Lab 1 Report

Tiantu Xu

1. Overall Design

The basic design falls into 3 parts:

(1) Read the command line

In the function, fgets() gets the string of the command.

(2) Parse the command line

In this step, a string is parsed into different categories as "token". Some normal tokens go into cmd->argv[], and some filenames that needs redirect to go into cmd->redirect_filename[] array. Some special tokens need special handling and is implemented by switch-case logic.

(3) Execute the command line

I execute the single command specified in the 'cmd' command structure. I use fork() and execvp() to execute the command in a child process. Some of the commands like [cd], [jobs],[exit] is in shell process, and is handled in special.

Use switch-case logic to deal with different circumstances differently in different cmd->controlop, and take into account the statues the command exit with (==0 or !=0).

I also add jobs command into my shell. I store the pid into an array, and in the parent process, if cmd->argv[0] = "jobs", it would fall into a while loop for waidpid ==0, and call report_background_job function.

2. Noteworthy features

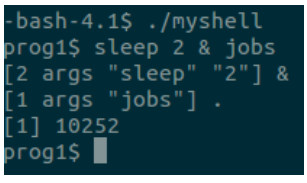
(1) When the environment (No zombie background activities) is clean, the test case would pass as 65/65.

(2) Except from passing all the basic test cases, my shell also implement [jobs] to report background.

3. Interesting Implementations

(1) Implementing [jobs]

If you type [sleep 2 & jobs] command, myshell would return a pid of background process [sleep 2]



(2) Some of the test case would fail if the background process is not cleaned up by others. If we type [ps –C sleep], some of the sleep process is still running in background, and I do not have permission to kill it. It is a little bit annoying when debugging.