Anand Chari
WES 237: Assignment 2
Dining Philosophers:

I started this lab by copying over code from previous labs to set up the LEDs, blinking, and the RGB LED. I then started to write the function to act as a philosopher and gave arguments as two forks which would be represented by locks and then after the amount of time to "nap" then a state of starvation until forks were available again. I considered the thread to represent the philosopher and then created 5 locks to represent the forks. I then created 1 thread for each professor and wrote the philosopher function to try and acquire two non-blocking locks; if the function could acquire the locks, the philosopher ate then slept, otherwise the philosopher napped. This did not work. The first philosopher would eat and nap and the rest were completely blocked. I then switched the algorithm to follow a sequence of eating, napping, sleeping and using the two locks to block threads rather than allowing all to run simultaneously and the threads ran the algorithm correctly. I found that napping must not be longer than eating to avoid the threads undergoing constant starvation. I initially used 10 blink at .5 seconds each for eating, 5 blinks at 2 second each for napping but because napping was longer than eating I ran into deadlock. To counter this I increased eating to 20 blink and decreased napping to 1 second per blink. This fixed the issue.

When implementing random.randint I chose eating bounds to be 15 to 20 and napping bounds between 1 and 5 keeping the time of being on/off the same. This would mean that eating would still take 7.5 to 10 seconds while napping would take between 1 and 5 maintaining that napping would be shorter than eating. This holds our truth from above and keeps our threads from deadlock.

Youtube: https://youtu.be/sihSRZ-xMlM
Github: https://github.com/achari23/wes237assignment2