(2) *(10 points)* Suppose we are given an instance of the stable matching problem, consisting of a set of n applicants $\{x_1, \ldots, x_n\}$ and a set of n employers $\{y_1, \ldots, y_n\}$, together with a list for each entity (applicant or employer) that ranks the entities of the opposite type from best to worst. This exercise concerns algorithms to solve the following problem: *decide whether there exists a stable perfect matching in which x_n is matched to y_n.*

(2a) A simple algorithm for this problem is the following: remove x_n from every employer's preference list, and remove y_n from every applicant's preference list. Run the Gale-Shapley algorithm (say, with employers proposing) to find a stable perfect matching, M, of the applicant set $\{x_1, \ldots, x_{n-1}\}$ and employer set $\{y_1, \ldots, y_{n-1}\}$. If $M \cup \{(x_n, y_n)\}$ is a stable perfect matching of the original $2n$ entities (with their original unmodified preference lists) then answer "yes"; otherwise, answer "no". Give an explicit input instance on which this algorithm outputs the wrong answer.

(2a) An explicit input on which the algorithm outputs the wrong answer is as follows:

Employer preferences: Applicant preferences:

$y_1 : x_1 > x_2 > x_3$ $x_1 : y_2 > y_3 > y_1$

$y_2 : x_3 > x_2 > x_1$ $x_2 : y_1 > y_3 > y_2$

$y_3 : x_1 > x_3 > x_2$ $x_3 : y_1 > y_3 > y_2$

When the algorithm is run starting with y_1 with the exclusion of y_3 and x_3 we first have the pairing of (y_1, x_1). Next, (y_2, x_2) becomes a pairing according to their preferences. Now according to the algorithm the pairing of (y_3, x_3) is introduced and in this case the answer no is produced as (y_3, x_1) forms an unstable pair as they each prefer each other to their current matchings. However, with these preferences a stable perfect matching does in fact exist as the set $[(y_1, x_2), (y_2, x_1), (y_3, x_3)]$ is a stable perfect matching as it has no unstable pairs and every applicant and and employer are paired.

(2b) Design a polynomial-time algorithm to decide whether there exists a stable perfect matching in which x_n is matched to y_n. Prove that your algorithm always outputs the correct answer and analyze its running time.

Hint: The solved exercises at the end of Chapter 1 in the textbook may provide a useful subroutine for your algorithm.

(2b)

To modify the G-S Algorithm for this solution we can simply create a Forbidden set to distinguish which pairs are forbidden to be made. The forbidden set will include all pairings of applicant x_n with every employer above y_n in the preferences of x_n. Similarly, all pairings of employer y_n with every applicant above x_n in the preferences of y_n. In addition, for each applicant above x_n in the preferences of y_n, the employers in their preferences below y_n and that applicant will be added to the forbidden list. The same technique is used to create forbidden pairings for those other employers in the preferences of x_n below y_n. After this, the G-S algorithm will be run with an altered condition, the condition will be run while there is an employer y who is free and has not

proposed to each applicant x where the pairing of the (y,x) is not on the forbidden list. If at the end there is an employer left unmatched then the algorithm outputs no meaning that there does not exist a perfect stable matching for the given preferences which includes the pairing (y_n, x_n). If at the end every employer has been paired then this is a stable perfect matching including the pair (y_n, x_n). Formally the algorithm is as follows:

Pre-Processing Step:

1.)Examine the preference list of x_n and add the pairing of (x_n,y) to the Forbidden List in decreasing preference until y_n (the desired pairing is reached) and then stop.

2.) For each y that has no been added to the forbidden list as a pairing with x_n, on their preference lists for each applicant x below x_n the pairing (x,y) should be added to the Forbidden List as well.

3.) Similarly, Examine the preference list of x_n and add the pairing of (x_n,y) to the Forbidden List in decreasing preference until y_n (the desired pairing is reached) and then stop.

4.) For each x that has no been added to the forbidden list as a pairing with y_n, on their preference lists for each employer y below y_n the pairing (x,y) should be added to the Forbidden List as well.

