ECE 40800/CSCI 40300
Project #1
Due Date: Feb. 16, 2021

To use for the entire semester, you will have a complete Linux system on your hand. You must have the root user privilege to the system and must be able to handle the startup sequence. This requirement cannot be satisfied by a remote Linux account or a Linux emulator such as WSL (Windows Subsystem for Linux), MinGW, Cygwin, MSYS. You are recommended to choose one of the following methods: 1) Raspbian on Raspberry Pi hardware, 2) Ubuntu on a virtual machine.

This project will implement a command line interpreter or shell. The shell should operate on both interactive mode and batch mode. In the interactive mode, the shell creates a child process that executes the command you entered. Once the child process terminates, the shell prompts for the next user input.

In the batch mode, the shell reads a text file where each line is executed as a command without displaying a prompt. Note that in a Linux system, a text line (aka shebang line) starting with "`#!/path/shell-interpreter`" is considered as a script, which can be directly executed from any Unix shell but the text content is executed by `/path/shell-interpreter` program. The shell specification are as follows:

1. All text of a line following a pond-sign (`#`) will be ignored and considered as a comment unless it is a part of a string.
2. Each line may contain multiple commands separated with semi-colons. Each command separated by a semi-colon should be run simultaneously or concurrently. Note that this is different behavior than standard Linux shells.
3. To exit the shell, the user can type a built-in command "`quit`" or press Ctrl-D.
4. The shell program is invoked as follows:

   ```
   shell script-file
   ```

   where "`script-file`" is an optional text file name. If present, the shell will read each line of the file for commands to be executed (batch mode). If not present, the shell will run in interactive mode by printing a prompt to the user at stdout and reading the command from stdin. If the script-file contains a shebang line, the script-file can be directly executed from a command line as if it is an executable program.
5. The shell program must respond to all input in a reasonable manner. By "reasonable", we mean print an understandable message and either continue processing (if not serious) or exit (if serious) depending upon the situation.
   a. The following situations are considered as errors so that it prints a message (to stderr) and exit gracefully:
      i. Incorrect command-line arguments to your shell program.
      ii. The batch file does not exist or cannot be opened.
   b. The following situation are considered as not-serious and the shell prints a warning message to the user (stderr) and continue processing:
      i. A command does not exist or cannot be executed.
      ii. To make coding your shell easier, you may print an warning message and continue processing for a very long command line (over 512 characters).
   c. The shell is able to handle the following scenarios as not an error or a warning (i.e., the shell should not print an error message):
      i. An empty command line.
      ii. Extra white spaces within a command line.
      iii. The script file ends without "quit" command or user types 'Ctrl-D' as command in interactive mode.
6. (Optional) The shell is very simple; it doesn't have a PATH variable, it doesn't support changing directories, there is no shell history of previous commands it has run, the user can't customize the prompt, etc. Feel free to play around with adding such features.

Notes

- If you have a question about the project, you will post a topic to Canvas/Discussions. You are encouraged to answer to other's questions in the forum. A programming is a social activity, though a programmer seems to be isolated. A programmer becomes a genuine programmer thru programmers.
- The project is a team effort of 4 members. A team will submit a single copy of the report.
- The main part of the report must be no longer than 3 pages including a brief description of your approach (no source codes in the main part)
- The appendix of the report (without page limit) includes program sources, and compile/execute traces to verify that your program has been compiled and executed. Don't forget to add proper comments to the sources for helping to understand them. All source codes and trace outputs are submitted as pure text.
- Each member must submit a Peer Rating Form individually and confidentially. Without the peer rating, you will be considered not to submit the project even though your team does. You should be fair in the peer evaluation.