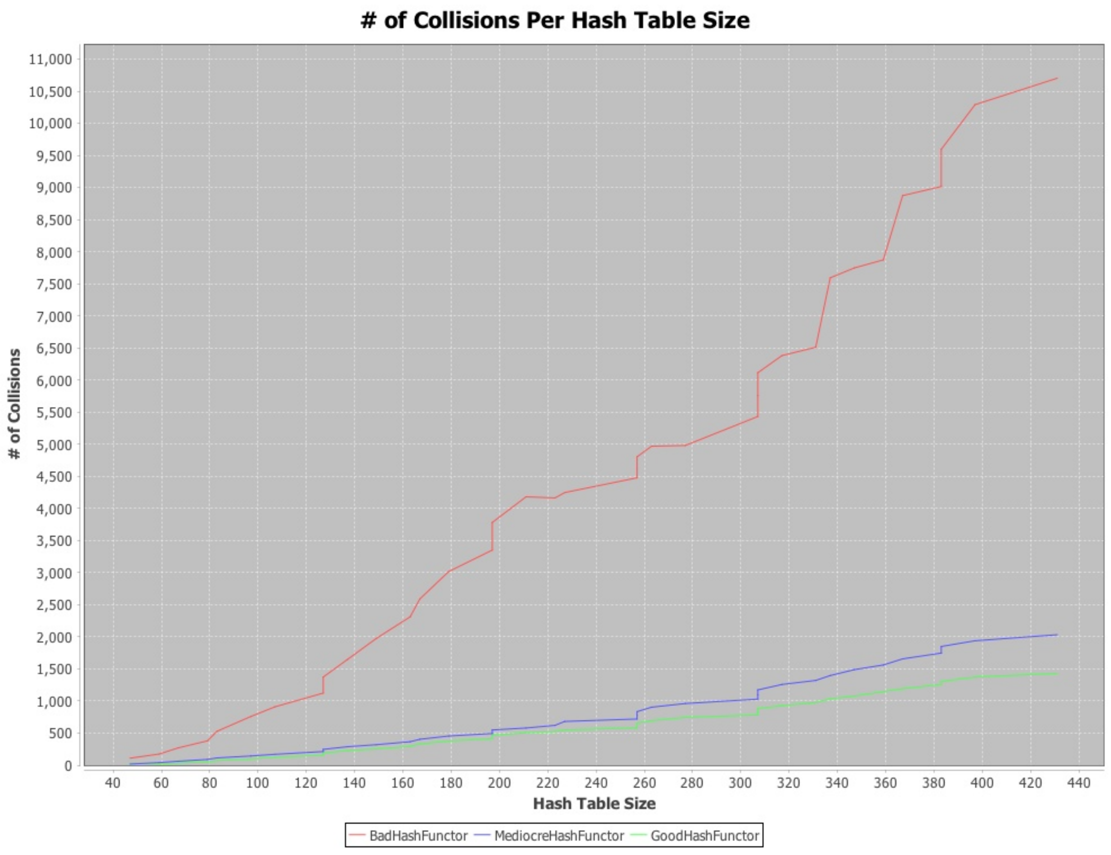
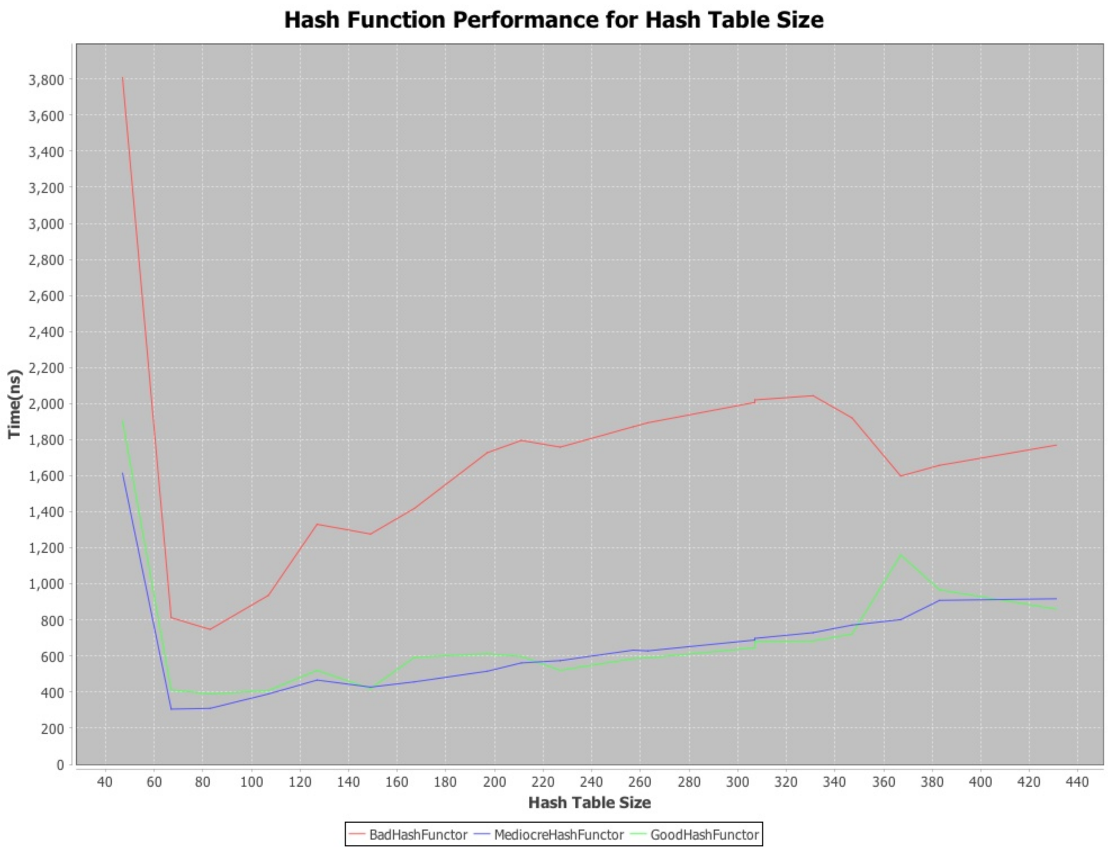
Cole Cruz
u1065058

