COMP304 – Operating Systems – Project 1 Report – Seashell
Kaan Türkmen – Eren Yenigül

## 1. Using execv instead of execvp command.

In this part, it is asked to use the execv command instead of execvp while executing user commands. Main difference between these commands is that execvp automatically detects the path of given commands while execv is taking a path location and other parameters as an argument.

Firstly, we are getting environment strings from the getenv method, then parsing from the ";" character. Then we are searching our command application in these folders. Before we start searching, we are sure it is not located in the current directory and its not executable file.

## 2. Creating a new command: shortdir.

In this part, it is asked to implement the new command `shortdir". This command should do the following operations:

- *shortdir set name*: associates *name* with the current directory. Overwrites an existing association.
- *shortdir jump name*: changes to the directory associated with *name*.
- *shortdir del name*: deletes the name-directory association.
- *shortdir clear*: deletes all the name-directory associations.
- *shortdir list*: lists all the name-directory associations.

When a shortdir set command is executed by the user a file named `.shortdir` is created storing alias names and corresponding directories line by line. Whenever there is an update (set/del), a new file is created named `.temp_shortdir`, and `.shortdir' is getting copied to this file with specified updates done on each line. Then this temporary file is renamed to `.shortdir`. When the user enters an alias name which is in use, the old alias gets deleted. shortname clear command deletes `.shortdir` file, shortname jump reads through `.shortdir` line by line, tries to find the alias user specified and shortname list again read through the file and prints the content.

```
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ shortdir set awesomedir
shortdir: awesomedir alias is set for  /home/kaan/Developer/seashell
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ cd ..
kaan@katurkmen:/home/kaan/Developer seashell$ shortdir set someotherdir
shortdir: someotherdir alias is set for  /home/kaan/Developer
kaan@katurkmen:/home/kaan/Developer seashell$ cd ..
kaan@katurkmen:/home/kaan seashell$ shortdir set awesomedir
shortdir: This alias was already in use (/home/kaan/Developer/seashell) and now it is overwritten!
shortdir: awesomedir alias is set for  /home/kaan
kaan@katurkmen:/home/kaan seashell$ shortdir list
Shortdir name        | Directory
someotherdir           /home/kaan/Developer
awesomedir             /home/kaan
kaan@katurkmen:/home/kaan seashell$ shortdir jump someotherdir
kaan@katurkmen:/home/kaan/Developer seashell$ shortdir del someotherdil
shortdir: someotherdil alias does not exist.
kaan@katurkmen:/home/kaan/Developer seashell$ shortdir del someotherdir
shortdir: someotherdir alias is deleted.
kaan@katurkmen:/home/kaan/Developer seashell$ shortdir list
Shortdir name        | Directory
awesomedir             /home/kaan
kaan@katurkmen:/home/kaan/Developer seashell$ shortdir del awesomedir
shortdir: awesomedir alias is deleted.
```

### 3. Creating a new command: highlight.

In this part, it is asked to implement a command that prints the content of a given file with a specified word being highlighted. The command has three color options for this highlighting operation.

