Question 2

How would you run the scheduler to reproduce each of the examples in the chapter?

```
8.2:     ./mlfq.py -n 3 -q 10 -l 0,200,0 -c
8.3:     ./mlfq.py -n 3 -q 10 -l 0,180,0:100,20,0 -c
8.4:     ./mlfq.py -n 3 -q 10 -l 0,175,0:50,25,1 -i 4 -c
8.5.1:   ./mlfq.py -n 3 -q 10 -l 0,100,0:100,50,2:102,50,2 -i 2 -S -c
8.5.2:   ./mlfq.py -n 3 -q 10 -l 0,150,0:100,50,2:102,50,2 -i 2 -S -B 50 -c
8.6.1:   ./mlfq.py -n 3 -q 10 -l 0,175,0:75,100,9 -i 1 -S -c
8.6.2:   ./mlfq.py -n 3 -q 10 -l 0,175,0:75,100,9 -i 1 -c
8.7:     ./mlfq.py -l 0,140,0:0,140,0 -Q 10,20,40 -a 2 -c
```

Question 3

How would you configure the scheduler parameters to behave just like a round-robin scheduler?

```
./mlfq.py -n 1 -q 10 -l 0,100,0:0,100,0:0,100,0 -c

  Job  0: startTime   0 - runTime 100 - ioFreq   0
  Job  1: startTime   0 - runTime 100 - ioFreq   0
  Job  2: startTime   0 - runTime 100 - ioFreq   0

Final statistics:
  Job  0: startTime   0 - response   0 - turnaround 280
  Job  1: startTime   0 - response  10 - turnaround 290
  Job  2: startTime   0 - response  20 - turnaround 300

  Avg  2: startTime n/a - response 10.00 - turnaround 290.00
```

Question 4

Craft a workload with two jobs and scheduler parameters so that one job takes advantage of the older Rules 4a and 4b (turned on with the -S flag) to game the scheduler and obtain 99% of the CPU over a particular time interval.

```
./mlfq.py -n 3 -l 0,100,0:0,50,9 -i 1 -S -c
```

Question 5

Given a system with a quantum length of 10ms in its highest queue, how often would you have to boost jobs back to the highest priority level (with the -B flag) in order to guarantee that a single long- running (and potentially-starving) job gets at least 5% of the CPU?

You need to boost every 200ms, since it would only run 10ms at a time before pushed down to starve. 10/0.05 = 200

Question 6

One question that arises in scheduling is which end of a queue to add a job that just finished I/O; the -I flag changes this behaviour for this scheduling simulator. Play around with some workloads and see if you can see the effect of this flag.

The -I Flag places the job that just finished at the start of the queue instead of the end.