# Mini-Homework

## Greedy/Dynamic & Graph

Since we are approaching midterm in Week 9 (Nov 5), this homework is designed with about 50% of regular workload to allow room for midterm preparation.

Please note that this mini-Homework is due by **7:00PM on Monday Oct 28**. The strategy is to make sure you have nothing else to be worry about and can focus on your review the week before your midterm.

Please note that all problems are to be completed **individually**.

So that there is no ambiguity, there are two non-negotiable rules. A violation of either rule constitutes plagiarism and will result in you receiving an F for this course.

**a)** If you meet with a classmate to discuss one of the Individual Problems, the articulation of your thought process (i.e., what you submit), must be an individual activity, done in your own words, away from others. Please remember that the solution-writing process is where so much of your learning will occur in this course: much more than anything we do in class, and even more than the time you spend on solving the problems. Do not be surprised if it takes you 3 to 5 times as long to write up a solution than it takes you to actually solve the problem.

**b)** This Problem Set has been designed to be challenging because struggling through problems is how we learn best. Your educational experience is cheapened by going online and finding the solution to a problem; even using the Internet to look for a "small hint" is unacceptable. In return, your wonderful TAs will be readily available during office hours (drop in or by appointment), and upon request, we will happily post hints to any questions you have on our class Canvas Page.

## Question 1: Happy Question - The Networking Adventure! [6 points]

Hello again! I'm the Networking Adventure, a special opportunity in your journey through the world of computer science and technology. You can earn 6 marks just by stepping out of your comfort zone and into the exciting realm of tech events!
In our fast-paced digital world, it's easy to forget the value of face-to-face interactions and real-world experiences. But remember our discussion about the importance of networking and engaging with the tech community? Well, here's your chance to put that into practice!
Invitation: We invite you to attend any tech event of your choice. It could be a local meetup, a tech conference, a workshop, or even a virtual event. The goal is to expose yourself to new ideas, meet fellow enthusiasts and professionals, and broaden your horizons.

**Task**:

1. Find a tech event that interests you happening either in Oct or Nov 2024.
    - **Hint**: To begin with you can just look at our campus calendar to see what's happening on our campus. Also you don't have to pay for this, many events are free, especially for students!)
2. Register for the event.
3. Attend the event (or at least part of it if it's a multi-day conference), network and build at least 3 connections.

**Submission**: To claim your 6 marks, simply submit the screenshot of your event registration or attendance proof. That's it!

Remember, in the world of technology, sometimes the most valuable algorithms are the ones that connect us with others in our field. Your career is not just about what you know, but also who you know and the experiences you gather along the way.
So go forth, explore, connect, and most importantly, enjoy the adventure of being part of the tech community!

## Question 2: Graph Design [6 points]

**Context:** In computer science and mathematics, we often encounter problems where we need to assign values or group items while satisfying certain conditions. This question explores a specific type of constraint satisfaction problem that can be represented using graph theory.

You are given a set of n variables $\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \ldots, \boldsymbol{x_n}$. These variables could represent anything from people in a social network to nodes in a computer network. Your task is to determine if it's possible to assign values to these variables while satisfying two types of constraints:

- **Equality constraints** ( $\boldsymbol{m_1}$ of them): These are of the form $\boldsymbol{x_i} = \boldsymbol{x_j}$, meaning variables $x_i$ and $x_j$ must be assigned the same value or be in the same group.
- **Inequality constraints** ( $\boldsymbol{m_2}$ of them): These are of the form $\boldsymbol{x_i} \neq \boldsymbol{x_j}$, meaning variables $x_i$ and $x_j$ must be assigned different values or be in different groups.

You are given a set of n variables $\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \ldots, \boldsymbol{x_n}$ and a set of $\boldsymbol{m_1}$ equality constraints of the form $\boldsymbol{x_i} = \boldsymbol{x_j}$ and a set of $\boldsymbol{m_2}$ inequality constraints of the form $\boldsymbol{x_i} \neq \boldsymbol{x_j}$. Is it possible to satisfy all of them?

For example, it is impossible to satisfy the constraints: $\boldsymbol{x_1} = \boldsymbol{x_2}, \boldsymbol{x_2} = \boldsymbol{x_3}, \boldsymbol{x_1} \neq \boldsymbol{x_3}$.

