Homework 1
CS601: Advanced algorithms

Instructions: In this assignment you will solve a problem related to stable matching. You may if you prefer solve the problem with one partner, but you need to make sure either one of you, if being asked, explain details of your solutions.

And I made another optional, which means extra credits, it's a slightly extended version of the med-school hospital matching problem stated in Textbook **Chapter 1 problem set 4**.

Don't write down your name(s) in case it will be peer graded. Do Clearly write down your answers! If you don't want to use latex or word to write your answer, you can hand write your solution, and scan or take pictures to submit, make sure it's clear to grade. Don't zip it, just submit your file (pdf prefered) or script directly to Canvas.

Problem 1. [Category: Coding] Here is another problem that is relate to stable matching.

You are given a list of preferences for n friends, where n is always even.

For each person i, $preferences[i]$ contains a list of friends sorted in the order of preference. Friends in each list are denoted by integers from 0 to $n - 1$.

All the friends are divided into pairs. The pairings are given in a list pairs, where $pairs[i] = [x_i, y_i]$ denotes x_i is paired with y_i and y_i is paired with x_i.

However, this pairing may cause some of the friends to be unhappy. A friend x is unhappy if x is paired with y and there exists a friend a who is paired with b but:

x prefers a over y, and a prefers x over b. Return the number of unhappy friends.

Example 1:

Input: n = 4, preferences = [[1, 2, 3], [3, 2, 0], [3, 1, 0], [1, 2, 0]], pairs = [[0, 1], [2, 3]]

Output: 2

Explanation:

Friend 1 is unhappy because: - 1 is paired with 0 but prefers 3 over 0, and - 3 prefers 1 over 2. Friend 3 is unhappy because: - 3 is paired with 2 but prefers 1 over 2, and - 1 prefers 3 over 0. Friends 0 and 2 are happy. Example 2:

Input: n = 2, preferences = [[1], [0]], pairs = [[1, 0]]

Output: 0

Explanation: Both friends 0 and 1 are happy.

Example 3:

Input: n = 4, preferences = [[1, 3, 2], [2, 3, 0], [1, 3, 0], [0, 2, 1]], pairs = [[1, 3], [0, 2]]

Output: 4

Example 4:

Input: n = 4, [[1,3,2],[2,3,0],[1,0,3],[1,0,2]]

[[2,1],[3,0]]

Output:0

Please submit a script including test cases (examples included above or more) and instructions how to run the script. **[25 points]**

Problem 2. [Category: Design and Coding]

Resident Matching, Chapter 1, problem 4(scanned on next page).

1. Please give a clear description of your algorithm in pseudocode, 1 page limit. **[5 points]**
2. Implement your algorithm in your favorite language, and provide test cases and instructions to run it. Python is highly recommended. A python implementation of the Standard Gale-Shapley algorithm is given to you which contains one example, it will help you implement the algorithm for this one. **[10 points]**
3. Follow the textbook, prove that your algorithm's correctness which contains two parts: **[5 points]**
    - Prove the algorithm will terminate
    - Prove output matching is stable.

    1 page limit.

4. Gale and Shapley published their paper on the Stable Matching Problem in 1962; but a version of their algorithm had already been in use for ten years by the National Resident Matching Program, for the problem of assigning medical residents to hospitals.

Basically, the situation was the following. There were m hospitals, each with a certain number of available positions for hiring residents. There were n medical students graduating in a given year, each interested in joining one of the hospitals. Each hospital had a ranking of the students in order of preference, and each student had a ranking of the hospitals in order of preference. We will assume that there were more students graduating than there were slots available in the m hospitals.

The interest, naturally, was in finding a way of assigning each student to at most one hospital, in such a way that all available positions in all hospitals were filled. (Since we are assuming a surplus of students, there would be some students who do not get assigned to any hospital.)

We say that an assignment of students to hospitals is *stable* if neither of the following situations arises.

- First type of instability: There are students s and s', and a hospital h, so that
  - s is assigned to h, and
  - s' is assigned to no hospital, and
  - h prefers s' to s.
- Second type of instability: There are students s and s', and hospitals h and h', so that
  - s is assigned to h, and
  - s' is assigned to h', and
  - h prefers s' to s, and
  - s' prefers h to h'.

So we basically have the Stable Matching Problem, except that (i) hospitals generally want more than one resident, and (ii) there is a surplus of medical students.

Show that there is always a stable assignment of students to hospitals, and give an algorithm to find one.