Midterm 2 **Monday, November 11, 2024: 7-9pm**

Location: CIF 0027/1025

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) Bring your i-card to the exam, so we can verify your identity.

(M) Do not write outside the framed area, do not remove the staple, and do not remove the last page of the booklet.

(N) ***Good luck!***

1 (20 PTS.) Short questions.

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

$$T(n) = \begin{cases} O(1) & n < 10 \\ T(\lfloor n/2 \rfloor) + T(\lfloor n/3 \rfloor) + T(\lfloor n/6 \rfloor) + O(n) & n \geq 10. \end{cases}$$

1.B. (10 PTS.) You are given a DAG $\mathsf{G} = (\llbracket n \rrbracket, \mathsf{E})$ with n vertices and m edges. The DAG G has the additional property that, for all $u \in \{2, \ldots, n-1\}$, there is a path from 1 to u in G, and a path from u to n in G.

Describe **shortly** an efficient algorithm (see **??** and **??** on cover page), that decides, if there are at least two distinct paths from 1 to n in G. Two paths π_1 and π_2 from 1 to n are *distinct* if they do not use the same set of edges. (Hint: Avoid DP if you can.)

2 (20 PTS.) ONE-WAY OR THE HIGHWAY.

You are given a directed graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$, and two vertices s and t, where $n = |\mathsf{V}|$ and $m = |\mathsf{E}|$. An edge $(u, v) \in \mathsf{E}$ is ***one-way*** edge if there is no path in G from v to u. Describe (see **??** and **??** on cover page) an algorithm that outputs all the one-way edges in G. ***Prove*** formally that the output of the algorithm is correct.

3 (20 pts.) Not a DP question.

You are given an array $A[1 \dots n]$, with $A[i] \in [\![n]\!] = \{1, \dots, n\}$, for all i. It the beginning of the tth round of the game, the player is at location $i_t \in [\![n]\!]$, with the starting location being $i_1 = 1$. The player needs to arrive to location n. During the tth round, the player must move to either $i_t + A[i_t]$ or $i_t - A[i_t]$ – it can choose arbitrarily which one to move to, but the new location i_{t+1} must be valid (i.e., in $[\![n]\!]$).

Describe (see **??** and **??** on cover page) an algorithm that, given k, n and A, decides if there is a sequence of ***exactly*** k moves by the player such that it moves (legally) from 1 to n.

4 (20 PTS.) (HOMEWORK QUESTION.)

Let G be a directed graph with n vertices and m edges, with weights $w(\cdot)$ on the edges (weights on edges can be any real number, including negative numbers). You are also given a start vertex s. Describe (see **??** and **??** on cover page) an algorithm that outputs all the vertices x in G, such that *all* the walks from s to x in G do *not* contain a negative cycle. Explain why your algorithm is correct.

5 (20 PTS.) THE BOTTLENECK (FROM EXTRA HOMEWORK PROBLEMS).

You are given a directed graph G with n vertices and m edges ($m \geq n$), with *real* weights $w(\cdot) \geq 0$ on the edges, and two vertices s and t. (You can assume that $\mathsf{V}(\mathsf{G}) = \{1, \ldots, n\}$, $s \neq t$, and all the weights are distinct.) The ***bottleneck*** of a path π in G is $b(\pi) = \max_{e \in \pi} w(e)$ – the weight of the heaviest edge in π.

Describe (see **??** and **??** on cover page) an algorithm that outputs a path from s to t with minimum bottleneck value. Formally, if Π is the set of all paths in G from s to t, your algorithm should output a path that realizes $\min_{\pi \in \Pi} b(\pi)$. Explain why your algorithm is correct.