Name: SGT Schoenwald-Oberbeck, Jesse
Date: 16JUN2017
Current Module: Networking using C
Project Name: fdr

Project Goals:

Build a server which listens on three ports based on UID. The purposes for these ports are taking in packets including a string of an indicator character (F, D, or R) each of which causes a corresponding action on the rest of the string.

Considerations:

- o Transmissions should be UDP.
- o C networking can be tricky.
- o Numbers expected can be much larger than existing data types will handle.
- o Roman numerals will <u>only</u> be handled correctly in the expected additive/descending form.

Initial Design:

The program will fork three processes, each on a separate port via the main function. Each forked process will create its socket, then loop indefinitely, waiting for and handling input. Input will be received via the listening/bound sockets.
A separate file, black_adder, contains the functions which digest/transform the data into the values returned to the client.

Data Flow:

UDP packets are received by forked child processes, run through a switch case based on the first character in the message, then sent to a corresponding function to transform the data in a manner in accordance with the indicator at the start of the user's string.

Communication Protocol:

UDP over bound port. Specifically on loopback.

Potential Pitfalls:

- o Networking in C
- o Very large numbers with no data types large enough by default.
- o Translating an outdated mode of numeric representation.

Test Plan:

User Test:

Multiple runs of the game, using every variation/combination of options and inputs the player can think of.

Test Cases:

All test cases completed with correct output.

Conclusion:

Networking with C is tedious and full of potential for all manner of error. Fortunately UDP is the simpler option, and much more appropriate for this use.

Handling large numbers was a significant challenge at first. I initially created my own method for handling them, as seen in the adder function and related functions. Later, the words “man bn” were written on the board, revealing an already built, potentially better way of handling the data. I left my own version in for the context it had already been built for.

Buffer sizes were adjusted to reduce stack size, but not within the allowance of the stack size warning, unfortunately, due to accounting for enormous numbers.

Forking processes was fairly straight-forward and manageable. It’s worthy of note how nice it is that a single function call causes the parent process to wait for all children.