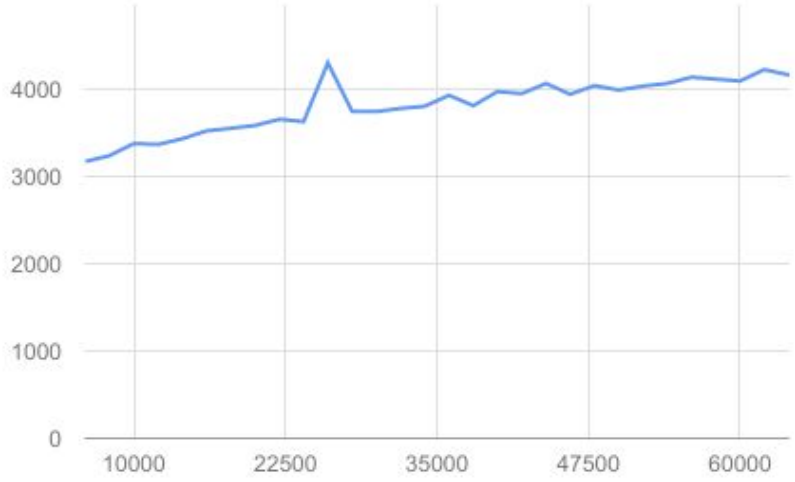
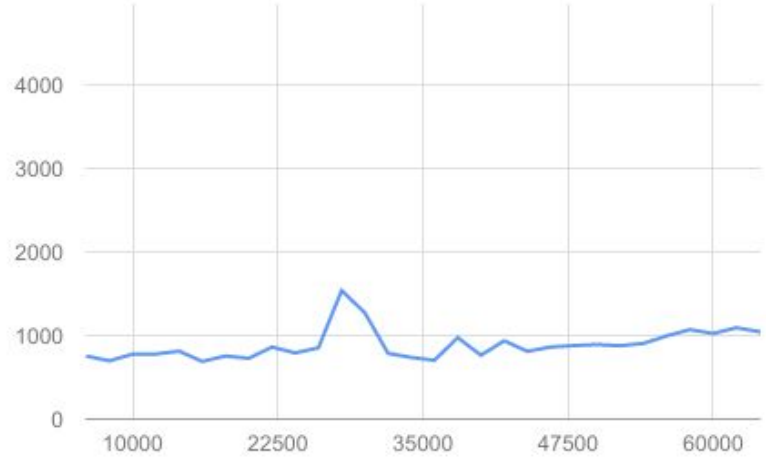
Final Report PA2
Kimiko Yamamoto
A13208241

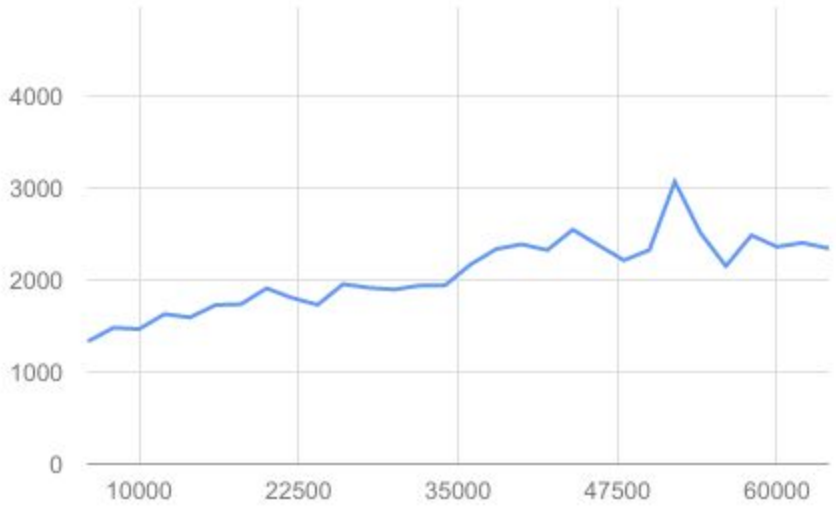
BST



Hash



TST



1) My empirical results seem to be consistent with the analytical running time expectations. In my graphs, the red lines show the running times of the find functions of the different programs. The BST graph (the first one) looked similar to that of a log(n) function. The Hashtable graph (the second one) had a fairly straight horizontal line with 0 slope which is similar to that of an O(n) function. Lastly, the TST graph (the last one) was similar to an O(n) function. This means that I may have used the worst case O(n) for my TST, but this is still an expected running time for a TST.

Benchhash

a) The first hash function finds the hash of a number as the sum of ascii values of each letter in the word mod by the table_size. The second hash function takes the ascii value of the first letter, then multiplies that value by a set number called seed. This product is added to the next letter's ascii value which is multiplied by the seed. The process repeats until all letters of the word are used. Then the hash is the final sum mod by the table_size. I found both of these hash functions on stackoverflow.

b) To verify the hash functions, I ran them in main and also computed the values by hand. In one test case, I found the hash for “tired” with a table size of 25. The hash was 536%25 = 11 for the first hash function and 34400010076%25 = 1 for the second hash function. In the second test case, I found the hash for “so” with a table size of 7. The hash was 226%7 = 2 for the first hash function and 15176%7 = 0 for the second hash function. For the last test case, I found the hash for “hungry” with a table size of 18. The hash was 669%18 = 3 for the first hash function and 4046972487109%18 = 7 for the second hash function.

	hash1	hash2
Freq1.txt 1000	#hits #slots receiving the #hits 0 742 1 160 2 68 3 20 4 8 5 1 7 1	#hits #slots receiving the #hits 0 629 1 276 2 83 3 10 4 2
Freq2.txt 1000	0 736 1 140 2 61 3 28 4 25 5 7 6 1 7 1 8 1	0 619 1 286 2 82 3 10 4 2 5 1
Freq3.txt 1000	0 702 1 123 2 55 3 60 4 27 5 20 6 7 7 3 8 3	0 616 1 298 2 75 3 10 4 1

c) In the above table, I ran benchhash 3 times and got these output.

d) The hash2 was better because there are less collisions. This was expected because the seed will separate words such as “ah” and “ha”.