Project 1 Report

Timeline of Implementation

(As only one in “group,” no division of labor possible)

- September 13
  Developed the basic framework. Added headers, set constants, implemented PrintPrompt(), GetInput(), and preliminary work on main().
- September 16
  Made the exit command modular instead of inline in main(), implemented ExitOrNo().
- September 18
  Made empty modules for each built-in. Began work on command parsing. Implemented GetCommand(), preliminary work on GetArguments().
- September 20
  Implemented Echo(), preliminary work on ViewProcess() and ChangeDir().
- September 21
  Completed implementation of GetArguments(), ViewProcess(), and ChangeDir().
- September 25
  Preliminary work on Execute(). Debugged ChangeDir() and GetArguments().
- September 27
  Made the main portion of each execution cycle its own module. Implemented MainExecutionModule(). Began work on time function.
- September 28
  Made time command handling its own function instead of inline in main(), implemented TimedCall().
- October 3
  Worked more on Execute(), adding redirection functionality. Wrote up README.
- October 6
  Went back to TimedCall() to try and determine why it was returning 0, began preliminary work on piping implementation.
- October 7
  Continued work on piping with little success, ran into problem with second command not executing.
- October 10
  Added second level piping functionality, still unable to determine source of execution issue. Updated README file.
- October 12
  Cut losses, drew up Project Report, tarballed, submitted.

Issues Discovered

I ran into some issues with my main execution areas. I found they had some bugs that were more easily dealt with after putting them into separate methods.

Another issue I noticed was with cd. I noticed that there was a problem if the user added an extra / at the end of their path (i.e. cd mydir/mydir2/). I solved this problem by adding a small algorithm at the end of my cd method to scrub an extra / off of the end, thus allowing the user to do so (or not do so) without breaking the program.

A third problem I came across when I implemented piping, where sometimes the filedescriptors wouldn't be cleared correctly, and later redirection wouldn't work properly. I solved this problem by obtaining the channel number from open() and then directly inserting that into my dup() and close().

I was unfortunately unable to solve a problem wherein my pipes' second commands would not execute despite having checking in place for the path, and being generally cautious with the way I implemented piping.

Things Learned

I learned, and was quite fascinated, that (when implemented correctly) you can not only call a shell from inside of itself, but you can also update the sourcecode, compile it, and then run the updated shell without ever exiting the original instance of the shell. This, of course, led to a lot of "exits" at the end.

I also learned that I/O streaming can be quite finicky and generally annoying.