*COMP 410 Fall 2016*

Midterm Exam

This exam is closed book, notes, calculators, cell phones, classmates, smart watches, everything but your own brain. You have 75 minutes to complete the exam. Do all your work on these exam pages. Please sign here (and print your name under it) pledging that the work you submit is your own:

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name (print): \_\_ **ANSWER KEY** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Q1:** Consider a minimum binary heap. In the array used to represent it, at what subscript will we find the parent of the element stored at subscript 43?

1. 42 **b) 21** c) 22 d) 86 e) 87

**Q2:** Consider a minimum binary heap. In the array used to represent it, at what subscript will we find the right child of the element stored at subscript 43?

1. 44 b) 45 c) 21 d) 86 **e) 87**

**Q3:** In an AVL tree, when the longest path and the shortest path differ by 2 or more, we always do a balance operation:

1. True **b) False** c) it’s not that simple

**Q4:** In an AVL tree, the delete operation is the only access operation (insert, delete, contains, findMin, findMax) that does not have a best worst-case time complexity of O(log N).

1. True **b) False** c) it’s not that simple

**Q5:** Which is the best worst-case time complexity for inserting an element by position into a list (linked cells) of N items:

1. O(1) **b) O(N)** c) O(N^2) d) O(log N) e) O(N log N)

**Q6:** Which is the best worst-case time complexity for finding an element by position in a list (linked cells) of N items:

1. O(1) **b) O(N)** c) O(N^2) d) O(log N) e) O(N log N)

**Q7:** Which is the best average-case time complexity for finding an element by value in a list (implemented as an array):

1. O(1/2) b) O(N) **c) O(1/2 N)** d) O(1/2 N^2) e) O(1/2 log N)

**Problems 8-12 apply to this tree**

mud

gem age ton

bet joy won far

pen sim

**Here are your answer choices for questions 8-12:**

1. mud gem age ton bet joy won far pen sim
2. mud gem bet joy pen won age ton far sim
3. bet gem joy pen won mud age ton sim far
4. bet pen joy won gem age sim far ton mud
5. none of the above

**Q8:** which sequence is a pre-order traversal? **B**

**Q9:** which sequence is a post-order traversal? **D**

**Q10:** which sequence is a bottom-up traversal? **E**

**Q11:** which sequence is a breadth-first traversal? **A**

**Q12:** which sequence is an in-order traversal? **C**

**Q13:** Which is the best worst-case time complexity for pushing an element onto a Stack (where the underlying list is linked cells):

1. **O(1)**  b) O(log N) c) O(N) d) O(2\*N) e) O(N^2)

**Q14:** Which is the best worst-case time complexity for removing an element from a Queue (where the underlying list is an array):

1. **O(1)** b) O(log N) c) O(N) d) O(N log N) e) O(2^N)

**Q15:** Consider a TRIE that contains N words, each of length between 1 and K characters. Which is the best worst-case time complexity to find a word stored in this trie:

1. O(N) **b) O(K)** c) O(K\*N) d) O(log N) e) O(log K)

**Q16:** What is the minimum number of nodes that might be in a complete binary tree with height K that is also not a full binary tree:

1. 2^(K+1) -1 b) 2^K -1 c) 2\*K - 1 d) K^2 **e) 2^K**

**Q17:** Which is the best worst-case time complexity for inserting an element into an unbalanced Binary Search Tree of N items:

1. O(1) **b) O(N)**  c) O(N^2) d) O(log N) e) O(N log N)

**Q18:** Which is the best worst-case time complexity for finding an element in an unbalanced Binary Search Tree of N items:

1. O(1) b) O(log N) **c) O(N)** d) O(N log N) e) O(N\*N)

**Q19:** Which is the best worst-case time complexity for deleting an element from an unbalanced Binary Search Tree of N elements:

1. **O(N)** b) O(1/2 N) c) O(1/2 \* (2^N)) d) O(N log N) e) O(log N)

**Q20:** Which is the best worst-case time complexity for getting (not removing) the smallest element in a minimum binary heap:

1. **O(1)** b) O(1/2 N) c) O(N^2) d) O(1/2 N^2) e) O(log N)

