

Assignment 3 – Simple Shell with Pipes

Description:

Creating a simple shell program for Linux that's capable of executing commands, parsing user input, handling errors, and supporting piping. The shell reads lines of user input, tokenizes them, and executes commands by creating new processes. It waits for child processes to complete before printing their PID and returning the result.

Approach:

I use C language to complete this assignment.

1. Using if statement to check if command-line arguments are provided. If additional arguments are provided, updates the prompt variable with the value of the second argument (`argv[1]`), allowing customization of the shell prompt.
2. Using a while loop that prompts the user with the current value of prompt until an EOF condition is encountered. And using the 'printf' function to display the prompt.
3. Use the 'fflush' function to ensure that flushes the standard output buffer to ensure the prompt is displayed immediately.
4. Reads a line of input from the user using 'fgets' function. And use if statement to check if the 'fgets' return NULL. If fgets returns NULL, it indicates an EOF or an error condition while reading input.
5. Using the `input[strcspn(input, "\n")] = '\0'` as an index to access the character in the 'input' string just before the newline character. Ensuring that the 'input' string is properly null-terminated after removing the newline character, making it suitable for further processing as a null-terminated string.
6. Using 'strtok' to tokenize the input string, splitting it into tokens separated by spaces. If the first token is NULL, it indicates an empty command or input consisting only of whitespace characters. Prints an error message to the standard error stream and continues to the next iteration of the loop. Otherwise, assigns the first token (command) to `args[0]`.
7. Using a loop do-while to tokenize the input string again to extract command-line arguments. Let the index 'i' assigns each token to the args array until reaching the maximum number of arguments or the end of input. '`args[i] = NULL`' is setting the next element of the args array to NULL to indicate the end of the argument list.
8. Using if statement to check for the presence of a pipe symbol ('|'). If a pipe symbol is found, sets the `pipe_flag` to 1 to indicate the presence of a pipe. And set the current element to NULL, splitting command into two separate commands at the pipe symbol. Additionally, stores the token after the pipe symbol in the `pipe_token` variable.
9. Create a variable 'exitRequested' and use 'strcmp' function to compare the first element of the args with the string "exit". If the result of `strcmp()` is not 0, it means that the two strings are not equal, indicating that the user did not enter "exit". If `exitRequested` is equal to 0, it means that the user input is equal to "exit", so it breaks out of the loop to exit the shell.
10. Using 'fork()' to create a child process. First, in the child process, executes either a piped command (EPIPE) if a pipe is present or a non-piped command (ESC) based on the value of

pipe_flag. Second, in the parent process, waits for the child process to finish using WaitCP and prints its exit status. If fork() fails, prints an error message using perror. Returns 0 to indicate successful execution of the program.

