Project 2: Unix/Linux Command Line Interpreter

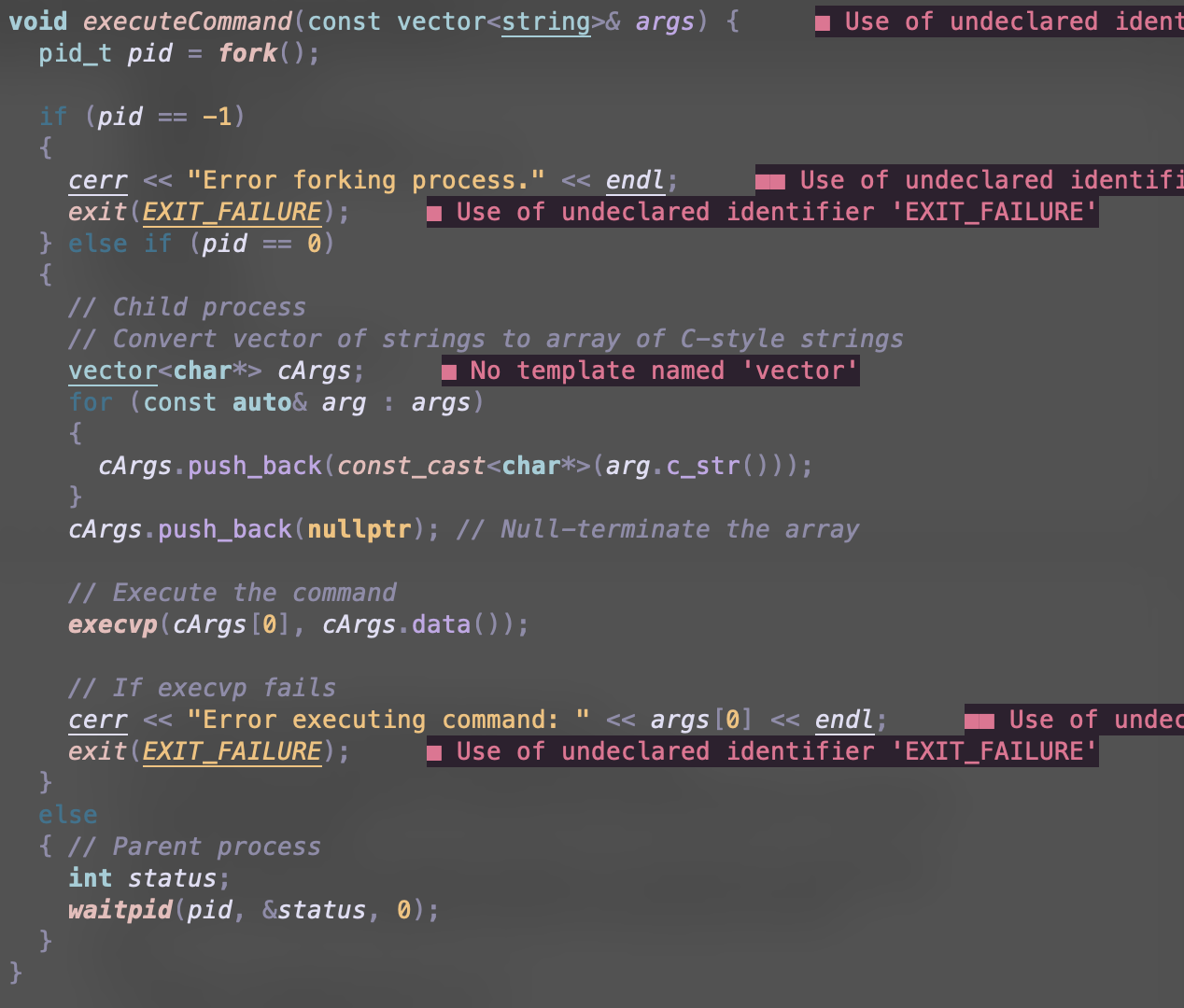
CST-315 Operating Systems

Angel Velazquez

February 11, 2024

In this project, I attempted to avoid using if statements to invoke commands. Instead, I tokenize the input into a C-style string that is sent to the executeCommand function. I continuously promt the user for input, and utilize the fork command to create child processes that wait for the parent process to finish executing.

In the screenshot below, I show the algorithm used to execute the shell commands.