**Q21:** Which is the best worst-case time complexity for adding an element into a minimum binary heap:

1. O(N) b) O(1/2 N) c) O(N^2) **d) O(log N)** e) O(1)

**Q22-23:** Consider this array of characters (string):

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | | | | | | | | | | | | | | |

| | **b** | **d** | **f** | **g** | **n** | **j** | **h** | **w** | **k** | **m** | **p** | **r** | | | | |

array: |\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|

1. 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

**Q22:** Is this a valid minimum binary heap? a) YES **b) NO**

**Q23:** Which is the index of the array element that first (left to right) violates the heap order property

1. none violate b) 7 c) 9 **d) 10** e) 11

**Q24:** Which choice is the best Big-Oh worst-case run-time analysis in terms of N for this code fragment:

public static long F (int N) {

if (N == 1) return 1;

return F( N - F(N-1) );

}

1. O(N) b) O(N^2) c) O(N log N) **d) O(2^N)** e) O(1)

**Q25:** Which choice is the best Big-Oh worst case run-time analysis in terms of N for this code fragment:

sum = 0;

for (int i = 0; i < N; i++)

for (int j = 0; j < i; j++)

for (int k = 0; k<N; k++) sum += i\*j\*k;

1. O(N) b) O((1/2) \* N^2) **c) O((1/2) \* N^3)** d) O(2^N) e) O(6 \* N^3)

*COMP 410 Fall 2016, Midterm Exam*

**Name (print): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**The following questions are NOT scantron answered.**

**PRINT your name in the space at top each page.**

**Please write out (clearly) your answers here, on the exam pages under the questions.**

**Q26:** Consider the AVL tree (balanced BST) below. Show its structure after “**insert( 5 )**” is complete. Show your work for partial credit.

**10**

**7 20**

**4 24**

**10 BST-style insert(5)**

**7 20 imbalance at 7**

**4 24**

**5**

**10 balance by rotating**

**5 20**

**4 7 24**

**Name (print): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Q27: Binary Search Tree**

1. Starting with an initially empty Binary Search Tree, show the tree that results after inserting the following values in the order given left to right:

**21, 9, 33, 13, 2, 44, 11, 36**

21

9 33

2 13 44

11 36

1. Starting with the result from (a), show the tree after a delete on 21.

33 *find min in R subtree*

9 44

2 13 36

11 **OR**

13 *find max in L subtree*

9 33

2 11 44

36

**Name (print): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Q28: Quaternary Search Tree (QST)**

Consider a data structure similar to a BST (unbalanced) but with 4 children at each node. We will add elements using rules similar to BST except our rules allow us to decide if a new value belongs down the far-left-child, down the left-mid-child, down the right-mid-child, or down the far-right-child. If we use the QST to store integers (unbounded range) the rules would be:

* far-left: value being added is less than current node value, and it is odd
* left-mid: value being added is less that current node value, and it is even
* right-mid: value being added is greater than current node value, and it is even
* far-right: value being added is greater than current node value, and it is odd

Answer the following questions, explain your answers as needed for partial credit. State any assumptions you make in your answers:

1. Describe the best worst-case time complexity of adding a new element to a QST with N items.
2. Describe the best average-case time complexity of finding an element in a QST with N items.
3. Describe the best worst-case time complexity of an in-order traversal of a QST with N items.
4. Describe the expected depth of a QST with N items in it.
5. Is there an advantage to using a QST over a BST? A disadvantage? Discuss the differences. Should we go to the extra coding effort to use QSTs?
6. **QST can in worst case be linear, one long list like a BST so O(N)**
7. **For average behavior we assume the QST is more or less balanced, with elements distributed down all possible branches. QST may seem like is has 4 branches out from each node (4 children) but if you work out some examples you see only the root has 4 children. After the root level, every other level has at most 2 children. So a QST is really 4 BSTs. We just ignore the root level, and so we are asking the average-case time complexity of finding an item in 4 BSTs with N items total… average N/4 items in each.**

**So it is O( log base 2 (N/4) ) which is O( log N ).**

1. **A traversal has to touch/deal-with every item, there are N items, so it is O(N) (assuming dealing with each item is O(1) activity).**
2. **Related to (b) a QST is really 4 BST’s hanging off a root. If the 4 BST’s are more or less balanced, and the N elements distributed over them, then we expect each of the 4 to have N/4 items in it, and to have a depth O( log (N/4) ) which is O( log N ).**
3. **There really is no advantage to coding up a QST. The coding is more complex, and we see the major operations are the same as for a BST except for a constant factor.**