CS 1632 - DELIVERABLE 3: Performance Testing

By: Joseph Reidell
GitHub: jreid2f
https://github.com/jreid2f/CS1632-Deliverable3

Summary

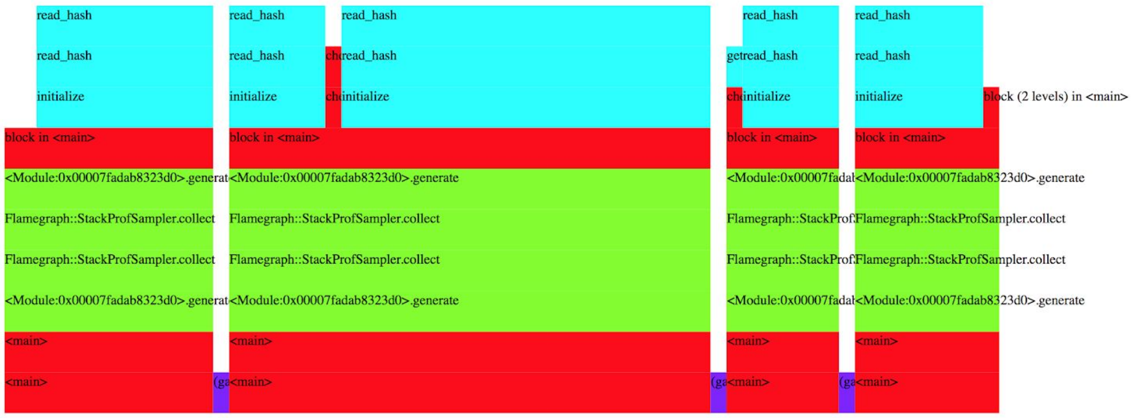
The biggest problem I had with this was figuring out a way to start writing the program. The second problem was also creating unit tests for the programs. I was not sure where to go because I ended up going a little longer than I might of needed too. I feel like I can get it down to fewer lines but, I would have to reconstructure everything to make it work. I had something going with the hash function we were given. At that point, I wasn't sure where to go. There was so much more that I needed to do. I was able to finally split everything up and check to make sure everything was working correctly.

From the unit tests, I figured the easiest and simplest edge cases to look for, were the amount of arguments that were needed to run the verifier. Instead of raising an error message, I thought it might be easier to check the validity and existence of the files. The program should only accept in one file. I wanted to check what would happen if no files, 1 file, or 2 files were entered. When entering no file, it failed and displayed ”Enter in a valid file that actually exists”. Same thing when two files were entered in.

I tried loading up the flame graph for "long.txt" but, my browser would not load it up. The picture on the next page shows the flame graph for "sample.txt". After looking over the flame graph, it seemed creating the flame graph was taking up the most CPU. Second behind that was "blockchain.rb" using the method "read_hash". The method unpacks and calculates the hash function for the blockchain. Sadly, getting rid of that function would greatly disintegrate most of the program. It would not be able to function properly. Best thing to probably do to shave some time off would be to disable creating the flamegraph.

To shave off some time for the verifier, I commented out the generation of the flame graph. The program originally was running at about 37 seconds. I shaved off about 3 seconds from the program for just getting rid of the flame graph. The program ran at about 34 seconds. Once that was removed, I got a mean of about 35 seconds. The time checks are on the last page.

Flame Graph



🟥 verifier.rb (59 samples - 93.65%)

🟩 flamegraph-0.9.5 (59 samples - 93.65%)

🟦 blockchain.rb (55 samples - 87.30%)

🟪 (3 samples - 4.76%)

Results of the three “long.txt” times:

1. 34.873 seconds
2. 34.803 seconds
3. 35.706 seconds

Mean: 35.127333333333
Median: 34.873