Authored by: Hakam Nabulssi

***Simple Shell***

# ==Description==

A **shell** is a program that provides the traditional, text-only user interface for Linux and other Unix-like operating system, And it’s an interface between the user and the internal parts of the operating system (at the very core of which is the kernel), and in this exercise we're aiming to make a Simple shell that execute the basic commands such as (man, echo, ls, cat, more, expr, sleep), and to invoke pipes.

A **pipe** is a form of redirection that used in Linux and other Unix-like operating systems to send the output of one command/program/process to another command/program/process for further processing. The Unix/Linux systems allow stdout of a command to be connected to stdin of another command. We can make it do so by using the pipe character **‘|’**.

# ==Functions==

This program has only nine functions and the main

|  |  |  |
| --- | --- | --- |
| Name of the function | type | description |
| 1. prompt | void | printing the prompt of the commands |
| 2. read\_line | void | getting the commands from the user, dividing them, counts the number of words |
| 3. remove\_end\_of\_line | void | removing the \n form the end of the string had entered by the user (by using fgets function) |
| 4. parse\_line | int | splitting the line entered by the user in the wanted manner |
| 5. commands\_in\_line | void | counts the number of commands in one line |
| 6. spacing | void | correcting the spaces for using pipe (make a space before and after the pipe) for making it works in all the situations |
| 7. replace\_char | void | replacing the quotations marks with a space for removing  them |
| 8. moving\_to\_arrays | void | moving the commands to an arrays as a first command, middle command, and last command for piping |
| 9. read\_parse\_line | int | calling all the methods in an ordered manner |

# ==Program Files==

EX3.c - contains only the functions and the main

**==compiling Steps==** compile: gcc EX3.c -o EX3 run: ./EX3

# ==Input:==

1. ls
2. ls -l
3. ls -l | wc -l
4. \n (empty command “clicking enter”)
5. cd
6. ls -l | sort | wc
7. echo “hi os” | wc -c
8. ls -l|wc -l
9. aaaaa | sort
10. done

# ==Output:==

1. ATester CMakeFiles hakamisgay.txt Makefile text.txt ATester.cbp cmake\_install.cmake hanna test x CMakeCache.txt file hanna.txt Testing x.txt
2. total 96

-rwxrwxr-x 1 student student 29912 Apr 30 16:54 ATester

-rw-rw-r-- 1 student student 5158 Apr 24 16:24 ATester.cbp

-rw-rw-r-- 1 student student 23117 Apr 24 16:24 CMakeCache.txt

drwxrwxr-x 6 student student 4096 Apr 30 17:01 CMakeFiles

-rw-rw-r-- 1 student student 1663 Apr 14 18:40 cmake\_install.cmake

-rwxrwxr-x 1 student student 0 Apr 6 05:38 file

-rw-rw-r-- 1 student student 0 Apr 26 14:25 hakamisgay.txt

-rw-rw-r-- 1 student student 0 Apr 25 20:47 hanna

-rw-rw-r-- 1 student student 13 Apr 13 17:00 hanna.txt

-rw-rw-r-- 1 student student 5321 Apr 24 16:24 Makefile

-rwxrwxr-x 1 student student 0 Apr 6 21:43 test

drwxrwxr-x 3 student student 4096 Apr 6 04:55 Testing

-rwxrwxr-x 1 student student 0 Apr 6 21:45 text.txt

-rwxrwxr-x 1 student student 110 Apr 6 21:49 x

-rwxrwxr-x 1 student student 8 Apr 6 21:48 x.txt

1. 16
2. NO COMMAND
3. Command Is Not Supported (Yet)!
4. 16 137 870
5. 6
6. 16
7. piping has failed to execute
8. Number of commands 9 Number of Pipes 6

See you Next time !