CSCI 5573: Advanced Operating Systems
Spring 2016

Homework 1
Due date: 02/14/2016

Goal: The main goal of this homework is to review OS programming background typically learned in an undergraduate course in operating systems. You will implement a command-line interpreter or shell. The assignment is divided into three parts: basic features, advanced features and a unique feature. Basic features include standard features that are typically provided by all Linux shells, advanced features include features that are not common in shells, and unique feature provides you an opportunity to come up with a cool feature that is not included in basic or advanced features.

Basic features:

1. When you type in a command (in response to its prompt), the shell creates a child process that executes the command you entered and then prompts for more user input when it has finished. In particular, your shell must use system calls such as **fork( )**, **execvp( )** and **wait( )**.

2. Your shell stops accepting new commands when it sees the quit command on a line or reaches the end of the input stream, i.e. the end of the batch file (see batch mode below) or the user types 'Ctrl-D'. The shell then exits **after** all running processes have terminated.

3. Your shell provides an ability to run a program in the background, whenever a command ends with the ampersand character '**&**'.

4. I/O redirection: Your shell allows redirection of standard input and standard output using '<' and '>'.

5. Pipe: Your shell allows piping of multiple commands using '|', where the output of a previous command becomes the input of the next command.

Advanced features:

1. Interactive and batch modes: Your shell can be run in two modes, *interactive* and *batch*. In interactive mode, you will display a prompt (any string of your choosing) and the user of the shell will type in a command at the prompt. In batch mode, your shell is started by specifying a batch file on the command line; the batch file contains the list of commands (each on its own line) that should be executed. In batch mode, your shell **does not** display a prompt, but it echos each line as it is read from the batch file back to the user before executing it.

2. Implement ‘process substitution’, which allow input or output of a command to appear as a file. The command is substituted in-line, where a file name would normally occur by your shell. The Wikipedia page on ‘process substitution’ provides a good explanation of this feature.

3. Implement a restricted version of ‘**fc**’ command:

fc –n1 –n2

Re-execute the last set of commands in the range from the last n1th command to the last n2th command, where n1 and n2 are positive integers.

A unique feature:

Design a new shell feature that you have not seen before. Describe why such a feature would be useful, provide a clear specification, and implement it. Grading for this part will be dependent on how well you can convince me via a writeup that your unique feature is useful.

Grading:

1. This assignment counts for 5% of your total course grade.

2. 40% of the assignment grade is for the basic features, another 40% is for the advanced features, and the last 20% is for your unique feature.

3. You may do this assignment in teams of size up to two students. Both students must be familiar with the complete assignment and able to answer any questions.

4. Grading will be done in an interview style, where each of you will sign up for a grading slot that will be posted on Moodle.

5. For each part (basic, advanced or unique features), 50% of the grade is for correct implementation and the other 50% is for correctly answering any questions during your interview.

6. Submit your assignment via the submission link provided on Moodle. Submit a single zipfile comprised of all source code files, a Makefile, a Readme file, and PDF file documenting your unique feature. Do not submit any binary file.

7. Submission deadline will be strictly enforced. Absolutely no extensions will be given unless there is a valid excuse. Moodle allows you to do multiple submissions, later submission overwrites the earlier submission. So, you are encouraged to submit partial submissions while working on the assignment and not wait till the last minute. Moodle site may get very busy towards the deadline, so keep that in mind.

Guidelines:

1. I am sure that all of you are familiar with the basic concept of shell. However, I advise you to read about it from various sources on the Internet. For example, you may check out http://web.stonehill.edu/compsci/cs314/assignments/assignment1.pdf that describes a shell programming assignment similar to this assignment and provides several useful hints.

2. Make sure that you understand what needs to be done before starting to write code. Design, design, design first!

3. You may find some code on the Internet that implements a shell. You are welcome to use that code. Please clearly cite your source in the Readme file. Of course, make sure that you understand the code before using it or making any changes to it. You should be able to answer all questions during your grading interview.

4. Start working on the assignment as soon as you can, preferably now, even if you find the assignment to be straight forward.

5. Use Moodle discussion board for any questions you have about the assignment.