Welcome to CS 256:

Algorithm Design and Analysis

Quick Logistics

Zoom lecture:

- Stay muted when not talking
- Make sure your name shows up (rename your ID)
- If you face any issues, let me know (worst case lecture is being recorded and will be posted on GLOW)

Recording disclaimer:

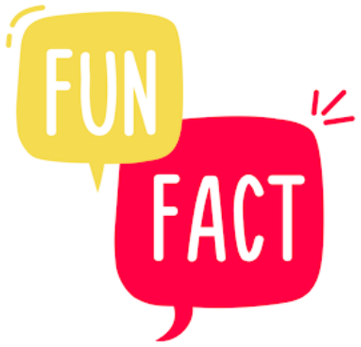
- All lectures will be recorded and posted on GLOW
- If you do not want your face/voice shown, you should disable video and ask questions via chat

About Me

Instructor: Shikha Singh (she/her/hers)

- Pronunciation: “Shi” is like in ship not sheep

- Reach me:
    - Email: shikha@cs.williams.edu
    - Course Slack !
- Zoom help hours:
    - Mon 2.30-4 pm, Tues 3-5 pm, Wed 1.30-3 pm



Learning and COVID

To make this course successful:

- Will have to work together
- Communication is key: if you any concerns or challenges please reach out to me

Participate in a way that you are comfortable with:

- Okay to join via Zoom if not feeling well
- If you prefer Zoom over in person, please talk to me

My goal is to support you during these difficult times

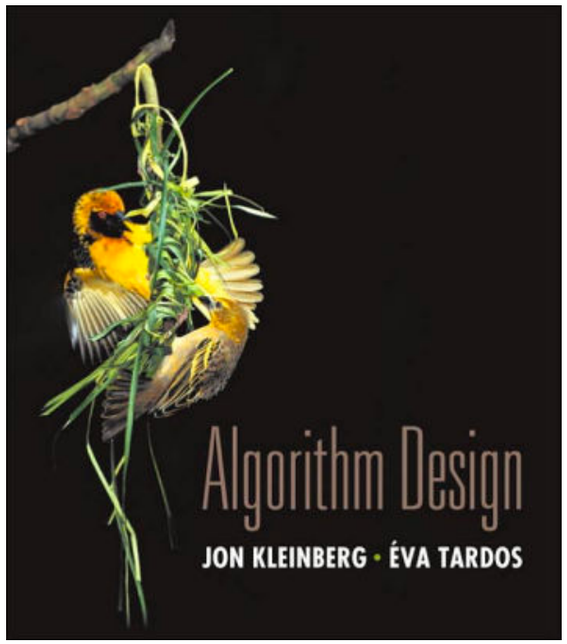
Hybrid Course

- In-person: **TCL 217A** | remote: via Zoom
- TCL 217A isn't the most spacious room (as you'll see)
- To facilitate the Zoom lecture:
    - Set up camera and try to stay in one place
    - Can only use a restricted area on the board
- Some desks might not have the best view of board
    - Okay to bring laptops and join the zoom call
- Students on Zoom cannot hear in-person students
    - Will repeat all questions/answer, but remind me if I forget

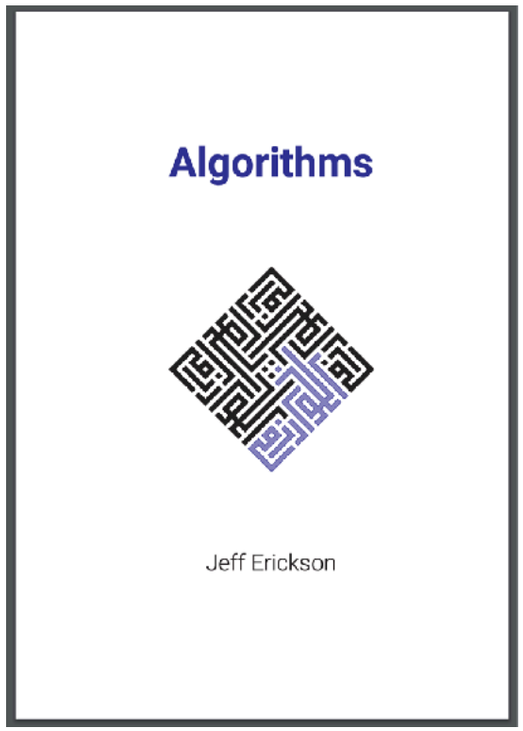
Questions?

