

CSE 417 25au Homework 1: Intro/Correctness

Released: Friday, September 26, 2025 @ 11:30am

First due by: Friday, October 3, 2025 @ 11:59pm

Last resubmissions by: Wednesday, October 15, 2025 @ 11:59pm

Instructions

For Problems 1 and 2, you have four options for submission:

- **Film a video in which you explain your solution.** See the [Homework Guide](#) for more details. A whiteboard is not a good format for Problem 1. Consider writing on paper or screen recording instead.
- **Use LaTeX to type your solutions.** A template is provided in the “Tasks” page of the course website, if you like.
- **Use Google Docs or Microsoft Word to type your solutions.** If doing so, please use the Equation Editor to ensure that any equations are legible and easy to read.
- **Handwrite your solutions on paper or digitally.** Please write neatly and if on paper, scan in black/white mode, not grayscale.

We prefer either video or LaTeX, but accept any of the 4 options.

A few more reminders:

- **Submit all problems on Canvas**, and programming assignment 1X.1 will redirect to **Gradescope**. Each problem should have its own submission. *Do not* submit one large file containing answers to several problems.
- **Suggested word counts/video length are rough guidelines.** We won’t actually count, but if your writing is verbose to the point of obscuring your main argument, we may ask you resubmit more concisely.
- **Review the [collaboration policy](#) in the syllabus.** Collaboration is encouraged but strict rules apply, and remember to cite your collaborators.
- If you don’t finish in time, we encourage you to be honest and just upload what you have so far. Resubmission won’t cost you anything, and we can give you timely feedback on your partial progress by submitting on time.

Happy problem solving!

Problem 1: Grading ChatGPT

The purpose of this problem is to dissuade you from overreliance on large language models for logical reasoning, and to practice concepts about stable matchings.

We posed a question about stable matchings to ChatGPT, using “Think longer” mode, which (as of September 2025) uses the GPT-5 Thinking Mini model. The question was:

Given a matching between proposers and receivers that matches $p \leftrightarrow r$ and $p' \leftrightarrow r'$, “swapping” these two matches means creating a new matching that matches $p \leftrightarrow r'$ and $p' \leftrightarrow r$, but otherwise is identical to the original. Give an instance of the stable matching problem with at least 3 people on each side and a stable matching, such that after swapping any pair of matches, the matching remains stable, or prove that such an instance cannot exist.

You can find the transcript of the conversation attached to the end of this PDF. Read the conversation and identify all places where ChatGPT made mistakes, such as faulty or unjustified logic. Point out precisely and explain each instance of such logical errors. If there are false claims, also explain why the claim is false. (suggested 100–300 words or 3–6 minutes video) (If choosing video, a whiteboard is not a good format for this problem. Consider writing on paper or screen recording instead.)

Problem 1X: Grading ChatGPT (Extensions)

This is an extension problem that builds on the ideas of Problem 1. **Pick one** of the following to complete:

1. The problem that we gave to ChatGPT is false, that is, with at least 3 people on each side, there does not exist an instance of the stable matching problem and a particular stable matching that always remains stable after swapping any two matches.

This might be surprising, but in order to prove this, we only need to consider stable matching instances with *exactly* 3 people on each side: If a larger instance existed with the stable-after-swapping property, pick any 3 matched pairs and delete everyone other than these 6 people from the preference lists. This results in a new instance with 3 people on each side, and it has the stable-after-swapping property, essentially because the original instance has the property. (That was a proof sketch—you are encouraged to think through the details yourself, or come discuss in office hours!)

In this problem, you will write a Java program that uses an exhaustive search among all instances of stable matching with exactly 3 people on each side to prove that no stable matching in any of these instances has the stable-after-swapping property. The provided starter code will have some parts of this already, leaving you to implement a couple of functions. It is available both on the course website and Canvas. Submit your code in the assignment on Canvas (which redirects to Gradescope) or Gradescope directly, where it will be autograded. Do not use libraries beyond the standard Java API.

2. Review all 7 questions in [this survey about university students' perspectives on ChatGPT](#), collected by Generation Lab in July 2025. Then synthesize your thoughts on this survey with reflections on your own experience solving Problem 1 and join the discussion on Canvas by responding to the following prompt, raising other questions of your own choosing, or replying directly to other students' responses. (suggested 150–300 words)

Large language models (LLMs) can be used for exploration, editing, reasoning, and many other purposes. What are the benefits and drawbacks of involving LLMs specifically in reasoning tasks, including short-term and long-term effects? For you personally, in what contexts, if any, would you refrain from using LLMs for reasoning? In what contexts, if any, do the benefits outweigh the drawbacks?

