Ray Trebicka, John Nguyen

Project 3

10/23/17

*Summary of sort and Search Times*

**Insertion Sort**

10x10 Grid

Wordlist 1:

**Time for Insertion Sort: 0.137114**

**Time for Binary Search: 0.00842**

**Total Time: 0.145534**

Wordlist 2:

**Time for Insertion Sort: 168.508**

**Time for Binary Search: .018429**

**Total Time: 168.526**

15x15 Grid

Wordlist 1:

**Time for Insertion Sort: 0.123976**

**Time for Binary Search: 0.025002**

**Total Time: 0.148978**

Wordlist 2:

**Time for Insertion Sort: 175.027**

**Time for Binary Search: 0.071296**

**Total Time: 175.099**

30x30 Grid

Wordlist 1:

**Time for Insertion Sort: 0.140447**

**Time for Binary Search: 0.149967**

**Total Time: 0.290414**

Wordlist 2:

**Time for Insertion Sort: 175.355**

**Time for Binary Search: 0.237667**

**Total Time: 172.592**

50x50 Grid

Wordlist 1:

**Time for Insertion Sort: 0.123165**

**Time for Binary Search: 0.389604**

**Total Time: 0.512769**

Wordlist 2:

**Time for Insertion Sort: 177.588**

**Time for Binary Search: 0.723234**

**Total Time: 178.311**

250x250 Grid

Wordlist 1:

**Time for Insertion Sort: 0.13145**

**Time for Binary Search: 10.5465**

**Total Time: 10.678**

Wordlist 2:

**Time for Insertion Sort: 169.586**

**Time for Binary Search: 15.8228**

**Total Time: 185.409**

**Quick Sort**

10x10 Grid

Wordlist 1:

**Time for Quick Sort: 0.004303**

**Time for Binary Search: 0.009167**

**Total Time: 0.01347**

Wordlist 2:

**Time for Quick Sort: 0.203215**

**Time for Binary Search: 0.02066**

**Total Time: 0.223875**

15x15 Grid

Wordlist 1:

**Time for Quick Sort: 0.004862**

**Time for Binary Search: 0.02627**

**Total Time: 0.031132**

Wordlist 2:

**Time for Quick Sort: 0.202345**

**Time for Binary Search: 0.068737**

**Total Time: 0.271082**

30x30 Grid

Wordlist 1:

**Time for Quick Sort: 0.004339**

**Time for Binary Search: 0.145879**

**Total Time: 0.150218**

Wordlist 2:

**Time for Quick Sort: 0.204692**

**Time for Binary Search: 0.252326**

**Total Time: 0.457018**

50x50 Grid

Wordlist 1:

**Time for Quick Sort: 0.003676**

**Time for Binary Search: 0.408902**

**Total Time: 0.412578**

Wordlist 2:

**Time for Quick Sort: 0.208218**

**Time for Binary Search: 0.710009**

**Total Time: 0.918227**

**250x250 Grid**

Wordlist 1:

**Time for Quick Sort: 0.003881**

**Time for Binary Search: 10.3252**

**Total Time: 10.3291**

Wordlist 2:

**Time for Quick Sort: 0.200218**

**Time for Binary Search: 16.3994**

**Total Time: 16.5996**

**Merge Sort**

10x10 Grid

Wordlist 1:

**Time for Merge Sort: 0.006554**

**Time for Binary Search: 0.009485**

**Total Time: 0.016039**

Wordlist 2:

**Time for Merge Sort: 0.33668**

**Time for Binary Search: 0.018812**

**Total Time: 0.355492**

15x15 Grid

Wordlist 1:

**Time for Merge Sort: 0.00611**

**Time for Binary Search: 0.029833**

**Total Time: 0.035943**

Wordlist 2:

**Time for Merge Sort: 0.337374**

**Time for Binary Search: 0.069755**

**Total Time: 0.407129**

30x30 Grid

Wordlist 1:

**Time for Merge Sort: 0.005488**

**Time for Binary Search: 0.1448**

**Total Time: 0.150288**

Wordlist 2:

**Time for Merge Sort: 0.340594**

**Time for Binary Search: 0.242214**

**Total Time: 0.582808**

50x50 Grid

Wordlist 1:

