Midterm 2 **Monday, November 7, 2022: 7-9pm**

Location: Foellinger Auditorium

Name		⇐ Please PRINT

NetID		⇐ Please PRINT

(A) Please print your name and NetID. Anonymous submissions would not be graded.

(B) There are five questions – you should answer all of them.

(C) If you brought anything except your writing implements, your double-sided **handwritten** (in the original, by yourself) $8\frac{1}{2}$" × 11" cheat sheet, and your university ID, please put it away for the duration of the exam. Please turn off and put away *all* medically unnecessary electronic devices.

If you are NOT using a cheat sheet, please indicate so. on this page.

(D) Read all the questions beforehand. Ask for clarification if questions are unclear.

(E) Describing an algorithm requires you to provide:

(i) a detailed description of the algorithm,
(ii) a detailed explanation of why it is correct,
(iii) analysis of its running time, and
(iv) stating the overall running time explicitly.

Failing to provide any of (i), (ii), (iii) or (iv) will result in a loss of points. Providing a pseudo-code is **recommended**. Pseudo-code without explanations is worth 0 points for the whole question.

(F) For all questions asking for a description of an algorithm, you need to provide an algorithm that is **as fast as possible**. Correct and efficient algorithms that are suboptimal would get partial credit. Obvious correct suboptimal naive algorithms, that are still efficient, would get at most 25% of the points. Inefficient algorithms are worth no points. Deficient algorithms are to be avoided.

(G) **This exam lasts 120 minutes.** The clock started when you got the questions.

(H) If you run out of space, use the back of pages – please tell us where to look.

(I) Give complete solutions, not examples. Declare all your variables. If you don't know the answer admit it and move to the next question. We will happily give 0 points for nonsense answers.

(J) **Style counts.** Please use the backs of the pages or the blank pages at the end for scratch work, so that your actual answers are clear.

(K) Please submit this booklet, your cheat sheet, and all scratch paper you used.

(L) ***Good luck!***

---

1 (20 PTS.) Short questions.

1.A. (10 PTS.) Give an asymptotically tight solution to the following recurrence:

$$T(n) = \begin{cases} O(1) & n < 10 \\ 8T(n/2) + O(n^3) & n \geq 10. \end{cases}$$

1.B. (10 PTS.) Given a DAG G with n vertices and m edges, describe an algorithm (see **??** and **??** on cover page), that decides, if there are two distinct vertices x and y, so that there is no path from x to y in G, and there is no path from y to x in G.

2 (20 PTS.) You are given a directed graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ with positive edge lengths, and two vertices s and t, where $n = |\mathsf{V}|$ and $m = |\mathsf{E}|$. For any edge $(u, v) \in \mathsf{E}$, let $\ell(u, v) > 0$ denote its length. An edge $e \in \mathsf{E}$ is ***meh*** if the cost of any walk from s to t that uses e costs at least 2Δ, where Δ is the length of the shortest path in G from s to t. Describe (see **??** and **??** on cover page) an algorithm that computes all the meh edges in G.

3 (20 PTS.) Suppose you are given an array $A[1 \ldots n]$, with n distinct numbers, that is a ***hill***. That is, the values of $A[1 \ldots i]$ are increasing, while the values of $A[i \ldots n]$ are decreasing, for some unknown value i. For example, consider the following array (here $i = 7$).

1	2	3	4	4	6	7	8	9	10
35	65	108	197	303	499	123,833	64	14	9.5

Given a number x, describe (see **??** and **??** on cover page) an algorithm, that decides if x appears somewhere in the array A. Provide a detailed analysis of the running time of your algorithm (i.e., stating the running time is not enough here). Provide pseudo-code for your algorithm.

4 (20 PTS.) Consider a sequence $\widehat{\alpha} \equiv \alpha_1, \ldots, \alpha_n$ of n distinct numbers, and a parameter $\delta > 0$. The sequence $\widehat{\alpha}$ is a ***δ-andén***, if there exists an index i, such that:

(A) For all t, we have $|\alpha_t - \alpha_{t+1}| \leq \delta$.
(B) For all $t < i$, we have $\alpha_t < \alpha_{t+1}$.
(C) For all $t \geq i$, we have $\alpha_t > \alpha_{t+1}$.

(I.e., a δ-andén is a hill where the difference between consecutive values is at most δ.)

The input is an array $A[1 \ldots n]$, and a parameter δ. Describe (see **??** and **??** on cover page) an algorithm that computes the length of the longest subsequence of A that forms a δ-andén. Your algorithm needs to output the number of elements in this subsequence (and not the subsequence itself).

5 (20 PTS.) You are given a directed graph G with n vertices and m edges ($m \geq n$). Describe (see **??** and **??** on cover page) an algorithm that computes the vertex in G that has the *smallest* number of vertices it can reach. Formally, for any $v \in \mathsf{V}(\mathsf{G})$, let

$$\mathrm{rch}(v) = \{x \in \mathsf{V}(\mathsf{G}) \mid \text{there is a path in } \mathsf{G} \text{ from } v \text{ to } x \text{ in } \mathsf{G}\}.$$

The task at hand is to compute and output $\min_{v \in \mathsf{V}(\mathsf{G})} |\mathrm{rch}(v)|$, and a vertex v that realizes this minimum.