boom, strlen(beep));
        exit(0);
    }
}

int main() {
    signal(SIGALRM, handler);
    alarm(1);
    while (1);
    exit(0);
}
```

1. How many seconds will this program remain approximately?

   ANS: 6 seconds

2. How many BEEPs and BOOMs will be printed if you run the above program?

   ANS: 1+2+4+8+16+32 = 63 BEEPs, 32 BOOMs

2 nonlocal

Implement a version of the standard I/O fgets function, called tfgets, that times out and returns NULL if it does not receive an input line on standard input within 5 seconds. Your function should be implemented in a package called tfgetsproc.c using processes, signals, and nonlocal jumps. It should not use the Linux alarm function. Test your solution using the driver program.

```c
#include "csapp.h"

char *tfgets(char *s, int size, FILE *stream);

int main() {
    char buf[MAXLINE];
    if (tfgets(buf, MAXLINE, stdin) == NULL)
        printf("BOOM!\n");
    else
        printf("%s", buf);
    exit(0);
}
```

ANS:

```c
sigjmp_buf buf;
void sigchild_handler(int sig) {
     siglongjmp(buf, 1);
}

char *tfgets(char *s, int size, FILE *stream) {
    if (Fork() == 0) {
        Sleep(5);
        exit(0);
    }
    switch (sigsetjmp(buf, 1)) {
        case 0:
            Signal(SIGCHLD, sigchild_handler);
            return fgets(s, size, stream);
        case 1:
            return NULL;
    }
}
```

3 Signal

```
1  volatile sig_atomic_t counter = 0;
2  void handler(int sig)
3  {
4      int olderrno = errno;
5      sigset_t hmask, hprev;
6      sigfillset(&hmask);
7      while (counter) {
8          waitpid(-1, NULL, 0);
9          sigprocmask(SIG_BLOCK, &hmask, &hprev);
10         sio_putl((long)(--counter));
11         sigprocmask(SIG_SETMASK, &hprev, NULL);
12         sio_puts("Children running\n");
13     }
14     errno = olderrno;
15 }
16
17 int main()
18 {
19     Signal(SIGCHLD, handler);
20     sigset_t mask, prev;
21     sigfillset(&mask);
22     for (int i = 0; i < 5; i++) {
23         if (fork() == 0) {
24             printf("Child\n");
25             exit(0);
26         }
27         sigprocmask(SIG_BLOCK, &mask, &prev);
28         counter++;
29         sigprocmask(SIG_SETMASK, &prev, NULL);
30     }
31     while (counter) pause();
32     exit(0);
33
34 }
```

The given code aims to create 5 children processes and reap them. Try to describe what unexpected problem may happen during execution, and give the solution.

ANS: If the last SIGCHLD comes before parent increases counter, the handler may fail to reap one of the children.

If the last SIGCHLD comes between while and pause, the program will never wake up.

Fix:

```c
volatile sig_atomic_t counter = 0;
void handler(int sig)
{
    int olderrno = errno;
    sigset_t hmask, hprev;
    sigfillset(&hmask);
    while (counter) {
        waitpid(-1, NULL, 0);
        sigprocmask(SIG_BLOCK, &hmask, &hprev);
        sio_putl((long)(--counter));
        sigprocmask(SIG_SETMASK, &hprev, NULL);
        sio_puts("Children running\n");
    }
    errno = olderrno;
}

int main()
{
    Signal(SIGCHLD, handler);
    sigset_t mask, prev;
    sigfillset(&mask);
    sigset_t one;
    sigemptyset(&one);
    sigaddset(&one, SIGCHLD);
    for (int i = 0; i < 5; i++) {
        sigprocmask(SIG_BLOCK, &one, &prev);
        if (fork() == 0) {
            printf("Child\n");
            exit(0);
        }
        //sigprocmask(SIG_BLOCK, &mask, &prev);
        sigprocmask(SIG_BLOCK, &mask, NULL);
        counter++;
        sigprocmask(SIG_SETMASK, &prev, NULL);

    }
    sigprocmask(SIG_BLOCK, &one, &prev);
    while (counter)
        //pause();
        sigsuspend(&prev);
    exit(0);
}
```

4 File

What is the output of the following program?

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int main(void) {
    int fd1, fd2, fd3;

    fd1 = open("foo", O_RDONLY |
               O_CREAT, S_IRWXU);
    fd2 = open("foo", O_RDONLY);
    close(fd1);
    fd3 = open("foo", O_RDONLY);
    printf("fd1=%d, fd2=%d, fd3=%d\n",
           fd1, fd2, fd3);
    close(fd2);
    close(fd3);

    return 0;
}
```

ANS: fd1=3, fd2=4, fd3=3