CS161 SU24: Homework 3 (Due July 22 11:00am)

❊ Problem 1

Consider the following **longest increasing subsequence** (LIS) problem.

Problem (LIS)**.**

Input: A sequence $S = (x_0, x_1, \ldots, x_{n-1})$ of n integers.

Outputs: An increasing subsequence $T = (x_{i_1}, x_{i_2}, \ldots, x_{i_k})$ such that $i_j < i_{j+1}$ and $x_{i_j} > x_{i_{j+1}}$.

More explicitly, a subsequence is a choice of elements from a sequence that retains the order. For example, "NESTS" is a subsequence of "U**N**IT**E**D **ST**ATE**S**" by choosing the highlighted elements, but "TENTS" is **not** a subsequence because it does not preserve the ordering.

Given a sequence $S = (x_0, x_1, \ldots, x_{n-1})$ of n integers, we want to design an algorithm to find the length of the **longest increasing subsequence**. That is, we want to find the length of a subsequence $T = (x_{i_1}, x_{i_2}, \ldots, x_{i_k})$ such that $i_j < i_{j+1}$ and $x_{i_j} > x_{i_{j+1}}$.

1. Write down an example instance of the above problem for $n = 8$. State the length of the longest increasing subsequence in your example, and what that subsequence is (You don't have to explain how you found it here).

2. How many possible subsequences are there of a sequence with length n?

3. For each element in the sequence, explicitly state its relation to the final solution. In other words, what are the choices we can make with respect to each element?

4. Design an $O(n^2)$ algorithm to compute the length of the longest increasing subsequence.

5. Modify your algorithm from above so that it runs in $O(n \log n)$ time.

6. Modify your solution so that you can also recover **which** subsequence yields the optimal solution.

※ Problem 2

Suppose you have two credit cards, Frankie's International (FI) and Shion's Capital (SC). FI gives you a $500 refund if the purchase exceeds $1000. SC will instead give you a $10 refund for any purchase of a flower. Let the "happiness index" be a number associated with the amount of joy some object brings you.

Problem.

INPUT: 2 integers representing the limits on each credit card, a list of n items with the items cost, happiness index, and whether or not the item is a flower.

OUTPUT: An array of length n, which in each index i records whether to 1) not buy item i, 2) buy item i with the FI credit card, or 3) buy item i with the SC credit card.

1. Write down an example instance of the above problem for $n = 5$. Write down the answer to your instance as well.
2. For each element in the sequence, explicitly state its relation to the final solution. In other words, what are the choices we can make with respect to each element?
3. Design an algorithm to solve this problem.
4. What is the running time of your algorithm?

❋ Leetcode

Here is a sample of some Leetcode problems related to dynamic programming that you should be able to start attempting now. As mentioned at the beginning of class, set a timer to try solving these on your own, then once the timer is up check a solution and try to understand why that works.

For any solution you write, try to come up with the recurrence relation and solve it to get your final run time.

- Climbing Stairs [Easy]

  `https://leetcode.com/problems/climbing-stairs/description/`

- Ugly Number ii [Medium]

  `https://leetcode.com/problems/ugly-number-ii/description/`

- Perfect Squares [Medium]

  `https://leetcode.com/problems/perfect-squares/description/`

- Jump Game [Medium]

  `https://leetcode.com/problems/jump-game/description/`