COP4338

Fall 2016

Caryl Rahn

## Assignment #3: Unix shell with redirects and pipes (due 12/3)

The purpose of this assignment is to learn to develop multi-process programs. You are expected to extend the myshell.c program and add pipelines and I/O redirections. In particular, your shell program should recognize the following:

1. > - Redirect standard output from a command to a file. Note: if the file already exist, it will be erased and overwritten without warning. For example,

COP4338$ ls > 1

COP4338$ sort myshell.c > 2

Note that you're not supposed to implement the unix commands (ls, sort, ...). You do need to implement the shell that invoke these commands and you need to "wire" up the standard input and output so that they "chain" up as expected.

2. >> - Append standard output from a command to a file if the file exists; if the file does not exist, create one. For example,

COP4338$ sort myshell.c >> 1

COP4338$ echo "Add A Line" >> 1

3. < - Redirect the standard input to be from a file, rather than the keyboard. For example,

COP4338$ sort < myshell.c

COP4338$ sort < myshell.c > 1

COP4338$ sort > 1 < myshell.c

The second and third commands are the same: the sort program reads the file named myshell.c as standard input, sorts it, and then writes to the standard output to file named 1.

4. | - Pass the standard output of one command to another for further processing. For example,

COP4338$ ls | sort

COP4338$ sort < myshell.c | grep main | cat > output

Make sure you have your parent process to fork the children and 'wire' them up using pipes accordingly

Your program should be able to run with all the unix commands as expected. Don't assume your program will only be tested by the above commands. Also, there could be more than two commands chained by pipes, like the second example above.

Your program needs to provide necessary sanity-check from the user input. Prompt meaningful errors accordingly as what you'd experience with your shell.

Hint: use strtok() functions to separate the tokens and use strstr() to locate sub-strings.

To start, first create a directory for this homework and name it FirstnameLastnameA3 (of course, you'd use your real name here). You should place the source code and the Makefile in the directory. One should be able to create the executable by simply 'make'. The Makefile should also contain a 'clean' target for cleaning up the directory (removing all temporary files, object files and executable files). Make sure you don't include intermediate files: \*.o, executables, \*~, etc., in your submission. (There'll be a penalty for including unnecessary intermediate files).

Please make sure you submit homework before the deadline. Extensions cannot be given.

NOTE THAT ALL HOMEWORK IS INDIVIDUAL WORK. NO CODE SHARING AND CO-DEVELOPMENT IS EVER ALLOWED. REFER TO CODE OF CONDUCT STATED IN SYLLABUS. ZERO TOLERANCE FOR ETHIC VIOLATIONS IN THIS COURSE.