Course Logistics

Textbooks:



Three copies reserved in the Schow library for reference



Available online at http://jeffe.cs.illinois.edu/teaching/algorithms/

Slides: Kleinberg and Tardos book has excellent slides for reference that I'll also be borrowing a lot from.

Course Technologies

slack

zoom

GLOW

Overleaf

LaTeX

gradescope

TA Team & Hours

- All TA hours will be held over Zoom
- You can find the schedule and link on the course calendar

Intro to CS256 Activities

Welcome to CS256! To get started, here are some start of semester activities for you:

- *Get oriented*: read and familiarize yourself with the course syllabus and policies
- *Keep on top of course schedule*: add the course google calendar to your calendar
- *Introduce yourself*:
  - Fill out the course introduction survey by Friday Feb 19, 5 pm
  - Sign up for a short Zoom hello (put your name against any slot that doesn't have two names already)
- *Set up course technology*:
  - Join the CS256-S21 course Slack using this invitation link
  - To submit assignments: join Gradescope using the course code **74XDKB**
  - To typeset HW in LaTeX: create an Overleaf account or install LaTeX on your machine

Course Logistics

Course website: https://williams-cs.github.io/cs256-s21/

Grading breakup:

- Weekly problem sets (45%)
- Midterm (20%)
    - Take-home 24 hour exam (~2nd April)
- Final (30%)
    - Take-home 24 hour exam (final exam period)
- Class participation (5%), includes attendance*

*Missing class if you are feeling ill is not only acceptable, but encouraged

About Class Participation

- I like interaction in my classes!
- Many ways to participate:
  - Ask & answer questions in class or on Slack
  - All Slack participation counts towards participation
  - Come to office hours
- Attendance policy:
  - Required but flexible: safety is the top-most priority!
- Class participation does not mean dominating classroom discussions or interrupting your peers

> **Bottom line:** *Help create a vibrant, positive and inclusive classroom environment!*

About Problem Sets

- Must be typeset in LaTeX **using template provided**
- Anonymized grading: No name/ID on homework
- PDF must be submitted via Gradescope
  - Join using course code **74XDKB**
    - **IMPORTANT.** **Assign ques to pages** of the PDF
- Assignments will usually be released on Wed and due the **following Wed at 11 pm**
- Assignment 0 is out! Due **Wed 02/24**

Demos on GLOW

- Go to GLOW course -> Course media gallery
- Demos:
    - how to use Overleaf/ LaTeX template
    - how to submit on Gradescope (assign pages, etc)

Late Days & Late Work

- Each student will get a total of **3 late days**
- Can use 1 late day for any problem set
- Gives you a no-questions asked 24 hour extension (automatic on Gradescope—no need to email me)
- Use them wisely!
- After late days have been used up, any late work will be penalized 20% per day
    - After 24 hours, need to email me your work
- Late work may be graded late as well

Academic Honesty Policies

- Read collaboration policies on course webpage
- Gist:
  - Collaboration is encouraged but *you should never submit a solution that you do not understand*
  - External sources okay for background study *but cannot search for problem-specific keywords*
  - Always cite your sources and collaborators
    - Cite sources/collaborators in the last section labeled "Acknowledgements" in template
    - Do not miss this part!

