Week – 1 Graded Homework (Solutions)

Solution 1:

The algorithm is very similar to the basic Gale-Shapley algorithm from the text. At any point in time, a student is either "committed" to a hospital or "free." A hospital either has available positions, or it is "full." The algorithm is the following:

```
While some hospital h_i has available positions
  h_i offers a position to the next student s_j on its preference list
  if s_j is free then
    s_j accepts the offer
  else (s_j is already committed to a hospital h_k)
    if s_j prefers h_k to h_i then
      s_j remains committed to h_k
    else s_j becomes committed to h_i
         the number of available positions at h_k increases by one.
         the number of available positions at h_i decreases by one.
```

The algorithm terminates in $O(mn)$ steps because each hospital offers a positions to a student at most once, and in each iteration, some hospital offers a position to some student.

Suppose there are $p_i > 0$ positions available at hospital h_i. The algorithm terminates with an assignment in which all available positions are filled, because any hospital that did not fill all its positions must have offered one to every student; but then, all these students would be committed to some hospital, which contradicts our assumption that $\sum_{i=1}^{m} p_i < n$.

Finally, we want to argue that the assignment is stable. For the first kind of instability, suppose there are students s and s', and a hospital h as above. If h prefers s' to s, then h would have offered a position to s' before it offered one to s; from then on, s' would have a position at *some* hospital, and hence would not be free at the end — a contradiction.

For the second kind of instability, suppose that (h_i, s_j) is a pair that causes instability. Then h_i must have offered a position to s_j, for otherwise it has p_i residents all of whom it prefers to s_j. Moreover, s_j must have rejected h_i in favor of some h_k which he/she preferred; and s_j must therefore be committed to some h_ℓ (possibly different from h_k) which he/she also prefers to h_i.

Solution 2:

The answer is Yes. A simple way to think about it is to break the ties in some fashion and then run the stable matching algorithm on the resulting preference lists. We can for example break the ties lexicographically that is if a man m is indifferent between two women w_i and w_j then w_i appears on m's preference list before w_j if $i < j$ and if $j < i$ w_j appears before w_i. Similarly if w is indifferent between two men m_i and m_j then m_i appears on w's preference list before m_j if $i < j$ and if $j < i$ m_j appears before m_i.

Now that we have concrete preference lists, we run the stable matching algorithm. We claim that the matching produced would have no strong instability. But this latter claim is true because any strong instability would be an instability for the match produced by the algorithm, yet we know that the algorithm produced a stable matching — a matching with no instabilities.

The answer is No. The following is a simple counterexample. Let $n = 2$ and m_1, m_2 be the two men, and w_1, w_2 the two women. Let m_1 be indifferent between w_1 and w_2 and let both of the women prefer m_1 to m_2. The choices of m_2 are insignificant. There is no matching without weak stability in this example, since regardless of who was matched with m_1, the other woman together with m_1 would form a weak instability.