Testing:
For this assignment, testing was done throughout the assignment, starting off with just the single-threaded server with the functions for handling in a different header file. Then I started working on the multi-threaded part where I tested if each connection fd, that gets sent, gets handled correctly on the queue and gets pulled off correctly. After this, I tested if it worked with just handling one request. After implementing the file locks, I tested using a script where I accessed the same files and made sure that there was not any concurrency. So this allowed me to be sure that multithreading worked. When testing the functions for logging, I first did individual testing and after implementing the functions I did a whole test using the server. When all this was done I did testing using Donovan's test script which was given in the class discord. This is where I found a couple of bugs and fixed an issue when stress testing. After all of this was done and passed all the tests, I felt confident that the program works.

Questions:
Repeat the same experiment after you implement multi-threading. Is there any difference in performance? What is the observed speedup?

After observing the difference in performance between the HTTP server from assignment 1 and assignment 2, there is a difference between speeds. Sadly assignment two is slower than assignment one which is about .1 second slower. This might be due to the dispatcher and the worker threads and with the locking of files and mutexes and checking if logging is enabled. Also because of the memory, I am allocating in general, and checks for logging.

What is likely to be the bottleneck in your system? How much concurrency is available in various parts, such as dispatch, worker, logging? Can you increase concurrency in any of these areas and, if so, how?

The bottleneck in the system is in the areas where there isn't any concurrency. Which is not too frequently but frequent enough to slow it down so much that assignment 1 is faster than assignment 2. There is a lot of concurrencies but there is more concurrency available that isn't used. In the dispatch, all the concurrency is used but in space like the worker threads and logging, I could have increased concurrency. But overall concurrency is almost at the maximum. I think the general reason why my code has a bottleneck is because of the shared data and amount of memory I am allocating.