# CS41 Homework 2

This homework is due at 11:59pm on Monday, September 13. 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, your 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. The only exception to this rule is work you've done with a lab partner *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 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. **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[1], but you should be able to clearly articulate your ideas in plain English[2].

3. **College visits.** A group of $n$ high school seniors $S = \{s_1, s_2, s_3, \ldots, s_n\}$ are touring a set of colleges $C = = \{c_1, \ldots, c_n\}$ over the course of $m \geq n$ days this autumn. Each student $s_j$ has an itinerary where he/she decides to visit one college per day (or maybe take a day off if $m > n$). However, these students are very pandemic-conscious prefer not to share college campus tours with other students. Furthermore, each student is looking for one college to call their own. Each student $s$ would like to choose a particular day $d_s$ and stay at his/her current college admissions office for the remaining $m - d_s$ days of the autumn (laying claim to 100% of the admissions officer's time). Of course, this means that no other students can visit $c_s$ after day $d_s$.

   Show that no matter what the students' itineraries are, it is possible to assign each student $s$ a unique college $c_s$, such that when $s$ arrives at $c_s$ according to the itinerary for $s$, all other students $s'$ have either stopped touring colleges themselves, or $s'$ will not visit $c_s$ after $s$ arrives at $c_s$.

   **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.

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.

[1] This is special permission granted *for this problem* only!
[2] math notation is also ok as long as it is readable