Sidharth Bambah

Professor Nachenberg

CS 32

Project 4 Report

1. MyMap

Associate(): If the map holds N elements, the time complexity to insert one new element is logN. This is because it exhibits the same characteristics of a binary search tree.

Find(): If the map holds N elements, the time complexity to find one element is logN. Once again, this is because this implementation follows the same time complexities of a standard binary search tree.

1. AttractionMapper

Init(): For N street segment with an average S attractions each, the time complexity of this method is O ( N \* S \* L \* logN). This is assuming that the average length of the attraction name is L characters and the time complexity to associate into a MyMap element is logN.

GetGeoCoord(): For an average of S attractions, the big oh of finding one attraction is logN \* L. This is because it is essentially finding one element of a MyMap data structure. Furthermore, there are L characters in the attraction name and each character is being converted to lower case.

1. SegmentMapper

Init(): The big oh of initializing a SegmentMapper object with N street segments is O (N \* A \* logN). This is because there are roughly A attractions associated with each street segment and it take roughly logN steps to find and associate each attraction with a StreetSegment.

getSegments(): The big oh, or time complexity, of calling the getSegments() function of a SegmentMapper object with N street segments is merely logN because it is finding the value associated with a key of the MyMap class. As mentioned earlier, the time complexity to find an element in a MyMap is logN.

1. Navigator

Navigate(): The big oh of calling the navigate function of a navigator object that is already loaded using the A\* algorithm is O ( S \* N ). Assuming that there are S successors of each parent node and it takes roughly N time to add each element to the open linked list.