CS41 Homework 2

This homework is due at 10pm on Sunday, September 20. Write your solution using LaTeX. Submit this homework using **github** as a file called **hw2.tex**. This is an individual homework. It's ok to discuss approaches at a high level. In fact, we encourage you to discuss general strategies. However, you should not reveal specific details of a solution, nor should you show your written solution to anyone else. The only exception to this rule is work you've done with a lab partner *while in lab*. In this case, note (in your **README file**) who you've worked with and what parts were solved during lab.

The main **learning goals** of this lab are to work with stable matching and the Gale-Shapley algorithm, and get comfortable analyzing it and applying it.

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 prefered 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. **Socially Distant Foodies.** A group of n foodies[1] $F = \{f_1, \dots, f_n\}$ are touring a set of n new restaurants $R = \{r_1, \dots, r_n\}$ over the course of $m \geq n$ days, in search of their new favorite restaurants. Each foodie f_j has an itinerary where he/she decides to visit one restaurant per day (or perhaps take a day off if $m > n$). However, because of the global pandemic, the foodies prefer not to share restaurants with each other. Furthermore, each foodie is looking for a favorite restaurant to call his or her own. Each foodie f would like to choose a particular day d_f and *stay* at his/her current restaurant r_f for the remaining $m - d_f$ days of the tour. Of course, this means that no other foodies can visit r_f after f arrives at r_f.

[1] a foodie is a food enthusiast

Show that no matter what the foodies' itineraries are, it is possible to assign each foodie f a unique restaurant r_f such that when f arrives at r_f according to the itinerary for f, all other foodies have either stopped touring restaurants themselves, or f' will not visit r_f after f arrives at r_f.

Hint: Solve this problem by **reducing** to stable matching. The input is somewhat like the input to stable matching, but at least one piece is missing. Find a clever way to construct the missing piece(s), run stable matching, and show that the final result solves this problem.

3. **Sorting to Half-Sorting**. In the HALF-SORT problem, you're given an array of n integers and must return an array that has the first $\lceil n/2 \rceil$ integers in sorted order. For example, if your array is $A = [5, 9, 1, 2, 6, 3]$, then a valid output of HALF-SORT(A) might be $[1, 2, 3, 9, 5, 6]$ since $1, 2, 3$ are the least elements of A.

   - Reduce the sorting problem to HALF-SORT. i.e., imagine you have an algorithm $\mathcal{A}$ for HALF-SORT, and use it to design a sorting algorithm.
   - Now, suppose that your friend claims to have an algorithm for HALF-SORT that runs in $10n$ time in the worst case. What is the runtime of your sorting algorithm? Is $10n$ a reasonable running time for HALF-SORT?

4. **(extra challenge problem)** In class, we discussed a version of the stable matching problem where we want to match n doctors to n hospitals. In this problem, we discuss the homogeneous version. The input is a set of students $A = \{s_1, \ldots, s_{2n}\}$ of size $2n$. Each student ranks the others in order of preference. A homework partner assignment of students into partners $M = \{(i, j)\}$ is a matching; it is unstable if there exists $(i, j), (i', j') \in M$ such that i prefers j' to j and that j' prefers i to i'. It is stable if it is a perfect matching and there are no instabilities.

   (a) Does a stable homework partner assignment always exist? Prove that such an assigment must always exist, or give an example where no stable assignment occurs. (Remember, you must have $2n$ students.)

   (b) Design and analyze an efficient algorithm that either returns a stable matching for homework partners or outputs that no such matching exists.