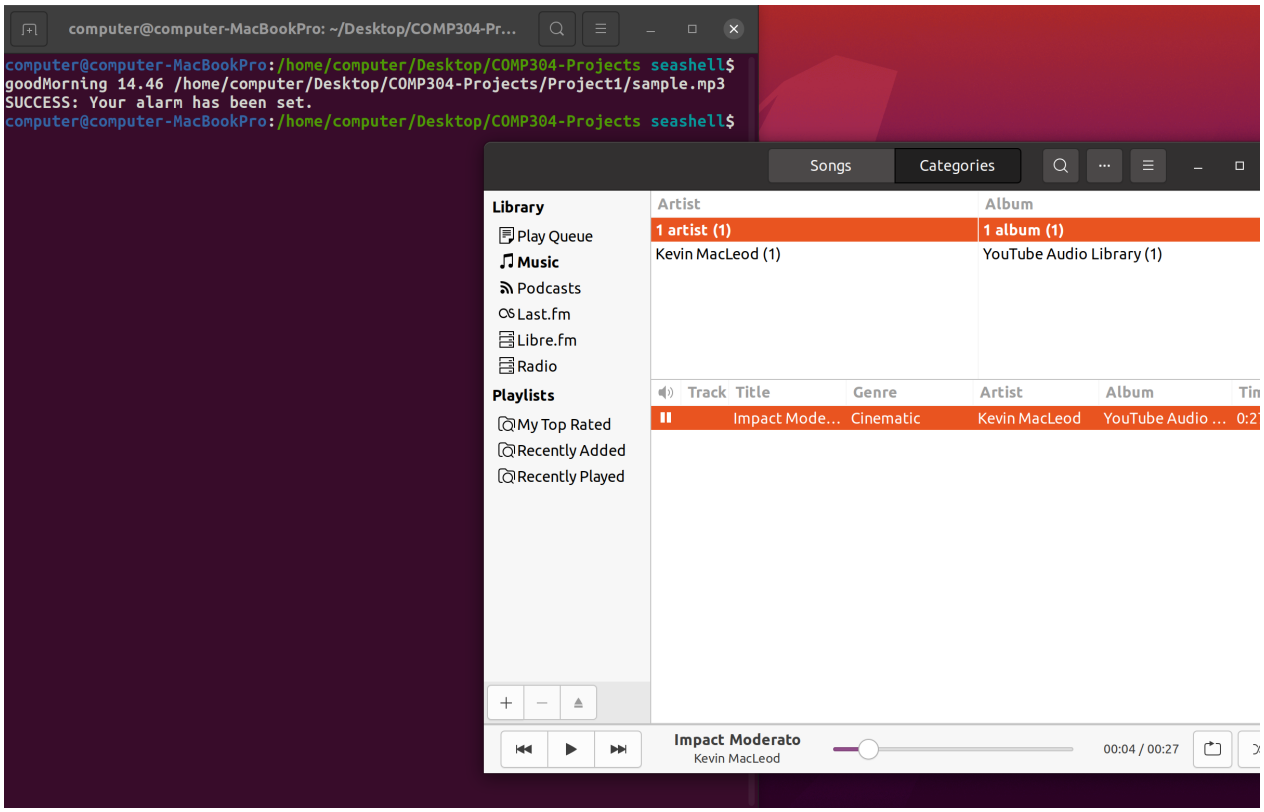
In our implementation, the program reads through the given file line by line, stores the current line in a buffer, then splits the words in the buffer by using strtok function and compares each token with the specified word. If the token matches with the word, it is printed with color codes surrounding it. If not, it is just printed normally. A word is printed at a time.

```
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ highlight was r file1.txt
Jeff Dean was born on December 31, 1969 at 11:42 PM. It took him twelve minutes to implement his first time counter.
Jeff Dean was forced to invent asynchronous APIs one day when he optimized a function so that it returned before it was invoked.
Jeff Dean wrote an O(N2) algorithm once. It was for the Traveling Salesman Problem.
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ highlight dean g file1.txt
Compilers don't warn Jeff Dean. Jeff Dean warns compilers.
The rate at which Jeff Dean produces code jumped by a factor of 40 in late 2000 when he upgraded his keyboard to USB 2.0.
Jeff Dean builds his code before committing it, but only to check for compiler and linker bugs.
When Jeff Dean has an ergonomic evaluation, it is for the protection of his keyboard.
gcc -O4 emails your code to Jeff Dean for a rewrite.
Jeff Dean once failed a Turing test when he correctly identified the 203rd Fibonacci number in less than a second.
The speed of light in a vacuum used to be about 35 mph. Then Jeff Dean spent a weekend optimizing physics.
Jeff Dean was born on December 31, 1969 at 11:42 PM. It took him twelve minutes to implement his first time counter.
Jeff Dean eschews both Emacs and VI. He types his code into zcat, because it's faster that way.
Unsatisfied with constant time, Jeff Dean created the world's first O(1/N) algorithm.
When Jeff Dean goes on vacation, production services across Google mysteriously stop working within a few days.
Jeff Dean was forced to invent asynchronous APIs one day when he optimized a function so that it returned before it was invoked.
When Jeff Dean designs software, he first codes the binary and then writes the source as documentation.
Jeff Dean wrote an O(N2) algorithm once. It was for the Traveling Salesman Problem.
Jeff Dean can beat you at connect four. In three moves.
When Jeff Dean fires up the profiler, loops unroll themselves in fear.
Jeff Dean is still waiting for mathematicians to discover the joke he hid in the digits of PI.
When Jeff Dean listens to mp3s, he just cats them to /dev/dsp and does the decoding in his head.
Google search went down for a few hours in 2002, and Jeff Dean started handling queries by hand. Search Quality doubled.
Jeff Dean puts his pants on one leg at a time, but if he had more legs, you would see that his approach is O(log(N)).
Knuth mailed a copy of TAOCP to Google. Jeff Dean autographed it and mailed it back.
Jeff Dean once shifted a bit so hard, it ended up on another computer.
Jeff Dean can losslessly compress random data.
Jeff Dean mines bitcoins. In his head.
Jeff Dean knows the last digit of Pi.
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ highlight so b file1.txt
Jeff Dean was forced to invent asynchronous APIs one day when he optimized a function so that it returned before it was invoked.
Jeff Dean once shifted a bit so hard, it ended up on another computer.
kaan@katurkmen:/home/kaan/Developer/seashell seashell$
```

