CS41 Homework 1

This homework is due (via `git`) by 11:59 PM on **Monday, September 14**. However, you are strongly encouraged to make an initial submission by *Thursday, September 10* (more details below).

You are expected to write your homework solutions using LaTeX, and submit using **git / github**.

The homework should be completed and submitted individually. It's ok to discuss approaches at a high level. In fact, you are encouraged to discuss general strategies. However, you should not reveal specific details of a solution, nor should you show your written solution to anyone else. Please ask your professor if you have any questions about the academic integrity policy.

The main **learning goals** of this homework are to (i) ensure your git repositories are properly setup, (ii) familiarize you with writing in LaTeX, and (iii) to begin to think algorithmically, and to formalize and analyze algorithms.

Getting Setup with Git and LaTeX

In order to make sure you are able to submit your completed homework through `git` without any issues, you are **strongly encouraged** to do the following *by Thursday, September 10*:

1. Follow the instructions at `https://cs.swarthmore.edu/~ngrammel/cs41/f26/git.html` to make sure you're able to access the CS41-F26 git organization, and reach out to the professor *as soon as possible* if you run into any trouble. Try to commit and push something.

2. Using the `LearningLaTeX.tex` file as a guide, add some displayed math to the `hw01.tex` file. Then, follow the submission instructions at the end of this homework to prepare a "practice submission", commit and push your submission.

Making this initial submission by Thursday, 9/10 will help you make sure that you are able to submit your LaTeX through git. After Thursday, I will check GitHub to see who successfully submitted something, and check whether the submitted `hw01.tex` file includes displayed math (added by you) that compiles successfully on the CS machines. I will follow up with anyone whose submission didn't come through or didn't compile successfully.

Problems to Solve

Solve the following problems, write up your solutions using LaTeX, and submit by the homework deadline.

1. **Find the Stable Matching.** Below is an input to the stable matching problem:

   - 5 hospitals: [Abington, Brandywine, CHOP, Delaware County Memorial (DCM), Einstein Medical Center (EMC)]
   - 5 doctors: [Alice, Bob, Chenye, Dmitri, Eva]
   - Hospital Preferences (in each list, doctors are ordered from most to least preferred, so e.g. Abington's top choice for doctor is Bob, and least preferred doctor is Alice)

  - Abington: [Bob, Eva, Chenye, Dmitri, Alice]
  - Brandywine: [Eva, Bob, Chenye, Alice, Dmitri]
  - CHOP: [Bob, Chenye, Alice, Eva, Dmitri]
  - Delaware County Memorial: [Chenye, Eva, Alice, Bob, Dmitri]
  - Einstein Medical Center: [Eva, Alice, Dmitri, Bob, Chenye]
- Doctor Preferences (in each list, hospitals are ordered from most to least preferred)
  - Alice: [Abington, Brandywine, CHOP, DCM, EMC]
  - Bob: [CHOP, Brandywine, Abington, EMC, DCM]
  - Chenye: [CHOP, Brandywine, Abington, DCM, EMC]
  - Dmitri: [DCM, Brandywine, CHOP, Abington, EMC]
  - Eva: [CHOP, Brandywine, EMC, DCM, Abington]

Give a stable (hospital-doctor) matching for this input.

2. **Stable Matching Runtime.** We showed in class that the Gale-Shapely Algorithm for stable matching terminates after at most n^2 iterations of the while loop.

   (a) For two sets A and B of size n, can a particular list of rankings actually result in a quadratic number of iterations? If so, describe what the rankings would look like. If not, argue why no set of rankings would ever result in a quadratic number of iterations. **Note:** the algorithm need not take exactly n^2 iterations, but *asymptotically* n^2 iterations, meaning $0.1n^2$ would be sufficient to show your claim.

   (b) Can a particular set of rankings result in strictly less than a quadratic number of iterations? Can you design an input that requires $O(n)$ iterations? If so, describe the structure of this input. If not, argue why this is not possible.

   (c) Finally, can you design an input that takes fewer than n iterations? Why or why not?

   Aim for clarity and conciseness in your write up of this problem. You should have all the necessary tools to express your solutions. You do not need formal proofs or pseudocode, but you should be able to clearly articulate your ideas in plain English.

3. **Sharing the Money.** The executive officers of Mega Corporation have just defrauded investors and shareholders out of $1 billion and are meeting to discuss how to divide up the misappropriated funds.

   The chairman of the board of directors currently controls the secret cryptocurrency wallet that holds all of the funds. There is a strict ranking of the remaining members of this conspiracy: after the chairman is the CEO, the CFO, the COO, the CTO, and finally Head Counsel from the legal department.

   They are all greedy and would all like to keep as much money for themselves as possible. However, they also believe in democracy (as long as they're the only ones voting). They have agreed that the chairman will propose how to divide the funds among the six of them, and they will then all vote on whether they approve of this division.

   If at least half of the conspirators vote to approve the allocation, then it passes. Otherwise, it fails. They have decided that an unpopular proposal should not go unpunished, so if the

proposal fails, the chairman will be ousted from the group and be permanently excluded from receiving any of the money. If this happens, the next person in line (the CEO) will take over and be given the chance to propose a division of the funds to the remaining conspirators. The process continues (with the leader proposing a way to divide the money, all remaining conspirators voting, and the leader being ousted if his proposal is not approved) until a division is accepted.

What distribution of funds should the chairman propose? You can assume that everyone is greedy and cares only about how much treasure they receive. Additionally, you may assume that allocations must be in whole dollars.

Note: If there are an even number of conspirators and a vote that is evenly split, the proposal is approved. If another conspirator believes they will end up with the same amount of money whether or not the proposal is accepted, they will vote to reject.

Submitting Your Solutions

Write up solutions to the above three problems in LaTeX, and place them in your `hw01.tex` file. Then, you will submit using `git`. When submitting, keep the following in mind:

- Please *don't include your name* in your file submission. This helps keep the grading anonymous.
- Make sure all your solutions are in your `hw01.tex` file, and that it *compiles on CS lab computers without errors.*
- Make sure your completed `hw01.tex` file is *pushed* to your github repository (e.g., with `git add`, `git commit`, `git push`).
- Don't submit a PDF. We will pull and compile the `.tex` file, so make sure all updates to this file are committed and pushed to your repo.

Handwritten Option (For HW1 Only) For this first homework only, you *may* submit handwritten solutions instead. You are encouraged to use LaTeX here as practice, since it will be required on future homeworks and so that I can offer feedback on your LaTeX code (e.g., formatting, etc.). However, if you are having trouble with LaTeX, you may (clearly and neatly!) write your solutions by hand (you should still not include your name), then scan or take photos of the handwritten solutions, add the images to your git repository, and commit+push those. If you do this, you should come speak with me before HW2 so that we can help get you started with LaTeX and make sure you're able to use it on HW2.