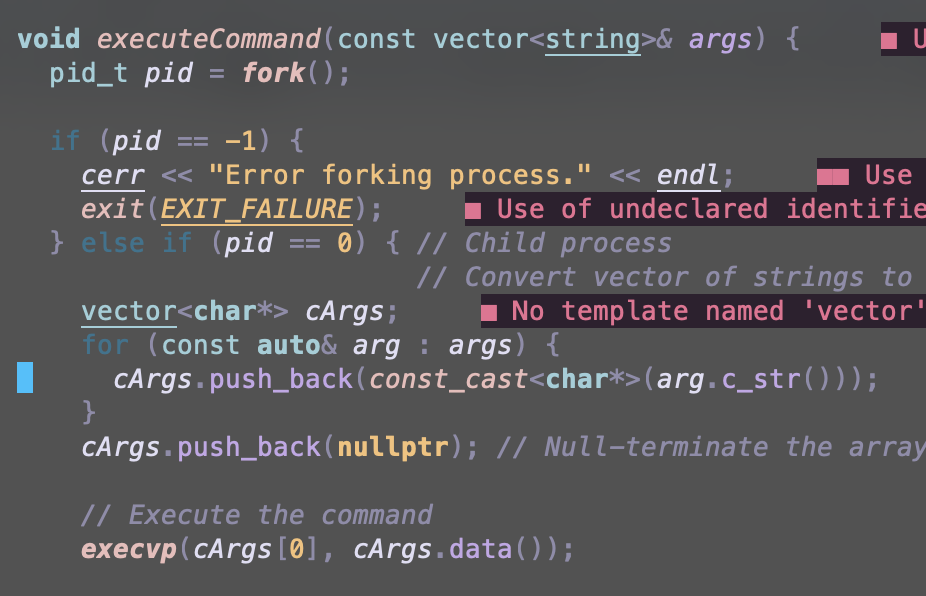
****

**Prompt the user for an input command.**

****

For this project, the shell name is just ‘SimpleShell>’. This makes it easy to understand that the shell is running.

