Alexander Young
CprE 308 - Section 7
Spring 2021 - Professor Matthew Wymore
Project 1 - Shell

Summary

I certainly learned a lot about C and lower level programming by making this shell. For this project, I did the obviously best thing to do, which was speedrun it. So on the day that it was due (originally March 8th, 2021), I got up at around 2:30 AM and just started writing code. After a few distractions, I did get a bulk of the main shell written out in about 7 to 9 hours. I had all the builtins implemented, and had foreground external process execution. I was also pretty happy with how extensible the builtin system worked, I can easily add more builtins if I wished. It only took a couple more hours to get backgrounding processes working, and jobs implemented. I would say that I did have some fair C experience before this class, but I still learned a lot. In particular, how to best handle dynamically sized multidimensional arrays. I store the initial args provided by the user, the builtin list, and the subprocess list in 2d arrays. When I was trying to implement piping (which I wasn't able to successfully do), I did experiment with 3d arrays, but that proved to be difficult to handle in a safe manner. All my arrays should be dynamically sized, so theoretically, the maximum number of builtins, subprocesses, and args, is limited by the OS or the system memory, however I haven't tested with a large number of any of those.

In addition to handling arrays, another thing I learned was handling function pointers in C. I implemented my builtins by storing the command name, and a pointer to the function in an array. That decision initially led to many headaches, as I had never really used function pointers in C before. It was kind of neat being able to just call a function with only a pointer to it. I did also have a bit of trouble with the syntax to typedef that function pointer definition, but I think it ended up pretty well. It is almost like an interface type of construct that you see in other languages, and I think that is pretty cool.