**Coding Assignment 6 Results**

CSE 3318

**Test 1**

**A.** How many rows are in your file/how many cells are in your hash table array?

34, 37

**B.** How many of those cells contained the head of a linked list?

23

**C.** What percentage of the array is being used?

67.647%

**D.** What is the length of the longest linked list?

*3*

**Test 2**

**A.** Did increasing the size of the hash table array give you different results than Test Question 1?

Yes

**B.** Explain why or why not.

Increasing the size of the hash table gave more space for the entries to go to making them more spread out with a lower percentage of the array being used. There was still one LL of length 3 but that’s down from 2 in test 1. 33.784% of the array indices were being used at the doubled size.

**Test 3**

**A.** How many rows are in your file/how many cells are in your hash table array?

34,19

**B.** How many of those cells contained the head of a linked list?

17

**C.** What percentage of the array is being used?

89.473%

**D.** How did decreasing the size of the hash table array affect the percentage of the array that filled?

The percentage went up.

**E.** Did your hash table get any linked lists that were longer than in Test 1? Why or why not?

The longest LL was 4, this is an increase from test 1. This is because there are less indices for hashes to goto causing more collisions.

**Test 4**

**A.** What was your average search time when your HASHTABLESIZE matched the number of records in the file?

1.2 tics

**Test 5**

**A.** What was your average search time when your HASHTABLESIZE was set to 1?

1.3 tics. I know this is supposed to be much higher. This is run on the linux vm since windows was just giving 0, but this isn’t much better. I’m assuming that going to 34 pointers is a very fast operation. No matter what I do the tics is always 1 or 2. If I had a couple thousand entries then I would see a larger difference.

**Test 6**

**A.** What was your average search time when your HASHTABLESIZE was set to 1 and you only searched for the last record of your input file?

1.3 tics. This is supposed to be larger than test 5 but since the computer is solving it so fast, the difference between 1 tic and 2 tics isn’t much.

**B.** Was this average different from your answer to Test 5. If yes, why?

The answer should be different by a factor of about 2 because test 1 on average will be the time for the center of the LL while Test 6 is testing the end of the LL.

**Bonus Question**

If your program was using Open Addressing rather than Separate Chaining, then how many cells of the hash table array would be used when HASHTABLESIZE is set to the number of lines in the file? How did you calculate this number? Show/explain your answer.

If the size of the array was the same as the number of entries then every spot in the array would be used to hold an entry. At first the entries go to their respective hashes but collisions start to happen and these entries go and fill in the remaining spots of the array. The later the entry was put into the hash table, the more likely a collision occurred and will have a higher likelihood of traveling further to reach the next empty spot.