**CS303 Lab 9 – HashMaps**

**Problem Specification:**

The problem we are given to solve in this lab is to create the get, put, linear probe, and quadratic probe functions for the HashMap class. The get function, using a specific hashing function, provides the string/value of the key that we input in. The put function will take in a key and value and place it into the HashTable. The linear probe essentially does what the put function does, but it hashes differently to decrease collisions. The quadratic probe has the same concept as the linear probe, but it once again has a different hashing function. Using the UPC data file and the input.dat file, I tested the time it took to put them and get them to and from the table.

**Program Design:**

The put function takes in a key, which I made into a double due to how large they would get and use a specific hashing function that multiplies the original index by 7 adds 1 and then uses the mod function with the size of the table. The get function uses the same hashing equation as the put function to find and bring the user the value given the key. The linear probe does the same thing but hashes differently. Rather than doing the multiplication and adding, the linear probe just adds one and then uses the mod function, so you essentially get the next index. The quadratic probe is similar to the linear, the only difference being that the sequences increase by perfect squares starting at 1.

**Testing Plan:**

To test this, I read in the UPC file and the Input.dat file. Then I split it to two different arrays, one held all of the keys while the other held all of the values. Then depending on the chosen putting method, it would take the keys and values and put it into the HashTable accordingly.

**Text

Description automatically generatedTest Cases:**

**Text

Description automatically generated**

**Text

Description automatically generated**

**Analysis/Conclusion:**

We can see from the above images that it took longer and longer to put the key and values into the table. Something surprising that occurred is that quadratic ended up being faster than the linear probing for this set of keys and values. It is expected that at some point the quadratic would be faster than the linear and at times the linear would be faster than the quadratic. It is just surprising that on this first attempt, the quadratic ended up being faster. The fastest time was for the original put function and that was the fastest because none of the keys and values stored in the array were being searched for, so all the algorithm did was go through the entire table enough times to match however many keys there were.

**References**:

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedThe only references I must make are to the pseudocode and my previous lab in which I used a similar format for writing this report. Below I have added images of all my code for this lab. I apologize but the images are not in exact order of the code for this report.

Text

Description automatically generated

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedText

Description automatically generated