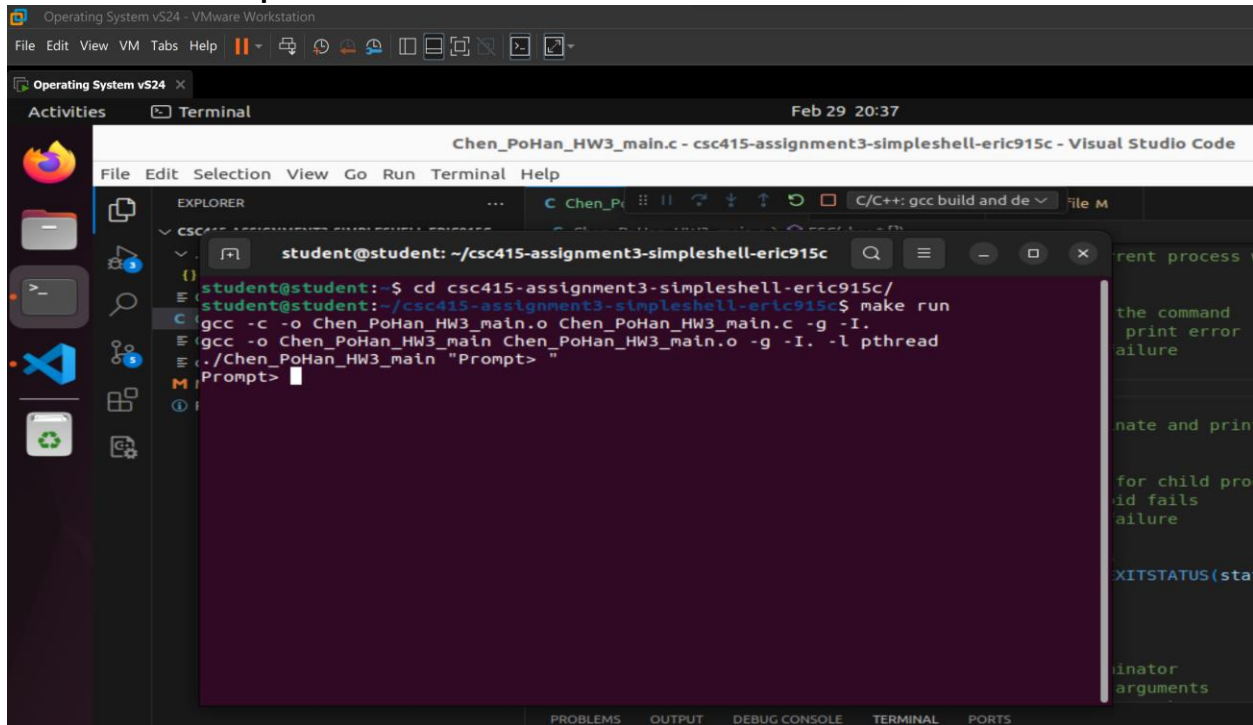
11. The function 'EPipeC' is responsible for executing two piped commands. And use if statement to check if the creation fails (pipe returns -1), it prints an error message using perror and exits the program with failure.
 - a. Forks the first child process (pid1) to execute the first command. If fork returns -1, indicating a failure, it prints an error message and exits the program. If pid1 is 0, it means the current process is the child process. In the child process, it closes the unused read end of the pipe, duplicates the write end of the pipe to the standard output (stdout), and closes the original write end of the pipe. And use 'execvp' executes the first command. If 'execvp' fails, it prints an error message using 'perror' and exits the child process with failure.
 - b. Forks the second child process (pid2) to execute the second command. If fork returns -1, indicating a failure, it prints an error message and exits the program. If pid2 is 0, it means the current process is the child process. Same as first child process, it closes the unused write end of the pipe, duplicates the read end of the pipe to the standard input (stdin), and closes the original read end of the pipe. And use 'execvp' executes the first command. If 'execvp' fails, it prints an error message using 'perror' and exits the child process with failure.
 - c. In the parent process, it closes both ends of the pipe. It waits for both child processes to finish using waitpid. Prints the child process ID and exit status of the second child process using printf.
12. The function 'ESC' is responsible for executing a single command. Use 'execvp' to execute the specified command. And attempts to replace the current process with the command specified by args[0] (the command name) and its arguments (args). If 'execvp' returns -1, it indicates an error occurred during execution. Use 'perror' to indicate the failure of execvp. Using exit(EXIT_FAILURE) to exit the process with failure status.
13. The function 'WaitCP' waits for a specific child process to terminate and then prints its exit status. 'waitpid' is used to wait for a specific child process with the process ID 'pid' to terminate. And stored the exit status of the child process in the variable 'status'. If 'waitpid' returns '-1', it indicates an error occurred during waiting. Use 'perror' to indicate the failure of 'waitpid' and prints an error message. Using 'exit(EXIT_FAILURE)' to exit the process with failure status. Using 'printf' to print the process ID of the child process 'pid' and its exit status. 'WEXITSTATUS(status)' extracts the exit status from the 'status' variable.

Issues and Resolutions:

My first issue was the "|" is an invalid option because I didn't add the if statement and use 'strcmp' function to let 'args[j]' to compare with the string "|".
I resolved it by adding the If statement and used the 'strcmp' function.