Academic Honesty Policies

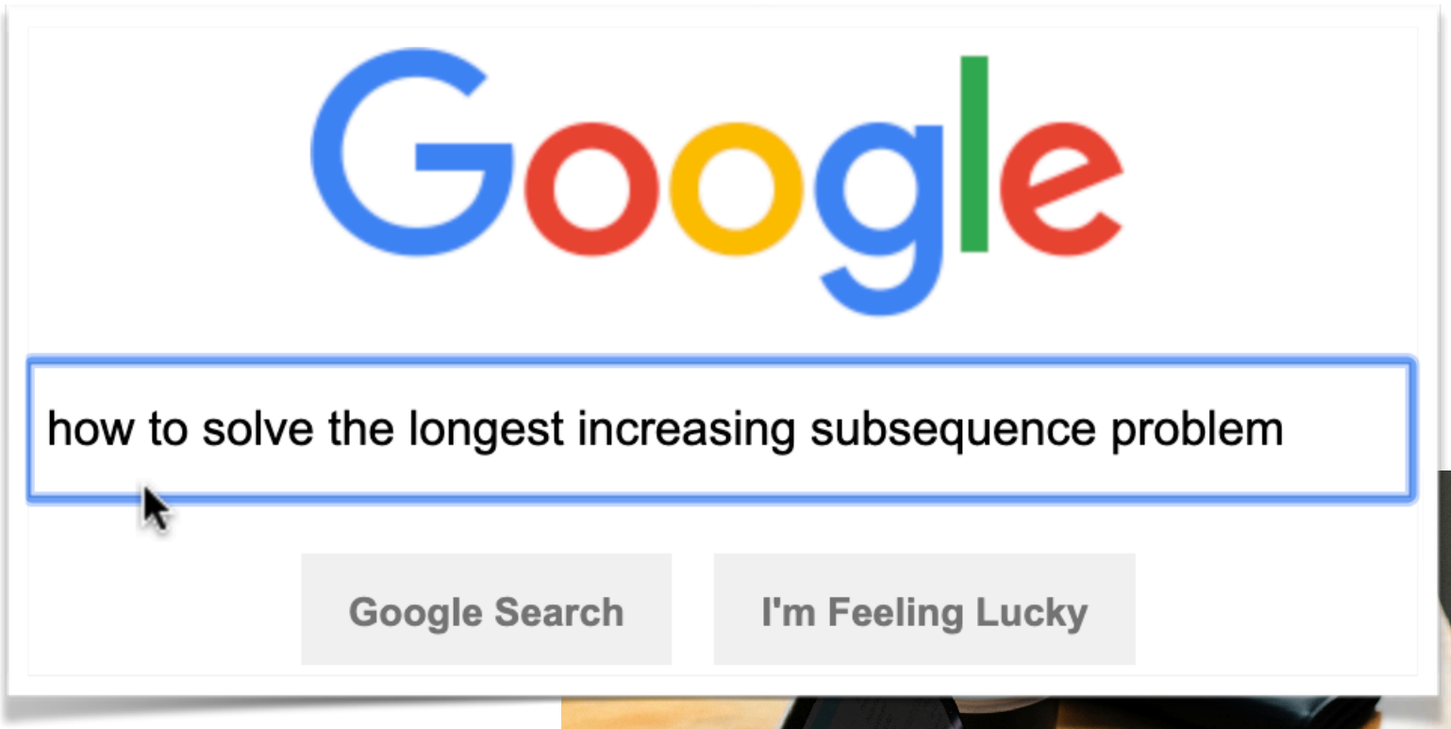
**I didn't full understand dynamic programming in class…
These MIT notes online look good, maybe I will read them to prepare for the assignment**