**4. Creating a new command: goodMorning.**

In this part, it is asked to implement the new command `goodMorning". This command should play music in the specific time selected by the user. It can be seen as an alarm clock application which is being used from the terminal.

To implement goodMorning, we are using the crontab (Linux Scheduler) and Rhytmbox-Client (Linux Music Player). First of all, we are making sure if the inputs are correct by creating regex. Our implementation is in a format that is XX:XX, so that 9.15 is not a valid input, it should be written as 09.15 to function properly. Furthermore, the path should be an absolute path. After the user enters a valid input, we are creating a txt file which contains related information for the cronjob. Since Linux Scheduler is always repeating a given process in the given time interval, we are clearing the crontab after 1 minute of execution.



**5. Creating a new command: kdiff.**

In this part, it is asked to implement the new command `kdiff". This command should have two different modes, one of which compares bit files while other one is comparing the .txt files. In the first mode, -a mode, which is also selected as default mode, program only allows *.txt files and compares content line by line. In the second mode, -b mode, it compares files byte by byte. Please keep in mind that, if one of the files reaches EOF, we are not adding any more differences. Our code can be thought of as "diff" and "cmp" commands exist in linux.

To implement kdiff, we are validating user input, after we are done with validation, we are opening each file separately, and comparing line by line or byte by byte. Since char is a 1 byte, we are getting help from the char array to store each byte.

```
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ kdiff file1.txt file2.txt

Difference spotted: Line 8: File1.txt Jeff Dean was born on December 31, 1969 at 11:42 PM. It took him twelve minutes to implement his first time counter.
Difference spotted: Line 8: File2.txt Jeff Dean was born on December 31, 1969 at 11:48 PM. It took him twelve minutes to implement his first time counter.


Difference spotted: Line 18: File1.txt Jeff Dean's keyboard has two keys: 1 and 2.
Difference spotted: Line 18: File2.txt Jeff Dean's keyboard has two keys: 1 and 0.


Difference spotted: Line 23: File1.txt Jeff starts his programming sessions with cat > /dev/null.
Difference spotted: Line 23: File2.txt Jeff starts his programming sessions with cat > /dev/mem.

Total different line count is 3
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ kdiff -a file1 file2
-seashell: kdiff: Please use valid paths or flags. (Use .txt extension only for the non-binary mode.)
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ kdiff -b binary1 binary2
Total byte difference between two file is 37
kaan@katurkmen:/home/kaan/Developer/seashell seashell$
```

**6. Creating a new command on your own: cstock**

In this part, we have designed a new command called `cstock'. This command shows a graph or table view of famous crypto currencies. Also, users can specify a time interval to graph only that time period.