Analysis: (If required for the assignment)

Screen shot of compilation:

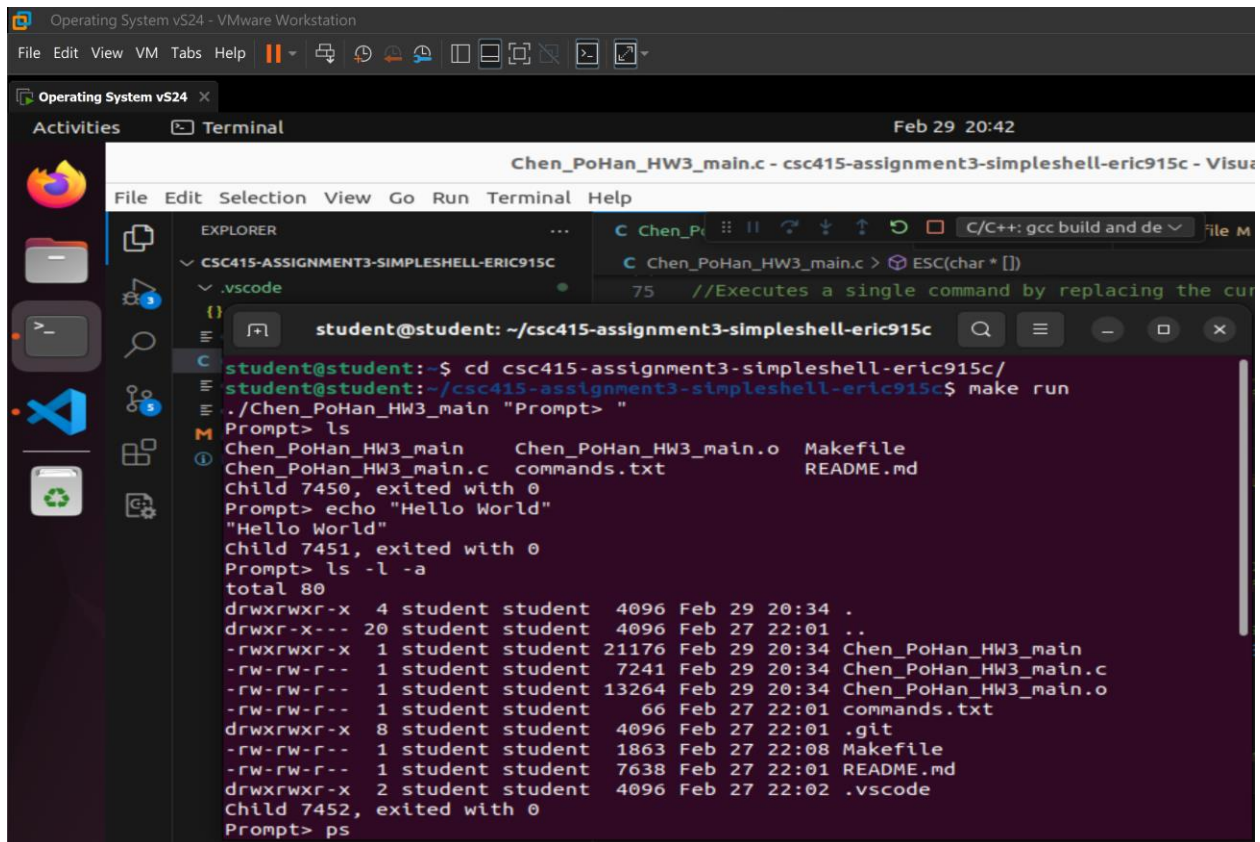


The screenshot shows a terminal window within a VMware Workstation environment. The terminal is titled "Operating System vS24" and "Feb 29 20:37". The user is in a directory named "csc415-assignment3-simpleshell-eric915c". The terminal output shows the following commands and their results:

