Project 1 Group 2

Randal Kwok
rkwok@hawk.iit.edu
> profile; traversal; alarm

John Paul Aldana
jaldana2@hawk.iit.edu
> history; shell; profile

Sufyan Menk
smenk@hawk.iit.edu
> parse-tree

Usage

In the MINIX shell "ash", `cd ./path_to_project` and `make clean` (if needed) then followed by `make shell` (or `make all` to include test executables). Running the shell can then be done by running `./shell`

Options

Setting the shell's `PROFILE` file
In the same directory as the shell executable, create (or edit if it exists) the file `PROFILE`.
The template for the `PROFILE` file is:

```
PATH=/usr/bin/
HOME=/usr/
ALARM=1
```

Each line should be terminated by either `\n` (new line) or `\r\n` (carriage return + newline).

Modifying the `HISTORY` file
The history of the shell is saved in the same directory as the `PROFILE` file and the shell executable in the file `HISTORY`. This file is created by the shell when a new command is entered. Each line in the file is delimited by `\n` represents a previously entered command. For each new command entered in the shell, it is appended to the end of the `HISTORY` file.

Commands within the `shell`
After running the shell via `./shell`, the user will be greeted with the prompt `shell>`. The user is then able to do the following:

- **Run single command**
  Write a command to execute
  i.e.: `ls`
- **Run multiple, sequential commands**
  Write multiple commands to be executed sequentially
  i.e.: `ls ; whoami`
- **Run multiple, parallel commands**
  Write multiple commands to be executed in parallel, subject to race conditions
  i.e.: `ls & whoami`
- **Run multiple, nested commands**
  Write multiple commands with nested parentheses
  i.e.: `(ls & whoami) ; (( pwd & ls -l ) ; uname)`
- **Abort command(s) that last more than five (5) seconds**
  If a command lasts for more than 5 seconds, the user will receive a prompt `Terminate process {PID}? (Y/N)` where a user can choose to hit 'Y' to terminate the process.
- **Use the 'tab-completion' feature**
  After (optionally) entering the beginning of a command, the user is able to hit the 'tab' key and the shell will replace the input with the first match in the `HISTORY` file. If no matches are found, no change will be made. If

a match is found, then the user will be able to hit the ‘up’ or ‘down’ arrow to traverse through the history list. WIP

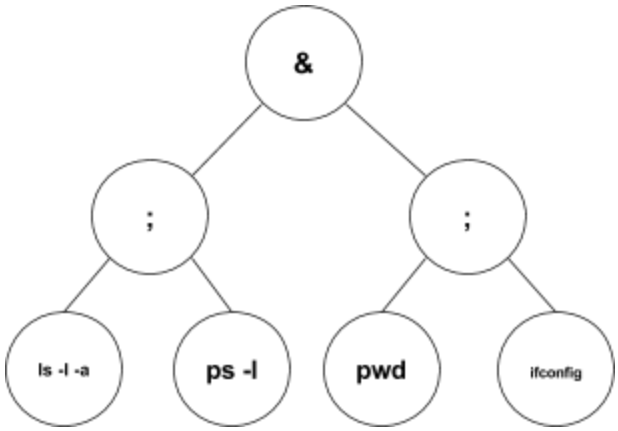
- **Exit the shell**
  Hitting `CTRL+Z` or `CTRL+X` will exit the shell.

Details

How our shell converts the user input into a traversable tree
First, our shell converts the infix expression into a prefix expression array. Using the prefix expression we make a parse tree. Then we traverse the tree and fork and exec when appropriate. After the command is complete or the user aborts, the tree and arrays are freed from memory. When a new command is issued the process is repeated.

Example:
Input: `(ls -l -a;ps -l)&(pwd;ifconfig)`
Prefix: `& ; ls *-l *-a ps *-l ; pwd ifconfig`

Code:
`parse-tree.h`
`parse-tree.c`



Exception Handling

PROFILE
If a PROFILE file is missing, an error is printed. The PROFILE file must be in the format seen above. If PATH, HOME, or ALARM is not found in the PROFILE file, an error is printed.

HISTORY
If no HISTORY file exists, print to screen No HISTORY file. If the file cannot be written to, the shell prints out Unable to write to HISTORY file: Permission denied.

Commands
When using parenthesis for command operators, if there are an uneven amount, an error will be thrown. If an unknown command is ran, an error will be printed.

Test Cases

Test	Description	Input	Result
1 ✔	Executables in PATH should work	`ls`	*returns a list of files in directory*

2 ✔	Show current working dir	`pwd`	*/usr* (or HOME in PROFILE)
3 ✔	Run multiple sequential commands in parallel	`(ls -l -a;ps -l)&(pwd;ifconfig)`	Runs *ls -l -a* and *pwd* in parallel. After *ls -l -a* executes, *ps -l* executes, and likewise, *ifconfig* executes after *pwd*.
4 ✔	Change directory (to project directory)	`cd /root/P1-working/`	Changes directory (chdir) to */root/P1-working/*.
5 ✔	Show current working dir (after cd)	`pwd`	*/root/P1-working* (or whichever directory was entered for T4)
6 ✔	Run the *hello-world* executable	`./hello-world`	*Hello, world!* (if this command is executed outside of this folder, then *./hello-world: command not found* is shown instead)
7 ✔	Run the *repeater* executable using various arguments	`./repeater 2 a &` `./repeater 3 b`	The console should have 2 *a*'s and 3 *b*'s followed by *Done!* twice. -- This command may print *shell>* before the command finishes.
8 ✔	Run the *sleeper* executable	`./sleeper`	If alarm is enabled, the console will print out *Terminate process: {PID}? (Y/N)* after 5 seconds. The user can choose to terminate the process by hitting 'Y' and then enter. If the user chooses to ignore the prompt or hits 'N' instead, the executable will continue to run until it completes.
9 ✔	Start shell without HISTORY file	`--`	The shell will prompt at the beginning *No HISTORY file*.
10 ✔	Start shell without PROFILE file	`--`	The shell will prompt at the beginning *PROFILE not found*.
11 ✔	Start shell with invalid PROFILE file	either set `HOME={invalid dir}` **or** make line 0 **!=** `PATH={path}` **or** make line 1 **!=** `HOME={path}`	(multiple) *Invalid PROFILE file. (Invalid HOME)* *'/usr2/' is not a directory*
12 ✔	Testing HISTORY	After beginning to type a command i.e. *who* and the HISTORY file has the entry *whoami*, then hitting tab will autocomplete to *whoami*.	Works. Hitting up/down arrow afterwards will allow you to 'cycle' through choices. -- Due to shell/tty implementation, this can be buggy.
13 ✔	Sequential expression with Nested Parentheses	`ls -l;(ps;pwd)`	Runs *ls -l* then *ps* and lastly *pwd*, which is the correct order.
14 ✔	Combination expression with Nested Parentheses	`(ls -l&(ps;pwd))&echo hello world`	Runs *ls -l* and *echo* in parallel, but does *ps* before *pwd*. Tabbing issue due to parallel printing. -- Might have tab/minor text display issues.