0 Overview

Instructions

Due Date: Tuesday, Nov 5 @ 11:59PM

Total points: 65

Your written solution may be either handwritten and scanned, or typeset. Either way, you must produce a PDF that is legible and displays reasonably on a typical PDF reader. This PDF should be submitted via autolab as WA4. You should view your submission after you upload it to make sure that it is not corrupted or malformed. Submissions that are rotated, upside down, or that do not load will not receive credit. Illegible submissions may also lose credit depending on what can be read. Ensure that your final submission contains all pages.

You are responsible for making sure your submission went through successfully.

Written submissions may be turned in up to one day late for a 50% penalty.

No grace day usage is allowed.

1 Questions

Part 1 - Graph Data Structures

For PA2, the StreetGraph class used to store the maps we were searching was implemented using an EdgeList data structure. The first function you had to implement created an external AdjacencyList that you could use for searching your graph.

1. [**5 points**] Derive the unqualified worst-case runtime (in terms of $|V|$ and $|E|$) to perform a BFS search on a Graph that is implemented using an EdgeList. **Justify your answer. Answers without justification will not receive credit.**

   Note: This is not what you implemented in PA2.

2. [**5 points**] For PA2, to perform a BFS search of the graph, we first computed the AdjacencyList as a HashMap (`get()` has an *expected* $O(1)$ runtime). Suppose that instead, we used a TreeMap (`get()` has an *unqualified* $O(\log(k))$ runtime for a map with k *keys*).

   What is the **total** unqualified worst-case runtime (in terms of $|V|$ and $|E|$) to perform the BFS search using an AdjacencyList stored as a TreeMap. **Justify your answer. Answers without justification will not receive credit.**

3. [**5 points**] Now suppose that, instead of creating an AdjacencyList, we instead sorted the Edge list on the `from` attribute, and stored the resulting sorted list in an `Edge[]` array. That is, we assume that can do binary search on the EdgeList to find the set of outgoing edges for a node.

   What is the **total** unqualified worst-case runtime (in terms of $|V|$ and $|E|$) to perform the BFS search using a sorted Edge array instead of an Adjacency List. **Justify your answer. Answers without justification will not receive credit.**

Part 2 - Pre-Computation

4. [**3 points**] Suppose we are searching an ArrayList of data. If the ArrayList contains n elements, what is the unqualified worst-case runtime in terms of n to:

   a) Find a specific value in the ArrayList?

   b) Sort the ArrayList?

   c) Find a specific value in the sorted ArrayList?

5. [**2 points**] Based on your answers for question 4:

   a) What is the runtime to perform n searches in row on the original ArrayList?

   b) What is the runtime to perform n searches in row on the sorted ArrayList?

6. [**5 points**] Base on your answers to the previous questions, is it worth it to sort the ArrayList first and then perform n searches? **Justify your answer. Answers without justification will not receive credit.**

7. [**5 points**] Now imagine ArrayList contains every song on Spotify, and you want to find 5 specific songs. Is it worth it to sort the list by song title before doing the 5 searches? **Justify your answer, answers without justification will not receive credit.**

Part 3 - Binary Trees

For the following questions, you will be creating various trees using the values in the Array: `A = [20, 23, 32, 46, 57, 9]`

8. [**5 points**] Draw a Min Heap containing the elements in `A` with 4 levels, or state that it is impossible and explain why.

9. [**5 points**] Draw a Max Heap containing the elements in `A` with 3 levels, or state that it is impossible and explain why.

10. [**5 points**] Draw a binary search tree containing the elements in `A` with 4 levels, or state that it is impossible and explain why.

11. [**5 points**] Draw a binary search tree containing the elements in `A` with 7 levels, or state that it is impossible and explain why.

12. [**5 points**] Write the Array that would result from calling Heapify on `A` to create a **Max** Heap. State the exact number of swaps that the algorithm performed (not the asymptotic bound).

Part 4 - Balanced Trees

13. [**10 points**] Draw the tallest possible AVL tree you can containing the values 1-10.

    Hint: Just focus on the structure first, and then fill in values 1-10 after.