***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 (***regular or double***) ,this shell also receives two types signals from the user the pausing signal (***SIGTSTP***) by clicking ***Ctrl-z*** , and the continuing signal (***STGCONT***) by entering ***fg***.

***SIGTSTP***: This signal is an interactive stop signal, it can be handled and ignored, the program should handle this signal if we have a special need to leave files or system tables in a secure state when a process is stopped.

This signal is generated when the user types ***Ctrl-z.***

***STGCONT***: You can send a ***SIGCONT*** signal to a process to make it continue. This signal is special, it always makes the process continue if it is stopped before the signal is delivered. The default behavior is to do nothing else. You cannot block this signal. You can set a handler, but ***SIGCONT*** always makes the process continue regardless.

This signal is generated when the user types ***fg***.

# ==Functions==

**This program has only ten functions and the main**

|  |  |  |
| --- | --- | --- |
| **Name of the function** | **type** | **description** |
| **sig\_handler** | **void** | **Register signal handle.** |
| **prompt** | **void** | **printing the prompt of**  **thecommands.** |
| **read\_line** | **void** | **getting the commands from the user, dividing them, countsthe number of words.** |
| **remove\_end\_of\_line** | **void** | **removing the \n form the endof the string had entered by the user (by using fgets function).** |
| **parse\_line** | **int** | **splitting the line entered by the user in the wanted manner.** |
| **commands\_in\_line** | **void** | **counts the number ofcommands in one line.** |
| **spacing** | **void** | **correcting the spaces for usingpipe (make a space before andafter the pipe) for making it works in all the situations.** |
| **replace\_char** | **void** | **replacing the quotations markswith a space for removing**  **them.** |
| **moving\_to\_arrays** | **void** | **moving the commands to anarrays as a first command, middle command, and last command for piping.** |
| **read\_parse\_line** | **int** | **calling all the methods in anordered manner.** |

# ==Program Files==

EX4.c - contains only the functions and the main

**==Compiling Steps==** compile: EX4.c -o EX4 run:./EX4

# ==Input:==

cat  clicking enter  Ctrl-z **(^z**) entering (**fg**)

# ==Output:==

After pausing the process, running the last order in the last suspended process.