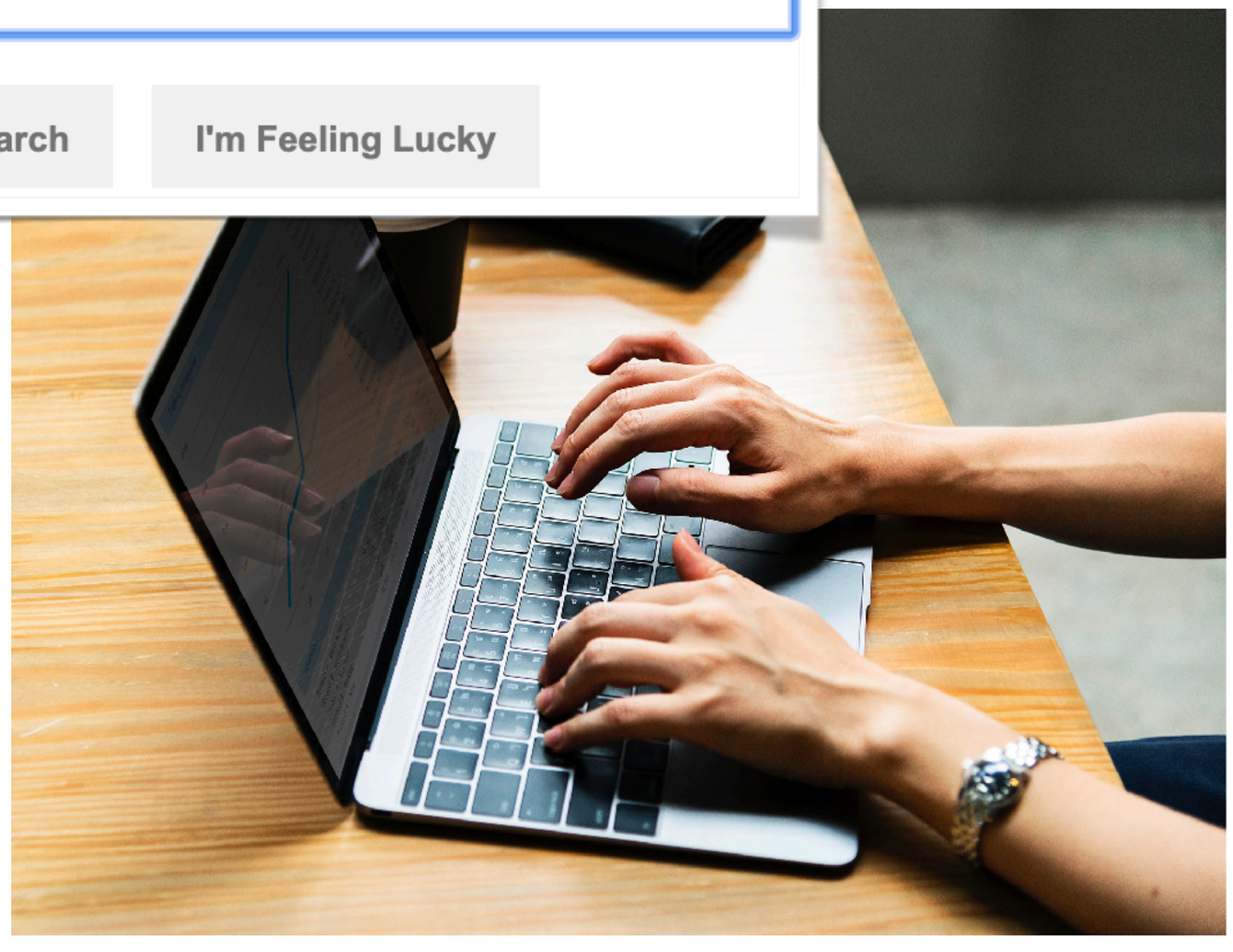
Academic Honesty Policies

**I didn't full understand dynamic programming in class…
These MIT notes online look good, maybe I will read them to prepare for the assignment**

