

Class: CSC-415-01 Spring 2020

Name: Pedro Souto

Student ID: 918412864

Project: Assignment 1 - Simple Shell

File: main.c

Description: Simple shell that runs on top of the regular command-line interpreter for Linux. The shell reads lines of user input, then parses and executes the commands by forking/creating new processes.

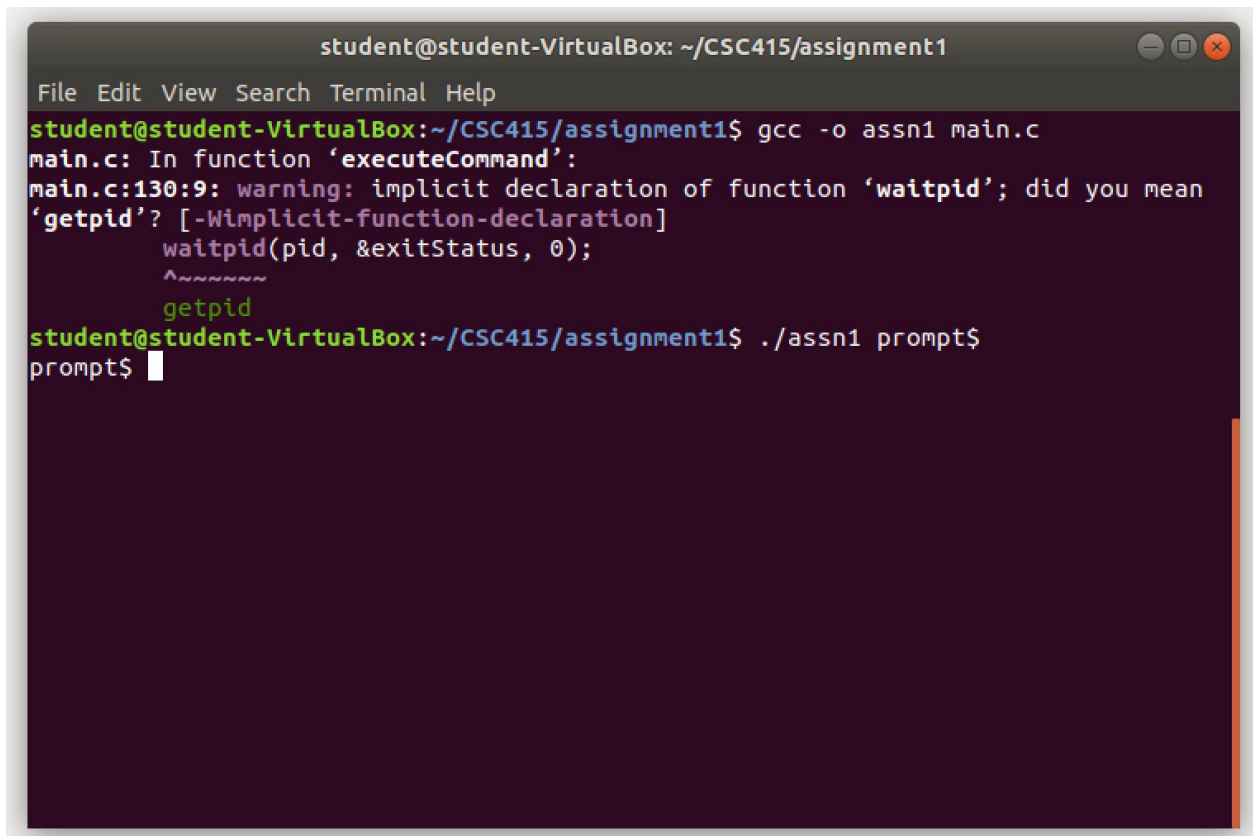
Homework 1 Writeup

For this shell program, I decided to break it down into different components I knew would be required. The first component was taking custom prompt from the user. This prompt came from arguments from the main launch of the shell. In order to solve this, I took the argument count, and if it was equal to 2, then the user passed something. The reason for this is that the size will always be at least one, since the program name is always passed in as an argument. This means that when a prompt is given, the argument count will be 2. With this knowledge, I was able to extract the second string from `argv`, and used that as the prompt. On the other hand, if the argument list was one, I was able to use the predefined ">" prompt.

The second component was tokenizing user input into arguments that needed to be executed. In order to do this, I get user input via `fgets()` and then tokenize it via `strtok()`. I insert those arguments into a `*char[]`, which `execvp()` can use. After null terminating the array, I pass that array into my final component - my `executeCommand()` function.

