Process Programming Assignment: A Unix Shell

Objectives

- To further develop your programming skills.
- Understand the Unix process model.
- Learn how to program using multiple processes on Unix.
- To learn how processes are handled (fork, exec, wait).

Specification

You are to design and implement a Unix shell, similar to existing Unix shells, such as sh, csh and bash. Your shell will not be fully functional but will be a simple shell. Minimally, it will handle the execution of commands, either a single command or multiple commands. The shell should execute commands that are entered at the command line or in a batch file. The shell will create a child process to execute each command entered. Your shell should be able to execute any existing Unix command or user-developed program (you should be able to run any program you previously wrote for a course). You do not have to implement any builtin shell commands, like cd, history, jobs, etc.

Interactive mode will allow the user to enter a command at the prompt, execute the command, and then display the prompt when the command completes. Batch mode will read commands from a batch file and execute each command. The prompt will not be displayed in batch mode. As a line is read from the batch file, display it to the screen and then execute the command(s). Before executing a command, display it to the screen.

In both interactive and batch mode, a line may exist of a single command or a list of commands, each command separated by a semicolon. Each command will be executed by a child process. A list of commands, separated by a semicolon is simply multiple commands entered on one line. There is no execution difference between two commands entered on two lines or the same two commands entered on one line, separated by a semicolon. In both interactive and batch mode, your shell should continue to run until it encounters the ***quit*** command or reaches the end of input (EOF). When the shell encounters the ***quit*** command or the end of input, then it should exit.

Usage Clause

The usage clause for your program will be:

shell [batchfile]

where the batchfile is an optional argument that will be a file containing a list of commands. Again, each line will be executed separately and the lines may be a single shell command or a list of commands.

Notes

- Be sure to do proper error handling throughout. The shell should exit immediately if the usage is incorrect or if the optional batchfile is specified but does not exist. If there is an error in executing a command, display an appropriate message to the user but continue processing (do not exit your program).
- You may implement a maximum size that any one command can be, if necessary, for array implementation. Be sure to document this value.
- Your shell should be able to handle an empty line in the batch file or a list of commands that have two semicolons next to each other (no command between two semicolons).
- Be sure to test your shell well!

Grading

- You will receive a 0 if there are any compilation errors.
- Your program should not core dump, hang or end prematurely.
- The following items will be considered when grading this assignment
    - Style and format
    - Documentation
    - Error checking
    - Implements specification as described

Turn in

- Source code
- Makefile
- Readme file that must include at least the following
    - Description of program (what it does)
    - How to compile
    - How to run
    - Design overview
    - How you may have handled any ambiguities in the specification
    - Any known bugs or problems