Richard Coffey
ECE 3220
4-9-18

Program 2 Writeup

The implementation of these three scheduling techniques were easy enough to understand in lecture, however when it came to implementing them, there were a few key areas that needed to be understood. First and foremost, Greg and I had to outline the specific areas that required separate code for the SJW cases. In particular, we isolated the load_process function, in which there had to be a separate else case to find the shortest job in the work queue every time there was a context switch. Additionally, thanks to Professor Martin, we correctly implemented a technique called a "searching for loop" which allowed us to run through the queues to find matches very quickly and easily. For the most part, we used GDB to find the areas that caused the program to exit falsely. In one instance we found that the we were leaving an empty hole in the queue when a process was loaded, and we fixed this by simply moving all the other processes up 1 space. Finally, after we managed to get our code to ouput the correct results, we were not getting any of the context switching numbers to print. We determined that we failed to increment the global value every time a process was preempted, and by putting switches++ in both cases of preempt_process(), the context switch value printed out as expected. Overall, this was a fantastic exercise in implementing scheduling techniques, and really helped us understand more about how job processing can be optimized in an operating system.