CS 4375

Professor Ward

**Shell Assignment**

Purposes: Gain experience weaving system calls into a functional program; learn to manipulate the properties of processes.

Goal: Build a user shell for a Unix operating system.

Functionality

1. Read a unix command from the user, execute it, and repeat. Handle at least the basic commands (**ls**, **cat**, **grep**, etc., typically found in **/usr/bin**), with all their normal parameters. [5 points]
2. Terminate if the input is **exit**. [1 point]
3. Before reading each line, print the prompt string specified by shell variable PS1, but if PS1 is not set, use the default prompt of **$$$$**. [1 pt]
4. Support background tasks, that is, tasks which run without requiring the user to wait before the next command, specified with **&**. [3 pts]
5. Accept commands from a file if one is specified; ignore lines starting with **#** [1 pt]
6. Also run commands that name a program anywhere in the path. [3 points]
7. If a command is not found, print an error message. [1 point]
8. If a command fails (with a non-zero exit value *n*), print "Program terminated: exit code *n*." [1 pt]
9. Change directories with the **cd** command. [1 pt]
10. Support redirection of output with **>** . [2 pts]
11. Support redirection of input with **<** . [1 pt]
12. Support simple pipes with **|** . [4 pts]

Grading: 30 points possible, including 6 points for general code quality and report quality.

Aim for 25 if you want an A, 20 for a B, and 15 for a C.

Hints

1. For the basic loop, you can borrow from someone’s chatbot code.
2. For each command, create a child process that uses **execve** to run it with its parameters.
3. You already know enough for a~c; the other topics will be covered soon.
4. Borrow code from Dr. Freudenthal’s demos in **https://github.com/robustUTEP/os-shell**
5. Read [**http://www.rozmichelle.com/pipes-forks-dups/**](http://www.rozmichelle.com/pipes-forks-dups/)

Constraints:

Use only the following python libraries:

* os
* sys
* re

From os, use only:

* pipe()
* fork()
* dup() or dup2()
* execve()
* wait()
* open() or create() and close()
* read() and write()
* chdir()

In particular, do not use system(), execvp(), or execvpe(). If you think you need to use another system call, consult with the instructor or TA.

Submit

1. a report including
   1. snapshots showing that your system works from the console: (function a ~ d)
   2. output showing that your system works in batch mode (function e)
   3. evidence that your system handles functions e ~ l. We will later provide test code to help you do this.
   4. a paragraph or two describing any interesting features of your system, and/or discussing where you had trouble and what you learned.
2. your code, as a py file that we can download and run

Due February 9