Lab exercise 3

Basic Shell

Due date

Sep 25, 2023 11:59 PM

Table of content

Academic Integrity

The following actions are strictly prohibited and violate the honor code. **The minimum penalty for plagiarism is a grade of zero and a report to the Aggie honor system office.**

- Uploading assignments to external websites or tutoring websites such as Chegg, Coursehero, Stackoverflow, Bartleby, etc.
- Copying code from external websites or tutoring websites such as Chegg, Coursehero, Stackoverflow, Bartleby, etc.
- Copying code from a classmate or unauthorized sources and submitting it as your work.
- Sharing your code with a classmate.

Keywords

Process, pipes.

Introduction

The objective of this exercise is to help you get ready for PA2 by expanding lab exercise 2 from hard-coded commands to a basic shell that supports pipes.

Expected Functionality

Your code should execute any number of piped commands that outputs the final result to the terminal. After the output has been printed, the shell will request additional commands from the user until the user enters ‘exit’.

You must prompt the user for the commands to run. Once you have the line from the user, **parse it** for individual commands and their arguments. Run all commands given similarly as in LE2. After the children have been completed, reset the main shell’s input and output and return to the user prompt.

Previously, we hardcoded the shell to call fork twice. Since there will be an unspecified number of commands for each user input, we must adjust how we call fork and execvp. This likely means a loop. See the video for details.

Starter Code

You are given the file shell.cpp to edit and the makefile required to compile the code. The file ‘shell.cpp’ will be the same as the starter code you received for LE2, please replace shell.cpp with the file you created for LE2. You are also given a Tokenizer class that you can use to parse the input from the user.

Rubric

There are public test cases checking that your code compiles correctly and performs appropriately. You need to make sure that the program can take multiple lines of input from the user and provides appropriate output.

1. Compilation (10 pts)
2. No stalling, returns to prompt after output (15 pts) [Manually Graded]
3. Correct output for two commands (25 pts)
4. Correct output for multiple lines (25 pts)
5. Correct output for three or more commands (15 pts)
6. No memory leaks (10 pts)

Getting started

The assignment template is hosted on GitHub classroom. Complete the following steps to get started:

1. Go to the assignment's GitHub classroom: https://classroom.github.com/a/tMxgth-j
2. Watch the Tutorial video (Disregard the title LE2).