Homework 7

The instructions below are the typical instructions I give for homeworks in prior semesters. In the Fall 2025 semester, I decided to do away with homeworks and simply calculate the course grade based on quiz and exam performance. However, I am still releasing these homeworks so students may have some extra study materials. We **are not** collecting/grading homework assignments in the Fall 2025 semester.

Word of advice: before working on these homeworks ask yourself: "Can I do all the lab problems right now (without looking at the solutions)?" If the answer is no, that is where you need to spend your time. If the answer is yes, then feel free to proceed. But again, don't be dumb and look at the solutions first. Lab/homework problems are opportunities to learn and you learn by struggling through them. Looking at a solution and telling yourself "Yeah, I get it" is the best way to do poorly in this course.

- **Every homework problem must be done *individually*.** Each problem needs to be submitted to Gradescope before 6AM of the due data which can be found on the course website: https://ecealgo.com/homeworks.html.
- For nearly every problem, **we have covered all the requisite knowledge required to complete a homework assignment prior to the "assigned" date.** This means that there is no reason *not* to begin a homework assignment as soon as it is assigned. Starting a problem the night before it is due a recipe for failure.

Policies to keep in mind

- **You may use any source at your disposal**—paper, electronic, or human—but you ***must*** cite ***every*** source that you use, and you ***must*** write everything yourself in your own words. See the academic integrity policies on the course web site for more details.
- **Being able to clearly and concisely explain your solution is a part of how you are graded in this course and elsewhere.** Before submitting a solution ask yourself, if you were reading the solution without having seen it before, would you be able to understand it within two minutes? If not, you need to edit. Images and flow-charts are very useful for concisely explain difficult concepts.

See the course web site (https://ecealgo.com/) for more information.

If you have any questions about these policies,
please don't hesitate to ask in class, in office hours, or on Piazza.

Extra Instructions Solutions to a dynamic programming problem have (at minimum) three things:

- A recurrence relation
- A *brief* description of what your recurrence function represents and what each case represents.
- A *brief* description of the memory element/storage and how it's filled in.

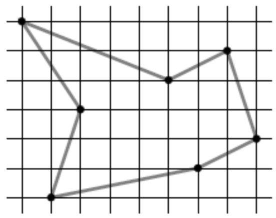
1. Assume we have a list of nuts $N = [n_1 \,..\, n_k]$ and a list of bolts $B = [b_1 \,..\, b_k]$, where each nut and bolt is of unique size. Each nut matches exactly one bolt and vice versa. The nuts and bolts are visually indistinguishable, therefore you cannot directly compare a pair of bolts or a pair of nuts. However, you can compare a bolt to a nut by trying to fit them, at which point you'll find if the pair is too loose, too tight, or perfectly fit. In addition, you are also given an $O(1)$ oracle $M(S)$ which will return the median of S, where $S \subseteq N$. Describe an $O(k \log k)$ algorithm that matches each nut to each bolt.

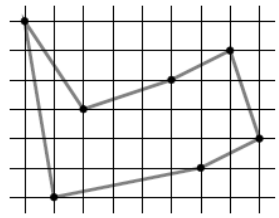
2. There is a group of n dogs labeled from 1 to n where each dog has a different level of loudness and a different level of smartness. You are given an array, denoted as clever, where clever$[i] = [a_i, b_i]$ indicates that a_i is smarter than b_i and an integer array where quite$[i]$ denotes the quietness of the i^{th} dog. You can assume all the input data is correct.

   Describe and analyze an algorithm that will output an integer array where ret$[x] = y$ if y is the least quiet dog among all dogs who have equal or more intelligence than dog x.

3. You are given a directed graph $G = (V, E)$ with positive length edges, as well as two vertices s and t. An edge $e \in E$, is considered bad if the cost of all walks from s to t that uses e costs at least 3β, where β is the length of the shortest path from s to t. Describe an algorithm that computes all the *bad* edges in G. Slower algorithms would earn 60% of the total points.

4. In the euclidean traveling-salesman problem, we are given a set of n points in the plane, and we wish to find the shortest closed tour that connects all n points. The figure above shows the solution to a 7-point problem. The general problem is NP-hard, and its solution is therefore believed to require more than polynomial time (we'll learn more about this in the third part of the course but for right now , this is just a bit of flavor).


(a)


(b)

J. L. Bentley has suggested that we simplify the problem by restricting our attention to bitonic tours, that is, tours that start at the leftmost point, go strictly rightward to the rightmost point, and then go strictly leftward back to the starting point. Figure 15.11(b) shows the shortest bitonic tour of the same 7 points. In this case, a polynomial-time algorithm is possible.

Describe an $O(n^2)$-time algorithm for determining an optimal bitonic tour. You may assume that no two points have the same x-coordinate and that all operations on real numbers take unit time. (Hint: Scan left to right, maintaining optimal possibilities for the two parts of the tour.)