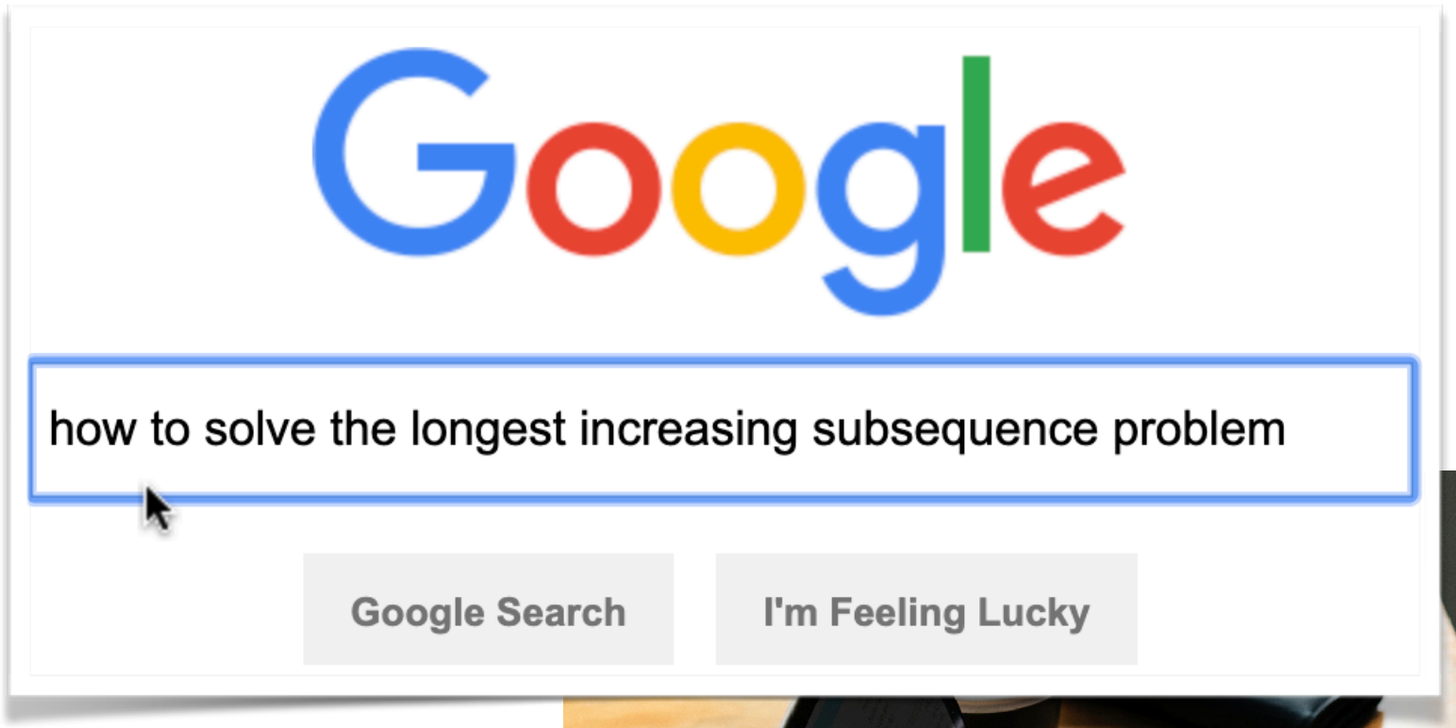
**This Is OK!
(if you cite)**

Academic Honesty Policies

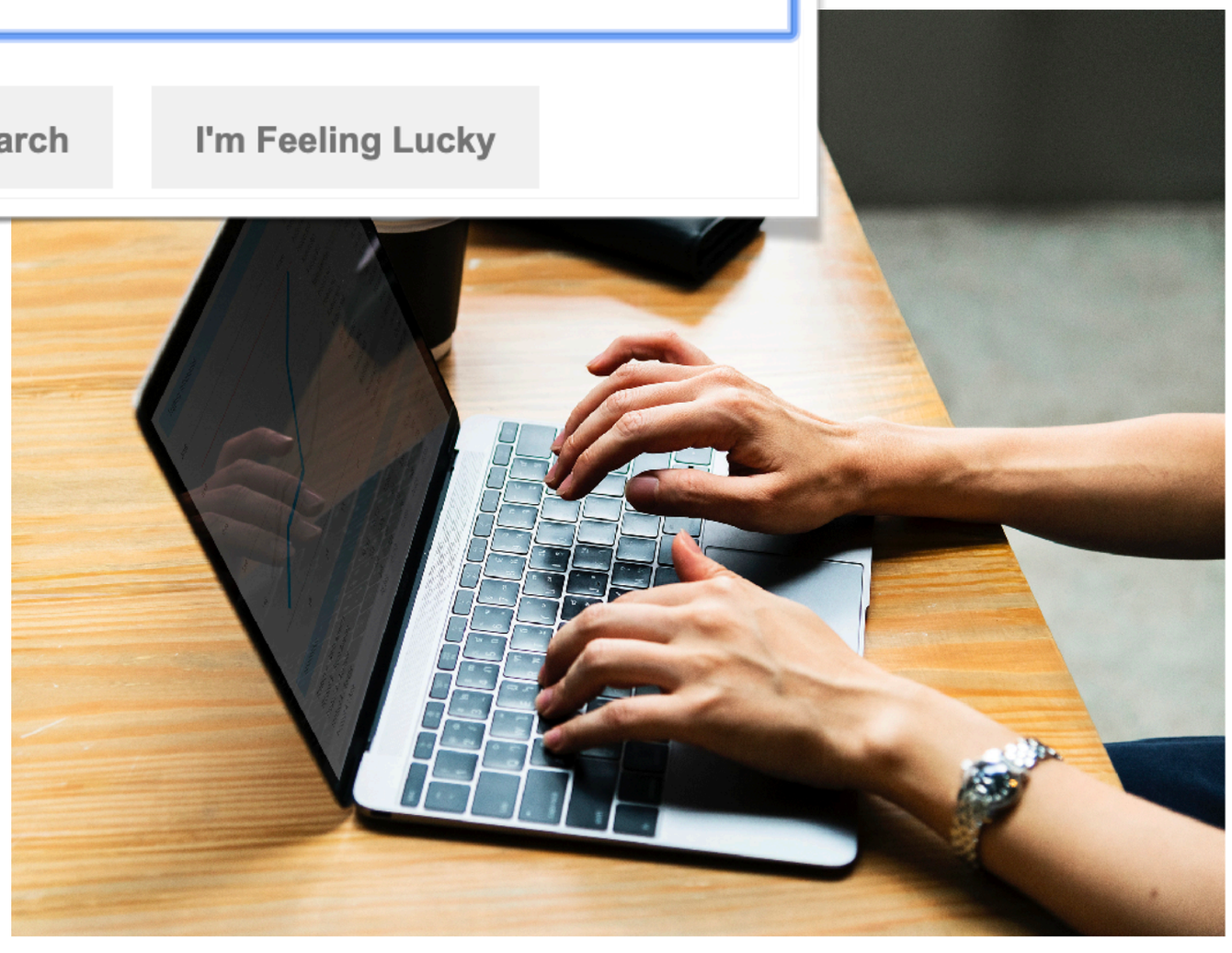




Academic Honesty Policies



This is NOT OK!
(even if you cite)



Academic Honesty Policies

What strategy did you use for Question 3?

I reduced the problem to network flows

https://www.wanderglobe.org/laptops-for-college-students/

Academic Honesty Policies

What strategy did you use for Question 3?

I reduced the problem to network flows

This is OK! (if you cite)

https://www.wanderglobe.org/laptops-for-college-students/

Academic Honesty Policies

Can you show me your solution to Question 3

Sure

https://www.wanderglobe.org/laptops-for-college-students/

Academic Honesty Policies

Can you show me your solution to Question 3

Sure

This is NOT OK!
(even if you cite)

https://www.wanderglobe.org/laptops-for-college-students/

Read More on Course Webpage

CSCI 256: Spring 2021

Algorithm Design and Analysis

Collaboration and Honor Code Policy

Sources. Algorithms is a core computer science field and there are multitudes of valuable resources out there, in the form of online lectures, videos, books, etc, that you are encouraged to use as reference to deepen your understanding of the material. However, when it comes to solving specific homework problems, you must be very careful about your sources and whether or not they violate academic honesty. The following guidelines state what is and is not allowed with respect to problem sets.

- **You must not search the internet or external resources using problem-specific keywords.** While it is okay to use external material for general reference, it is academically dishonest to use them with the intention to obtain solutions to specific problems. Copying (or approximately copying) solutions from the internet or other external sources, is plagiarism, *even if you cite your source*.
- **You must always cite external resources used for background reading.** If you use external materials for background reading to prepare for a problem set (and it does not violate the policy above), you must cite the source in the *Acknowledgments* section of your submission.
- **If you unintentionally come across a solution to a homework problem, you must request an alternate problem.** Solutions to some classic algorithms problems may be available on the web or other books. If you unintentionally encounter a solution to a homework question during background reading, you must inform me and I will replace the question and assign you an alternate problem on the same topic.