**Time for Merge Sort: 0.005327**

**Time for Binary Search: 0.413657**

**Total Time: 0.418984**

Wordlist 2:

**Time for Merge Sort: 0.341797**

**Time for Binary Search: 0.681435**

**Total Time: 1.02323**

250x250 Grid

Wordlist 1:

**Time for Merge Sort: 0.005876**

**Time for Binary Search: 10.8299**

**Total Time: 10.8358**

Wordlist 2:

**Time for Merge Sort: 0.348914**

**Time for Binary Search: 16.9805**

**Total Time: 17.3294**

**Heap Sort**

10x10 Grid

Wordlist 1:

**Time for Heap Sort: 0.004604**

**Time for Binary Search: 0.010218**

**Total Time: 0.014822**

Wordlist 2:

**Time for Heap Sort: 0.294035**

**Time for Binary Search: 0.018327**

**Total Time: 0.312362**

15x15 Grid

Wordlist 1:

**Time for Heap Sort: 0.004634**

**Time for Binary Search: 0.027266**

**Total Time: 0.0319**

Wordlist 2:

**Time for Heap Sort: 0.286997**

**Time for Binary Search: 0.068852**

**Total Time: 0.355849**

30x30 Grid

Wordlist 1:

**Time for Heap Sort: 0.00436**

**Time for Binary Search: 0.154318**

**Total Time: 0.158678**

Wordlist 2:

**Time for Heap Sort: 0.29129**

**Time for Binary Search: 0.24392**

**Total Time: 0.53521**

**50x50 Grid**

Wordlist 1:

**Time for Heap Sort: 0.005595**

**Time for Binary Search: 0.404397**

**Total Time: 0.409992**

Wordlist 2:

**Time for Heap Sort: 0.281786**

**Time for Binary Search: 0.722414**

**Total Time: 1.0042**

250x250 Grid

Wordlist 1:

**Time for Heap Sort: 0.00455**

**Time for Binary Search: 10.2272**

**Total Time: 10.2317**

Wordlist 2:

**Time for Heap Sort: 0.265946**

**Time for Binary Search: 15.8878**

**Total Time: 16.1537**

**Hash Table**

10x10 Grid

Wordlist 1:

**Time for Hash Sort: 0.002331**

**Time for Search: 0.011665**

**Total Time: 0.013996**

Wordlist 2:

**Time for Hash Sort: 0.080222**

**Time for Search: 0.011342**

**Total Time: 0.091564**

15x15 Grid

Wordlist 1:

**Time for Hash Sort: 0.00248**

**Time for Search: 0.0344**

**Total Time: 0.03688**

Wordlist 2:

**Time for Hash Sort: 0.084537**

**Time for Search: 0.034901**

**Total Time: 0.119438**

30x30 Grid

Wordlist 1:

**Time for Hash Sort: 0.001736**

**Time for Search: 0.196655**

**Total Time: 0.198391**

Wordlist 2:

**Time for Hash Sort: 0.082994**

**Time for Search: 0.207118**

**Total Time: 0.290112**

50x50 Grid

Wordlist 1:

**Time for Hash Sort: 0.001754**

**Time for Search: 0.533166**

**Total Time: 0.53492**

Wordlist 2:

**Time for Hash Sort: 0.076671**

**Time for Search: 0.594067**

**Total Time: 0.670738**

250x250 Grid

Wordlist 1:

**Time for Hash Sort: 0.002211**

**Time for Search: 13.562**

**Total Time: 13.5642**

Wordlist 2:

**Time for Hash Sort: 0.071425**

**Time for Search: 14.8222**

**Total Time: 14.8936**

\*All times are in seconds

Overall, the best performing sorting method appears to be using the hash table. The average runtime for the hash table is Θ(1), so it would be expected to be the fastest. When it came to sorting wordlist 2, the hash table had a sizeable advantage over the other search methods. The second best performing algorithm was quicksort, followed closely by heap sort, and then merge sort. It is not surprising that these three algorithms had very similar performance, because they all have Θ(nlogn) average runtimes. By far the worst performing algorithm was the insertion sort method. While the other algorithms were all sorting the second wordlist in well under 1 second, insertion sort was taking close to 3 minutes. This is also not surprising because it the average runtime for insertion sort is Θ(n2), which would lend itself to very high runtimes with high n values.