COSC 201 – Lab #6

Look Ma, I'm a Job Scheduler!

Purpose: to practice using Priority Queues to do a core OS task

Tasks:

1.) Open Eclipse and start a new Java Project called Queue Project

2.) Add a new class called Job. Job will have two fields, a String called command and two integers, one called priority and one called timerequired. priority indicates what level of priority that the Job has. timerequired notes how much time is required for the job to finish.

3.) Add a new class called JobScheduler. JobScheduler will have one field, a PriorityQueue and three methods, one called ScheduleThis which will add elements to the Queue, RunFIFO which will run the jobs in priority order until complete and RunAdjusted which will run the jobs using an adjusted shared CPU time algorithm (see below). For each, you should print at what time a job completes (we assume that we're starting at a 0 time point) and then the average completion time. Main can go here if you'd like.

4.) The adjusted shared CPU time algorithm will work as follows: a job, once selected can only run up to 25 time ticks before having to give up the CPU. This means that if a job needs 45 ticks, it'll have to go through the queue twice. If a job needs less than the full 25 time ticks it simply forfeits the rest of its time and we dequeue something else and start it "running". To handle priorities, we will first sort everyone by priority (using the PriorityQueue) then allot each job 25*n time where n is the priority level of the job. If the job has a priority 2, this means the job can use 50 time ticks. Priority ordering should only be used to initially decide the order, otherwise it should ignore priority for ordering. Due to this, you may need an additional data structure to mimic a "normal" queue.

5.) Turn in your code via the Digital Dropbox in Blackboard.