Problem 1. [20 points] KT Problem 1-1 and 1-2
Decide whether you think the following statements are true or false. If it is true, give a short explanation. If it is false, give a counterexample.

(1) In every instance of the Stable Matching Problem, there is a stable matching containing a pair \((m, w)\) such that \(m\) is ranked first on the preference list of \(w\) and \(w\) is ranked first on the preference list of \(m\).

(2) Consider an instance of the Stable Matching Problem in which there exists a man \(m\) and a woman \(w\) such that \(m\) is ranked first on the preference list of \(w\) and \(w\) is ranked first on the preference list of \(m\). Then in every stable matching \(S\) for this instance, the pair \((m, w)\) belongs to \(S\).

Problem 2. [20 points] KT Chapter 1 Problem 4
The stable matching problem, for matching medical residents to hospitals. Please read the whole description from the textbook.

Problem 3. [30 points]
In order to asymptotically analyze the running time of an algorithm, one has to think about how the data will be represented and manipulated in an implementation of the algorithm, so as to bound the number of computational steps it takes. In this problem, you will use arrays to implement the different high-level operations in the Gale-Shapley algorithm presented in Chapter 1, and shown in Figure 1.

Initially all \(m \in M\) and \(w \in W\) are free
While there is a man \(m\) who is free and hasn’t proposed to every woman
Choose such a man \(m\)
Let \(w\) be the highest-ranked woman in \(m\)’s preference list to whom \(m\) has not yet proposed
If \(w\) is free then
\((m, w)\) become engaged
Else \(w\) is currently engaged to \(m’\)
If \(w\) prefers \(m’\) to \(m\) then
\(m\) remains free
Else \(w\) prefers \(m\) to \(m’\)
\((m, w)\) become engaged
\(m’\) becomes free
Endif
Endif
Endwhile
Return the set \(S\) of engaged pairs

Figure 1: The Gale-Shapley algorithm.
(a) Rewrite the algorithm, with the appropraite data structures.
(b) Analyze the worst case running time for your algorithm. Explain your answer.

Problem 4. [35 points] KT Chapter 1 Problem 5. [Proof Problem]
Consider a version of the stable matching problem where there are $n$ men and $n$ women as before. Assume each man ranks the women (and vice versa), but now we allow ties in the ranking. In other words, we could have a man that is indifferent two women $w_1$ and $w_2$, but prefers either of them over some other woman $w_3$ (and vice versa). We say a woman $w$ prefers man $m_1$ to $m_2$ if $m_1$ is ranked higher on the $w$’s preference list and $m_1$ and $m_2$ are not tied.

Part 1

With indifferences in the rankings, there could be two natural notions for stability. And for each, we can ask about the existence of stable matchings, as follows.

(a) **Strong Instability.** A strong instability in a perfect matching, $S$, consists of a man $m$ and a woman $w$, such that each of $m$ and $w$ prefers the other to their partner in $S$.

Does there always exist a perfect matching with no strong instability?
Either give an example of a set of men and women with preference lists for which every perfect matching has a strong instability; or give an algorithm that is guaranteed to find a perfect matching with no strong instability.

(b) **Weak Instability.** A weak instability in a perfect matching, $S$, consists of a man $m$ and a woman $w$, such that their partners in $S$ are $w'$ and $m'$, respectively, and one of the following holds:
* $m$ prefers $w$ to $w'$, and $w$ either prefers $m$ to $m'$ or is indifferent between these two choices; or
* $w$ prefers $m$ to $m'$, and $m$ either prefers $w$ to $w'$ or is indifferent between these two choices.

In other words, the pairing between $m$ and $w$ is either preferred by both, or preferred by one while the other is indifferent.

Does there always exist a perfect matching with no weak instability?
Either give an example of a set of men and women with preference lists for which every perfect matching has a weak instability; or give an algorithm that is guaranteed to find a perfect matching with no weak

Part 2 - Proof problem part

If you defined an algorithm to either one of the types of instability above in Part 1, prove its correctness, and submit it separately.