Aaron Rosenberger  
COMP 340 OS Project 2

I experienced, again, that simulators are very difficult to code. Also, I learned that it’s not very efficient to implement a ton of cores (more than 8, probably) because things get too complicated. In terms of this project, it costs a lot of overhead, looping, and stack to store all those data structures that represent each core.

Different inputs can cause all kinds of phenomena. For us, different inputs mostly just broke our code and forced us to develop more. Obviously, as the number of cores increased, the wait time for each process decreased; for example, one of my tests involved a processor with 30 cores. The wait time for each of its 8 processes was 0 (fp4.txt).

The file ‘fp3.txt’ exhibits an almost exponential increase in wait time. Because I had the duration same as the arrival time, it took an increasingly long amount of time to not only execute each process, but even get to it. This file had some of the biggest numbers for wait and turnaround time.

In the file ‘fp5.txt’, wait time was almost nothing, even though the number of cores was 8 and there were 8 processes. Curiously, there was still some wait time for this test.

I learned that if I want to implement a simulator again, I should use C++ or even Python. ☺