Quiz 1 Practice Questions

Multiple Choice Questions

[M1]

What is the recurrence relation for Merge Sort?

A. $T(n) = T(n-1) + 1$

B. $T(n) = T(n-1) + n$

C. $T(n) = T(n/2) + n$

D. $T(n) = 2T(n/2) + 1$

E. None of the above

[M2]

What is the solution to the recurrence $T(n) = T(n-1) + 1$

A. $O(n)$

B. $O(n^2)$

C. $O(n^3)$

D. $O(2^n)$

E. None of the above

[M3]

What is the lower bound for sorting algorithms based on comparison?

A. $\Omega(n \log n)$

B. $\Theta(n \log n)$

C. $\Omega(n^2)$

D. $\Theta(n^2)$

E. None of the above

[M4]

Suppose you always pick median as the pivot in the QuickSort. What is the recurrence relation for this variant?

A. $T(n) = T(n-1) + 1$

B. $T(n) = T(n-1) + n$

C. $T(n) = T(n/2) + n$

D. $T(n) = 2T(n/2) + 1$

E. None of the above

[M5]

What is the time complexity of Radix sort? Suppose that the array has n elements with d digits each. Also each digit can take values between 1 and k

A. $O(n + k)$

B. $O(n + k + d)$

C. $O(n(k + d))$

D. $O(d(n + k))$

E. None of the above

Algorithmic Questions

Topic : General

[G1]

You are given 9 identical looking balls and are told that one of them weighs a bit less than the rest of the eight balls. The only operation you are allowed is to compare a set of balls against another set of balls. Determine the lighter ball using 3 comparisons. Generalize your answer to more that 9 balls if possible.

[G2]

You are given 12 balls, and are told that one of them is of a different weight from the rest - i.e., you don't know if it is heavier or lighter. Determine this ball using 4 comparisons.

Topic : Asymptotics

[A1]

Solve the following recurrence relations:

A. $T(n) = 2T(n/3) + n^2$

B. $T(n) = 2T(n-1)$

C. $T(n) = T(n/2) + 1$

D. $T(n) = 4T(n/2) + n$

[A2]

Compare the following functions in terms of orders. In each case, say whether $f(n) = O(g(n))$, $f(n) = \Theta(f(n))$ or $f(n) = \Omega(f(n))$

A. $f(n) = \sqrt{n}$ and $g(n) = (\log n)^5$

B. $f(n) = n^2 \log n$ and $g(n) = n(\log n)^2$

C. $f(n) = \log n$ and $g(n) = (\log n)^2$

D. $f(n) = n\,2^n$ and $g(n) = 3^n$

Topic : Order Statistics

[OS1]

The majority of a set of numbers is defined as a number that repeats at least $\frac{n}{2}$ times in the set. Design a linear time algorithm to find the majority, **if** one exists.

[OS2]

Show exactly why if we grouped elements into groups of 3 each, the median finding algorithm that we discussed in class will not work in linear time. What would be the running time of the algorithm in this case?

[OS3]

Which is better in the median-finding algorithm, grouping into groups of 5, or into groups of 7? Explain your answer.

[OS4]

Let X and Y be two arrays of n numbers each, both already sorted. Give a $O(\log n)$ algorithm to compute the median of $X \cup Y$.

Topic : Sorting

Which sorting algorithm would you use for the following cases?

[S1]

The array is almost sorted.

[S2]

The array is sorted in reverse order.

[S3]

Sorting UTA student ids (1000nnnnnn).

[S4]

Sorting mp3 files in your player based on the name.

Hint: *Note: This is a case where you want to minimize number of swaps - each file is, say 5MB. Then doing unnecessary swapping will degrade the SSD of your drive.*

[S5]

Your array is too huge to hold in memory.

[S6]

You are given an array that was randomly shuffled.

[S7]
The size of your array is close to the size of your RAM.

[S8]
Sorting cheques in a bank based on the cheque id.
Hint: *Note that most customers use cheques in the order given in the cheque book.*

Topic : Trees

[G1]
Suppose you are given a completely balanced binary search tree (a completely balanced tree means that each path from root to leaf is exactly the same). What is the time required to find the median of the elements of such a tree ?

[G2]
Suppose you are given a completely balanced binary search tree (a completely balanced tree means that each path from root to leaf is exactly the same). What is the time required to find the max of the elements of such a tree ?

[G3]
Suppose some one gave you a binary tree and claimed that it is a BST. Design and analyze an efficient algorithm to verify if it is indeed the case.

[G4]
Given a BST T and two integers a and b, print all elements in T between a and b.

[G5]
Suppose you are given two trees T_1 and T_2. Design an efficient algorithm to determine if they contain the same elements.

[G6]
Design an algorithm to convert a *sorted* array to a *balanced* binary tree.

Topic : Potpourri

[P1]
The input is a set S containing n numbers, and a number x

(a) Design an algorithm to determine whether there are two elements of S whose sum is exactly x. The algorithm should run in $O(n \log n)$ time.

(b) Suppose now that the set S is given in a sorted order. Design an algorithm to solve the above problem in time $O(n)$.

[P2]
Suppose you are given a sorted array A. Design an efficient algorithm to find if an index i exists such that $A[i] = i$.