Final exam–May 3, 9:00 AM

Open book section (50 points)

The exam is to be turned in at 12:00 noon. The closed book section should be turned in before you open your books and notes to work the open book section. For the open book section, write your answers on separate pieces of paper.

Problem 1. (5 points)

You are the only person logged into a Unix computer. You connect to another user's directory and type

```
% ls -i foo
```

The system replies with

```
 2050 hi
```

informing you that the directory `foo` has a file `hi` with inode 2050. You then type

```
% cat foo/hi
```

and receive the error message

```
cat:  cannot open foo/hi
```

Name a way the directory `foo` could be protected to make this behavior possible.

Name a way the file `foo/hi` could be protected to make this behavior possible.

In each case, where does the operating system detect the protection violation?

Problem 2. (10 points)

Presently, Unix directories contain file names and their associated inode numbers. Suppose we wanted to "improve" the file system by abolishing inodes and moving the information presently stored in inodes into the directory.

What would be lost and what would be gained by this change?

What system calls would need to be changed?

Problem 3. (5 points)

Suppose you type the command

```
% cat .login
```

to the shell. Describe the system calls the shell uses in executing `cat`.

How would your answer change if the command was

```
% cat <.login
```

Problem 5. (5 points)

Suppose the grades for a course are stored in a read and write protected file as a series of lines each of the format:

```
("242-12-3456" "Pooh, Winnie" 10 10 13 15 30 96 71)
```

Describe how the features of the Unix operating system can be used to implement a single program `mygrades` which will allow each student to see the line containing his or her grades.

Problem 6. (10 points)

Suppose the tape drive on `napoleon` melts this morning and we wish to modify `napoleon`'s operating system kernel so that when a program attempts a tape I/O operation on `napoleon`, the actual tape operation is performed on `dopey`.

Outline (in two or three paragraphs) how the kernel must be changed to accomplish this. Be sure to point out how your solution will use the networking system calls.

Problem 7. (15 points)

Suppose the program on the following page has been successfully compiled and run on a Unix system. (Incidently, it has and executed with no errors.)

Draw a picture showing the pipes created by the processes.

Describe the possible results of executing the program. Pay particular attention to the races between the two children processes. Merely "describing" the possible character(s) output on standard output does not constitute an adequate answer to this question.

By the way, the most likely outputs of executing the program seem to be "`ACa`", "`AaC`", and "`Aae`".

Hint: Identifying the races that are more or less independent makes the problem more manageable.

```
/* The routine shuttle has three parameters:
   * fdin:      a file descriptor which can be read
   * fdout:    a file descriptor which can be written
   * incr:      a small integer
   * shuttle(fdin, fdout, incr)
   *   reads a character from fdin
   *   writes that character to standard output
   *   adds the input parameter incr to the character
   *   writes the incremented character to fdout
*/
shuttle(fdin, fdout, incr)
     int fdin, fdout, incr;
   {
     char c = 'a';
     read(fdin, &c, 1);
     write(1, &c, 1);
     c = (char) ((int) c + incr);
     write(fdout, &c, 1);
   }

main(argc, argv)
     int argc;
     char *argv[];
   {
     int p[2], q[2];
     pipe(p);
     if ( fork() != 0) {
         close(p[0]);
         write(p[1], "A", 1);
         close(p[1]);
         wait(0); }
     else {
         close(p[1]);
         pipe(q);
         if ( fork() != 0) {
             shuttle(p[0], q[1], 2);
             shuttle(q[0], q[1], 3);
             wait(0); }
          else {
             close(q[0]);
             shuttle(p[0], q[1], 4); }}}
```