

Homework 7

Problem I

```
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) //a mistake in the problem
        //pause();
        sigsuspend(&prev);
    exit(0);
}
```

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**.

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.

Problem II

<pre>1 int main(){ 2 int fd1, fd2; 3 char c; 4 fd1 = open("c.txt", O_RDONLY, 0); 5 int i = 0; 6 if(fork() == 0){ 7 read(fd1, &c, 1); 8 } 9 read(fd1, &c, 1); 10 printf("%c\n", c); 11 exit(0); 12 }</pre>	a.txt 12345
--	---------------------------

Please give **all** the possible output and one execution order for each. You can use line Cx or line Px to distinguish the same line of code executed by child and parent.

1\n3\n: P9->P10->C7->C9->C10

3\n1\n: P9->C7->C9->C10->P10

2\n3\n: C7->C9->C10->P9->P10

3\n2\n: C7->C9->P9->P10->C10

Problem III

<pre>int main(){ int fd1, fd2, fd3; char *buf1=(char*)malloc(10); char *buf2=(char*)malloc(10); fd1 = open("a.txt", O_RDWR, 0); fd2 = open("b.txt", O_RDWR O_APPEND, 0); fd3 = open("a.txt", O_RDWR, 0); if(fork()==0){ read(fd2, buf1, 2); dup2(fd1, fd2); read(fd2, buf1, 1); exit(0); } waitpid(-1, NULL, 0); read(fd2, buf1, 3); write(fd1, buf1, 3); read(fd1, buf1, 10); }</pre>	a.txt abcdefg
	b.txt 0123456789

<pre>printf("%s\n", buf1); read(fd3, buf2, 10); dup2(fd2, 1); printf("%s\n", buf2); free(buf1); free(buf2); exit(0); }</pre>	
--	--

1. What will the contents of a.txt and b.txt be after the program completes?

a.txt: a234efg

b.txt: 0123456789a234efg

2. What will be printed on stdout?

efg