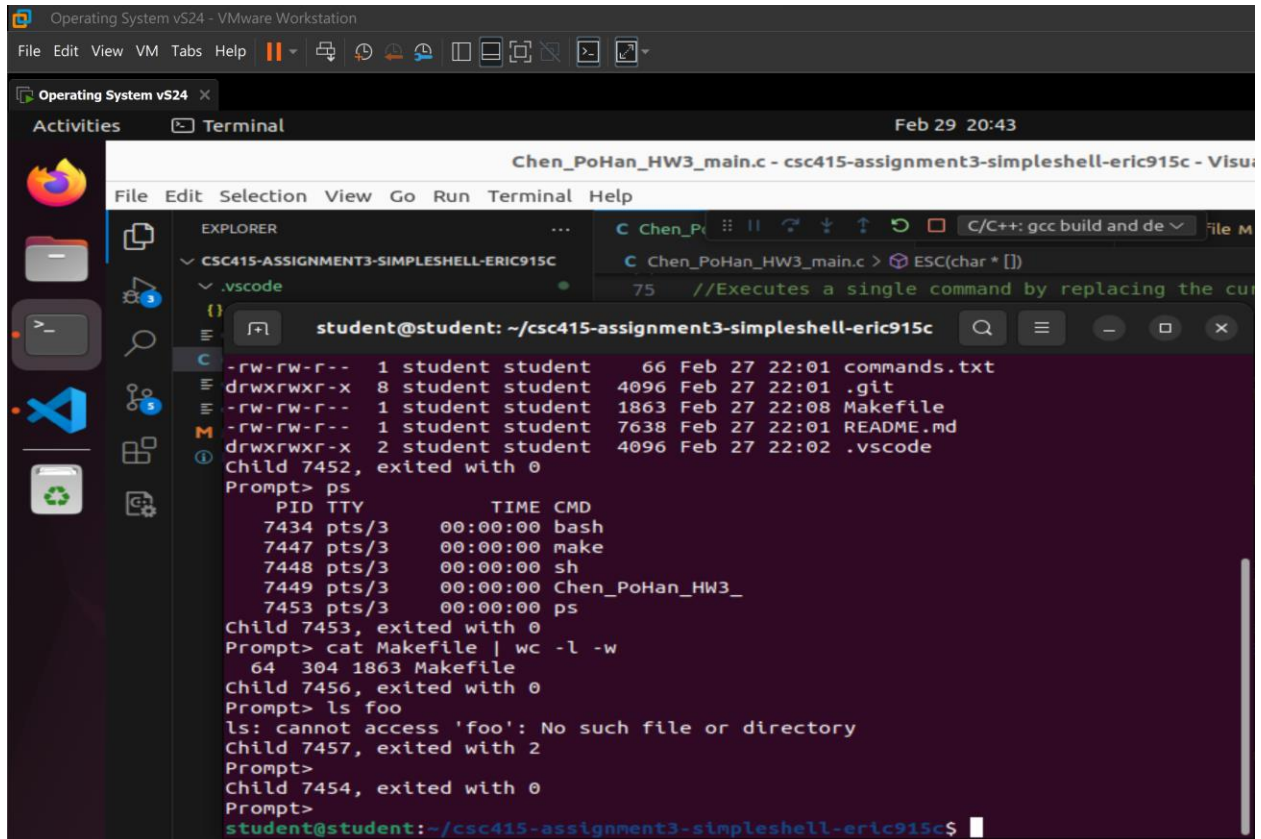
```
student@student:~$ cd csc415-assignment3-simpleshell-eric915c/  
student@student:~/csc415-assignment3-simpleshell-eric915c$ make run  
gcc -c -o Chen_PoHan_HW3_main.o Chen_PoHan_HW3_main.c -g -I.  
gcc -o Chen_PoHan_HW3_main Chen_PoHan_HW3_main.o -g -I. -l pthread  
./Chen_PoHan_HW3_main "Prompt> "  
Prompt>
```

The terminal window is part of a larger application window titled "Chen_PoHan_HW3_main.c - csc415-assignment3-simpleshell-eric915c - Visual Studio Code". The Visual Studio Code interface is visible, showing the Explorer panel on the left with the file "Chen_PoHan_HW3_main.c" selected. The main editor area shows the C code for the program, which includes a loop that reads input from the user and prints it back. The terminal output shows the program running successfully and printing the prompt "Prompt>".