You are *not* required to cite the survey or any other sources, but please support your response with evidence, which may be anecdotal. Disagreeing with the course staff or course policies will not be penalized. **Resubmissions will not be available for this option.**

Problem 2: Business profit

The purpose of this problem is to practice complete proofs of simple algorithms, formally verifying all aspects of an algorithm's correctness.

You are given an array of integers $A[1 \dots n]$ representing the daily profit or loss of a company. Your task is to determine if the company's cumulative change in money was ever positive (an overall profit), and if so, output the index of the first day where this happened.

More formally, if $A[1] + \dots + A[k] > 0$ and $A[1] + \dots + A[j] \leq 0$ for all $j \leq k-1$, you should output that the first profit occurred on day k . On the other hand, if $A[1] + \dots + A[j] \leq 0$ for all j , output "never".

For example, if $A = [-3, 2, -1, 4, 2]$, you would say that the company first had an overall profit at the end of day 4, since $-3 + 2 - 1 + 4 = 2$, which is positive, and this never happened before day 4.

(This problem statement used 1-indexing. Feel free to use either 1-indexing or 0-indexing in your solution, but make this clear. Keep n as the length of the array.)

1. Write a simple Java program that solves the above problem. Since this part is mostly for your own benefit, we will not run your code. Paste the source code in your written submission, or in the comments box if recording a video.
2. Translate your Java program into pseudocode intended for human reading, as if you were describing the algorithm on a whiteboard to a friend who is currently taking the equivalent of CSE 12X at another university that may not necessarily use Java. Aim to efficiently communicate what you are doing and why, not the particular details of Java syntax.
3. Provide a full proof of correctness for your algorithm, following the structure covered in class. If you make a claim that is a loop invariant, make sure to say that it is an invariant and briefly outline why it stays true. (target 100–300 words)

(Video submissions are suggested to take 4–8 minutes total.)

Given a matching between proposers and receivers that matches $p \leftrightarrow r$ and $p' \leftrightarrow r'$, "swapping" these two matches means creating a new matching that matches $p \leftrightarrow r'$ and $p' \leftrightarrow r$, but otherwise is identical to the original. Give an instance of the stable matching problem with at least 3 people on each side and a stable matching, such that after swapping any pair of matches, the matching remains stable, or prove that such an instance cannot exist.



Thought for 37s

Short answer: **Such an instance exists.**

Below I give a concrete 3×3 example and a short general proof that the construction works for any $n \geq 2$ (so in particular for "at least 3").

Concrete 3×3 instance

Matching $M = \{(p_1, r_1), (p_2, r_2), (p_3, r_3)\}$. Preferences (strict):

Proposers:

- $p_1 : r_1 \succ r_2 \succ r_3$
- $p_2 : r_2 \succ r_1 \succ r_3$
- $p_3 : r_3 \succ r_1 \succ r_2$

Receivers:

- $r_1 : p_2 \succ p_3 \succ p_1$ (here p_1 is last)
- $r_2 : p_3 \succ p_1 \succ p_2$ (here p_2 is last)
- $r_3 : p_1 \succ p_2 \succ p_3$ (here p_3 is last)

Check: M is stable because no proposer prefers some other receiver to his matched one (every proposer has his matched receiver first), so there cannot be a mutual blocking pair.

Now pick any two matched pairs and swap them (for example swap (p_1, r_1) and (p_2, r_2) to get $(p_1, r_2), (p_2, r_1), (p_3, r_3)$). In the swapped matching any proposer who

was not involved still has his top choice, so he will not prefer anyone else. Any proposer involved in the swap prefers his original partner to his new partner, but the corresponding original receiver prefers her new partner over that proposer (because every receiver ranks her matched proposer last), so the two cannot form a blocking pair. Thus the swapped matching is stable. The same reasoning works for any choice of two pairs.

General construction and proof (works for any $n \geq 2$)

Let proposers be p_1, \dots, p_n , receivers r_1, \dots, r_n , and take the matching $M = \{(p_i, r_i) : i = 1, \dots, n\}$. Make every proposer p_i rank r_i as his top choice, and make every receiver r_i rank p_i as her **worst** choice (the rest of each agent's ordering may be arbitrary, subject to strictness).

1. M is stable: no proposer wants to leave his matched receiver (each already has his top choice), so there are no blocking pairs.
2. Fix any two indices $i \neq j$ and swap those two matches to get M' . Any proposer p_k with $k \notin \{i, j\}$ still holds his top choice and thus will not participate in a blocking pair. For p_i (now matched to r_j) the only receiver he could prefer to his current is r_i , but r_i prefers p_j (her new partner) to p_i (since p_i was placed last), so (p_i, r_i) is not a blocking pair; similarly for p_j . No other pair can be mutually preferable. Hence M' is stable.

Therefore the requested instance exists (the concrete 3×3 example above), and the simple general recipe gives infinitely many such instances for every $n \geq 2$.