C-Shell

This is an implementation of the Linux/Unix Shell in C Language. Similiar to the Linux Shell, it allows the users to enter various commands, and then it works accordingly. It supports a list of internal and external commands which are described briefly below. The implementation doesn't use the inbuilt external command implementations, instead it uses exec and fork to call the programs written for the external commands. This has been made as a part of the Operating Systems - CSE231 Course at IIIT Delhi.

Features

The various commands supported by the C-Shell is: #### 1) cd This is used to change the current directory of the program. The flags supported are:
* cd .. : takes you to the parent directory
* cd ~ : takes you to the root directory
* cd [dirname] : takes you to the directory name specified

2) echo This displays a line of text. The flags supported are:
* echo -n : do not output the trailing newline
* echo -e : enable interpretation of backslash escapes

3) history Many programs read input from the user a line at a time. The GNU History library is able to keep track of those lines, associate arbitrary data with each line. The flags supported are:
* history -c : This clears the history
* history [n] : Will output the last n commands entered by the user

4) pwd This prints the name of current/working directory. The flags supported are:
* pwd --help : display this help and exit
* pwd --version : output version information and exit

5) exit This causes normal process termination.

6) mkdir This make directories. The flags supported are:
* mkdir -v : print a message for each created directory
* mkdir -p : creates the parent directories as well
* mkdir --help : display this help and exit

7) rm This remove files or directories. The flags suuported are:
* rm -d : remove empty directories
* rm -f : ignore nonexistent files and arguments, never prompt
* rm -v : explain what is being done

8) ls This list directory contents. The flags supported are:
* ls -U : do not sort; list entries in directory order
* ls -a : do not ignore entries starting with .

9) cat This concatenates files and print on the standard output. The flags supported are:
* cat -E : display $ at end of each line
* cat -n : number all output lines

10) date This prints or set the system date and time. The flags supported are:
* date -u : print or set Coordinated Universal Time (UTC)
* date -r : display the last modification time of FILE

Setting it up

Clone the Repository and set it up on your Desktop. Open Terminal(Ctrl+Alt+T) and then type:
`make` ```` ./newshell.exe ```

Corner Cases Handled

1) cd
* If a user gives no argument after cd, it successfully takes the user to the root directory.
* In case of the argument being specified as a wrong directory, it prompts it to be an invalid directory.

2) echo * In case of no argument being specified by the user, it prints a blank line.
* It doesn't consider the "" while printing to the console.

3) history * It retains all the user entered commands(without a limit) and is accessible as per the user's requirement.
* On entering the number greater than the length of the history, it doesn't give any errors and displays all the commands in the history.

4) pwd * Even on specifying an argument that doesn't match with the flags supported, it displays the present working directory.
* It always gives the absolute path from the root and not relative path.

5) exit * On specifiying any kind of argument, it gives a prompt of exit, and successfully exits the code.
* It lets prgram to exit with a status code of 0 and not 1.

6) mkdir * In case of no argument specified, it prompts the user about missing operand
* If a directory already exists, it prompts the message that directory already exists.

7) rm * In case of no argument specified, it prompts the user about missing operand
* If the directory doesn't exists, it prompts the user that no such directory exists.
* If the user is trying to remove non-empty directories, it prompts and says that it can only remove empty directories.

8) ls * It displays all the directories in an alphabetical order, unless specified not to.
* In case of the -a argument, and an empty directory it displays . and .. correctly.

9) cat * In case of no argument specified, the shell doesn't print anything which signifies no output to print.
* If the file passed as an argument doesn't exist, then it prompts that the file doesn't exists.

10) date * In case of an unsupported argument specified, the shell prompts invalid argument.
* In -r argument, the shell prompts if the file doesn't exists.

Testing

To test the code, open the file test.txt. A few test cases have been mentioned in the file. Those commands can be typed in line by line to check the correctness of the shell.

Licensing

Author : Meghna Gupta
Feel free to point out issues! Pull Requests will be welcomed!