Runtimes 1.1 - 1.9 & 2.1 - 2.4

Rubric

+ Correct simplified tight O
- Deductions were given for not specifying or giving the incorrect the bound type.

Hints

- Pay attention to how the input affects the performances of each data structure
- It's important to note that the in-practice performance of a specific data structure in isolation is NOT the same as the in-practice of the same data structure in THIS scenario.

Free Response 1.10

Suppose the order of the students added to the course was completely randomized rather than in registration order. How, if at all, would that impact the performance of each of the above data structure choices?

Rubric

+ Correct analysis of impact on BST asymptotic runtime
+ Correct analysis of impact on AVL asymptotic runtime
+ Correct analysis of impact on Trie asymptotic runtime

Hints

- Consider how the order of insertion in this scenario will impact the structure of each given data structure
- Some students forgot to discuss impact of all three structures

Free Response 1.11

In this given randomized scenario, among the given data structures, select one that is most optimal in terms of runtime and spatial complexity. Justify your decision.

Rubric

+ Selection of most optimal data structure (one officially recognized option)
+ Correct analysis of time complexity
+ Correct analysis of space complexity
+ Partial credit given for correct analysis applied to suboptimal data structure

Hints

- Students should consider the in-practice case for the randomized scenario given.
- Students may briefly compare selected data structure to other options as a method of justification.
- Partial credit was given for discussion of other data structures as a means of error carried forward.

Free Response 1.12

Which data structure is overall best suited for this? Describe how it would work in conjunction with our existing program, as well as how it would be populated and utilized. You may choose any data structure from the course as you see fit.

Rubric

+ Selection of best suited structure (two officially recognized options)
+ Selection enables association of IDs with given name

+ Student discusses how selected structure would be filled with data within the existing program (which existing method(s) would be impacted)

Hints

- Be sure to clearly explain the fields and how the data would be inputted and used within your chosen data structure
- Be sure to discuss how you would incorporate your chosen data structure into the existing design including discussing specific methods

Free Response 1.13

What drawbacks, if any, does implementing `printIDs` with an additional data structure have, in comparison to only using the previous ID -> Name dictionary?

Rubric

+ Student discusses memory usage implications
+ Partial credit given for discussion of drawbacks within their chosen structure from 1.12

Hints

- Discuss the negative consequences of using an additional data structure to implement `printIDs` as opposed to NOT using an additional data structure to implement `printIDs` i.e. the design before the additional structure was added.

Free Responses 2.5 & 2.6

Each of these solutions has tradeoffs. Discuss under what conditions or scenarios either solution might be more effective or efficient than the other and why.

Rubric

+ Describes a scenario where the optimized method is called more frequently
+ Discussion of why data structure used makes method more efficient for this solution
+ Discussion of memory usage optimizations
+ Partial credit given for identifying any method as more runtime optimized (correct or incorrect analysis)
+ Partial credit given for identifying that solution is more effective when a more optimized method is called more frequently (correct or incorrect method optimality analysis)

Hints

- Many students lost points for not being specific about which methods are optimal and in what use cases those methods would be called more frequently
- When justifying optimality be sure to compare runtime of specific methods
- When justifying optimality be sure to discuss how a given method can be optimized due to specific data structure choices
- When justifying optimality be sure to discuss space complexity / memory usage when appropriate
- Please note that “large” or “small” number of players is not a scenario for case analysis, as this puts limits on `n` instead of doing analysis as `n` tends to infinity