CMSI 387-01

Operating Systems

Spring 2013

Assignment 0321 Feedback

Britain Southwick

1a — You were able to work the command line and terminal windows effectively to get the requested shots. Congratulations on figuring out how to get `ps` to display threads, on both Linux and Mac OS X! (+)

1b — In terms of working with processes and threads, your screenshots were on the money. You caught the primary detail with respect to comparing Linux and Mac OS X approaches, which was the presence of a "LWP" (lightweight process) ID for Linux threads. You didn't think that this made for a relevant difference between the two, but it is actually a key detail. You would have totally nailed it if you had observed that those LWP IDs *are in the same sequence as the main processes*. But even without that, you've done enough for the maximum proficiency **:)** (+)

1d — Check my comments in your points to ponder to see what's missing in your answers. Your responses are all sensible, but they are not completely reflected in your code. Also, #2 was unanswered, so I dropped a humongous hint in my inline comments. Remember, that was associated with this outcome *1d* regarding I/O. Big hint there! (/)

2c — Your shell is still in a somewhat preliminary state, so this proficiency necessarily can't be very high. It's better than having nothing yet though, that's for sure. I have a few inline comments that might help you along in the right direction. (/)

3a — You were able to accomplish this process business successfully on two platforms. (+)

4a — As mentioned, your shell is still in a pretty early state, but it is certainly moving in the right direction with your initial command loop and generally decent `&` behavior (except for how you detect it). As mentioned in my points-to-ponder comments, it can almost run scripts **:)** (/)

4b — The code is not far enough in for a separation-of-concerns review, but as you move on, make sure to keep in mind that you want your variable names to be clear and descriptive, and be open to defining helper functions that keep self-contained computations separated, in order to make the main line of code more compact and easier to follow. (/)

4c — Even in its current unfinished state, your code is decently easy to read. Just keep in mind going forward that C is syntactically a lot like Java and JavaScript, so whatever is stylistically correct in those languages is likely appropriate in this language also. (/)

4d — You did very well with your process/thread work, but of course have some way to go with your shell. Still a lot of documentation and web lookups to do there. (/)

4e — Your version control usage is appropriate for the process screenshots, and given the state of your shell it is reasonable to be at the current number of commits. Assuming that you keep working and committing at appropriate points, I think you'll end up fine for this outcome. (+)

4f — Submitted on time, although the shell is extremely preliminary. But, still, a little bit of everything was there by the due date. (+)

Updated feedback based on commits up to May 3:

1d — Your revised points to ponder answers show an understanding of how I/O can be redirected into and out of processes, and your shell (to a degree) demonstrates this understanding. (+)

2c, *4a*, *4b*, *4c*, *4d* — You have shell! However, it is not functionally complete (argument handling is severely limited, your `&` parsing has flaws, and you don't have any "secret system call" code), the program is one monolithic `main` function, and your code needs comment cleanup. This adds up to | across the board. (|)