Create a Simple Shell

Create a simple shell with Java that can run in Windows, Linux and OS X. In the Windows operating system you need to install Cygwin. Use the default Cygwin installation configuration. This installation will provide the Unix-like system utilities you need for this project. Make sure that you configure Windows such that the PATH environment variable includes the directory path of those utilities.

The simple shell must give a ">" prompt such that the user knows that it is ready to receive user commands.

The utilities or commands that may be entered by the user on the simple shell are:

(1) ls
(2) cd
(3) echo
(4) ping
(5) ipconfig (Windows)
(6) ifconfig (Linux y OS X)

The simple shell at all times will continue to be ready for more commands showing the ">" prompt once all the previous entered commands have been executed until the command "exit" is entered by the user. Once the command or command sequence is typed in the simple shell, the key "enter" shall be hit for the command or commands to start executing.

Command sequence: The user can concatenate various of the above shell commands utilizing the character "$" as a command-concatenation character. Once the command or sequence of commands have been typed and entered on the simple shell, **the output associated with each typed command must be as follows:**

(1) The name of the commands with their respective arguments and a colon (:) all on the same line.
(2) Immediately underneath the output described in item (1), you must list the output of the command associated with stdout, if there were no errors, or stderr if there were errors. For example, if the user typed "ls -l" and afterwards "enter", the output must be as follows:
    a. ls -l:
    b. In the following lines from the output in item (a), all the files associated with the current directory must be listed as follows:

```
-rwxr-xr-x 1 Daniel None    2038 Aug 26  2016  zforce

-rwxr-xr-x 1 Daniel None    7444 Aug 26  2016  zgrep
```

```
-rwxr-xr-x 1 Daniel None    2154 Aug 26  2016  zless
```

(3) The sequence of output referred to in item (2) for every command must be followed in all cases. The fact that the user executes one or a sequence of commands linked with the character “$” does not change the requirement in item (2) for every single command.

The simple shell must be able to process any combination and permutation of all the 6 utilities listed above and linked with the character “$”.

Some examples of possible command sequences to the simple shell:

(1) cd ..\..\..\..\..\ $ ls $ cd basura $ cd D:\ $ ipconfig $ echo something $ ls
(2) cd C:\ $ ls $ ipconfig $ cd D:\temp $ ls

The commands must be executed from left to right in terms of how the user enters a command sequence. The output of the shell must be in that order, first the output of the leftmost command, then the output of the second leftmost command, etc. Following this order becomes critical to get consistent, predictable results in commands such as “cd ..\..\..\..\..\ $ ls -l”.

In addition to the requirements already mentioned, the simple shell must be able to execute the command “history” that must list the last commands made by the user up to a maximum of the last 20 commands. These last commands, up to a maximum of the last 20 commands, must be listed whether they produced errors or not. The format of the output of this command must have one command per line and each line must have a number; from the oldest command to the last command. The history of commands must also include sequences of commands. Each sequence of commands has to be in one line. As far as the “history” command is concerned, a sequence of commands is treated as one compound command.

Example (there are only 11 commands at this point entered by the user in this session):

```
History:
   1  ls -l
   2  pwd
   3  cd pwd
   4  pwd
   5  java
   6  java --version
   7  java -version
   8  pwd
   9  cd /
```

```
10 pwd
11 cd ..\..\..\..\..\ $ ls -l
```

The simple shell must be able to execute any command or sequence of commands listed as the output of the “history” command as follows:

a. If the user types “**!1**” and “**enter**”, command number 1 is executed.
b. If the user types “**!#**” and “**enter**”, the last command of the history listing is executed.

Very important; be sure that you include a readme.txt with the following information:

1. All the members of your team
2. Any relevant comments associated with the project.
3. If you use another separator for concatenating commands other than **“$”, I will assume that you used work that was not your own.**