[H]

1: Modified G-S Algorithm:
2: Begin with all employers(y's) and all applicants(x's) free
3: While there is a free employer y who hasn't proposed to each applicant x for which the pairing is not on the Forbidden List
4:    Choose this employer y
5:    Find the most preferred applicant x for which y has not already proposed to and (x,y) is not on the Forbidden List
6:       If x is free, (x,y) become a pair
7:       Else x is already paired to another employer y'
8:          If x prefers y to y' then
9:             x switches and pairs with y making (x,y)
10:             y' becomes free
11:          Else x prefers y' to y then
12:             y stays free
13:          EndIf
14:       EndIf
15:    EndWhile
16:    If there exists an employer that is free then "No" is returned
17:    Else "Yes" is returned
18:    EndIf
19:

Example:
Using the example introduced in (2a) in the pre-processing steps in step 1 the pair (x_3,y_1) would be added to the forbidden list. In step 2 no pairs would be added. In step 3, the pair (x_1, y_3)

would be added to the forbidden list. In step 4, the pair (x_1, y_1) would be added to the forbidden list. Then the algorithm would begin first pairing (x_2, y_1). Then (x_3, y_2) would be paired. Next (x_3, y_3) would be paired leaving y_2 free. Then (x_1, y_2) would pair. finally the post processing step would output "Yes".

Proof of Correctness:

Let M be the matching generated by the algorithm

Observation 1: Given the algorithm outputs "Yes" then M is a perfect stable matching.

Part 1: The output is yes, then M is a perfect matching.

Proof: By definition the algorithm only outputs "Yes" if every employer has been paired. Each employer can only be paired to one applicant and there are equal number of applicants as there are employers. Thus, as all employers are paired it follows that all applicants are also paired and each can only be paired once, thus M is a perfect matching.

Part 2: The output is yes, then M is a stable matching

Proof: Suppose for the sake of contradiction that there exists and unstable pair (x,y). Then we know that x prefers y to its current partner y' and that y prefers x to its current partner x' and that (x,y) is not currently a pairing in M. Then we know that y must have either proposed to x as it is currently paired with x' or (x,y) must have been an element of the forbidden list. In the case that y proposed to x is would have had to have been displaced by some y" which was higher ranked by x, thus x: y">y. However, as x is currently paired with y', x must have preferred y' to y", thus x: y'>y". This a a contradiction as this means that x: y'>y and out assumption that (x,y) was an unstable pair means that x: y>y'. The case that (x,y) is a forbidden pair is in fact impossible. In the case that the unstable pair was put n the forbidden list by pre-processing step 2 or 4 the pair cannot possible be unstable. As was stated both x and y were previously paired. The forbidden pairs generated by these pre-processing steps are the least desirable matches for those specific x and y and thus they cannot possibly be preferred to their current pairs. Similarly, if the forbidden pair was put on the list by steps 1 or 3 these cannot possibly be unstable as all less x and y which were more desired by y_n and x_n respectively, had all of their less desirable pairings forbidden in steps 2 and 4. As these matches cannot be preferred by both x and y in the pairing and unstable pairing cannot possibly be a forbidden pair. For this reason clearly M contains no unstable pairs.

Observation 2: The output is "No", then the matching M is not a perfect stable match. Proof: By definition in the algorithm, "No" is only output if there is an employer who is left unmatched at the end. This means that by definition the matching M is not perfect and in turn it cannot be a perfect stable matching.

Run time Analysis Pre-Processing Step:

In steps 1 and 3 each preferences could possibly be accessed once in a list containing n elements thus the maximum number of preferences accesses made is n. In steps 2 and 4 which happen for each preference accessed in steps 1 and 3, at a maximum the entire preferences list must be accessed in succession thus there would be n accesses. With the actual additions of pairing to the Forbidden list taking constant time the overall run time for the pre-processing step would be

$\text{O}(n^2)$

Run time Analysis Processing:
As at a minimum there may be no forbidden pairs, the algorithm could at a maximum have the same run time as the G-S algorithm. As in the G-S algorithm there are n employers who may potentially have to make a total of n proposals each thus the overall run time is $\text{O}(n^2)$.

Run time Analysis Post-Processing Step:
This step is simply checking to see if each employer has been paired, this would at the maximum involved iterating over each of the n employers and in constant time checking if they are paired or not. Thus, the run time for this step is O(n).

Overall Run-Time is $\text{O}(n^2)+\text{O}(n^2)+\text{O}(n)=\text{O}(n^2)$