1) **Give an example of two words that would hash to the same value using stringHash1() but would not using stringHash2().**

word: sent, hash1: 442, hash2: 1111
word: star, hash1: 442, hash2: 1094

2) **Why does the above make stringHash2() superior to stringHash1()?**

Because comparing with stringHash1() the result of stringHash2() is more dispersed. There would not be many keys in the same bucket.

3) **When you run your program on the same input file but one run using stringHash1() and on the other run using stringHash2(). Is it possible for your size() function to return different values?**

No. It is not possible that for two different hash function have impact on size() function because the count of items is fixed.

4) **When you run your program on the same input file using stringHash1() on one run and using stringHash2() on another, is it possible for your tableLoad() function to return different values?**

No, because both hash maps have same number of words and same table size.

5) **When you run your program on the same input file with one run using stringHash1() and the other run using stringHash2(), is it possible for your emptyBuckets() function to return different values?**

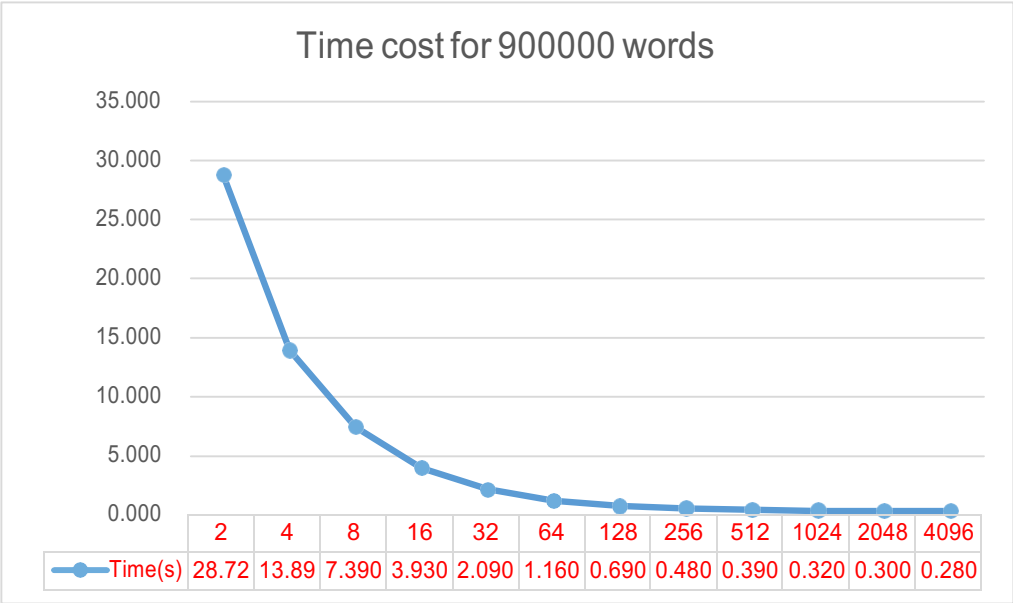
Yes, the result of stringHash2() is more dispersed than stringHash1(), so it would be less empty buckets.

6) **Is there any difference in the number of 'empty buckets' when you change the table size from an even number, like 1000 to a prime like 997?**

Yes, the number of empty buckets will decrease.

7) **Using the timing code provided to you, run you code on different size hash tables. How does changing the hash table size affect your performance?**

Please show results as a graph for various table sizes. For this test, remove the "resize" capability of the table.



	2	4	8	16	32	64	128	256	512	1024	2048	4096
Time(s)	28.72	13.89	7.390	3.930	2.090	1.160	0.690	0.480	0.390	0.320	0.300	0.280