Shehab Ellithy

Peter Brooks

CSS 143 B

Design Document

March 14, 2020

Trie Solution Design Document

To start writing this project, I thought of a trie exactly like a tree. So, I created a trie node class and made it a private static subclass of a trie class. Basically, the trie class is a wrapper for the node class.

First, I defined some constants in the trie class to formalize my code a little bit. The Alphabet size one is simple to understand, 26 for the 26 letters of the English alphabet. Then I defined a beginning and terminating characters. I chose a zero-width space for beginning and the null character for terminating. The final member is the root node containing the beginning character.

The first two methods are wrapper methods for add word and complete in the root node. The char index method just returns the ASCII position of the character to look up in the array. The wrapper methods are for checks before working with the words. I have yet to fully implement them the way I want, but they work for now, so I won’t change anything.

Second, there is the main class which holds all the functionality: trie node. For the node class I choose to create an array of trie nodes as a references to the next letters. You can think of the array as a hashed array and the charIndex() function as the hashing function. Essentially, the array for quick access and editing of the nodes. The algorithm for adding is pretty simple, check if a node exist, if it does just add the word. It is recursive, passing in the same word except the first letter.