CIS 1210 — Data Structures and Algorithms
Homework Assignment 3

Assigned: February 4, 2025 **Due:** February 10, 2025

Note: The homework is due **electronically on Gradescope** on February 10, 2025 by 11:59 pm ET. For late submissions, please refer to the Late Submission Policy on the course webpage. You may submit this assignment up to 2 days late.

A. **Gradescope**: You must select the appropriate pages on Gradescope. Gradescope makes this easy for you: before you submit, it asks you to associate pages with the homework questions. Forgetting to do so will incur a 5% penalty, which cannot be argued against after the fact.

B. **LaTeX**: You must use the LaTeX template provided on the course website, or a 5% penalty will be incurred. Handwritten solutions or solutions not typeset in LaTeX will not be accepted.

C. **Solutions**: Please write concise and clear solutions; you will get only a partial credit for correct solutions that are either unnecessarily long or not clear. Please refer to the Written Homework Guidelines for all the requirements. Piazza will also contain a complete sample solution in a pinned post.

D. **Algorithms**: Whenever you present an algorithm, your answer must include 3 separate sections. Please see Piazza for an example complete solution.

1. A precise description of your algorithm in English. No pseudocode, no code.
2. Proof of correctness of your algorithm
3. Analysis of the running time complexity of your algorithm

E. **Collaboration**: You are allowed to discuss **ideas** for solving homework problems in groups of up to 3 people but *you must write your solutions independently.* Also, you must write on your homework the names of the people with whom you discussed. For more on the collaboration policy, please see the course webpage.

F. **Outside Resources**: Finally, you are not allowed to use *any* material outside of the class notes and the textbook. Any violation of this policy may seriously affect your grade in the class. If you're unsure if something violates our policy, please ask.

Note the following: $\lg n$ means $\log_2 n$.

1. [15 pts] Fly, Eagles Fly!

Krystof and Atharv are huge Birds fans, but they have an ongoing argument on who the *biggest* fan truly is. They need a way to settle this debate once and for all. Hoping to determine the ultimate fan, Krystof proposes a true test of fandom: Who can chant "Fly, Eagles Fly" faster?

Agreeing on this, they each proceed to record the speed at which they chant as functions in terms of n during different games.

For each occasion in parts a and b, help them decide and prove the relationship between the pair of functions with the definitions of $O, o, \Omega, \omega, \Theta$ (please provide the most precise relationship. For example, if both Ω and ω apply, provide your answer in ω). For part c only, prove or disprove the given statement. For any proof, give values of constants for which the definitions hold — **that is, without the use of limits.**

a. $\lg(n)^{\lg(n)},\ n^{\lg(\lg(n))}$

b. $2^{2^n},\ 2^{2^{n+2}}$

c. $2^{7\lg n+\lg\lg n}\lg(n^{10}) = O(8^{3\lg n})$

2. [10 pts] Fastest Run Time Wins!

Nearly everyone knows and loves Saquon Barkley, but few recognize the real hidden gem of the Philadelphia Eagles' offense: Kevin Zhou, a highly skilled backup running back. To settle who deserves the starting spot on Super Bowl Sunday, the two decide to race across the length of the field. Kevin and Saquon's yards ran over time can be modeled by two functions f and g respectively. Kevin and Saquon, having both taken CIS 1210, confidently conclude by the end of the race that $f(n) = O(g(n))$.

However, they're feeling a bit rusty and could really use your help analyzing the following statements. For each of the following statements, decide whether you think it is true or false, and give a brief supporting argument or a counterexample.

a. $20^{f(n)} = O(25^{g(n)})$.

b. $\sqrt{f(n)} = O(\sqrt{g(n)})$.

3. [15 pts] Rig the Big Screens!

Josh has been hired by the NFL to program scripts for the Jumbotrons inside the Caesars Superdrome for Sunday night. As an Eagles fan, he wants to rig them to display Eagles propaganda for as long as possible. To do this, he needs your help analyzing the runtimes of his scripts. Provide him with a tight $\Theta(f(n))$ bound for the runtimes. If this is not possible, provide both the tightest possible upper bound and tightest possible lower bound. Assume that the Jumbotrons can run print statements in constant time.

Don't forget to provide all the steps for your answer!

(a)

```
for (i = 1; i < n; i = i * 3)
    for (j = i; j < n + i; j = j + 1)
        print (" It's a Philly Thing ");
```

(b)

```
for (i = 12; i ≤ n; i = i + 1)
    for (j = 16; j ≤ i^2; j = j + 1)
        for (k = 1; k ≤ |j − i|; k = k + 1)
            print (" Trust the Process ");
```

(c)

```
for (i = 1; i < n; i = i * 4) do
    for (j = 8n; j > i; j = j − 7) do
        print (" E-A-G-L-E-S, Eagles! ");
```

4. [15 pts] Jalen Hurts-Zong's Recurrences

Solve the following recurrences using the method of expansion (iteration), giving your answer in Θ notation. Keep the constants to show your analysis, and then use the Θ notation to express your final answer. You must prove your Θ bound.

a. $T(n) = T(n-1) + 4 \lg n$
Assume that $T(n) = 1$ for $n \leq 1$.

b. $T(n) = \sqrt[4]{n^3}\, T(\sqrt[4]{n}) + n$. Assume $T(n) = 1$, for all $n \leq 4$.

