CPSC 320 Assignment #1

January 4, 2015

Due date: Monday, 2015/01/12 at 5PM

Staple your solution behind the CPSC 320 cover page and submit in our handin box.

1. In this problem, you will prove or disprove this statement about the Gale-Shapley stable marriage algorithm with **women proposing to men**: All possible orderings of proposals by women result in the same number of iterations of the loop in the algorithm.

   As you work **every** problem, you should consciously think about strategies for working problems effectively. For example, working the problem in steps: trying to disprove it, trying to prove it, working examples, simplifying the problem, and finally trying to solve the whole thing.

   (Marking note: we will mark at least one part of the first two subproblems of our choice and the third subproblem. Label your parts **VERY** clearly or risk losing marks!)

   (a) Imagine the statement is false and work on disproving it, looking for insights as you go.

      i. Give an instance of the stable marriage problem with a **very** small number of women and men and trace the algorithm with two different possible orderings, counting up the number of iterations of the loop in each one.

      ii. Give a different instance (still very small!) and trace with a different number of iterations of the loop.

      iii. Note any patterns you see in the number of times each particular woman is the proposer in iterations of the loop.

      iv. Note any patterns you see in the total number of iterations of the loop.

   (b) Next, imagine the statement is true, still looking for insights.

i. Give an instance in which some woman w marries her most preferred man m in the stable match produced by the algorithm. For two different possible orderings of proposals, how many times does w propose? (Using one of your answers from the previous part is encouraged, if it has the right property.)

ii. Prove or disprove (for any instance) that a particular woman w who marries her most preferred man m in the stable match produced by the algorithm only proposes once.

iii. Give an instance in which some woman w marries her second most preferred man m in the stable match produced by the algorithm. For two different possible orderings of proposals, how many times does w propose? (Again, reusing an answer is encouraged!)

iv. Imagine that in an arbitrary instance a women w marries her k^{th} most preferred man m in the stable match produced by the algorithm. Suggest an upper- and lower-bound on the number of proposals she makes (ideally an exact number of proposals). Sketch a proof of your bound. (A "proof sketch" is not a proof; it's just a rough outline of how a proof might flow.)

(c) Build on your work from the previous two parts to either formally prove or formally disprove the statement.

2. Two nations have chosen teams to compete in determining which is better at a one-on-one sport (like tennis, speed algorithm design, or Hearthstone—um, "sport" defined loosely). Each has chosen a team of n players. The competition will have n games—each pairing a player from the first nation against a player from the second, with no player playing twice—and the nation whose players win the most games will win overall.

   Every player on both teams is ranked globally, where no two players have the same rank. Both nations believe that in any given game, the higher-ranked player will beat the lower-ranked player.

   You have been asked to produce an algorithm that will generate a match-up of players from the two sides (a solution). Ideally, the solution will be such that each nation will agree to the matchup rather than proposing an alternate match-up it likes better.

   (Marking note: we will mark at least one part of this problem of our choice. Label your parts **VERY** clearly or risk losing marks!)

(a) Give a clearly defined reduction from this problem to the stable marriage problem. In your reduction, players should correspond to people. (Note: a reduction of problem P_1 to problem P_2 should be a pair of algorithms: one that takes an instance of P_1 and transforms it into an instance of P_2 and another that transforms a solution to that instance of P_2 back into a solution to the instance of P_1. In this case, we do not expect your reduction to produce ideal solutions to P_1, as noted in the third part below!)

(b) Now, give a (very small!) instance—set of players for each country and their rankings—such that there is no match-up that meets the "ideal" criterion described above.

(c) Briefly explain what feature of this problem caused the reduction to "fail".

3. Give and briefly justify good best- **and** worst-case Θ-bounds on the runtime of (i.e., number of equality comparisons in) the algorithm below that determines whether an array of n numbers contains duplicates.

```
HAS_DUPS(array):
  For each index i from 1 to the length of the array:
    For each index j from (i+1) to the length of the array:
      If array[i] is equal to array[j]:
        Halt and return false
  Halt and return true
```