COMP304 Project#1

Ata Sayın 64437, Sinan Keser 76982

03 April 2022

1 Introduction

The project simulates an Unix-style operating system shell. Basic commands of an unix operating system shell and custom commands of the followings are implemented. git repository of the project can be found in here. The filesearch, take and factors commands are done by Sinan Keser and the cdh, joker, automata commands are done by Ata Sayın. External files such as shellfyre.txt or folders are to test if the functions work.

1.1 Basic Commands

execv command complies if the argument command name isn't one of the custom commands.

1.2 Custom Commands

1.2.1 filesearch

filesearch command can take different inputs such as having -r or -o and a file name that is the user wants to search or it can just have an input which is again the file name the user wants to search.The function checks every file and folder in the directory if they have the input name as a substring and if they do, then it prints the files' name. If it has -r as an input, it looks also all the subfolders for the file. If it has -o as an input it opens the files according to their extention by passing xdg-open[17][16] and the file's name to execvp[6][7]. If it has no other inputs other than the file's name, then it searches the file only in that directory.

1.2.2 cdh

cdh commands allows user to change directory whose path are saved in a history file. Firstly, a history file must be created to read and write the history for all computers. getenv with "$HOME$" arguments gets the home path from the current computer, and a history file is created to that directory to be read from all directories[8]. History should be updated in any command therefore a save_history helper function read from the history file and sets the value in to

global string array. Then the current path is add to to history variable while old values are shifted up[5], therefore oldest path will be diminished if the history file is full. Old strings are written without any updates, however for newest path is added no to miss last string while reading the file.

When the user calls cdh command, the child process prints out the history variable and ask directory from the user. Since its done in the child process when the cdhir is called the directory of the parent won't be change, consequently the input should be send to parent process through child process. Shared memory is used for the communication which sends the index of the history file that will be the desired directory[14].

Argument $-r$ deletes the history file with remote function [13]

1.2.3 take

take command takes an input in form of A/B/C. What the function does is it creates folders according to the input in the direction it is in. To create a new folder with appropriate permissions[12] it uses mkdir[9] and to go into the newly made folder it uses chdir[2] For example, if we are in a folder named shellfyre, typing take A/B/C creates a folder named A inside the shellfyre folder and goes into folder A and creates another folder and creates a new folder named B and does the same for the C. In the end we are now located in ./shellfyre/A/B/C .

1.2.4 joker

joker command takes a "a dad joke" from `https://icanhazdadjoke.com/` in every 15 minutes and prints out it to display. To schedule a command, crontab command is used[3][4]. The desired command must be added to the crontab file which is received with -l argument. The desired command echo'ed to a cat'ed crontab -l file, lastly crontab -l command writes the printed old crontab file and new scheduled command into new crontab file, therefore the desired command is added. To add the command in a correct way, firstly the curl function should be called when the crontab calls it, otherwise a same joke would be printed out in every 15 minutes. Moreover system calls mix up the quotation marks, therefore one line inner quotation mark is used for curl command which is called with $ symbol[11][10][15].

1.2.5 factors

factors command take an integer and outputs its factors. The function uses a loop which starts from 2 and in that loop it checks if their modulo is 0. If it is 0 then it divides the given number and doesn't change the current number the loop is in. If the modulo is not 0, then it increments the current number by 1. Since it starts from 2 and increases it by 1 each until it reaches the given number, the function works without any error. Not to confuse with the outputs of factor command inside the UNIX, it is renamed as factors.

1.2.6 automata

automata command simulates the text scenes from a game called Nier Automata. The scenes goes through a text file while printing slowly char by char. Example(03:30:51 time). The texts are gotten from [1] and took only M*.txt files. The text files are renamed as "automata_ < *number* > *.txt*". set_random_automata gets a random a file from the automata directory and reads to an automata string array. For printing animation, 2 nested for loops are used, first one goes through lines of the text and second one prints the a temp string whose terminal char is modified by incremented one by one, therefore the user sees as an animation. For each printing, a 10 new line char used for seeking the oldest prints. Texts have characteristic features, for example <bt_waits>refers to a waiting process in the animation. If its in the end of the line, its not printed. In addition sleep function is added to give the animation. <page><\page>refers to new page, therefore extra new line char are added. They also won't be printed.

The command could be improved while creating another process for an audio or the features <bt_waits>that is present in the middle of line <i>can be modified too.

1.3 Kernel Modules

References

[1] AutomataNieR. URL: `https://github.com/AutomataNieR/NieR-Automata-Text-Files`.

[2] *Change Dir*. URL: `https://stackoverflow.com/questions/1293660/is-there-any-way-to-change-directory-using-c-language`.

[3] *Crontab*. URL: `https://tecadmin.net/crontab-in-linux-with-20-examples-of-cron-schedule/`.

[4] *Crontabguru*. URL: `https://crontab.guru/`.

[5] *currentDir*. URL: `https://stackoverflow.com/questions/298510/how-to-get-the-current-directory-in-a-c-program`.

[6] *execv*. URL: `https://linux.die.net/man/3/execv`.

[7] *execvp*. URL: `https://stackoverflow.com/questions/27541910/how-to-use-execvp`.

[8] *getEnvExample*. URL: `https://www.tutorialspoint.com/c_standard_library/c_function_getenv.htm`.

[9] *Make Dir*. URL: `https://stackoverflow.com/questions/53600297/how-to-create-folders-using-mkdir-in-c`.

[10] *Medium Nested commands*. URL: `https://levelup.gitconnected.com/a-guide-to-bash-commands-redirecting-chaining-and-nesting-f0c752c62eca`.

[11] *Nested commands*. URL: https://stackoverflow.com/questions/3746079/can-i-execute-nested-or-chained-commands-in-unix-shell.

[12] *Permissions*. URL: https://www.guru99.com/file-permissions.html.

[13] *Remote*. URL: https://www.geeksforgeeks.org/c-program-delete-file/.

[14] Galvin Silberschatz and Gagne. *Operating System Concepts - Ninth Edition*. 2013.

[15] *Variable Assignment*. URL: https://tldp.org/LDP/abs/html/varassignment.html.

[16] *Xdg-Open Geeks*. URL: https://www.geeksforgeeks.org/xdg-open-command-in-linux-with-examples/.

[17] *Xdg-Open Unix*. URL: https://unix.stackexchange.com/questions/340531/launch-executable-with-xdg-open.