1. StringHash1() simply sums the values of all characters in the input string, so any two words that have the exact collection of letters in them would have the same hash. Therefore, a word such as Rotator would have the same value.

2. StringHash2() includes the index of each character in generating the hash value, making it more resilient against reordering. Also, by offsetting the ASCII value of each character relative to its position in the string, issues with words with the same letters are resolved.

3. Size returns the number of hashlinks, so there shouldn't be a change for the same word set.

4. The tableLoad() function are designed to return the number of hashLinks divided by the number of buckets. The only related factors are that are relevant with tableLoad() are the size of the hash table utilized **and** the number of unique entries in the input file.

5. I believe the answer is, yes. Different hash functions assign input keys to different buckets, so it is possible for two hash functions to have different outputs.

6. Yes, because the table size is used in the modulo operator based on the hash. When changing to a prime number, this decreases the probability of a collision by eliminating the possibility of common factors. In short, we are assigning more objects to a smaller bucket.

7. I believe that in doing this process, testing will show that when we decrease the number of buckets directly correlates to decreased performance of the hashMap. I was right, we see this when we are forced to decrease the number of buckets by 100, this directly lead to more than a 30 times runtime.