Cairo University Operating system Course

**Faculty of Computers & Information.**

**Third Year**

**Dr. Khalid Wassif**

**2014/2015**

**Assignment #1**

**Command Line Interpreter**

**Purpose**

An operating system interfaces with a user through a Command Line Interpreter

(CLI). A CLI is a software module capable of interpreting textual commands coming either from the **user’s keyboard** or from a **script file**. A CLI is often referred to as a shell.

**Description**

In this assignment, you are to write a Command Line Interpreter (CLI) for your operating system. Your CLI should prompt the user to enter the input through the keyboard. After a sequence of characters is entered followed by a return, the string is parsed and the indicated command (s) executed. The user is then again prompted for another command.

Your program implements some built-in commands**; the list of required commands is listed below.** This means that your program must implement these commands directly by using the system calls that implement them. Do not use **exec** to implement any of these commands. The **exit** command is also a special case: it should simply cause termination of your program.

For this assignment, the following are essential features for your work:

1. Your CLI should be written in **Java** and as a task function (CLI commands may be written as functions or tasks).
2. All commands and parameters should be entered from the keyboard and **parsed** by your program, **verified**, and then **executed**. If the user enters wrong command or bad parameters the program should print some error message. For example, if the user writes **mkdir**, the program should response by an error message as the command **mkdir** should have one parameter.
3. Your program should handle different parameters for each command. For example, if the user writes **cd C:/** then the program should change to directory **C:/** in case of the current directory is **D:/.** On the other hand, if the user writes **cd** only then the program should change to default directory (defined in your program) which may be **D:/**
4. Command parameters are either strings or quoted.
5. You should implement the following commands: **clear, cd, ls, cp, mv, rm, mkdir, rmdir, cat, more, less, pwd.**
6. Others commands should be implemented also:
   1. **args** - list all parameters on the command line, numbers or strings for specific command.
   2. **date** - output current system date and time.
   3. **help** - list all user commands and the syntax of their arguments. For example, if the user write **help** command, the program output should be like the following :

**help**

|  |
| --- |
| args : List all command arguments  date : Current date/time  exit : Stop all |

1. Redirecting should also be implemented (i.e. > and >>) to output the result of command to some file.
2. Commands can be combined using "**;"** operator**.** Your program should implement this function. For example, if the user enters **cd C:/ ; pwd** the program should first change the current directory to **C:/** and then display to the user the content of the current directory which is **C:/.**

**Bonus:**

In addition, implement additional functionality to your CLI from the following list:

* Implement two more commands :
  1. **find** <fileName> <directory>: Search for files with the given name in the given directory and all subdirectories and then display their full paths.
  2. **grep** <text> <fileName>: Search for the given text in the given file and write each matching line to standard output. Ex. grep 'word' lab1.txt will display all the lines in lab1.txt file contain string ‘word’
* Command line recall (using the up and down keyboard arrows).

**Submission instructions:**

**1. Submission date starting Sunday 2 November, each in her/his lab time.**

**2. The assignment is submitted in group of maximum 3 persons or 4 persons but bonus part would be mandatory and the total grade would be the same for group of 3 without doing bonus part as we discussed in lab**.