**Create a child process (use execv() and fork()).**

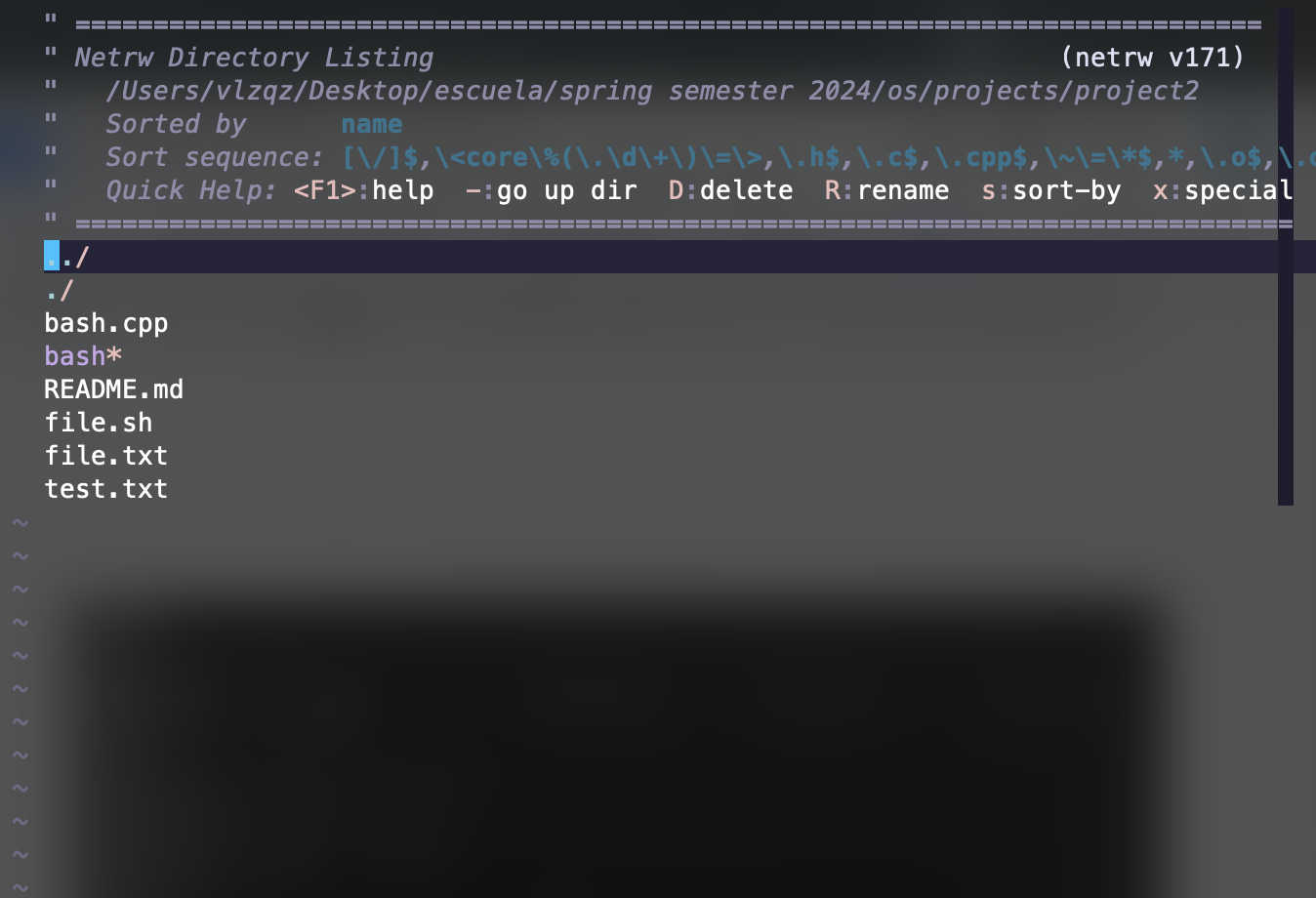