Collaboration. Collaboration is an integral part of learning and research in computer science. In this class, you are encouraged to collaborate with your classmates, form study groups, and to exchange high-level ideas on problem-solving techniques. Working with your peers can not only help you avoid common pitfalls and blind alleys, but also help you develop a deeper understanding of the material.

While open exchange of ideas is encouraged, you must adhere to the following academic integrity policies. Violation of these policies will be treated as an honor-code violation.

Advice on Collaboration

- Collaborations policies of this course are generous
- What do you see as the benefits of this?
- What do you see as the downsides?
- Problem set advice:
  - HW problems tend to have solutions that require some insight to discover
  - “If you immediately start working on the assignments in a group, you will miss out on the opportunity to come up with these insights on your own.”
  - Attempting problems yourself first is the single most important practice for the exams

What to Expect from this Class

- Expect **challenging and fun problems**
  - Expect to spend a lot of time playing with the problems!
  - Sense of accomplishment on finally solving them
- Expect to **make mistakes**
  - Making mistakes is the best way to learn
  - If you knew everything, you wouldn't be in this class
- Expect to **go out of your comfort zone**
  - Learning is uncomfortable, but in a good way
  - Common and OK to be frustrated by false starts!
- Expect to develop **"algorithmic thinking"**

Practice with CS Proofs

- Huge component of this class (the “Analysis” part of the course name)
- We will learn how to write computer science proofs
    - Slightly different style than Mathematics proofs
- Programming assignment vs proofs: common roadblock: how do you know your proof is “correct”?
    - **No autochecker for proofs!** Need to debug yourself
        - Go line by line and ask “why is this true?”
    - Ask me or TAs for guidance
    - You’ll build more intuition with practice

How to Succeed

- Read the handout on Problem Set Advice
- Do the readings
    - Ok to skim before class, **read more closely after**
    - Two textbooks give complementary presentations
- Go to TA and office hours
    - Treat them as "lab time"
- Form study groups!
    - Both in person and remote
    - I'd be happy help coordinate this

Questions?

