CS41 Homework 2

This homework is due at 11:59PM on Sunday, September 17. Write your solution using LaTeX. Submit this homework in a file named `hw2.tex` using **github**. This is an individual homework. It's ok to discuss approaches at a high level. In fact, I 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/group *while in lab.* In this case, note (in your post-homework survey) who you've worked with and what parts were solved during lab.

The main **learning goals** of this homework are to work with stable matching and the Gale-Shapley algorithm, to get comfortable analyzing it and applying algorithmic thinking, and to practice writing thorough arguments.

1. **Practice with proofs.**

   (a) Sometimes a proof which follows the correct structure can have a subtle problem. Consider the following proof.

   Claim 1. *All dogs are the same color.*

   If we want to think of this as an inductive statement, the statement is "In any group of n dogs, all of those n dogs are the same color."

   Proof. (by induction)

   - base case: Let's say we have just $n = 1$ dog. Then clearly all the dogs we're considering are the same color.
   - inductive hypothesis: Assume that if we have $n = k$ dogs in a group, they are all the same color, for some arbitrary constant k.
   - inductive step: Let's say we have $n = k + 1$ dogs in a group. We want to show that they are all the same color.
     Let's consider the group if we take out one dog ("Albus Dumbledog"). Now we have a group of k dogs, so by the inductive hypothesis, they are all the same color.
     Now add Albus back in, and let's take out a different dog ("Hairy Potter"). Now again we have a group of k dogs, so by the inductive hypothesis, they are all the same color.
     Since the group with Potter but not Albus is all the same color, and the group with Albus but not Potter is all the same color, it must be that Albus and Potter are the same color. Therefore all $k + 1$ dogs in the group are the same color.

   Therefore, by induction, all dogs are the same color. □

   It seems like something is probably wrong with this argument, since dogs exist in many different colors. Explain what the issue is with this proof.

   (b) Prove or disprove the following claim:

   Claim 2. *All CS professors wear an outfit consisting of a plaid button-down shirt and khakis when they teach.*

2. **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) perfect matching for this input. (You do not have to show any work or argue why it is stable, but the matching you give must be a stable perfect matching!)

3. **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 (math notation is also ok as long as it is readable).

4. **(extra challenge problem) Sorting and 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.

(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. Briefly explain why your algorithm is correct.

(b) Reduce HALF-SORT to the sorting problem. i.e., imagine you have a sorting algorithm $\mathcal{B}$, and use it to design an algorithm for HALF-SORT. Briefly explain why your algorithm is correct.

(c) 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?

5. **(extra challenge problem) Homework Partner Matching.** 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.