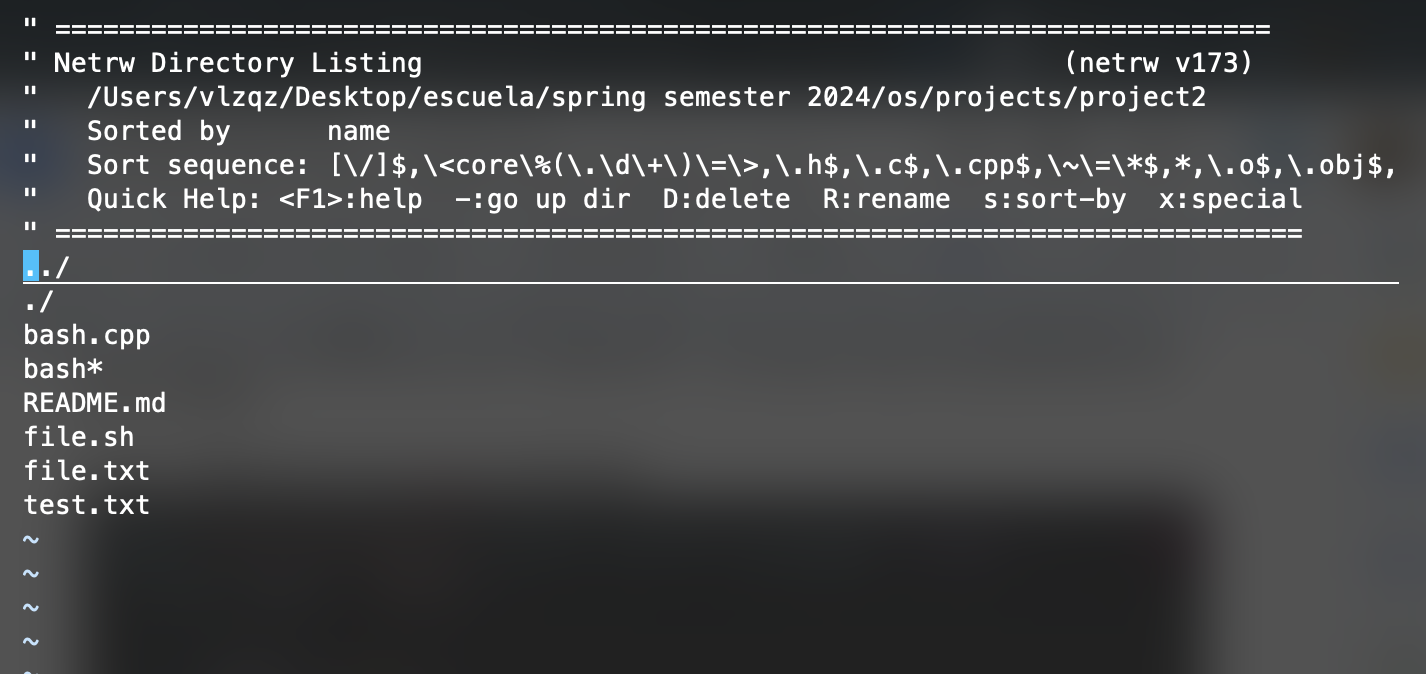
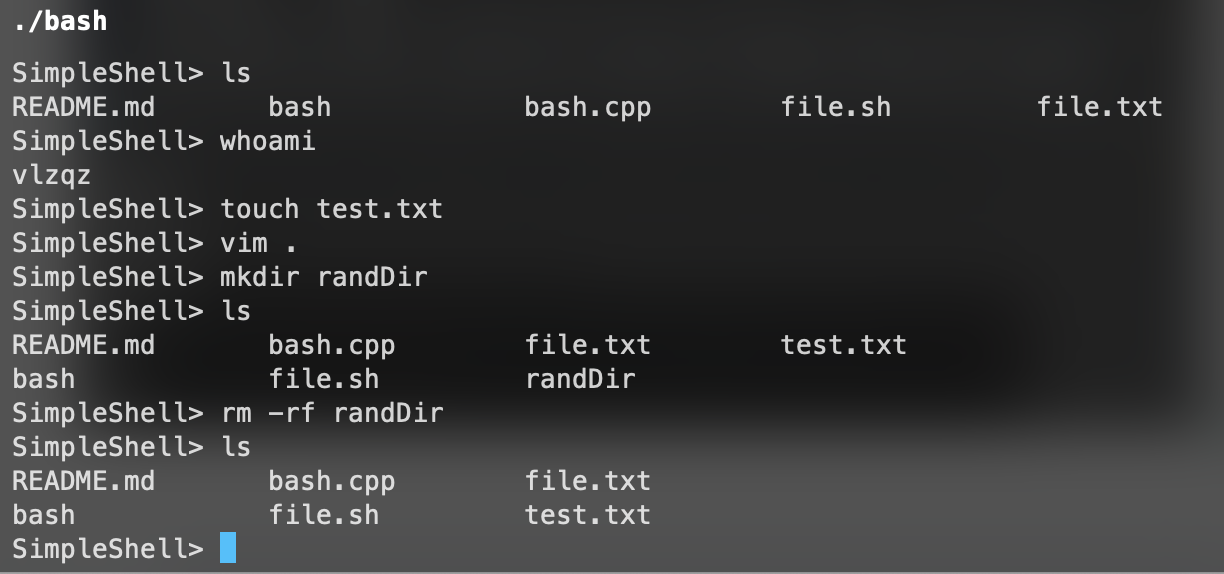