Course Overview

- In CS 136, you (likely) learned:
    - almost any computational problem can be solved by breaking into small, digestible pieces
    - the asymptotic performance of those pieces can have a very significant impact
- In this class we take this further
    - Learn various algorithm design paradigms
    - How to model and efficiently solve computational problems
    - Analysis and proofs: why does the algorithm work (correctness)? how long does it take (running time)?
    - Study known algorithms and how to use them

Course Outline

- Review of asymptotic complexity
- Graph Traversals: BFS, DFS and applications
- Greedy algorithms
- Divide and conquer (recursion)
- Dynamic programming (recursion without repetition)
- Network-flow algorithms
- Intractability: NP hard, NP completeness & Reductions
- Randomized and approximation algorithms

Stable Matchings

An Illustrative Example:

The Stable Matching Problem

Applications

- Assigning first year students to advisors
- Pairing job candidates with employers
- Matching doctors to hospitals

Fundamental Problem

- Given preferences of both sides, find a matching that is resilient against *opportunistic swapping*

State the Problem

- Two groups: hospitals and students
- Students have preferences over hospitals
- Hospitals have preferences over students

Goal. Match hospitals to students that is “stable”, that is, no pair has an incentive to break their match!

The Redesign of the Matching Market for American Physicians: Some Engineering Aspects of Economic Design

By Alvin E. Roth and Elliott Peranson*

We report on the design of the new clearinghouse adopted by the National Resident Matching Program, which annually fills approximately 20,000 jobs for new physicians. Because the market has complementarities between applicants and between positions, the theory of simple matching markets does not apply directly. However, computational experiments show the theory provides good approximations. Furthermore, the set of stable matchings, and the opportunities for strategic manipulation, are surprisingly small. A new kind of “core convergence” result explains this; that each applicant interviews only a small fraction of available positions is important. We also describe engineering aspects of the design process. (JEL C78, B41, J44)

THE MATCH
NATIONAL RESIDENT MATCHING PROGRAM®

National Resident Matching Program

Matching Med-Students to Hospitals

Input. A set H of n hospitals, a set S of n students and their preferences (each hospital ranks each student, each students ranks each hospital)

	1st	**2nd**	**3rd**
MA	Aamir	Beth	Chris
NH	Beth	Aamir	Chris
OH	Aamir	Beth	Chris

	1st	**2nd**	**3rd**
Aamir	NH	MA	OH
Beth	MA	NH	OH
Chris	MA	NH	OH

Perfect Matchings

Definition. A matching M is a set of ordered pairs (h, s) where $h \in H$ and $s \in S$ such that

- Each hospital h is in at most one pair in M
- Each student s is in at most one pair in M

A matching M is **perfect** if each hospital is matched to exactly one student and vice versa (i.e., $|M| = |H| = |S|$)

	1st	**2nd**	**3rd**
MA	Aamir	Beth	Chris
NH	Beth	Aamir	Chris
OH	Aamir	Beth	Chris

	1st	**2nd**	**3rd**
Aamir	NH	MA	OH
Beth	MA	NH	OH
Chris	MA	NH	OH

Unstable Pairs

Definition. A perfect matching M is **unstable** if there exists an unstable pair $(h, s) \in H \times S$, that is,

- h prefers s to its current match in M
- s prefers h to its current match in M

Can you point out an unstable pair in this matching?

	1st	**2nd**	**3rd**
MA	Aamir	Beth	Chris
NH	Beth	Aamir	Chris
OH	Aamir	Beth	Chris

	1st	**2nd**	**3rd**
Aamir	NH	MA	OH
Beth	MA	NH	OH
Chris	MA	NH	OH

Unstable Pairs

Definition. A perfect matching M is **unstable** if there exists an unstable pair $(h, s) \in H \times S$, that is,

- h prefers s to its current match in M
- s prefers h to its current match in M

Can you point out any unstable pairs in this matching?

- E.g. (Beth, MA) better-off together: no incentive to follow M

	1st	2nd	3rd
MA	Aamir	Beth	Chris
NH	Beth	Aamir	Chris
OH	Aamir	Beth	Chris

	1st	2nd	3rd
Aamir	NH	MA	OH
Beth	MA	NH	OH
Chris	MA	NH	OH

Stable Matching Problem

Problem. Given the preference lists of n hospitals and n students, find a stable matching, that is a matching with no unstable pairs.

