Cairo University
Faculty of Computers & Information.
Operating Systems 1 Course
Third Year
Dr. Khalid Wassif
2018/2019

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 will 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 maybe written as functions or tasks).
2. Your application should contain 2 major classes (Parser, Terminal).

```
// Interface for parser
public class Parser{
        String[] args;  // Will be filled by arguments extracted by parse method
        String cmd;  // Will be filled by the command extracted by parse method

        // Returns true if it was able to parse user input correctly. Otherwise false
        // Incase of success, it should save the extracted command and arguments
        // at args and cmd paramters
        public boolean parse(String input)
}

// Interface for Termianl
public class Terminal{
        public void cp(String sourcePath, String destinationPath );
        public void mv(String sourcePath, String destinationPath);
        public void rm(String sourcePath);
        public void pwd();
        // Add any other required command in the same structure…..
}
// Main implementation is up to you.
```

3. 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 messages. For example, if the user writes **mkdir**, the program should response by an error message as the command **mkdir** should have one parameter.
4. 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:/**
5. Command parameters are either strings or quoted.
6. You should implement the following commands: **clear, cd, ls, cp, mv, rm, mkdir, rmdir, cat, more, pwd.**
7. Other commands should be implemented also:
    a. **args** - list all parameters on the command line, numbers or strings specific command. For example, args cp should print
    Number of args is 2: Source Path, Destination Path
    b. **date** - output current system date and time.
    c. **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
    ```

8. Redirecting should also be implemented (i.e. > and >>) to output the result of command to some file.
9. the interpreter allows any “possible” combination of all the above features using "|"pipeoperator**.** 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:/.**

Submission instructions:

1. **Submission deadline date 20/10/2018 on Acadox.**
2. **The assignment is submitted in group of maximum 3 students**.