Author: Sumukh Hallymysore Ravindra
Date: 02/02/2016

Building A shell

Important implementation flow-

- ➔ Got the input string
- ➔ Parsed the input as tokens and build commands by segregating appropriate tokens
- ➔ Forked a new child process and executed the commands in the child using execvp sys call (except internal commands line cd, jobs and exit)
- ➔ Implemented Pipes, Redirection, sub shell, conditional, and sequential execution features
- ➔ Based on the command operation and child execution status (obtained using waitpid sys call), took necessary actions, like whether continue execution or not
- ➔ In subshell, called the command_line_exec() function again to execute the commands by spawning new children separately and collect the return status of the complete subshell execution
- ➔ Handled internal commands in parent process

Additional Implementation:

In 'cd' internal shell command:

- ▪ cd: Change directory to home. Used the getenv("HOME") function to get the path of HOME and change it to that diectory
- ▪ cd ~/<filename>: Replace '~' with the "HOME" path if it is present in the start of the

Wildcard implementation:

- ▪ Implemented wildcard substitution ('*') for handling pattern matching files.

eg: ls -l main*, grep cmd myshell*

Jobs:

- ▪ Implemented a global struct to maintain the background process and update the status of the process on completion by doing waitpid() along with WNOHANG flag at the start of the command and display the process status on 'jobs' command and destroy the completed and printed entries from the list.

Error Handling for conditional operators:

- ▪ Handled error for conditional operators without a normal token following them

Free memory:

- ▪ Free command_t (commands, and the commands pointed by the subshell) and job_t (which handle Jobs) structures once done