**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
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**

**// In case of success, it should save the extracted command and arguments**

**// to args and cmd variables**

**// It should also print error messages in case of too few arguments for a commands**

**// eg. “cp requires 2 arguments” public boolean parse(String input); public String getCmd(); public String[] getArguments();**

**}**

# // Interface for Terminal

**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(); public void cat(String[] paths);**

**// Add any other required command in the same structure….. }**

1. Your CLI should be written in **Java** and as a task function (CLI commands maybe 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 messages. 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. You should implement the following commands: **cd**, **ls, cp, mv, rm, mkdir, rmdir, and cat.**

**Note:** *for the* ***cat*** *command, you only need to implement the case of one parameter; you are not required to implement the case of two parameters. For the* ***mv*** *command you are not required to implement the option part such: “****rm*** *–r folderName”.*