Solve the following recurrence using induction, giving your answer in Θ notation. Note for this problem you only need to show that the Θ-bound holds over the positive integer domain, you do not need to show it holds for all real numbers.

c. $T(n) = T(\lfloor \alpha n \rfloor) + T(\lfloor (1-\alpha) n \rfloor)$ where $0 < \alpha < 1$
Assume that $T(n) = 1$ for $n \leq 1$. .

5. [15 pts] Dylan's Super Bowl Watch Party

Dylan was placed in charge of organizing a Super Bowl watch party for all of the CIS 1210 TAs! He declares two TAs t and t' "buddies" if they wish to watch the Super Bowl together. Note that a TA t can be buddies with multiple other TAs. Additionally, two TAs being buddies does not indicate their relationship with any other TA.

Dylan was originally excited to plan the best super bowl watch party ever! However, as an avid Bills fan, he was devastated to find they did not make it to the super bowl. Extremely bitter, he books two rooms for the party, vowing that no pair of buddies will watch the Super Bowl together, and everyone will be sad. He wants to split the TAs between the two rooms in a way such that the number of pairs of buddies that are split is maximized.

In his efforts to split up the TAs, Dylan proposes the following algorithm:

1. Randomly and independently assign each TA a label of green or red with probability $\frac{1}{2}$ each.
2. Output the group defined by the green/red split.

Let us represent the TAs as vertices of a graph $G = (V, E)$, where two vertices are adjacent if Dylan believes they are buddies. Let us additionally define the random variable X as the number of edges whose endpoints are different colors.

a. Compute $\mathbf{E}[X]$ as a function of $|E| = m$, and deduce that $\mathbf{E}[X] \geq \frac{\text{OPT}}{2}$, where OPT is the maximum/optimal number of separated pairs of buddies possible.

b. Let p be the probability that the number of separated pairs output by the algorithm has size at least $\frac{49}{100}$OPT. Using Markov's inequality show that $p \geq \frac{1}{51}$.
(Markov's inequality states that for any non-negative random variable X and any for any $a > 0$, $\Pr[X \geq a] \leq \mathbf{E}[X]/a$)

c. Compute the variance $\text{Var}[X]$.

d. How would you modify the algorithm so that it *always* finds the number of separated pairs to be at least $\frac{49}{100}$OPT but has only expected linear running time?

6. [15 pts] Amy's Ads

Amy doesn't know much about football, but loves watching Super Bowl commercials and looking up how much they cost. For last year's game between the 49ers and the Chiefs, Amy kept track of the costs of the n commercials played during broadcast in an array *Commercials* of size n. Amy was curious to see how the cost of each commercial compared to the total spent on other commercials, so she devised a method to calculate an array *Cost*, where each *Cost*$[i]$ contains the sum of all entries in *Commercials* except *Commercials*$[i]$. Amy implements the following function, and because she was feeling festive, does not use subtraction.

```
for (i = 1; i ≤ n; i++)
    Add up array entries in Commercials except Commercials[i]
    Store the result in Cost[i]
```

a. Help Amy by giving a bound of the form $\Theta(f(n))$ on the running time of her method on an input of size n. Justify your answer.

b. Give another algorithm to help Amy that has an asymptotically better running time, still without subtraction of values from the *Commercials* array.

For this problem only, you may give us properly formatted pseudocode with your English explanation of your new algorithm. **Psuedocode alone will receive no credit**.
Hint: you may use up to $O(n)$ of extra space

7. [15 pts] Lori's Super Eagles Super Bowl Quintet

Lori, a devoted Philadelphia Eagles fan, is on a mission to discover which of her friends is the most enthusiastic Super Eagles fan ahead of their Super Bowl matchup against the Kansas City Chiefs. She is overjoyed to have secured six tickets to see the game live at the Caesars Superdome in New Orleans on February 9th. However, with so many passionate Eagles fans in her friend group, she must choose which five to accompany her.

To make her decision, she ranks her friends according to the number of Eagles games they have seen in person. She collects each of her n friends' attendance records and randomly arranges them in an

array. She then uses her personalized sorting method: $fiveSort$. $fiveSort(A)$ takes as input an array A of integers. The method $fiveSort(A)$ works by calling $five(0, n, A)$ where n is the length of A and where $five(lo, hi, A)$ is a recursive algorithm. The arguments lo and hi of $five$ delimit the portion of the array A that $five$ sorts, namely $A[lo], A[lo+1], \ldots, A[hi-1]$.

The recursive function $five(lo, hi, A)$ works as follows:

1. If $hi - lo$ is 0 or 1, return. Otherwise go to the next step.
2. If $hi - lo$ is 2, 3, or 4, put $A[lo]$ and $A[hi-1]$ in order (using *quick-sort*) then return. Otherwise go to the next step.
3. Divide the array portion between lo and hi into five (approximately) equal parts. Call *insertion-sort* to order the middle fifth, then recursively calls $five$ on the first, second, fourth, and fifth quin-tiles of the (sub)array.
4. Merge all five sorted parts.

Let $T(n)$ be the worst-case running time for $fiveSort$ on an array of length n (assume that n is an exact power of 5). Write a recurrence relation for $T(n)$. On the right side of the recurrence relation do not include the terms of the form $c\,n^p$ where $p \geq 0$, *except for the term of highest degree.* Explain your answer briefly.

Find the running time of the algorithm. Prove your answer.