CIS 160 Final - Kelbe Faast

I think one of the most major concerns I've come to acquire about the software development scene is the apparent reliance on code that has been passed around for so long without any meaningful updates that it is not only nearly impossible to unravel the convoluted structure of the code, but that this inability to alter the code in a reliable way makes the developer get stuck with what the program will allow them to create rather than what they can create on their own. This was most evident from the reading '*Big ball of mud'* which illustrates just how damaging these lazy programming habits can be, as they slowly seep into thousands of created projects everyday. While the reading '*Is computer programming all about coding*' shows that while coding is a major part of the process, the planning and execution of that planning is really where most of the work comes from. And if people are taking shortcuts to create these programs by using pre-existing mud balls with no room for change, then the project as a whole will suffer. Continuing on that point is the article '*Inevitable Pain*' which details just how frustrating forgetting to plan ahead can be as you need to backtrack and constantly rewrite the structure of the code to allow these new ideas to function, potentially breaking everything else in the process. If you started off with a mud ball, then going back to change some of that code wouldn't even be a reasonable option given all of the time that would need to go into unraveling the mess. Another thing to consider about this type of work is just how long it takes to complete projects that have a lot going for them in the first place. Taking the example from the reading '*Dreaming in code*' and its summary '*Software is Hard*' you can see that even with exceptional talent and forethought going into the project isn't enough to keep things running smoothie or quickly, as the years pass by in a mere blink of an eye and you barely have a 1.0 version for release. The larger the company and resources, however, can lead to an exception to the rule as they can consistently put out major programs in record times with little to no flaws, which are usually patched on user feedback in regular updates anyways. This is mainly in reference to the DevOps reading in which you can see the foresight of these major production companies and just how refined and strategic they are in their chosen craft.

Having completed most of the assignments from this term, I am confident in displaying which projects inspired me to keep going and which ones were a slog to work through. I resonated most with the python and greenfoot type material because I like doing work in which there is a clear goal, and I just need to create the structure for that goal to work as intended. It's like looking at a math problem and you have the starting numbers and the answer, but you have to create the equation that makes them equal. In a similar vein, I initially didn't like the code circuit portion of the assignments, but after switching from logisism to digital, and doing some online research to better understand how the circuits worked, by the time the hdl circuits came I was actually having fun trying to figure out how to create a new circuit from an existing one. The assignments that I had the most trouble with and the ones I often didn't even complete were the ones that left too much up to me to decide how they would work. Whether I like it or not I am simply not a very creative person and if I have to come up with the project like the processing js programs or the college picks, I either get stumped on trying to think of an idea, or just get really bored with the whole process of designing it that I end up scrapping the whole idea. I think I just rather prefer someone to tell me exactly what to do, but perhaps only a little bit of guidance on how to do it so I can research for myself and create the thing they wanted rather than come up with something myself. Overall though I don't really see myself pursuing a career in software development so much so as something like hardware repair/upkeep, security or networking.