**Q2 [6 points]:**

**a)** **[4 points]** Design an algorithm (via pseudocode and 1 paragraph description (few sentences only)) that takes as input the $\boldsymbol{m_1} + \boldsymbol{m_2}$ constraints and decides whether the constraints can be satisfied.

**b)** **[2 points]** Provide a brief analysis of the running time.

**Clarification:**

- You don't need to find the actual values or groupings; you just need to determine if it's possible to do so.
- The variables don't have a specific range of values. You can think of them as being able to take on any value or belong to any group, as long as the constraints are satisfied.
- A constraint system is considered satisfiable if there exists at least one way to assign values or group the variables that doesn't violate any constraint.

**Additional Examples:**

- Satisfiable example:
  - Variables: $x_1$, $x_2$, $x_3$, $x_4$
  - Constraints: $x_1 = x_2$, $x_3 = x_4$, $x_1 \neq x_3$
  - This is satisfiable because we can group $x_1$ and $x_2$ together, and $x_3$ and $x_4$ together, satisfying all constraints.
- Unsatisfiable example:
  - Variables: $x_1$, $x_2$, $x_3$, $x_4$
  - Constraints: $x_1 = x_2$, $x_2 = x_3$, $x_3 = x_4$, $x_1 \neq x_4$
  - This is unsatisfiable because the equality constraints force all variables to be equal, contradicting the last constraint.

## Question 3: Greedy & Dynamic Programing [6 points]

You work for a small manufacturing company and have recently been placed in charge of shipping items from the factory, where they are produced, to the warehouse, where they are stored. Every day the factory produces $n$ items which we number from **1 to $n$** in the order that they arrive at the loading dock to be shipped out. As the items arrive at the loading dock over the course of the day they must be packaged up into boxes and shipped out. Items are boxed up in contiguous groups according to their arrival order; for example, items **1 .. 6** might be placed in the first box, items **7 .. 10** in the second, and **11 .. 42** in the third.
Items have two attributes, value and weight, and you know in advance the values $v_1 .. v_n$ and weights $w_1 .. w_n$ of the $n$ items. There are two types of shipping options available to you:

**Value-Restricted Boxes:** One of your shipping companies offers insurance on boxes and hence requires that any box shipped through them must contain no more than $V$ units of value. Therefore, if you pack items into such a "value-restricted" box, you can place as much weight in the box as you like, as long as the total value in the box is at most $V$.

**Weight-Restricted Boxes:** Another of your shipping companies lacks the machinery to lift heavy boxes, and hence requires that any box shipped through them must contain no more than $W$ units of weight. Therefore, if you pack items into such a "weight-restricted" box, you can place as much value in the box as you like, as long as the total weight inside the box is at most $W$.

Please assume that every individual item has a value at most $V$ and a weight at most $W$. You may choose different shipping options for different boxes. Your job is to determine the **optimal way to partition the sequence of items into boxes** with specified shipping options, so that shipping costs are minimized.

**Q2.1 [6 points]:** Suppose limited-value and limited-weight boxes each cost $1 to ship.

- **a)** **[2 points]: Briefly describe** a *Greedy Algorithm* that can determine a minimum-cost set of boxes to use for shipping the items
- **b)** **[4 points] Write a pseudocode** for your algorithm and **explain the time complexity** of your solution (don't just say O(…), justify that briefly).

**Note:** Your algorithm should produce an optimal solution, but you do NOT need to prove that!

---

**For Practice ONLY [No Need to submit]:** Suppose limited-value boxes cost $\$C_v$ and limited-weight boxes cost $\$C_w$. Assume you want to design $O(n^2)$ *Dynamic Programing* algorithm that can determine the minimum cost required to ship the $n$ items.

