

Name: _____

Directions: Neatly write your response to each question in the space provided on the exam or on separate scratch paper. This is an open book/open notes exam worth 100 points. Although it is an open exam I expect you to understand the material and phrase answers in your own words. You will be downgraded if you copy material directly from the book. You have one hour and fifteen minutes to complete the exam. If you finish early, check your work thoroughly and hand in materials including any scratch paper. Then, quietly leave the room. Good luck on the exam.

1. (10 Points) Describe one situation where symbolic links would be useful, but hard links would not be as appropriate:

- (1) Forming links across mounts.
- (2) Forming links to files that might change.

2. Given the following file descriptor tables for a two-process ring formed by process1 and process2:

	Process1 File Descriptor Table
[0]	read pipe 1
[1]	write pipe 2
[2]	standard error

	Process2 File Descriptor Table
[0]	read pipe 2
[1]	write pipe 1
[2]	standard error

- (a) (10 Points) List code that would make this a three-process ring by adding process3.

Process 2 executes the following code.

```
int FD[2]
pipe(FD);
if (childpid = fork())
    dup2(FD[1], STDOUT_FILENO);
else
    dup2(FD[0], STDIN_FILENO);
close(FD[0]);
close(FD[1]);
```

- (b)(10 Points) Show the changes to the process1, process2 and process3 File Descriptor Tables as key elements of the code in 2(a) above are executed. The most recent changes should be in the rightmost columns.

process1 File Descriptor Table Sequence				
[0]	read pipe1			
[1]	write pipe 2			
[2]	std error			
[3]				
[4]				

process2 File Descriptor Table Sequence				
[0]	read pipe 2	read pipe 2	read pipe 2	read pipe 2
[1]	write pipe1	write pipe1	write pipe1	write pipe3
[2]	std error	std error	std error	std error
[3]		read pipe 3	read pipe 3	read pipe 3
[4]		write pipe 3	write pipe 3	write pipe 3

process3 File Descriptor Table Sequence				
[0]		read pipe 2	read pipe 3	read pipe 3
[1]		write pipe1	write pipe1	write pipe1
[2]		std error	std error	std error
[3]		read pipe 3	read pipe 3	
[4]		write pipe 3	write pipe 3	

3. (20 Points) Write code that performs the following: A parent process attempts to create a named pipe that may already exist called *npipe* and then it creates two child processes. Then, have one child open the named pipe for read and the other child open the named pipe for write. Finally, have the writing process send "hello world" over the pipe to the reading process and then terminate both child processes leaving no zombies.

```

...
char buf[] = "hello world";
strsize = 12;
mode_t mode = S_IRUSR | S_IWUSR;
if ((mkfifo (npipe, mode) == -1) && (errno != EEXIST)) exit(1);
if (childpid1 = fork() > 0) {
    if (childpid2 = fork() == 0) {
        if (fd = open(npipe, O_RDONLY | O_NONBLOCK) == -1)
            perror(..); }
        if (read(fd, buf, strsize) < 0) {
            perror(...);
            exit(1); }
    }
else {

```

```

    if (fd = open(npipe, O_WRONLY) == -1) {
        perror(...);
        exit(1); }
    if(write(fd, buff, strsize) {
        fprintf(... "write to named pipe failed\n");
        exit(1); }
}
if (waitpid(-1, NULL, 0); == -1) perror(...);

```

4. (20 Points) Write code which causes an alarm goes off after 10 seconds that in turn causes the message "ALARM" to be output to standard output.

```

...
struct sigaction act;
...
void catchit() {
    printf("alarm"); }
...
act.sa_handler = catchit;
sigemptyset(&act.sa.mask);
act.sa_flags = 0;
if (sigaction (SIGALRM, &act, NULL) < 0)
alarm(10);
...

```

5. (10 Points) Assume that it took 23521 CPU clock ticks to execute a program component. Write a short program that converts the clock ticks to time in seconds.

```
23525/sysconf(_SC_CLK_TCK)
```

6. (20 Points) Write code in which a parent creates a child process and the child process creates a child of its own (grandchild to the parent). Have the grandchild send a signal terminating the parent, and then have the child and grandchild to terminate without leaving any zombies.

```

parentid = getpid();
if ((childpid = fork()) == 0) {
    if ((gchildpid = fork()) == 0)
        kill(parentid, SIGKILL);
    wait(NULL) };
/* child catches grandchild */

```

/* Note that child doesn't need to be caught because parent is dead */