CIS 303 Analysis of Algorithms Midterm 2 Spring 2022

You may not use your book, notes, or a calculator for this exam. **Show all work for full credit. Failure to show work will result in loss of credit. Use the back of the page as needed for work space. Please clearly label your work if you use the back of the page.**

Name: ____________________

Scoring Summary

Question	Points	Score
1	8	
2	12	
3	14	
4	10	
5	6	
6	10	
Total:	60	

1. (8 points) **Storage considerations in list and tree implementations.** Show your process/reasoning/work for full credit.

   (a) (4 points) Compute the **break even point** for the **array-based list and the linked list implementation** of lists when sizes of the data field, a pointer, and the array-based list's array are as follows: The data field is 8 bytes, a pointer is 16 bytes, and the array holds 30 elements.

   (b) (4 points) Compute the **overhead fraction** for the **full binary tree implementation** with the following design: Leaf nodes store only data (no pointers); internal nodes store only 2 child pointers (no data). The data field requires 16 bytes and each pointer requires 16 bytes.

2. (12 points) Consider the possible underlying representations for a List ADT (array-based list, linked list). When discussing a linked list representation, you may assume the list has head and tail pointers, but not a "current position" pointer.

   (a) (6 points) For both representations, analyze in detail the time cost of: (1) appending (that is, adding an item to the end of the list), (2) inserting (to an arbitrary position), and (3) searching in an *unsorted* List.

   (b) (6 points) Repeat the analyses in part (a) for a *sorted* List.

3. (14 points) **Which Data Structure?** You must keep track of some data. Your options are:

   - Linked-list (with a tail pointer) maintained in sorted order (the list is always sorted).
   - Linked-list of unsorted records.
   - AVL-balanced binary search tree.
   - Array-based list maintained in sorted order (the list is always sorted).
   - Array-based list of unsorted records.

   For each of the following scenarios, explain:
   (i) which ONE structure you choose and why you chose it, and
   (ii) why you rejected each of the other structures.

   Additional considerations for your solutions:

   - Your answer **must** include both the time and space efficiency of your choice and must give detailed analysis for both.
   - You know all the given constraints as described ahead of time, as given in each problem, including the maximum size of the data set.
   - Assume each insertion is a different data item (*i.e.,* no removals take place and for simplicity you may assume no duplicate data values).
   - You have no information about the distribution or ordering of the data values.
   - Keep in mind that you must consider **all costs** of using the selected structure, including the overhead of maintaining the structure itself. Overhead includes both storage overhead and algorithmic overhead (such as maintaining a sorted list). **Nothing is free of cost.**

- You must use your selected data structure for ALL operations: you may not treat a structure like a different structure based on your knowledge of the data, and you may not run additional algorithms (*e.g.,* sorting) to manipulate the data apart from the operations of the data structure. However, methods may be implemented differently as is appropriate to the particular data structure (*e.g.,* insert is different for a sorted list *vs.* an unsorted list). When such changes are made, you must explain your strategy for the implementation (such as specifying what sorting algorithm you will use on a sorted list).

(a) (7 points) 100,000 insertions are interspersed with 100,000 searches.

(b) (7 points) 100,000 insertions are performed, followed by 1,000,000 searches.

4. (10 points) **Binary Search Trees.**

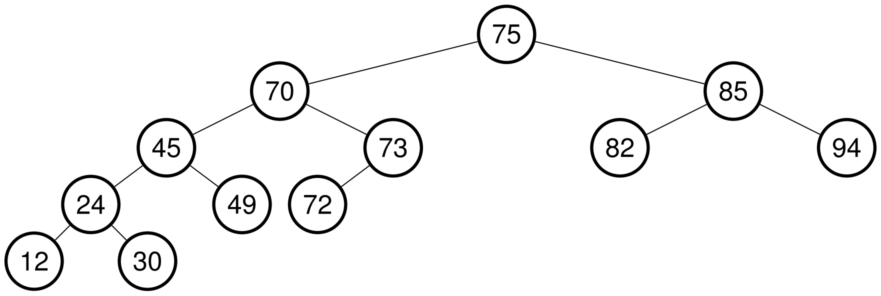

Figure 1: Tree for Problem 4.

(a) (3 points) Draw the BST that results from inserting the value 42 into the BST shown in Figure 1.

(b) (3 points) Draw the BST that results from removing the value 70 from the BST shown in Figure 1. You may start with either the original tree or the tree after inserting the value 42.

(c) (4 points) Write the enumerations of the tree in Figure 1 for a **preorder traversal** and a **postorder traversal**. Use the original tree before you added or deleted a value.

5. (6 points) **AVL trees.** Draw the tree that results from inserting the value 48 into the tree shown in Figure 2, including all necessary rotations. For full credit, show all steps of the process. Your solution will receive no more than 2 points if you provide only the finished tree (if correct).

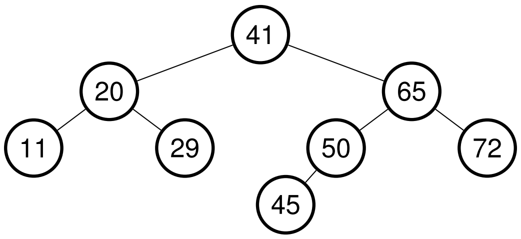

Figure 2: Tree for Problem 5.

6. (10 points) **Heaps.**

(a) (4 points) Describe/define a heap and explain how it differs from a Binary Search Tree. Include considerations of structure and usage in your discussion.

(b) (6 points) Consider the following list: $30, 34, 12, 38, 22, 16, 40$. Show the MaxHeap that results if the the heap is built beginning with the entire list already placed in the heap (in the order shown above) and the heap is built using the `siftdown` algorithm. For full credit, show all the steps required to build the heap. Draw the conceptual heap in each step, not the underlying array. Your solution will receive no more than 2 points if you provide only the finished heap (if correct).