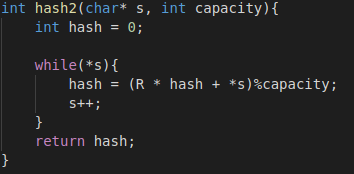
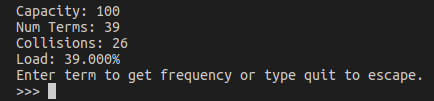
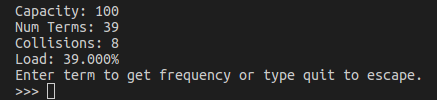
**3D5A Lab5&6**

1. I designed a hash table to store the frequency of names in an ordered list of Irish surnames. I used char arrays as the keys and stored that name and the frequency of that name in that index. I completed this out by writing a structure with those two characteristic and, for simplicity, allocating a fixed amount of memory for an array of this structure. I used linear probing to handle collisions.
2. In the part1, we were given a simple hash function that adds up the values of each character in the string. The problem with is that if two words are anagrams of each other then they will have the same hash value. There is no weight to the different bits. I improved this design and added weight to the different bits by adding R, where R is a small prime integer. This is because the product of a prime number has a good chance of being more unique. I choose 37 instead of 31 because a prime number because I wanted to avoid a prime number that was too close to 2n. As a result, the number of collision decreased from 26 to 8 at a capacity of 100. This is shown in the two collision diagrams below.

Hash2

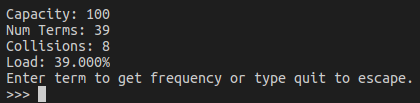
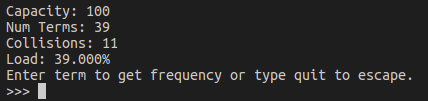


Part 1. Part2.



1. In part2 I wrote a new hashing function, hash1. In part3, I wrote another hashing functions , where M’ is slightly smaller than M and called these two hashing functions using . Similarly, to the previous task the number of collision decreased from 26 to 8 at a capacity of 100, because I replaced linear probing with double hashing. This is shown in the two collision diagrams below.

Part 2. Part 3.



1. I designed a chaining hash table to complete this task by augmented a copy of my solution to the previous task. I created two structures, Surnames and People. The surname structure had two characteristics, a char array for people’s surnames a pointer to the head of a linked list those people. The program filled the hash table by creating an instance of a person, reading in the data about that person and storing that data in the structure. It then calculated the hash of their surname to find the index of that surname in the hash table and add that person to the beginning of the linked list in that index.

References

1. <https://www.ics.uci.edu/~dan/class/165/notes/memory.html>
2. https://stackoverflow.com/questions/2624192/good-hash-function-for-strings
3. https://www.youtube.com/watch?v=jtMwp0FqEcg