CSE Ph.D. Qualifying Exam, Spring 2022
Theory & Algorithms

PRINT NAME: _____

PRINT PSU EMAIL: _____ PSU ID: _____

SIGNATURE: _____

INSTRUCTIONS:

- Do 5 of 6 problems.
- Indicate on the grading table below which problem is NOT to be graded by crossing out the corresponding column in the grade box below, otherwise an arbitrary question will be chosen not to grade.
- No aids allowed.
- Neatly and coherently **justify** your answers in the space allotted.
- Work on the backs of pages will **not** be graded.

QUESTION:	1	2	3	4	5	6	TOTAL
SCORE:							
MAXIMUM:	20	20	20	20	20	20	100

1 Induction

Prove by induction that, $\forall n \geq 1$:

$$1 + \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{3}} + \frac{1}{\sqrt{4}} + ... + \frac{1}{\sqrt{n}} \quad > \quad 2(\sqrt{n+1} - 1).$$

Hint: $\sqrt{2} \approx 1.4$.

2 Recurrence Relations & Asymptotic Growth

Consider a bowl with n noodles, each a strip with two tips. Suppose two of those tips are selected uniformly at random from among the $2n$ tips in the bowl. If the two tips belong to the same noodle, the noodle is removed from the bowl, otherwise the two tips are joined (so merging two noodles) and the (longer) resulting noodle is returned to the bowl. Note that, either way, the number of noodles in the bowl is reduced by one. This random experiment is independently repeated n times until the bowl is empty.

(a) Find a recurrence relation for the mean number of noodles V_n removed from the bowl.

(b) "Solve" the recurrence relation to find a non-recurrent expression for V_n.

(c) Find a simple function $f(n)$ such that the asymptotic rate of growth of $V_n = \Theta(f(n))$.

3 Data Structures

Consider an array of N numbers $[x_1, x_2, ..., x_N]$ that you will be receiving one-by-one in a single pass (i.e., as a stream of numbers). You are not allowed to revisit previous numbers. Design an effective data structure involving one or more heaps that, after receiving the nth number, reports the median of the numbers $x_1, x_2, ..., x_n$ observed so far. The time complexity of your algorithm should be $\mathrm{O}(\log n)$ per number received in the worst case. Write down the pseudocode of your algoritm and analyze its time complexity. For simplicity, assume there are no duplicates in the stream.

Hint: If n is odd, the median of the data stream $[x_1, x_2, ..., x_n]$ is the middle element of the *sorted* data stream, else the median is the average of the middle two elements of the *sorted* data stream.

4 Divide and Conquer

For this problem, a *subtree* of a binary tree means any connected subgraph. A binary tree is *complete* if every internal node has two children, and every leaf has exactly the same depth. Describe and analyze a recursive algorithm to compute the *largest complete subtree* of a given binary tree. Your algorithm should return the root and the depth of this subtree.

5 Greedy

Alice and Bob are playing a computer game. Each of them has an army with m units. They will line up their respective units in two lines, with each of Alice's units facing exactly one of Bob's and vice versa. Then, each pair of units who face each other will fight and the stronger one will win, while the weaker one will be captured. If two opposing units are equally strong, Alice's unit will lose and will be captured. Alice knows how Bob will arrange his units beforehand, and must decide how to line up hers. Your task is to help Alice maximize the sum of the strengths of her units that are not captured during the battle.

You will be given two arrays of positive integers, A and B, each of size n, that specify the strengths of the units of Alice and Bob, respectively. Design an algorithm that computes the maximum total strength of Alice's units that are not captured.

6 Dynamic Programming

Given two strings $x = x_1 x_2 \cdots x_m$ and $y = y_1 y_2 \cdots y_n$, we wish to find the length of their longest common substring. That is, the largest k such that there are indicies i and j such that $x_i \cdots x_{i+k-1} = y_j \cdots y_{j+k-1}$.

(a) Write a dynamic programming algorithm to accomplish this task. Be sure to include all initializations and show that the algorithms runs in time $\mathrm{O}(mn)$.

(b) How might you modify your algorithm so that the space complexity is better than $\mathrm{O}(mn)$? Pseudocode is not necessary, but explain your solution in detail.