CS 3310 Design and Analysis of Algorithms
Midterm
(3/23/2023)

Name : Loc Nguyen

Last 4 digits of your Student ID #: 2624

Read these instructions before proceeding.

- Closed book. Closed notes. You can use calculator.
- You have 50 minutes to complete this exam.
- *Important Notes:*
  - *During the exam, all students need to join the Zoom meeting*
  - *No questions will be answered during the exam about the exam questions. Write down your assumptions and answer the best that you can. In the last 15 minutes of the exam, if you have additional questions about submitting your exams, you can ask your questions on the Chat room of Zoom Meeting and your questions will be answered there.*
  - *Just in case you have trouble of submitting your exam here @Canvas, alternative way is to submit your completed exam to Prof. Young by emailing*
    qsyoung@cpp.edu
- You can choose your own way to work on your exam, but you are required to submit your completed exam in **ONE PDF FILE** here @ Canvas.
- Two popular ways students used are:
  - Print out the exam paper. Write your answers on the exam paper. Scan your completed exam papers or take photos of them. Then turn in **one PDF file** here @ Canvas.
  - Read the exam from the computer screen and answer questions on your own white papers (number your answers). Scan your exam answers or take photos of them. Then turn in **one PDF file** here @ Canvas.

Q.#1 (24)	Q.#2 (20)	Q.#3 (20)	Q.#4 (18)	Q.#5 (18)	Total(100)

1. (24 pts) (3 pts each)

(a) and (b): Give the ***best-case complexity and the worst-case complexity*** for each of the following algorithms. No justifications are required.

(a) ***Quicksort*** for *Sorting*. Input: a list of *n* numbers.

Best-case Complexity	Worst-case Complexity
O(nlogn)	$O(n^2)$

(b) ***Strassen's Algorithm*** for *Matrix Multiplication*. Input: two n×n matrices.

Best-case Complexity	Worst-case Complexity
$O(n^{2.81})$	$O(n^3)$

(c) f(n) = O (g(n)) if and only if g(n) = _____ (f(n)).

Answer: if and only if $f(n) \le c * g(n)$ where $n \ge n_0$

(d) If f(n) = O (g(n)) and g(n) = Ω (f(n)) then g(n) = _____ (f(n)).

Answer: then, $c_1 * g(n) \le f(n) \le c_2 * g(n)$. Therefore, g(n) is the tight bound of f(n)

(e) If f(*n*) = O(g(*n*)) and f(*n*) = θ(h(*n*)), then h(*n*) = _____ (g(*n*)).

Answer: If g(n) is upper bound of f(n) and h(n) is the tight bound of f(n), h(n) is less than or equal to g(n)

(f) $\sum_{i=1}^{n} a^i =$ $\frac{(a^{n+1} - a)}{(a-1)}$

2. (20 pts)

Rank the following functions by order of growth ***from low to high***. If two or more are of the same order (f(n) and g(n) are in the same class if and only if f(n) = θ(g(n)), group them together. **No proofs are necessary.**

2^n	7×lg n	2^{n+2}	(n-1)!
ln n	n!	lg (lg n)	$3\times n^3 + \lg n$

lg(lg n) 7*lg(n), ln(n) $3*n^3 + lg(n)$ 2^n, 2^{n+2} n!, (n-1)!

low high

3. (20 pts)

Prove that f(n) = O(g(n)) where f(n) = 1234×3^n and g(n) = n^n.
(Hint: Give a witness pair and show that it works)

$n^n \geq 1234 * 3^n$ for all $n \geq 8$. Witness pair is c = 1 and $n_0 = 8$. Substitute these values in, we can get $8^8 \geq 1234 * 3^8$ => 16,777,216 ≥ 8,096,274. Since n^n will grow faster due to increase in both base and exponential power whereas f(n) only increases in exponential power, the upper bound for f(n) is g(n) based on the witness pair of c = 1 and $n_0 = 8$.
Therefore, c = 1 and $n_0 = 8$. So, f(n) = O(g(n)) is true.

4. (18 pts)

Solve the following recurrence relations using ***successive substitutions***, where *c* is a constant.

$$T(n) = \begin{cases} T(n/3) + c & \text{if } n > 1 \\ c & \text{if } n = 1 \end{cases}$$

T(n) = T(n/3) + c
= T(n/9) + c + c = T(n/9) + 2c
= T(n/27) + 3c
= T(n/81) + 4c
.....
= $T(n/(3^p)) + p*c$
When n grows, we can assume $3^p = n$ so p = log(n)
= T(n/n) + log(n)*c
= T(1) + log(n)*c
= c + log(n)*c
= c(1+log(n))
Discarding all constants, T(n) = O(log(n))

5. (18 pts)

Binary search algorithm splits the set into 2 sets of equal sizes (½ and ½). ***"MidtermSearch"*** *algorithm* is a variant of binary search algorithm, and it splits the set into 2 sets of different sizes of one-third (1/3) and two-thirds (2/3) instead.

(a) (9 pts) Give the recurrence relations of ***"MidtermSearch"*** *algorithm.*

The recurrence relations is as follow:
$T(n) = T(n/3) + T(2n/3) + 1$ if $n > 2$ and $T(n) = 1$ if $n = 1$

(b) (9 pts) Give the best-case, worst-case and average-case complexity of ***"MidtermSearch"*** *algorithm*? **No Justifications needed.**

Best-case Complexity	Worst-case Complexity	Average-case Complexity
O(1)	O(lg n)	O(lg n)