Assignment10 Analysis

1. The load factor for hash tables is a performance indicator. For quadratic probing the load factor is (# of items in the hash table/ capacity of the hash table) and it is good when it is under 0.5. For separate chaining the load factor is the average length of the chains and it is good when it is a small number (the chains are shorter and easier to navigate).
2. For my bad hashing function I just returned the length of the string provided. I thought this would cause a lot of collisions because if two strings were to be the same length then they would collide.
3. For my mediocre hashing function I returned the sum of the integer values of the first and last characters of the string. I believed this would cause fewer collisions because the first and last characters of the two strings would have to be the same in order to return the same number.
4. For my good hashing function I returned the sum of the integer values of all characters of the string. I thought this would cause little to no collisions since it would be difficult to have the same number returned by the hash function (the strings would have to made up of the same characters).
5. Create a quadratic probing hash table with one of the three hash functions. For a variety of hash table sizes, add a list of strings and keep track of the collisions also track the running time of the add method. Plot the number of collisions compared to the size of the hash table and the running time of the add method for the size of the hash table. Repeat for each hash function.

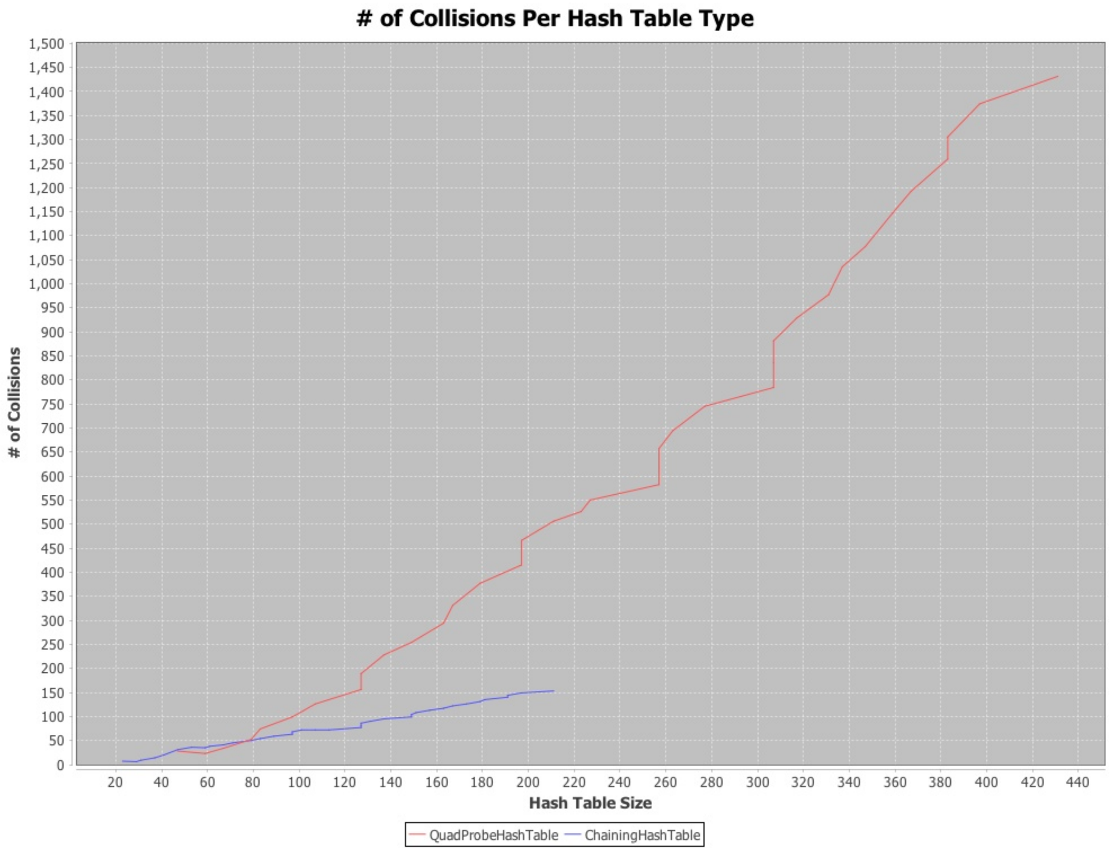
Cole Cruz
u1065058



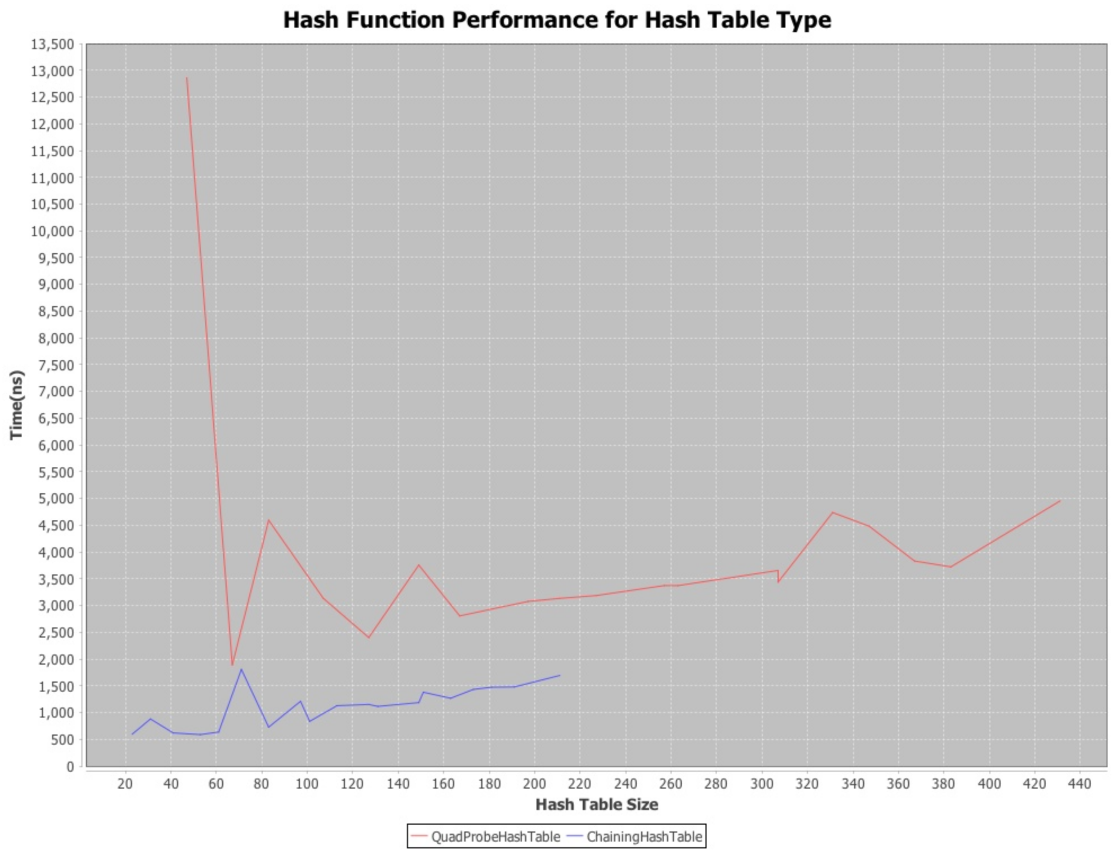


Cole Cruz
u1065058

