README

The indexer was designed around a hashtable to hold keys as the words and values as heaps containing files and frequency nodes. Each insert of a new word should process in about O(1) time given that there is a good hashing function. When a node's frequency is updated, it will take O(log n) to find its new location in the heap through the siftup process. To find a node in the hash table it should be O(1) but we ran into an issue with the implemented hashtable and had to linearly search through the hash table for the apporpriate key, increasing the running time to O(n).

Overall runtime of the program is O(n^2 * log f) where n is the number of unique words and f is the number of files. This is assuming there are many more unique words than files. The n^2 comes from iterating through the hash table linearly n times (for each word). The log f comes from having found a word, each time siftup can be worst case log f runtime.

Error note: Could not support more than 4 files that have the same word. Some issue with the first file of the heap getting screwed up. All the rest o the files in that heap remain alright.