Practice Midterm Exam

Your Name:

Instructions

Solve each of the following problems to the best of your abilities.

Good luck!

1. Prove that f(n) = 3 log(n) is O(log n). Show your work.
2. Analyze the computational complexity of T(n) = 4T(n/2) + log n using the Master Theorem.
3. The elements 4, 5, 3, 2, 1, 7, 9 are inserted into a min heap in that order. Sketch a diagram of the final state of the min heap.
4. Evaluate the following postfix expression: 1 3 1 4 2 + * + *
5. What is a sentinel node in a linked list? What problem do they solve?
6. Write the pseudocode for a stack data structure which is built on top of a static array with size N. Specifically, you should write the code for push(T x), pop(), and peek(), making sure that you handle the boundary conditions.
7. Consider a naive implementation of the Fibonacci sequence without memoization. Sketch out a recursive trace for the calculation of the fourth Fibonacci number.
8. Suppose I create a two-dimensional array with four rows and eight columns. What is the index of the element in row r = 0 and column c = 3 if the language uses row-major ordering?
9. How does tail call optimization prevent stack overflows in recursive functions?
10. What is an iterator? Why are they useful?
11. What is a comparator / comparable interface? Why is it useful?
12. Draw the binary tree representation of the following arithmetic expression:

    (5 + 2) / (2 + 9) + 7 * 2 - 1

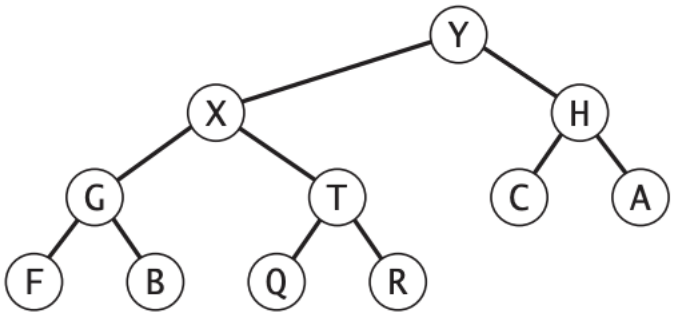
13. Prove the following statement using induction:

$$\sum_{i=1}^{n}(3i-2) = \frac{n(3n-1)}{2}$$

Suppose you have a singly linked list of unknown size that is populated with integers. You have access to the HEAD and TAIL pointers of the list, as well as operations to access and traverse nodes in the list.

1. Write the pseudocode for an algorithm that creates a duplicate of the list. Be sure to handle the edge case when the list is empty.
2. Write the pseudocode for an algorithm that finds the element at the midpoint of the list. Be sure to handle the edge case when the list is empty.

Consider the binary tree shown in the diagram below.



1. Is this a complete binary tree? How do you know?
2. What would I get if I printed out the elements of the binary tree using a post-order, depth first traversal?
3. What would I get if I printed out the elements of the binary tree using a breadth-first search?
4. Write the pseudocode for an algorithm that, given a node in the tree, it swaps the node with its largest child.
5. Suppose I wanted to write an algorithm that deletes all of the nodes in the tree, given its root. Would I want to base my algorithm on a pre-order, post-order, or in-order traversal? Why?