6. Create a quadratic probing and a chaining hash table with the good hash function. For a variety of hash table sizes, add a list of strings and keep track of the collisions also keep track of how long it takes to add the list. Plot the number of collisions compared to the size of the hash table for each hash table and the running time of the addAll method for the size of the hash table.



Cole Cruz
u1065058



7. The expected Big-O notation for my hash functions are O(c), O(c), and O(N) for the bad, mediocre, and good hash functions respectively. I tested their performance by testing a variety of strings on various capacity hash tables and counting the number of collisions that occurred and found they all performed as expected.
8. The closer the load factor is to 1 the worse the performance of the quadratic probing hash table. This is because there are less open spaces in the hash table for new data to be added and you must go further to find the data you want. The higher the load factor the worse the performance of the separate chaining hash table. It is worse for the same reasons.

Cole Cruz
u1065058

9. Implementing a remove for the chaining hash table would be simple and you could just remove the item from the linked list at the correct index from the hash. On the other hand, a remove for the quadratic probing hash table would require removing all items with the same index from the hash and then adding all but the original item to remove back to the hash table.
10. It would be possible to make the hash tables generic however you'd have to change the hash functions to be able to work on any type and you'd have to provide a comparator for any objects that aren't comparable.
11. I spent a total of 6 hours on this assignment.