Boris Topalov, bnt4yb

3/24/2020

IBCM was initally much harder than a "traditional", higher-level language like C++, but once I got the hang of it it wasn't that bad. Since there are so many less operations, there seems to be a lot more repetition with IBCM so you start to pick up on patterns and certain sets of instructions that serve a specific purpose. I never would have thought that I would be able to write a sorting algorithm with just sets of bits, but I see that it is possible. It was cool to be able to see how computers/machines operate on a low level, since I haven't really had any exposure to something like this.

The most tedious part of this lab was definitely debugging, since making a small change in a code often meant rewriting half of the instructions since all of the memory addresses changed. I started using nop instructions way too late, so for the entire pre-lab and some of the in-lab I was getting pretty frustrated when changing my code. I am not sure how this could be combatted in the simulator, though. Having instructions and variables be indistinguishable, at least for the machine, seems like a very powerful tool, and I am sure that it could be exploited to make code easier. We saw it in writing a quine. It ended up being much less complicated than I expected because of the lack of actual variables. IBCM seems like it has a high skill floor because of how abstract it is, but also a very high skill ceiling because there are less things to worry about at such a low-level.