Name: Daniel Kim

ID: 10235420

CSCI 3104, Algorithms **Profs. Chen & Grochow**
Final Exam S2–S5 **Spring 2020, CU-Boulder**

Instructions: This quiz is open book and open note. You **may** post clarification questions to Piazza, with the understanding that you may not receive an answer in time and posting does count towards your time limit. Questions posted to Piazza **must be posted as PRIVATE QUESTIONS.** Other use of the internet, including searching for answers or posting to sites like Chegg, is strictly prohibited. Violations of these are grounds to receive a 0 on this quiz. Proofs should be written in **complete sentences. Show and justify all work to receive full credit.**

TIMING: If you are not attempting all the standards in a given quiz, please only use the ordinary amount of time for the number of standards you attempt. For example, if you are only attempting one standard on a 4-standard quiz, please only use 30 min (or 38 for 1.5x, 45 for 2x).

YOU MUST SIGN THE HONOR PLEDGE. Your quiz will otherwise not be graded.
Honor Pledge: On my honor, I have not used any outside resources (other than my notes and book), nor have I given any help to anyone completing this assignment.

Your Name: Daniel Kim

Quicklinks: 2 3 4 5

---

2. **Standard 2.** *Let* $f(n) = 3(n+10)^2$ *and let* $g(n) = n^2$. *Determine which relation* ***best*** *applies:* $f(n) = \mathcal{O}(g(n))$, $f(n) = \Omega(g(n))$, *or* $f(n) = \Theta(g(n))$. **Show all work to receive full credit. This is the only problem on which you (may) need to specify details of calculus problems, such as limits.**

   (BEGIN YOUR ANSWER ON THE NEXT PAGE.)

YOUR ANSWER HERE FOR STANDARD 2. (YOU CAN DELETE ALL THIS TEXT IN CAPS.)

IF YOU ARE HANDWRITING AND INSERTING AN IMAGE, SEE THE COMMENTED CODE BELOW IN THE .TEX FILE. PLEASE BE SURE TO ROTATE YOUR IMAGE TO THE CORRECT ORIENTATION (CAN BE DONE IN THE LATEX DIRECTLY; SEE COMMENTS.)

3. **Standard 3.** *Write down a closed form expression in* Θ *notation for the runtime of the following algorithm.* **Show all of your work**.

```
// Assume A is an n x n matrix of real numbers
function DiagSumSquare(A):
    n = length(A)
    sum = 0.0
    for i = 1 to n {
        sum += A[i][i] * A[i][i]
    }
    return sum
```

(BEGIN YOUR ANSWER ON THE NEXT PAGE.)

YOUR ANSWER HERE FOR STANDARD 3. (YOU CAN DELETE ALL THIS TEXT IN CAPS.)

IF YOU ARE HANDWRITING AND INSERTING AN IMAGE, SEE THE COMMENTED CODE BELOW IN THE .TEX FILE. PLEASE BE SURE TO ROTATE YOUR IMAGE TO THE CORRECT ORIENTATION (CAN BE DONE IN THE LATEX DIRECTLY; SEE COMMENTS.)

4. **Standard 4.** *Write down a closed form expression in* Θ *notation for the the runtime of the following algorithm.* **Show all of your work**.

```
// Assume A is not empty, and length(A) = n
function MaxGap(A):
    maxGapSoFar = 0
    for i = 1 to length(A) {
        for j = i+1 to length(A) {
            gap = A[j] - A[i]
            if (gap > maxGapSoFar) { maxGapSoFar = gap }
    }}
    return maxGapSoFar
```

(BEGIN YOUR ANSWER ON THE NEXT PAGE.)

YOUR ANSWER HERE FOR STANDARD 4. (YOU CAN DELETE ALL THIS TEXT IN CAPS.)

IF YOU ARE HANDWRITING AND INSERTING AN IMAGE, SEE THE COMMENTED CODE BELOW IN THE .TEX FILE. PLEASE BE SURE TO ROTATE YOUR IMAGE TO THE CORRECT ORIENTATION (CAN BE DONE IN THE LATEX DIRECTLY; SEE COMMENTS.)

Since each line is atomic, we need to figure out the number of times each line executed. Starting at line 2, it is executed just one time. In line 3, it will execute n times and since length(A) = n, the numbers in A does not matter. In addition, it won't exit the loop early. For line 4, the loop will go through n - i iterations in the 0-th iteration from line 3. In line 5 and 6, since they're both atomic, they are count as theta(1) in each iterations. To figure out the number of iterations, we need to use this information: i goes from 1 to n and it is n - i in the i-th iteration. Using these information, the total number of steps will be..

$$c \cdot \left((n-1)+(n-2)+(n-3)+\cdots+(n-n)\right) = c \cdot \sum_{i=1}^{n}(n-i)$$

$$= c \cdot \sum_{i=1}^{n} n - c \cdot \sum_{i=1}^{n} i$$

$$= c \cdot n^2 - c \cdot \frac{n(n+1)}{2}$$

$$= c \cdot \left(n^2 - \frac{n(n+1)}{2}\right)$$

$$= c \cdot \left(\frac{2n^2}{2} - \frac{(n^2+n)}{2}\right)$$

$$= c \cdot \left(\frac{n^2}{2} - \frac{n}{2}\right)$$

In line 7, it is executed just one time.
Therefore, we can use the limit test with n^2 n the denominator and conclude with theta(n^2)

5. **Standard 5.** *Consider the following algorithm.*

```
def Fun(A, lo, hi):
       if hi > lo {
           mid1 = floor (lo + (hi-lo)/4)
           mid2 = floor (lo + (hi-lo)/2)
           mid3 = floor (lo + 3(hi-lo)/4)
           for i = lo to mid2 {
                A[i] += 1
                A[lo+hi-i] -= 1
           }
           Fun(A, lo, mid1)
           Fun(A, mid1+1, mid2)
           Fun(A, mid3+1, hi)
       }
       else {
             return 1
       }
```

Write down a recurrence relation describing the number of steps the above algorithm performs on an input of size $n = length(A)$. You do ***not*** *need to solve the recurrence you obtain.*

(BEGIN YOUR ANSWER ON THE NEXT PAGE.)

YOUR ANSWER HERE FOR STANDARD 5. (YOU CAN DELETE ALL THIS TEXT IN CAPS.)

IF YOU ARE HANDWRITING AND INSERTING AN IMAGE, SEE THE COMMENTED CODE BELOW IN THE .TEX FILE. PLEASE BE SURE TO ROTATE YOUR IMAGE TO THE CORRECT ORIENTATION (CAN BE DONE IN THE LATEX DIRECTLY; SEE COMMENTS.)