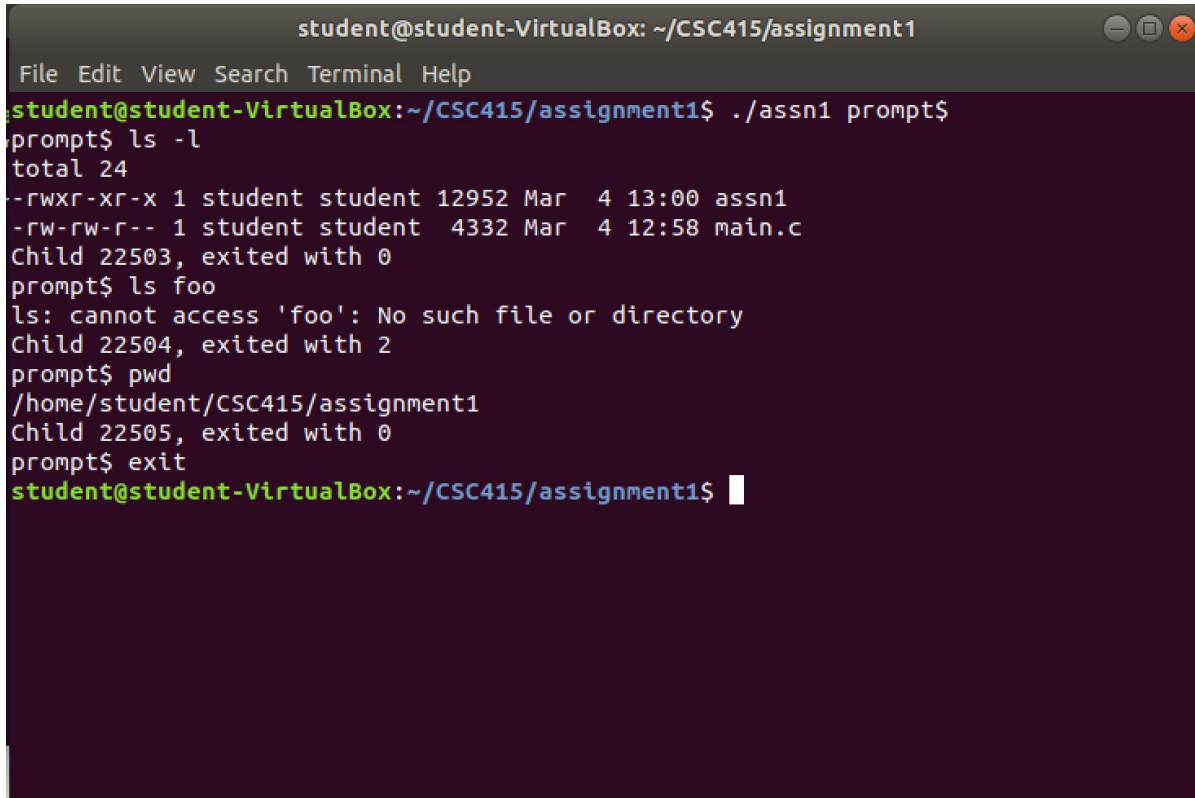
This executor function takes in an argument list, then forks a child. Assuming the fork was successful, the child calls `execvp()` using the passed in argument list to run the command. Meanwhile, the parent is waiting for the child to finish this process. Once the child finishes executing, the parent is done waiting, and we move back to the user input component, where the loop starts over.

Compiling the C program was easy enough, as it was only one file. I choose to name my file main.c. In order to compile it, the *command* `gcc -o assn1 main.c` can be executed. Then, to run the executable, the command `./assn1` can be used. Furthermore, you can pass in a prompt by using the command `./assn1 "prompt"`. Below is a screenshot describing the compilation and execution of my simple shell.



