KOÇ UNIVERSITY
SPRING 2021

COMP304
PROJECT 1
REPORT

STUDENTS:

Sedat Çoban

Burcu Özer

INSTRUCTOR:

Didem Unat

TA:

Najeeb Ahmad

SUBMISSION DATE:

04/04/2021

Part 1)

The aim of this part was to use execv system call by searching the path for the command and execute it. For getting the path of the command, we executed system function[1] in our implementation. System function executes the following command written into it:

which command->name > pathToCommand

By this execution, the path for the command is written into the pathToCommand file. Next, we read that file and get the first line which is our path. Later, inside the path variable, we remove all of the white spaces and replace new line ("\n") with 0 which is null. Then, we executed execv with the path and arguments of the command.

Part 2)

The aim of this part was to create name directory associations. Separate arrays are used to store name and directories (keys and values arrays respectively). Shortdir_count variable was used to store the number of associations. Keys, values and short_dir variables were stored in a struct named association. We created a binary file (shortdir) in the home directory[7] to store the struct. Thus, the associations are saved across shell sessions. "shortdir" command with "set" argument was used to create new name directory associations by adding them to the keys and values arrays. "jump" argument was used to change to the directory of the name given by searching the name within the keys array and finding the corresponding directory value from the values array. "del" argument was used to delete the specified name and directory from the keys and values arrays. "list" argument gave all the name-directory associations found in the keys and values arrays. "clear" argument was used to delete all the name-directory associations found in the keys and values arrays. In each command call, these keys and values are updated according to the argument and saved into the struct which is then saved into the binary file (shortdir)[2]. All of the example commands with shortdir are shown in below:

```
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ shortdir set down
down alias already used
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ shortdir list
name: down directory: /home/sedat/Downloads
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ cd ../
sedat@sedat-VirtualBox:/home/sedat seashell$ shortdir set sed
sed is set as an alias for /home/sedat
sedat@sedat-VirtualBox:/home/sedat seashell$ shortdir list
name: down directory: /home/sedat/Downloads
name: sed directory: /home/sedat
sedat@sedat-VirtualBox:/home/sedat seashell$ shortdir jump abc
sedat@sedat-VirtualBox:/home/sedat seashell$ shortdir set sb
sb is set as an alias for /home/sedat
sedat@sedat-VirtualBox:/home/sedat seashell$ shortdir list
name: down directory: /home/sedat/Downloads
name: sb directory: /home/sedat
sedat@sedat-VirtualBox:/home/sedat seashell$ shortdir jump down
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ shortdir del sb
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ shortdir list
name: down directory: /home/sedat/Downloads
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ shortdir clear
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ shortdir list
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ 
```

Figure 1: Shortdir Command

Part 3)

The aim of this part was to highlight the given word within the file and print the content of the file accordingly. For this purpose, we first open up the file and started to read the file line by line. For each line, we try to find the occurrence of the word within the line by comparing character by character. We stored the occurrence index of the word within the line in the occurrence array. Next, we checked whether the location of the string within the line is considered as separate word or not (not part of another word). After finding the occurrences of the word within the line, we travelled through the line again and we used ANSI-color codes[3] to print the words in specified color type. The example of highlight command can be seen in below:

Figure 2: Highlight Command

Part 4)

The aim of this part was to set an alarm with specified time and music. For this purpose, we create shell script which plays the song[4], which is found in the given path, when executed. The content of the shell script is given below:

DISPLAY=:0 rhythmbox-client --play /home/sedat/Downloads/backToBlack.mp3
(DISPLAY=:0 was taken from the TA)

Later, we create another file (crontab file) which is composed of the time and the path for the .sh file.[5] The content of the file is given below:

50 15 * * * /home/sedat/Downloads/goodMorning.sh

Next, we used execvp to execute the crontab command with the constructed crontab file.

Part 5)

The aim of this part was to compare two files line by line and print different lines and the total number of line difference or byte by byte and print the total number of byte difference. For this purpose, we checked the comparison type first. If the comparison type is line by line (-a or nothing is given for the comparison type), we opened two files and counted their line numbers by travelling through the files. Next, we compared two files line by line by getting the same row at the same iteration. This line comparison was performed through char comparison of the lines. If there are at least 1 char difference between the lines, the lines were printed otherwise nothing was printed. At the end, total number of lines was printed. If the

comparison type is byte by byte (-b), we opened two files and counted their line numbers. Next, we compared two files byte by byte by travelling the files. If we see any difference, we increased the total number of byte difference by 1. At the end, byte difference between the files were printed. The example of kdiff command can be seen in below:

```
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ cat file1
gdec
cccc
can we do it?
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ cat file2
gdec
cccc
can we do it?
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ cat diffFile
gdec
cccc
No, we could not!!
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ kdiff -a file1 file2
Files are identical
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ kdiff -a file1 diffFile
file1:Line 3:can we do it?
diffFile:Line 3:No, we could not!!
1 different lines are found
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ kdiff -b file1 file2
Files are identical
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ kdiff -b file1 diffFile
The two files are different in 14 bytes
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ kdiff file1 file2
Files are identical
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$ kdiff file1 diffFile
file1:Line 3:can we do it?
diffFile:Line 3:No, we could not!!
1 different lines are found
sedat@sedat-VirtualBox:/home/sedat/Downloads seashell$
```

Figure 3: kdiff Command

Part 6)

We created a new command called "hist". The aim of the command was to show the executed command history of the seashell with date and username information. In each execution of a command, we stored the current user, current time, command name and command arguments in a string. For getting the current time, we used time.h library and localtime function. We also used strftime function to store the date, day of the week and time information.[6] For getting the user environment variable, we used getenv function.[7] We appended this string into a history.txt file which was created in the home directory. We have 4 different arguments for hist command which are "all", "user", "date" and "clear". For "all" argument, we printed all of the history.txt file. The example execution of "hist all" command is given below:

Figure 4: Hist All Command

For "user" argument, we took another argument, which is user name, and filtered the history.txt with the given user argument and printed that user's history. The example execution of "hist user" command is given below:

Figure 5: Hist User Command

For "date", we took another argument which is the date (day/month/year) and we filtered the history.txt according to the date and printed the history of that day. The example execution of "hist date" command is given below:

Figure 6: Hist Date Command

For "clear" argument, we cleared the history.txt file. The example execution of "hist clear" command is given below:

Figure 7: Hist Clear Command

REFERENCES

1) System() Function in C/C++. (2021). Retrieved 4 April 2021, from https://www.tutorialspoint.com/system-function-in-c-cplusplus#:~:text=The%20system()%20function%20is,included%20to%20call%20this%20function.

2) The fread() fwrite() function in C. (2021). Retrieved 4 April 2021, from http://www.tutorialdost.com/C-Programming-Tutorial/C-file-io-fread-fwrite-function.aspx

3) Git shell coloring. (2021). Retrieved 4 April 2021, from https://gist.github.com/vratiu/9780109

4) (2021). Retrieved 4 April 2021, from http://manpages.ubuntu.com/manpages/hardy/man1/rhythmbox-client.1.html

5) Hope, C. (2021). Linux crontab command help and examples. Retrieved 4 April 2021, from http://www.computerhope.com/unix/ucrontab.htm

6) Bodnar, J. (2021). C Date time tutorial. Retrieved 4 April 2021, from https://zetcode.com/articles/cdatetime/

7) C library function - getenv() - Tutorialspoint. (2021). Retrieved 4 April 2021, from https://www.tutorialspoint.com/c_standard_library/c_function_getenv.htm