

ECES 338 Assignment #2
(100 points)

Due: February 15, 2001

Spring 2001
Prepared by E. Markensohn, B. Karas and B. Coate

In this assignment, you will work with pipes. For pipes, please read chapter 5 of the Unix textbook.

(1) Unnamed Pipes: Write a C program to create a process P0 and two child processes, P1 and P2. Have two unnamed pipes PP1 and PP2 created between the parent and its children such that communication is set up between the parent and each of its children. Have P0 set up a log file where it should send a copy of everything it outputs. Have P1 and P2 send their PID and PPID through its pipe. P1 should use dup2() to redirect its stdout to the logfile, then exec() and run the Unix command 'df'. P2 should use dup2() to redirect its stdout to the PP2, then exec() and run the Unix command 'who'. The parent should read the data written to the pipes and output a copy to stdout and to the log file.

Make sure to (i) close/duplicate file descriptors to properly associate the ends of each pipe, and (ii) close the unneeded ends of each pipe.

(2) Named Pipes: Create two programs, P0 and P1. P0 will read lines from the input file shown below. P0 should write the format strings to a named pipe, PP0, and to stdout. Each format string can be used in a printf-like function and will later be used to format a single variable. P1 will open PP0 and read the format strings from the pipe. P1 should use printf/fprintf/snprintf (whichever is applicable) with each format string, and provide a variable of the appropriate type for the function to output. (i.e. , fmt="%f" would be passed to printf(fmt, myfloat);) Choose 'interesting' values to pass to these function calls. The output of P1 should be sent to another pipe, PP1. Use the Unix utility 'cat' to extract the output from PP1.

Hint: You may want to take advantage of the number of each type of format string. There are 6 doubles, 7 ints, 2 chars, and 2 strings, in that order.

Note: The equals are intentional -- they will let you see spaces in the resulting output.

-----begin input file

```
=% f=  
=%15f=  
=%10.3f=  
=%-10.3f=  
=%010.3f=  
=%+10.3f=  
=% d=  
=%15d=  
=%10.3d=  
=%-10.3d=  
=%10.3d=  
=%x=
```

```
=%#.8x=
```

```
=%10c=
```

```
=%-10c=
```

```
=%42s=
```

```
=%-42s=
```

```
-----end input file
```

Turn in your codes and a sample output. Please make a note of your recitation section and your login id.

(3) Named Pipes through the Unix Shell: Create a named pipe from the shell with the `mknod` command. Do `'ls -l'` to list the pipe as a file. Take note of the file permissions on the file, they tell you that it is a pipe. Redirect the output of a command into the pipe (e.g., `last`), and redirect the input of a command (e.g., `grep`) from the pipe. Script the session, and turn it in.

(4) Unnamed Pipes through the Unix Shell: Using unnamed pipes, issue shell commands that achieve all you have done in question 3 above. Script the session, and turn it in.