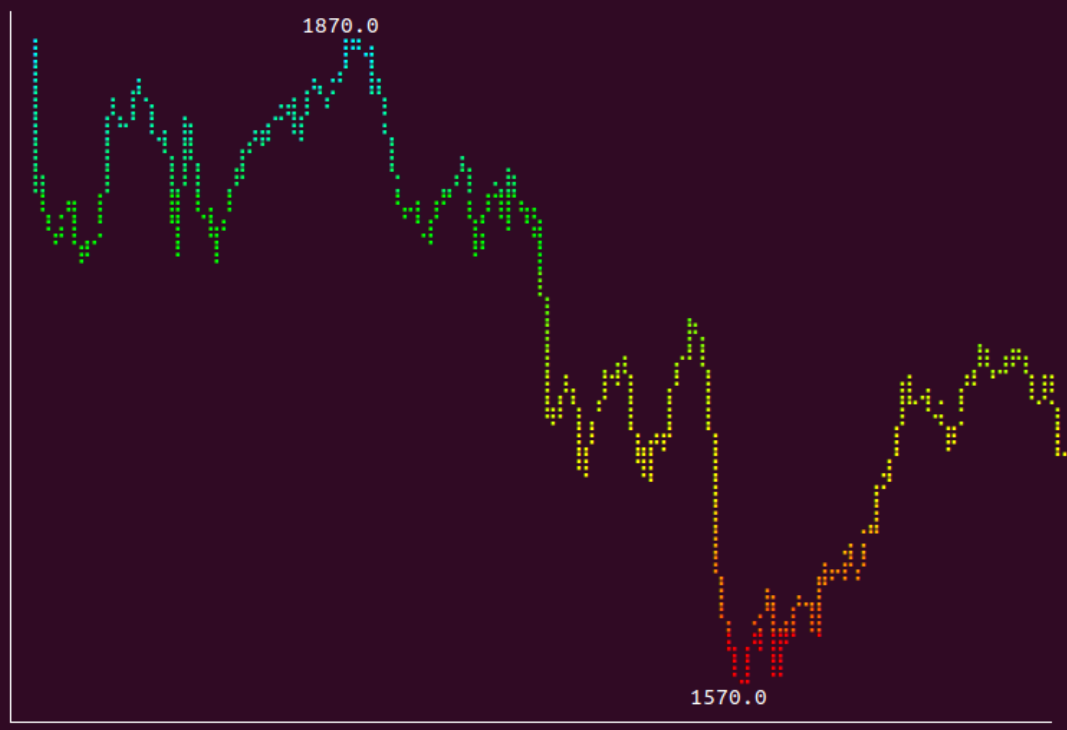
To implement cstock, we are web scraping with the "curl" command. We are first validating the input, then creating a final string with the user's selected mode / time period. Then we are accessing the site with the final url. By using the site, we are printing the graph on the screen. Type "cstock --help" to get valid usages of the command.

```
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ cstock --help

cstock: Graph Mode -> cstock [Crypto Currency Name] [Day (Optional) Range: [1-90]]
cstock: Example: cstock eth 4
cstock: to view last 4 day activity of ethereum.


cstock: Table Mode -> stock -a
cstock: to view available currencies' table.

kaan@katurkmen:/home/kaan/Developer/seashell seashell$ cstock -a
                                                                  _
                                                       _ _ __ _| |_ ___  ____ __
                                                      | '_/ _` |  _/ -_)(_-< \ /
__.................................................._________|_| \__,_|\__\___()__/_\_\____
 '          '          '          '          '

Market Cap: $1,742,835,951,895 =
24h Vol: $108,144,184,185 =
BTC Dominance: 59.4% =
```

