CS2750 **System Programming and Tools**
Assignment # 7 Due Date: May 09, 2018

C Programming – Building a Small Shell

This assignment is optional.

Write the code in C. I want to see the use of `git` and `Makefile`. Use the `README` file to indicate what is deing done, especially if you have done something new, and how to execute the program.

Use top-down design to design and code the implementation. Use good documentation.

In this assignment, you will create a minimal shell in Unix, called `mysh`. You will start with creating a prompt on the terminal for the user. The user will enter a command to be executed. You will search for the executable corresponding to the command in all the directories specified in the environment variable `MYPATH`. This variable should be set with at least three directories before you start `mysh`. In case the variable is not set, use the variable `PATH` to search for the executable.

You can set the variable `MYPATH` using `:` to separate directories. For example, in `bash`, you can set it as `export MYPATH=/bin:/usr/local/bin:${HOME}/bin:.`

You will search the directories in sequence. As soon as you find a filename corresponding to the command, check to make sure that it is an ordinary file and its execution bit is set for you. If these conditions are met, execute the command by `fork`ing and using `execl`. Otherwise, keep on searching in the directories. If you cannot find the command, issue the message saying `Command not found` preceded by the name of the command.

After executing the command, issue the prompt again. If the command entered is `exit`, terminate your shell.

Take the prompt from environment variable `PS1` if set. Otherwise, use `$ ` as the default prompt.

Implementation

You will start by receiving the prompt `PS1` from you environment by using `getenv`. You should parse it by a separate function so that you can modify the prompt from within your shell. In case you are using `bash` and have your prompt already set, I'll expect you to handle these modifiers: `\u`, `\h`, `\w`, and `\n`. If `PS1` isnot specified in your environment, make sure to use `$ ` for your prompt (`$` followed by a space). If `PS1` contains any other characters, they should be used as literals.

Parse the path (either from `MYPATH`, if set, or from `PATH`) and store it in an array of tokens. You will get that using `makeargv`.

After handling the prompt, you should start an infinite loop. In each iteration, issue the prompt. If the user types `exit` or `logout`, or `CTRL-D`, break from the loop and terminate `mysh`. If the user changes directory, handle that using `chdir`; make sure to verify that the user is changing to a valid directory for which he has permissions. If needed in PS1, modify the prompt (hence the function to handle this part). Continue in the loop.

At this point, you know that the user has entered a command. Search for the command using the tokens from your path. For each element in the path, append the command name to the token and use stat to see if the application exists. If it does exist, verify that you have appropriate execute privilege. If not, go to next token in the path. If the command is not found, issue appropriate message and continue in the loop.

Once the command is found, fork and in the child, execute the command using your favorite version of exec. The parent waits for the child to finish and once that happens, continues in the loop.

Make sure that you have a Makefile, a README file, and use version control.

Extra Credit

Allow your shell to kill the currently executing command by typing a CTRL-C. When the user hits CTRL-C, the command that was executed from within mysh is terminated. The only time you terminate mysh is when the user types a CTRL-D or exit command.

Submission

Create a directory *username*.7 in your $HOME where *username* is your user name on delmar. Keep all programs and datafiles for this assignment in this directory.

You do not have to submit any hard copy of the code. Write the code in C on delmar. Follow good programming principles and document your code well.

After you are done with the assignment, execute the following commands:

```
% cd
% chmod 755 ~
% ~bhatias/bin/handin cs2750 7
% chmod 700 ~
```