- **Give the formula for the optimal substructure of this problem.** Here is **an example of such formula** for LCS.
- **Write a pseudocode** for your algorithm.

$$c[i,j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0, \\ c[i-1, j-1] + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j, \\ \max\{c[i, j-1], c[i-1, j]\} & \text{if } i, j > 0 \text{ and } x_i \neq y_j. \end{cases}$$

## Question 4: Leetcode on BFS, DFS, MST or Shortest Paths [6 points]

In this homework, you will pick your own choice of only ONE LeetCode problem from the following site (click on "Breadth First Search" or "Depth First Search" or "Minimum Spanning Tree" to filter the results):

- https://leetcode.com/problem-list/breadth-first-search/
- https://leetcode.com/problem-list/depth-first-search/
- https://leetcode.com/problem-list/shortest-path/
- https://leetcode.com/problem-list/minimum-spanning-tree/

**[Read Carefully] This HW offers a lot of flexibility but there are four important rules:**

I. If the problem took you less than 30 – 40 minutes to solve, you may not include it in your portfolio. (Since then the question was an exercise for you, rather than a problem or a challenge.)

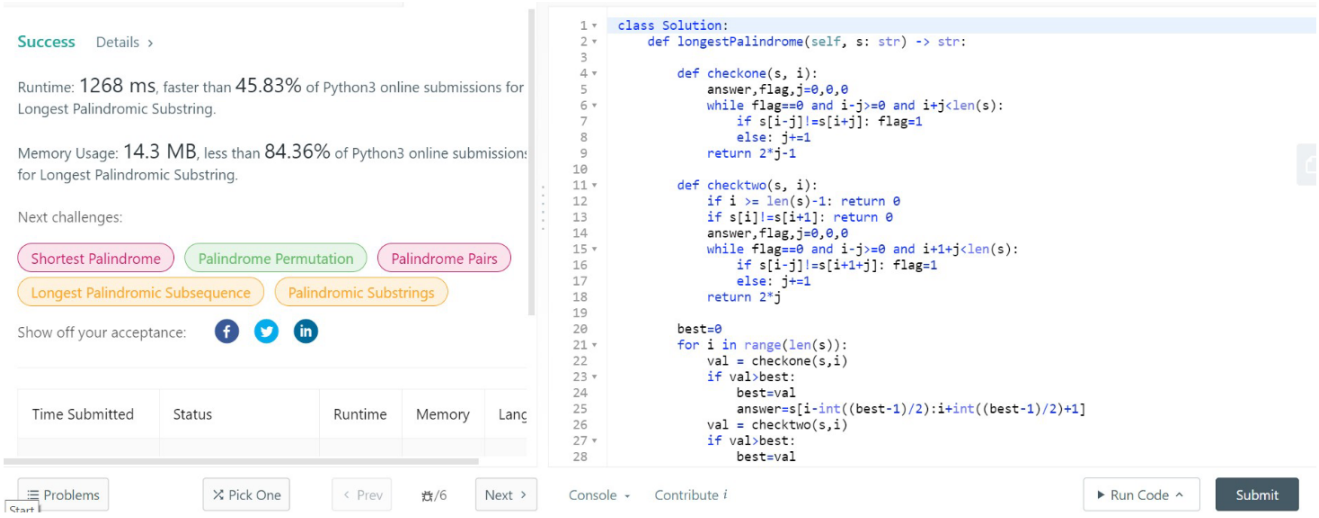
II. You may only include problems you have solved after Oct 16, 2024.

III. (III If you click on the LeetCode "Solution" or "Discuss" button before you solve the problem or look at any online resources for hints to solve a LeetCode problem, especially sites such as Chegg, GitHub, Stack Overflow, and Quora, you cannot include it in your portfolio. (You may look at these sites after you solve the problem.)

IV. Under no circumstance may you use Co-Pilot, LLMs or any other AI-assisted programming tools.

**(a)** **[2 points]:** For the selected problem provide the problem number, problem title, difficulty level, and the screenshot (showing all the details, submission time, etc) of you getting your solution accepted by LeetCode. You will receive up to 4 marks for each problem you submit.

Here is an example from a sample portfolio: Longest Palindromic Substring (medium)




You will get full credit for any correct solution accepted by LeetCode, regardless of the difficulty of the problem, and regardless of how well your runtime and memory usage compares with other LeetCode participants.

**(b)** **[4 points]:** Explain the various ways you tried to solve this problem, telling us what worked and what did not work. Describe what insights you had as you eventually found a correct solution. Reflect on what you learned from struggling on this problem, and describe how the struggle itself was valuable for you. Reflect on what might be causing the obstacle (e.g. lack of familiarity with a particular data structure, lack of knowledge about a fast and efficient algorithm, etc.)

**Note**: For your reflections, write a minimum of 250 words.