Cairo University
Faculty of Computers & Information.
Operating Systems 1 Course
Third Year
Dr. Khalid Wassif
Summer 2023

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 or terminal.

Description

In this assignment, you will write a simulator for 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 are 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**
2. Your code **MUST** be divided into at least **2 major classes:**
    a. **Parser** class which is responsible for performing a parse for the user input to extract the command and its arguments. It's also responsible for reporting errors if the command is invalid or incorrect number of arguments.
    b. **Terminal** class which is divided into functions, each function is responsible for performing certain command. for example:
        i. public void date(); // This function prints the dat
        ii. public void pwd(); // This function prints the working directory
        iii. ….

    Submissions that do not contain these 2 classes will be graded Zero
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 commands or bad parameters the program should print some error messages. For example, if the user writes **mkdir**, the program should respond 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:/**
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, date.**
7. Other commands should be implemented also:
    a. **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

pwd : Current work directory date : Current date/time exit : Stop all

8. Redirection 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 the "**&**" operator**.** 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 is 7th August on Google Classroom.**
2. **The assignment is submitted in a group of maximum 4 students**.
    a. Students from **G1** and **G2** can form groups from both **G1** or **G2** students
    b. Students from G3 **MUST** form groups from **G3** Students only