CS 101 Fall 2023 - Quiz 4

11/06/2023 - 20 Minutes **Name:** **ID number:**

For the whole Q1 and Q2(a), write your answers in the following table.

For multiple answer questions (Q1(a)-(d) and Q2(a)), please fill in **ALL correct choices**: there may be more than one correct choice, but there is always at least one correct choice. **NO partial credit** on multiple answer questions: the set of all correct answers must be checked.

For True or False questions (Q1(e)-(f)), please fill in "**T**" or "**F**" correspondingly.

1(a)	1(b)	1(c)	1(d)	1(e)	1(f)	2(a)
BC	BC	BC	A	F	F	AC

1. (12 points) Multiple Answers / True or False

(a) (2') Which of the following could be the run-time of applying the randomized quick-sort algorithm (i.e. choose an element from $\{a_l, \cdots, a_r\}$ randomly as the pivot when partitioning the subarray $\langle a_l, \cdots, a_r \rangle$) to sort an array of length n? A. $o(n)$ B. $\Theta(n \log n)$ C. $\Theta(n^2)$

(b) (2') There exists an algorithm that multiplies two n-bit integers in ____ bit-operation complexity. A. $o(n)$ B. $\Theta(n^{\log_2 3})$ C. $\Theta(n^2)$

(c) (2') There exists an algorithm that multiplies two n-by-n matrices in ____ arithmetic operation complexity. A. $o(n^2)$ B. $\Theta(n^{\log_2 7})$ C. $\Theta(n^3)$

(d) (2') If we apply the randomized quick-sort algorithm (i.e. choose an element from $\{a_l, \cdots, a_r\}$ randomly as the pivot when partitioning the subarray $\langle a_l, \cdots, a_r \rangle$) to sort the sequence $\langle 8, 5, 7, 1, 4, 2 \rangle$, then the probability that the element 8 and 2 are compared is ____? A. 0.4 B. 0.5 C. 1

(e) (2') Both the quick-sort algorithm and the merge-sort algorithm are stable sorting algorithms.

(f) (2') Quick-sort is an in-place sorting algorithm, while merge-sort is not since it requires $\Theta(n \log n)$ extra space.

2. (4 points) Recurrence Relation and the Master Theorem

Suppose the run-time $T(n)$ of a divide-and-conquer algorithm satisfies $T(n) = aT(\frac{n}{b}) + f(n)$ with $T(0) = 0$ and $T(1) = 1$.

(a) (2') $f(n)$ is related to the time complexity of <u>fill your answer in the table above</u>?

A. Dividing the original problems into several sub-problems

B. Recurring all sub-problems

C. Merging solutions of sub-problems into the overall one.

(b) (2') Using Master Theorem, solve the close form (i.e. $T(n) = \Theta(g(n))$) of $T(n)$ in the case where $a = 27, b = 3, f(n) = \Theta(n^3)$: <u>$T(n) = \Theta(n^3 \log n)$</u>

3. (9 points) Count β-α-Inversions

Given an array $A = \langle A_1, \cdots, A_n \rangle$, a pair of elements (A_i, A_j) form **a β-α-inversion** ($\beta > 0$) if

$$i < j \text{ and } A_i > \beta \cdot A_j + \alpha$$

For example, a 1-0-inversion is the inversion introduced in our lectures when $\beta = 1$ and $\alpha = 0$.

The following is a piece of pseudocode of an algorithm for counting β-α-inversions, which is modified from the enhanced merge-sort mentioned in lecture. Fill in the blank of the pseudocode below.

Algorithm 1 Counting β-α-Inversions

```
function COUNT-β-α-INVERSIONS-L-R(L = ⟨A_l, ··· , A_mid⟩, R = ⟨A_mid+1, ··· , A_r⟩ )
    cnt ← 0
    i ← l
    for j ∈ {mid + 1, ··· , r} do
        while i ≤ mid  and A_i ≤ β · A_j + α do
            i ← i + 1
        end while
        cnt ← cnt + (mid − i + 1)
    end for
    return cnt
end function
function COUNT-β-α-INVERSIONS(A)
    if n = 1 then
        return A, 0
    end if
    Cut A in the middle and divide it into L_0 and R_0
    L, cnt_L ← COUNT-β-α-INVERSIONS(L_0)
    R, cnt_R ← COUNT-β-α-INVERSIONS(R_0)
    cnt_LR ← COUNT-β-α-INVERSIONS-L-R(L, R)
    A_sorted ← MERGE(L, R)
    return A_sorted, cnt_L + cnt_R + cnt_LR
end function
```