Knicholas Kennedy
Operating System Simulation Final Project

Software Required: Java 6 or greater
Hardware Required: Very little to none

I implemented a multi-level parent-child process relationship simply by having a pointer to the parent process and a list of children. This allows me access as far up or down in the process matrix as I want. I used a multi-level priority-based round robin scheduler overall, but I also tested a shortest job first algorithm. The shortest job first algorithm was actually faster, but I deemed it unfair because in real life you would not know the size of the process to be checked against. I used binary semaphores to get around the critical section problem. This, combined with other aspects of my machine, assures that only one process can be in its critical section at a time. I used the banker's algorithm with a reference bit to solve process resource management issues. I also avoided deadlocks using the round-robin approach with priority aging to ensure that processes do not sit at the back of the queue for the entirety of their existence. My memory was separated into virtual memory, which I labeled as secondary memory, main memory, cache, and registers. I only used registers to simulate the use of them, but everything else has proper algorithms associated with it. I handle multi-threading using hardware, each of my cores has four hardware threads, and I use two cores. This gives me a total of eight threads. They will not always be running at the exact same time, due to the nature of thread starts in Java, but they consistently run 7-8 at a time. I used a simple java swing gui to display all of the relevant information. I was very worried about this project, but in retrospect it was one of the most fun individual projects I've ever done! I wish I had started slightly earlier, as I'm sure most students wish, but overall, I feel that I did very well on this assignment and took it seriously.