Dear TA,

I’m writing to explain my project, the thought process behind, and the steps I took to implement it. My program is designed to take two files, a dictionary file and an input file, and determine the number of misspelled words in the input file based off the dictionary. I implemented this with two different hashing approaches: quadratic probing and chain hashing. I created two different classes to represent each of these unique approaches. Both used a vector to implement the array and had insertion, contain, print, and hashing functions. I chose to use a vector because it came with an automatic resizing function which I thought would be helpful when I would need to rehash. My chain hash did not hold a vector of strings but rather than a vector of Nodes that had a string member variable. Essentially, I had a vector of linked lists in my chain hash. When reading the files in, I handled removing punctuation and capitalization in my main. All strings that were passed into my hash tables were lowercase. To hash them I iterated through every character and found their ASCII values. Before adding the ASCII values to a sum value, I multiplied them by a large prime number(691, really chose that number arbitrarily) and also by two to the power of the character’s index within the string. This avoided clumping and clustering within my code which reduce the need for chaining and probing hence improving efficiency. Both my contains and insertion functions relied on these hash functions to find the initial location of the string and then either chained or hashed depending on which class it’s in. Finally, I decided on a load factor of 0.5 in order to do my rehashing. I also chose to find the smallest prime number larger than twice my previous size of my array in rehash.

\*\*\*\*\*Misspells\*\*\*\*\*

Chain: 136

Quadratic: 136

\*\*\*\*\*Runtimes\*\*\*\*\*

Chain: 1.46

Quadratic: 1.832

Chain Hashing is more efficient.

-Will

╔══╗

╚╗╔╝

╔╝(¯`v´¯)

╚══`.¸.[Computer Science]