MiniShell

Authored by Laith Al-mouhtaseb

211353297

==Description==

The program start in infinite loop that stopped when the user enter “done” to the input, every loop the program print the directory where the program found,

Then the program get a string input from the user, check if the string start with “cd” or it’s only “done”,if it’s “cd” the program print “the command not supported yet”, if it’s “done” the90 program stopped and print how many words in all commands that entered until done and how many commands, and every other input the program create a new process using fork() method and run the command using execvp() method if execvp() can’t run the command the program print error “the command not supported yet” in addition to the system error that returned by execvp() method.

after that save the input string in a text file I named "file.txt",

and if the user entered "history",the program print all the strings that the user have entered it.

functions:

splitToArray: the function received 2 args, the input string and 2d array of chars, it’s split the input string every word in it entered in cell in the array that received.

How It works: run at the input string save update 2 int every time one for the start index of the first character in the word and one for the last index in the word, and if end int is not zero so we arrived to the index of the last char in the word, so we allocate memory for the cells of the array with length of the word and + 1 for the ‘\0’ (end - start +1), if there’s wrong with allocate the program free what it allocate and exit from the program, and the last cell of the array is NULL.

Loop: void function that run all the program, infinity loop.

history: its void function with no input args, that open the txt file "file.txt" and print the file to the console.

count: the parameters that require it is string, pointer of int and another pointer of int, and return a string,

the function edit the value of the two ints that were given to it, it takes ints by reference.

one int to count how many words in the string,

and the second one to count how many letters in the string,

the string that given to the function is the string that the user give in the main.

how it works: define a new string with length of 8 because the longest word that will return is "history" and the '\0' at the end of it.

it runs throw the string and if the char at index i isn't space " " so it's a legal char so the letter counter++.

if the char at index i is space " " and there's 1 or more legal chars so the word count++ and at the end if the char counter is larger than 1.

add 1 for the word counter, because the word count actually count the splits between words.

at the end the function check if the first word equal "exit" or "history" and return what is if its else return ""

and in main they check if the returned value of count function equal "exit" or "history" and the wordcount equal to 1 so the main do what the word have to do.

and if it equals to "" so that a normal input, and we have to add it to history file.

Ex and exHistory: it received 4 parameters, int how many words in the string , the string and two pointer of int one for the command number and one for the word of the command, first it split the string to the array with size of wordCount +1, +1 for the NULL, it make a new process using fork() then execute the input string using execvp() function, after execute the command the father free the array.

countLine: function that run throw the file and count how many rows in the history file.

writeToFile: its receive a string, open the “file.txt” and add the string to the file.

readFromFile: its receive int that contain which line the Function should execute. Using exHistory Function.

==Program Files==

ex2a.c ex2b.c

==How to compile==

compile: gcc ex2a.c -o ex2a

run: ./ex2a

compile: gcc ex2b.c -o ex2b

run: ./ex2b

==input==

string from user

==output==

file.txt file

The executed command

Total command

Total word in all commands