Taking a screenshot from my vim text editor. You can see that the program utilizes the fork() and execvp() commands to create child processes.

**Execute the command entered by the user**

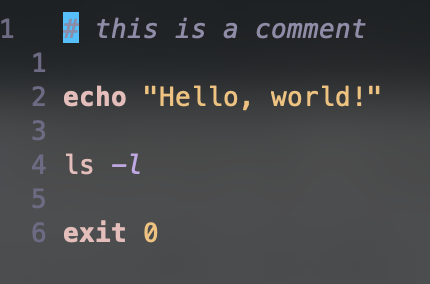
In the screenshots below, I am testing different commands to make sure that they work the way that they are intended. I will list 5 commands below.

In the screenshot below, you are able to see the commands implemented into my bash. I use ls, whoami, touch to create files, vim to edit files, mkdir to create directories, and rm to remove contents from my current directories. I included two screenshots of my vim homepage to show that the vim command works in my shell. The second screenshot below is the vim homepage ran on the shell. It does not support syntax highlighting. However, the third screenshot does have syntax highlighting, because it is running on my system’s shell, not on my shell.



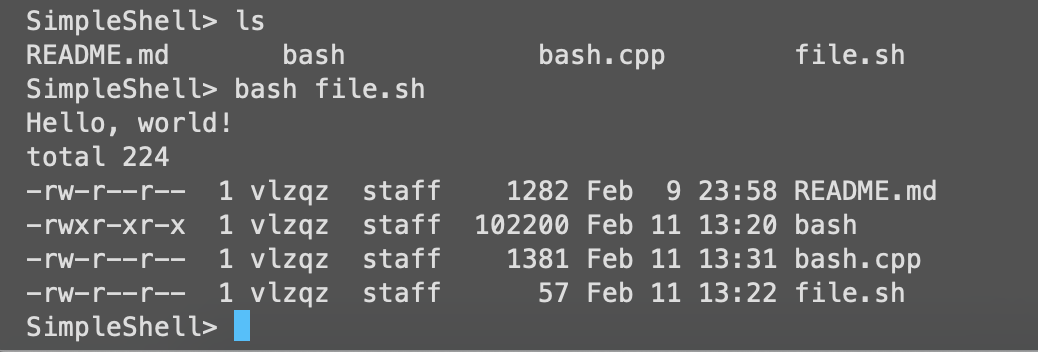
**The user creates a batch file with a list of commands.**

In the screenshot below, I created a .sh file with a list of simple commands to run.



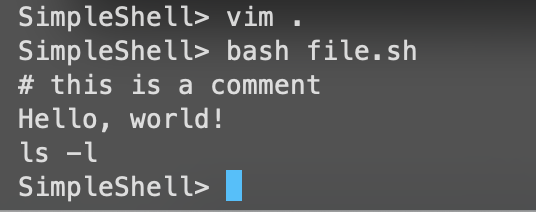
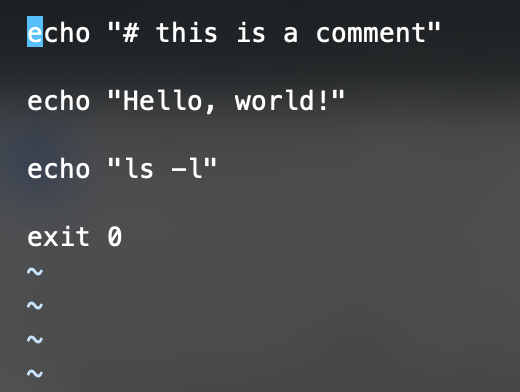
**Start the shell by providing the name of the batch file as argument. When in batch mode, do not display the prompt until execution of ALL commands is complete.**

In the screenshot below, I use ls to find the .sh file. Then, I run bash file.sh in my CLI to run the file. You can see that the bash outputs all commands at once.



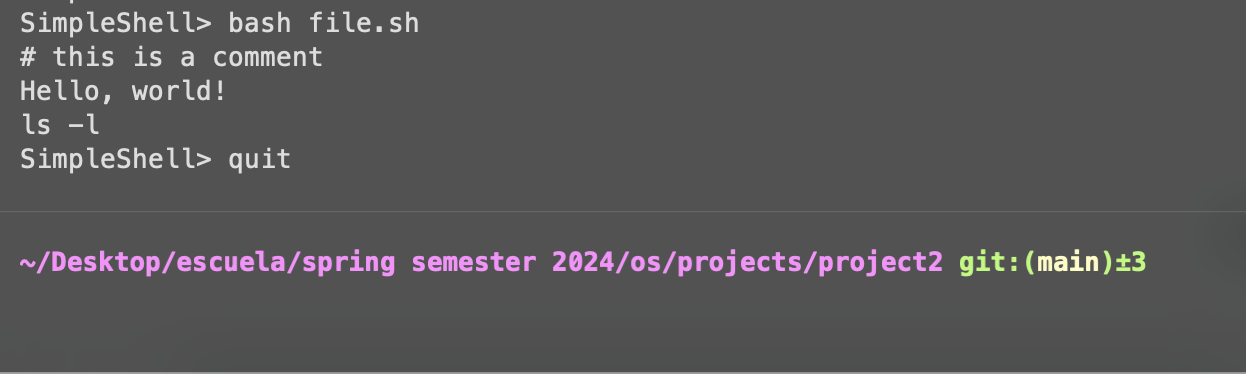
**Display (echo) each line in the batch file and execute it.**

In the screenshots below, I put all arguments into an echo command and it shows the commands printed out into the terminal.



**Show execution of the exit command**

In the screenshot below, you can see that I simply enter ‘quit’ to quit the bash and return to the systems CLI



**Github Repo:** [Click Here](https://github.com/angel-vlzqz/Operating-Systems/tree/main/projects/project2)