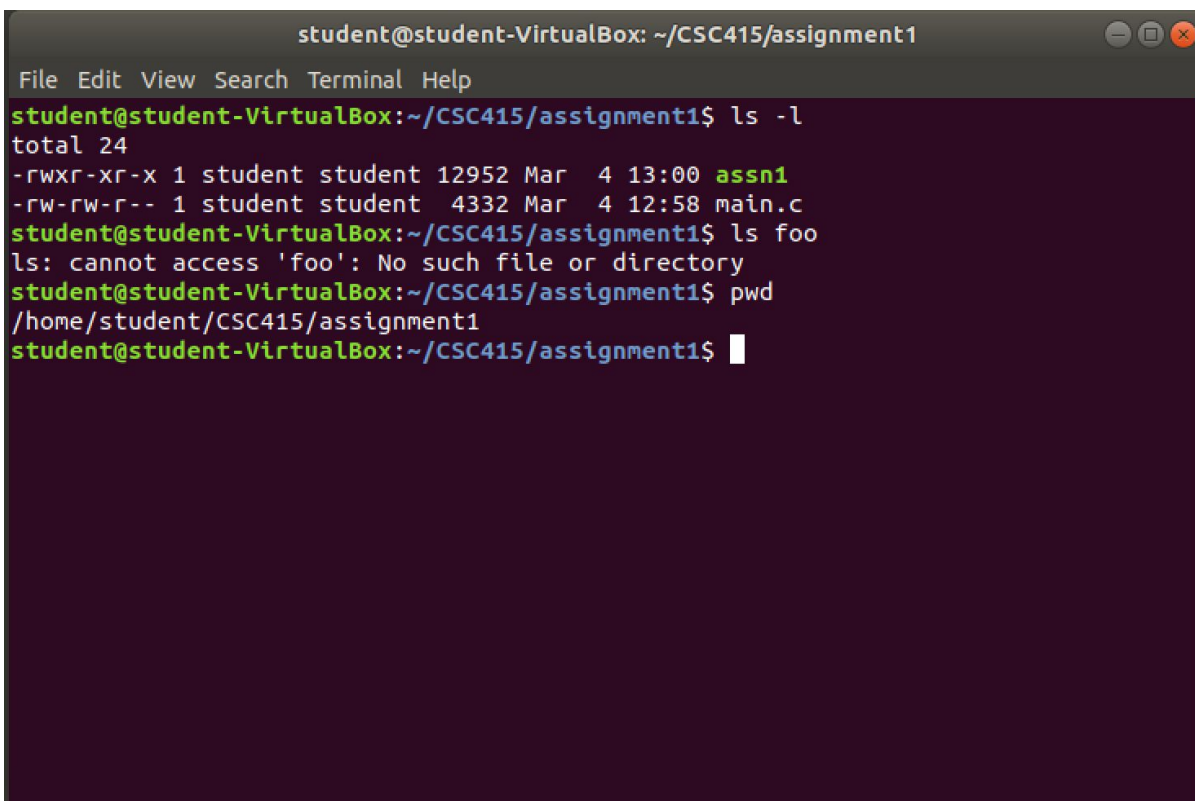
```
student@student-VirtualBox: ~/CSC415/assignment1
File Edit View Search Terminal Help
student@student-VirtualBox:~/CSC415/assignment1$ gcc -o assn1 main.c
main.c: In function 'executeCommand':
main.c:130:9: warning: implicit declaration of function 'waitpid'; did you mean
'getpid'? [-Wimplicit-function-declaration]
      waitpid(pid, &exitStatus, 0);
      ^~~~~~
      getpid
student@student-VirtualBox:~/CSC415/assignment1$ ./assn1 prompt$
prompt$
```

Once the simple shell is running, it will take in command to execute, just like the regular terminal. Below is a screenshot of sample output from the simple shell.



```
student@student-VirtualBox: ~/CSC415/assignment1
File Edit View Search Terminal Help
student@student-VirtualBox:~/CSC415/assignment1$ ./assn1 prompt$
prompt$ ls -l
total 24
-rwxr-xr-x 1 student student 12952 Mar  4 13:00 assn1
-rw-rw-r-- 1 student student  4332 Mar  4 12:58 main.c
Child 22503, exited with 0
prompt$ ls foo
ls: cannot access 'foo': No such file or directory
Child 22504, exited with 2
prompt$ pwd
/home/student/CSC415/assignment1
Child 22505, exited with 0
prompt$ exit
student@student-VirtualBox:~/CSC415/assignment1$
```

The output should be the same as the regular terminal commands, which can be verified by executing commands on the regular terminal and comparing the outputs. Below is a screenshot of sample output from the regular terminal.



```
student@student-VirtualBox: ~/CSC415/assignment1
File Edit View Search Terminal Help
student@student-VirtualBox:~/CSC415/assignment1$ ls -l
total 24
-rwxr-xr-x 1 student student 12952 Mar  4 13:00 assn1
-rw-rw-r-- 1 student student  4332 Mar  4 12:58 main.c
student@student-VirtualBox:~/CSC415/assignment1$ ls foo
ls: cannot access 'foo': No such file or directory
student@student-VirtualBox:~/CSC415/assignment1$ pwd
/home/student/CSC415/assignment1
student@student-VirtualBox:~/CSC415/assignment1$
```