Screen shot(s) of the execution of the program:



```
Operating System vS24 - VMware Workstation
File Edit View VM Tabs Help
Operating System vS24
Activities Terminal Feb 29 20:42
Chen_PoHan_HW3_main.c - csc415-assignment3-simpleshell-eric915c - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
CSC415-ASSIGNMENT3-SIMPLESHELL-ERIC915C
.vscode
Chen_PoHan_HW3_main.c
Chen_PoHan_HW3_main.o
Makefile
README.md
Chen_PoHan_HW3_main.c
75 //Executes a single command by replacing the cur
student@student: ~/csc415-assignment3-simpleshell-eric915c
student@student:~$ cd csc415-assignment3-simpleshell-eric915c/
student@student:~/csc415-assignment3-simpleshell-eric915c$ make run
./Chen_PoHan_HW3_main "Prompt> "
Prompt> ls
Chen_PoHan_HW3_main      Chen_PoHan_HW3_main.o  Makefile
Chen_PoHan_HW3_main.c    commands.txt           README.md
Child 7450, exited with 0
Prompt> echo "Hello World"
"Hello World"
Child 7451, exited with 0
Prompt> ls -l -a
total 80
drwxrwxr-x  4 student student 4096 Feb 29 20:34 .
drwxr-x--- 20 student student 4096 Feb 27 22:01 ..
-rwxrwxr-x  1 student student 21176 Feb 29 20:34 Chen_PoHan_HW3_main
-rw-rw-r--  1 student student 7241 Feb 29 20:34 Chen_PoHan_HW3_main.c
-rw-rw-r--  1 student student 13264 Feb 29 20:34 Chen_PoHan_HW3_main.o
-rw-rw-r--  1 student student  66 Feb 27 22:01 commands.txt
drwxrwxr-x  8 student student 4096 Feb 27 22:01 .git
-rw-rw-r--  1 student student 1863 Feb 27 22:08 Makefile
-rw-rw-r--  1 student student 7638 Feb 27 22:01 README.md
drwxrwxr-x  2 student student 4096 Feb 27 22:02 .vscode
Child 7452, exited with 0
Prompt> ps
```



```
Operating System vS24 - VMware Workstation
File Edit View VM Tabs Help
Operating System vS24
Activities Terminal Feb 29 20:43
Chen_PoHan_HW3_main.c - csc415-assignment3-simpleshell-eric915c - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
CSC415-ASSIGNMENT3-SIMPLESHELL-ERIC915C
.vscode
Chen_PoHan_HW3_main.c
75 //Executes a single command by replacing the cur
student@student: ~/csc415-assignment3-simpleshell-eric915c
-rw-rw-r-- 1 student student 66 Feb 27 22:01 commands.txt
drwxrwxr-x 8 student student 4096 Feb 27 22:01 .git
-rw-rw-r-- 1 student student 1863 Feb 27 22:08 Makefile
-rw-rw-r-- 1 student student 7638 Feb 27 22:01 README.md
drwxrwxr-x 2 student student 4096 Feb 27 22:02 .vscode
Child 7452, exited with 0
Prompt> ps
  PID TTY          TIME CMD
 7434 pts/3        00:00:00 bash
 7447 pts/3        00:00:00 make
 7448 pts/3        00:00:00 sh
 7449 pts/3        00:00:00 Chen_PoHan_HW3_
 7453 pts/3        00:00:00 ps
Child 7453, exited with 0
Prompt> cat Makefile | wc -l -w
 64 304 1863 Makefile
Child 7456, exited with 0
Prompt> ls foo
ls: cannot access 'foo': No such file or directory
Child 7457, exited with 2
Prompt>
Child 7454, exited with 0
Prompt>
student@student:~/csc415-assignment3-simpleshell-eric915c$
```