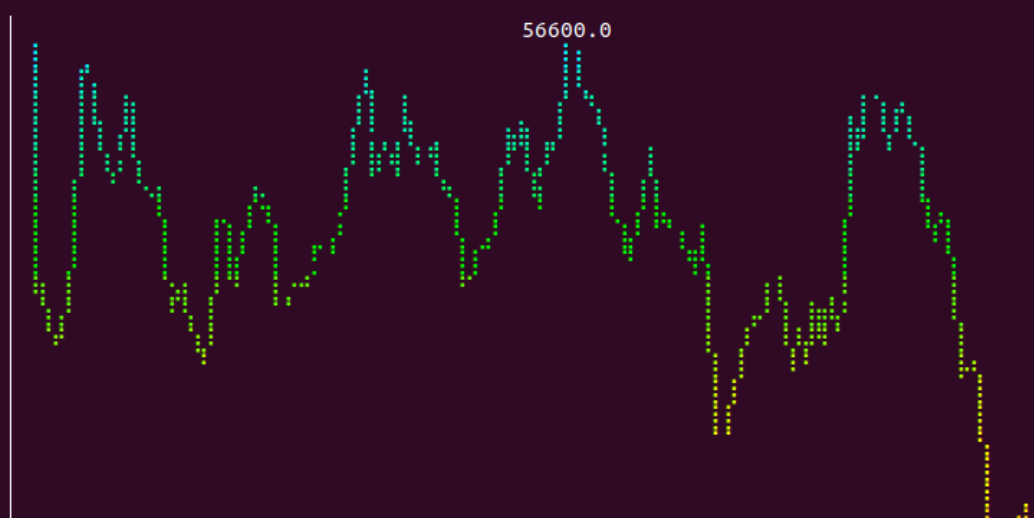
| Rank | Coin | Price (USD) | Change (24H) | Change (1H) | Market Cap (USD) | Spark (1H) |
|---|---|---|---|---|---|---|
| 1 | BTC | 55283.1 | -1.17% | 0.12% | 1.032T | |
| 2 | ETH | 1678.32 | -1.95% | 0.14% | 193.410B | |
| 3 | BNB | 265.735 | -0.21% | 0.20% | 41.065B | |
| 4 | USDT | 1.00095 | -0.01% | 0.02% | 40.514B | |
| 5 | ADA | 1.17743 | -1.08% | -0.08% | 37.617B | |
| 6 | DOT | 31.6517 | -3.03% | -0.10% | 29.248B | |
| 7 | XRP | 0.546777 | -1.66% | -0.16% | 24.826B | |
| 8 | UNI | 27.6112 | -1.82% | -0.31% | 14.413B | |
| 9 | THETA | 12.7997 | 6.95% | 0.74% | 12.800B | |
| 10 | LTC | 182.411 | -0.72% | -0.04% | 12.176B | |

```
2021-03-28 20:40:21.142523 UTC

NEW FEATURE: to see cryptocurrency exchange rate, do curl rate.sx/eth (or any other coin instead of ETH)
[Follow @igor_chubin for rate.sx updates] [github.com/chubin/rate.sx]
kaan@katurkmen:/home/kaan/Developer/seashell seashell$
```

```
kaan@katurkmen:/home/kaan/Developer/seashell seashell$ cstock eth 12

▶ Ethereum (ETH) ▶ 16 Mar +1w5d -6.67%
```




```
begin: $1798.2 (16 Mar 21:20) // end: $1678.3 (28 Mar 20:40)
high: $1868.8 (20 Mar 13:20) // low: $1565.8 (25 Mar 02:10)
avg: $1738.4 // median: $1717.3 // change: -119.92 (-6.67%)
See rate.sx/:help for help and disclaimer

kaan@katurkmen:/home/kaan/Developer/seashell seashell$ cstock btc

▶ Bitcoin (BTC) ▶ Sat 27 +1d -1.29%
```



**References**


https://stackoverflow.com/questions/27303062/strstr-function-like-that-ignores-upper-or-lower-case

https://stackoverflow.com/questions/60484638/crontab-file-is-not-created-when-called

https://stackoverflow.com/questions/5947742/how-to-change-the-output-color-of-echo-in-linux

https://stackoverflow.com/questions/21248840/example-of-waitpid-in-use

https://linux.die.net/man/1/rhythmbox-client

https://www.geeksforgeeks.org/crontab-in-linux-with-examples/

https://crontab.guru/

https://man7.org/linux/man-pages/man5/crontab.5.html

https://man7.org/linux/man-pages/man1/crontab.1.html

https://github.com/torvalds/linux

http://www.informatika.bg/jeffdean

https://www.geeksforgeeks.org/regular-expressions-in-c/

https://www.educative.io/edpresso/how-to-write-regular-expressions-in-c

https://stackoverflow.com/questions/1085083/regular-expressions-in-c-examples

https://stackoverflow.com/questions/1631450/how-to-use-regular-expressions-in-c

https://stackoverflow.com/questions/41740157/unable-to-match-regex-in-c

https://www.gnu.org/software/libc/manual/html_node/Regular-Expressions.html

https://www.tutorialspoint.com/c_standard_library/c_function_fgetc.htm