**UNIX Programing Assignment 7**

Due date: 2016.12.13 23:59

Demo time: 2016.12.13 18:30 ~ 21:30

10 points

In this assignment, we will refine our tsh with job controls. In our previous homework, we have implemented a tiny shell that supports basic built-in functions (cd and pwd), command executions, and background processes. In this assignment, you have to implement **Terminal Control and Signal Handling** to support **Job Control.** More precisely, your shell shall support the following tasks:

* Signal Handling: user can terminate the running process with Ctrl+C without quitting from your shell
* Background processes without resulting in zombies
* Foreground/Background Switching: Your shell is able to put the process to the background (“&”) and bring background process back to the foreground by “fg” command
* Built-in functions: jobs, fg
  + **“Jobs”** allows users to view all the processes
  + **“fg** <shell\_assigned\_process\_id>” (You will need to handle the id by yourself)

To achieve these features, we give a brief guideline for this assignment.

**Guideline for implementing job control in the shell**

**Initializing the shell**

When your shell performs job control, you need to make sure that your shell stays in the foreground and doesn’t terminate. This can be done, e.g., as follows:

1. Getting your shell initial process group ID with the [getpgrp(2)](https://linux.die.net/man/2/getpgrp) function, and comparing it with the process group ID of the current foreground job associated with its controlling terminal ( [tcgetpgrp](https://linux.die.net/man/3/tcgetpgrp)function).
2. Setting your shell to ignore all the job control stop signals so that it doesn’t accidentally stop itself.

**Lunching jobs**

To perform job controls of child processes, please refine the following features of your shell:

* Handle process group id for each child process ([setpgid()](https://linux.die.net/man/3/setpgid) function)
* Set signal handling in each child process after the fork

**Foreground and background**

When a foreground job is launched, the shell must first give it access to the controlling terminal ([tcsetpgrp](https://linux.die.net/man/3/tcsetpgrp)() function). When the child process is terminated, the shell should regain control of the terminal for its own process group. If the process group is launched as a background job, your shell needs to keep tracking of the process by setting the signal handler, so that the process will not become zombie.