Yiduo Ke

1.

a. Yes. Consider a table called STATES that's a slightly modified REJECTED table.

   STATES[m][w] = null if either man m or woman w has not proposed to the other.

   STATES[m][w] = 0 if man m and woman w are tentatively matched.

   STATES[m][w] = 1 if either man m rejected woman w or w rejected m.

   All cells of STATES start out as null.

   Each cell can only have the following changes:

   1) From null to 0 (accept a proposal)
   2) From null to 1 (reject a proposal because the proposer is worse than the proposed's current spouse)
   3) From 0 to 1 (dumping someone)

   Thus, there can at most be n^2 proposals because

   - Case 1 needs precisely one proposal and disqualifies case 2 from happening to the same cell
   - Case 2 needs precisely one proposal and disqualifies cases 1 and 3 from happening to the same cell, making the cell unable to be operated on again
   - A cell (m,w) that goes through both cases 1 and 3 needs two proposals, the first one matching m and w, and the second one making w dump m and rematching her with with another man (or vice-versa), incurring another cell to go through case 1.

   The number of proposals hence always equals the number of non-null cells, and since there are only n^2 cells in STATES, there are at most n^2 proposals.

b. No. Since perfect matching means everyone is matched to exactly one other person, we'll show a counterexample where not everyone is matched:

Preferences of man in decreasing order	man	woman	Preferences of woman in decreasing order
(A, B)	X	A	(Y,X)
(B, A)	Y	B	(X,Y)

   One possible order of proposals and what happens:

   1) X proposes to A. A accepts.
      Matches: (X,A). Rejections: none
   2) A proposes to Y. Y accepts. A dumps X. X is single.
      Matches: (Y,A). Rejections: (X,A)
   3) Y proposes to B. B accepts. Y dumps A. A is single.

Matches: (Y,B). Rejections: (X,A), (Y,A)

4) B proposes to X. X accepts. B dumps Y. Y is single.
   Matches: (X, B). Rejections: (X,A), (Y,A), (Y,B)
5) No one can propose anymore because there is 1 match and 3 pairs of rejections.

We end with only one match, but there are two people of each gender, so this algorithm cannot always produce a perfect matching.

c. Yes. Let's say there is an instability, a pair (m,w) that prefer each other over their current partners w_prime and m_prime, respectively. This can only happen because of one of the four following reasons:

1) w rejected m in favor of m_prime

2) m rejected w in favor of w_prime

3) w never proposed to m

4) m never proposed to w

Cases 1 and 2 are impossible because everyone only accepts better and better offers. So this means m and w don't actually prefer each other over their final partners, contradicting our initial assumption.

Cases 3 and 4 are impossible because everyone goes down their preference list to propose. If case 3 or 4 happened, that only means either w_prime is higher than w on m's preference list or m_prime is higher than m on w's preference list, contradicting our assumption that m and w prefer each other over their final partners.

d.

a) Yes, it's still guaranteed to end after n^2 proposals. Priority doesn't matter. Everything we proved before doesn't change just because some cells are filled before others.

b) Yes, it's guaranteed to output a perfect matching. This is essentially the Gale-Shapley algorithm, except women get to propose after men are done, and we proved in class that Gale-Shapley produces a perfect matching.

c) Yes, it's guaranteed to be stable. This is in fact the exact same as the Gale-Shapley algorithm; the women proposing after men are done part doesn't matter at all because we proved in class that each man already get the best valid partner, so they will reject all proposals from women after they're done. Since this is the same as the Gale-Shapley algorithm, which is stable, this is also stable.