Question. Does such a matching always exist?

This does not seem obvious!

	1st	**2nd**	**3rd**
MA	Aamir	Beth	Chris
NH	Beth	Aamir	Chris
OH	Aamir	Beth	Chris

	1st	**2nd**	**3rd**
Aamir	NH	MA	OH
Beth	MA	NH	OH
Chris	MA	NH	OH

How Can We Show This?

- Want to prove: a stable matching always exists
- One way:
  - Give an algorithm to find a stable matching
  - Prove that it is always successful
  - Constructive method

False Starts

Proceed greedily in rounds until matched. In each round,

- Each hospital makes offer to its top available candidate
- Each student accepts its top offer (irrecoverable contract) and rejects others

What goes wrong?

	1st	**2nd**	**3rd**
MA	Aamir	Chris	Beth
NH	Aamir	Beth	Chris
OH	Chris	Beth	Aamir

	1st	**2nd**	**3rd**
Aamir	OH	NH	MA
Beth	MA	OH	NH
Chris	MA	NH	OH

False Starts

Proceed greedily in rounds until matched.

- (Round 1) MA → Aamir, NH → Aamir, OH → Chris

	1st	2nd	3rd
MA	Aamir	Chris	Beth
NH	Aamir	Beth	Chris
OH	Chris	Beth	Aamir

	1st	2nd	3rd
Aamir	OH	NH	MA
Beth	MA	OH	NH
Chris	MA	NH	OH

False Starts

Proceed greedily in rounds until matched.

- (Round 1) MA → Aamir, NH → Aamir, OH → Chris
- (Round 1) Aamir rejects MA, accepts NH, Chris accepts OH

	1st	2nd	3rd
MA	Aamir	Chris	Beth
NH	Aamir	Beth	Chris
OH	Chris	Beth	Aamir

	1st	2nd	3rd
Aamir	OH	NH	MA
Beth	MA	OH	NH
Chris	MA	NH	OH

False Starts

Proceed greedily in rounds until matched.

- (Round 1) MA → Aamir, NH → Aamir, OH → Chris
- (Round 1) Aamir rejects MA, accepts NH, Chris accepts OH
- (Round 2) Only Beth and MA left, and must match

	1st	2nd	3rd
MA	Aamir	Chris	Beth
NH	Aamir	Beth	Chris
OH	Chris	Beth	Aamir

	1st	2nd	3rd
Aamir	OH	NH	MA
Beth	MA	OH	NH
Chris	MA	NH	OH

False Starts

Proceed greedily in rounds until matched.

- (Round 1) MA → Aamir, NH → Aamir, OH → Chris
- (Round 1) Aamir rejects MA, accepts NH, Chris accepts OH
- (Round 2) Only Beth and MA left, and must match

Is this a stable matching?

	1st	2nd	3rd
MA	Aamir	Chris	Beth
NH	Aamir	Beth	Chris
OH	Chris	Beth	Aamir

	1st	2nd	3rd
Aamir	OH	NH	MA
Beth	MA	OH	NH
Chris	MA	NH	OH

False Starts

Proceed greedily in rounds until matched.

- (Round 1) MA → Aamir, NH → Aamir, OH → Chris
- (Round 1) Aamir rejects MA, accepts NH, Chris accepts OH
- (Round 2) Only Beth and MA left, and must match

Is this a stable matching?

- Unstable pair: (MA, Chris). What could have avoided it?

	1st	2nd	3rd
MA	Aamir	Chris	Beth
NH	Aamir	Beth	Chris
OH	Chris	Beth	Aamir

	1st	2nd	3rd
Aamir	OH	NH	MA
Beth	MA	OH	NH
Chris	MA	NH	OH

Next Time: Deferred Acceptance

Proceed in rounds until all hospitals matched. In each round,

- Each free hospital offers to its top choice among candidates it hasn't offered yet
- Each free student *retains but defers accepting* **top offer**, rejects others
- If a student receives a better offer than currently retained, they reject current and retain new offer (trade up)

	1st	**2nd**	**3rd**
MA	Aamir	Chris	Beth
NH	Aamir	Beth	Chris
OH	Chris	Beth	Aamir

	1st	**2nd**	**3rd**
Aamir	OH	NH	MA
Beth	MA	OH	NH
Chris	MA	NH	OH