Greg Pregulman & Eric Wenski
EECS 678
Project 2

The first function that we implemented was enqueue_task_other. This function adds a task to the running queue. This is done by setting the time slice of the task, adding it to the queue, and then incrementing the number of tasks in the queue. The next function was dequeue_task_other which removed a task from the queue. It updates the running queue, removes the task from the queue, and decrements the number of tasks in the queue. Then we implemented yield_task_other_rr which is called when a task yields control of the CPU. After that, we implemented the *pick_next_task_other_rr, which selects the next task to run. It sets the next task to run to be the one at the front of the queue unless the queue is empty. It also sets a timer to maintain correct runtime statistics. The next function was task_tick_other_rr, which checks the quantum to see whether it should run a FCFS scheduler or a round robin scheduler. If the quantum is 0, then it runs FCFS. If not, it runs round robin until the time slice is 0. It also updates the runtime statistics. We also added code to make this other round robin scheduler valid. We modified __sched_setscheduler to allow for this new scheduler. We also defined the setquantum system call. This just takes in a quantum value that the time slice is set to. We also added the system call for setquantum to the list of system calls as call 338 in both the syscall table and the unistd table.

We tested the program using the thread_runner code provided with different quanta, threads, and allocations of memory. We started the testing with quantum 0 to make sure FCFS worked and that the quantum option was being accepted. We looked for the program to run and give output similar to the test included in the makefile. Then we moved on to different quanta with varying buffer sizes and numbers of threads. Our biggest problem is that for some reason it wouldn't accept other_rr as a valid scheduler option on one of our KVMs, but it worked fine on the other. We also had a small problem with the program hanging, but that was fixed when we realized the yield function wasn't correct. It also took us a while to realize that we needed to rebuild and re-install the kernel. All functions seem to work correctly.