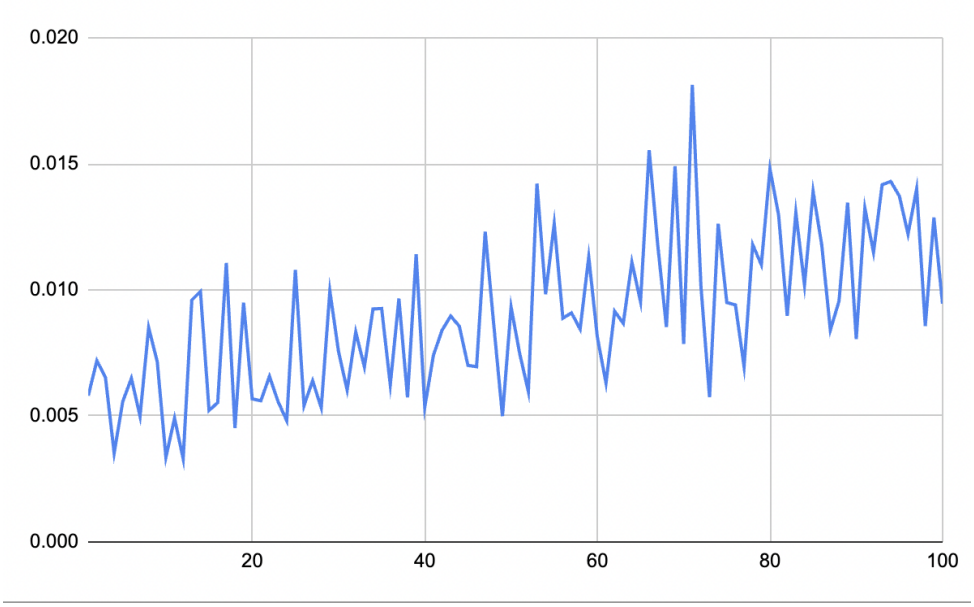
Homework 3

1a. This will create a starvation for writers as the up(&mutex) is happening before up(&writer). The up(mutex) call allows another reader to enter who does down(writer), blocking the writer again before the writer has the chance to enter. Thus it is important that up(&writer) is before up(&mutex) as it checks whether the last reader has left and then allows writers to enter preventing a starvation of writers.

1b. This code doesn't work correctly either since in the if statement, the up(&mutex) and up(&writer) allows both readers and writers to enter the critical section simultaneously.

1c. If the last reader is leaving the critical section and mutex is 0, the up(&writer) will unblock a writer. The down(&mutex) in the writer() function will cause mutex to become -1, and even when the writer leaves the mutex becomes one. A reader entering after the writer will do down on mutex and become forever blocked creating a starvation.

2c.



The graph above shows a generally increasing trend for computation time with noise. The generally increasing trend can indicate that beyond a certain number of threads (in this case, 12) there is little profitability in increasing the number of threads as the cost of increasing the threads to computation time is greater than any time it saves.