CSE 410
Summer 2018
Project #1
Due: Friday, May 25th, 11:59 PM
Points: 30

The Objective. For this assignment, you are to design and implement a C/C++ program that will serve as a rudimentary command line interpreter (shell).

Questions. Work on the self-study code and understand the examples we studied in class. If you have a difficulty with the project, please attend the office hours, contact either me or a TA, or post a question on Piazza.

How to submit the assignment. Submit your solution via Handin. You can find the link on this course's website.

Assignment Specifications.

1) The program will repeatedly display a prompt containing the sequence number of the current command (starting at 1) and the username of the user executing the program. This information will be enclosed in the characters '<' and '>'. For example:

<1 cse410>

The sequence number and username will be separated by a single space.

2) After displaying the prompt, the program will read one line of input from the user. An input line is defined as a sequence of zero or more tokens (character strings), separated by one or more delimiters (blanks and tabs), ending with a newline character. The ampersand character ('&') is defined to be a separate token. There will be no more than 128 characters in a line.

If the first token is the name of a **built-in command** (listed below), then the program will take the appropriate action. Otherwise, the program will assume that it is an **external command** (the name of a file containing an executable program).

3) For ***external commands***, the program will create a child process; the child process will use a member of the "**exec**" family of system calls to execute the external command.

If the last token is an ampersand ('&') and the command is an external command, the program will execute the command in the background. That is, the program will not wait until the completion of the given external command before continuing with its processing.

4) The program will recognize the following ***built-in commands*** :

quit terminate the shell process
date display current date and time
curr display absolute pathname of current directory
env display all environment variables
cd Dir move from the current directory to Dir
hlist display recent input lines
dlist display recent directories

Built-in commands will be completely processed by the program (the program will **NOT** create a child process to perform the processing).

4-1) The command "**date**" will display the current date and time.

4-2) The command "**curr** " will display the absolute pathname of the current working directory.

4-3) The command "**env** " will display the name and value of all environment variables (such as USER, LOGNAME, PATH, and so on).

4-4) The command "**cd Dir**" will reset the working directory to be "Dir", where that token must be a relative pathname or an absolute pathname.

The command "**cd ~xxx**" will reset the working directory to be the home directory of the user with username "xxx". For simplicity, you may assume that all home directories reside in "/user". As a special case, the symbol "~" represents the home directory of the current user.

4-5) The command "**hlist** " will display the most recent input lines entered by the user (along with the associated sequence number). An example of partial "hlist" output:

197 cd /user/cse410/Projects
198 curr
199 dtae
200 date

The history list will hold up to ten non-null input lines, including any lines with misspelled commands (such as command #199).

The command "**!N**" will cause the program to use command number N from the history list as its input. The contents of that input line will be displayed before the program processes the command(s) within the line.

4-6) The command "**dlist** " will display (in reverse order) the most recent directories that were the working directory, along with the associated index. An example of partial "dlist" output:

1 /user/cse410
2 /usr/include/sys 3 /usr/include

The directory list will hold up to ten absolute pathnames, with the current working directory always at index 1.

The command "**cd #N**" will reset the working directory to be the directory which is held at index N in the directory list.

5) The program will perform appropriate error handling. It will display an appropriate message if the user's command fails in any way.

Assignment Deliverables. The deliverables for this assignment include the following files:

1. project01.makefile -- a make file which produces the executable "project01"
2. project01.*.c -- the source code file(s) for your solution
3. Project01.*.h -- the interface file(s) for your solution

Be sure to use the specified file names, and to submit your files for grading via the "handin" program.

Assignment Notes. There are a number of system calls and library functions which may be useful for this project.

man -s 2 fork man
-s 2 wait man -s 2
waitpid man -s 2
exec man -s 2
chdir man -s 2
time man -s 3

getcwd man -s 3
ctime man -s 3
cuserid man -s 3
string

Your solution must compile, link and execute on "cse410" using "g++".