Professor: Al EhabELmallah

CMPUT 379 - Midterm Exam (20%)

Date: November 1, 2000

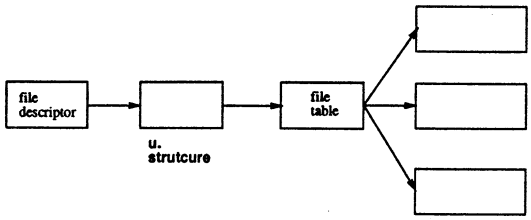
Time: 50 minutes
Total Marks: 50
Questions: 7

Closed Book
Calculators Allowed

- ***No questions during exam time.***
- ***If you are unsure, write down your assumptions.***

Answer the following questions (briefly, and in-point form).

1. [6 marks] The figure below aims at illustrating the relationship between some data structures used in managing *file descriptors* in a typical UNIX environment. The leftmost box corresponds to a user declaration; the rightmost boxes refer to structures that deal directly with the relevant hardware. Complete the figure by (a) labelling the empty boxes, and (b) identifying boxes that should be protected by the kernel. Explain your answer.



2. [6 marks] Recall that in UNIX the *first* file descriptor assigned by `pipe()` is for *reading*. Complete lines 10, 11, 14, and 15 in the following program fragment so that the *parent* redirects its output to the pipe, and the *child* reads from the pipe and writes to the terminal.

```
4 ...
5 #define MAXBUF 128
6 int main (int argc, char *argv[]) {
7     char buf[MAXBUF];    int n, fd[2];    pid_t pid;
8     pipe(fd);    pid= fork();
9     if (pid == 0) {
10        close(...);
11        while ((n= read(..., buf, MAXBUF)) > 0) write(..., buf, n);
12    }
13    else {
14        close (...);
15        dup2(..., ...);
16        execl("/usr/bin/who", "who", (char *) 0);
17    }
18 }
```

3. [6 marks] List at least two disadvantages of the "older" Unix `signal()` environment that have been considered in the newer POSIX `sigaction()` environment.

4. [5 marks] List two requirements that justify implementing UNIX pipes using system calls (as opposed to implementing them as library functions).

5. [5 marks] Suppose that a scheduler has k ready processes at time 0, and that no new processes are created after time 0. Process i ($1 \leq i \leq k$) requires i units of computing time. For a preemptive, round-robin scheduler with a scheduling quantum of 2 time units, what is the mean response time for these processes? (Recall, the *response time* of a process is the interval elapsed from arrival to receiving the CPU for the first time.)

6. [10 marks] Consider the Banker's algorithm on a system with 9 identical units of some resource type. Each of *two* high-demand processes has declared a maximum need of 3 units. Each of the remaining (low-demand) processes has declared a maximum need of 2 units. Within the declared limits, processes can request resource units in any order.

   (a) Assuming that the two high-demand processes are running, what is the maximum number of the low-demand processes that can be accommodated without ever having the possibility of being in an *unsafe* state? Explain your answer.

   (b) Using a *resource allocation graph*, illustrate an unsafe state that may exist if the number of processes exceeds your answer to part (a) by one process. Identify the two high-demand processes on the graph.

7. [12 marks] The following is a proposed solution to solve the mutual exclusion problem where critical sections exist inside the `for(;;)` loops, and protected by the `mutexBegin(i)` and `mutexEnd(i)` code (i is the process number).

```
int  turn= 1, trying[2];
```

```
P1(void) {
#define my_turn 1
#define his_turn 2
for (;;) { ... }
}
```

```
P2(void) {
#define my_turn 2
#define his_turn 1
for (;;) { ... }
}
```

- `mutexBegin(i):`

```
trying[i]= 1;
while (the other is trying) {
     trying[i]= 0;
     wait until it is my turn;
     trying[i]= 1;
};
```

`mutexEnd(i):`

```
turn= his_turn;
trying[i]= 0;
```

   (a) Is the above solution deadlock free? Explain your answer.

   (b) Does it guarantee mutual exclusion? Explain your answer.
