ESO207A/211: Data Structures and Algorithms	Due: 8 Aug 2014
Homework *1*	*Introduction: Stable Matching*

Problem 1. Consider n boys and n girls in a school seeking to be paired. Each boy has a preference list that ranks all girls and each girl has a preference list that ranks all boys. The set of n boys is divided into two categories: *normal* and *weird* and similarly for the girls. Suppose that there is a number k between 1 and n (included) such that there are k weird boys and k weird girls: thus there are $n-k$ normal boys and $n-k$ normal girls.

Suppose that the preference lists satisfies the following property. A weird person prefers a weird partner to a normal partner, and, a normal person prefers a normal partner to a weird partner.

Show that in every stable matching, every weird boy is paired with a weird girl.

Solution.We will show that if in any matching a weird boy is not paired with a weird girl, then such a matching is not stable.

Consider a matching where there is weird boy b_w who is paired with a normal girl g_n. Hence there must be a weird girl g_w who is paired with a normal boy b_n, since the number of weird boys and weird girls are both k. Consider the pair b_w, g_w. Each prefers the other to their current partner, since each is weird, and weird people only prefer weird partners. This is an unstable pair. This proves the statement.

Problem 2. *True or False.* Prove your answer or give a counterexample.

1. In every instance of the Stable Matching problem (done in the class), there is a stable matching containing a pair (b, g) such that g is ranked first on the preference list of b and b is ranked first on the preference list of g.

   Solution.*False.* See the instance and the matching in the Slide 7 (titled "The Problem") in Lecture 1. The instance has the property that for every boy, his first preference girl does not have him as her first preference. Hence, in every stable matching of that instance, there will be no pair (b, g) such that they are ranked first on each other's preference lists.

2. Consider an instance of the stable matching problem where there is a boy b and a girl g such that g is ranked first in the preference list of b and b is ranked first in the preference list of g. Then in every stable matching S for this instance, b is paired with g.

   Solution.*True.* Suppose there is a matching in which b and g are not paired with each other. Then, b, g form an unstable pair, since each prefers the other (first preference) to their current partner. Hence the matching is not stable. Thus in every stable matching, b is paired with g.

Problem 3. *College Counselling.* Consider a scenario where there n students and m colleges, with the ith college having k_i seats. Each student has his/her own ranking for the m colleges. Each college has a ranking of the students in order of preference. Assume that the total number of seats in the m colleges is less than the number of students.

You have to design an algorithm that assigns students to colleges such that all available positions are filled and the assignment is *stable*. The assignment of students to colleges is stable if neither of the following two kinds of situations (or instability) arises.

- *Instability 1.* There are students s and s' and a college c such that
  - s is assigned to c.
  - s' is not assigned to any college.
  - c prefers s' to s.
- *Instability 2.* There are students s and s' and colleges c and c' such that
  - s is assigned to c.
  - s' is assigned to c'.
  - s prefers c' to c.
  - c' prefers s to s'.

An assignment of students to colleges is said to be *stable* if neither kind of instability arises.

Show that there is always an assignment of students to colleges that is stable and give an algorithm for finding such an assignment.

Solution. The algorithm is a direct variant of the Gale-Shapely algorithm for stable pairing discussed in the class.

Algorithm **College-Student Assignment**
Initially all students have no offers and no college has made an offer
while there is a college c with a position to offer and
hasn't made an offer to every student do {
College c makes an offer to the highest ranking student s in c's preference list
to whom c has not yet made an offer
if s has no offer in hand {
s keeps the offer
}
else {
let s have an offer from college c'
if c is more preferable to s than c' {
c''s offer is rejected by s–position of c' becomes free
and s keeps offer from c
}
else {
s prefers its current offer
c's offer is rejected–position of c remains free
}
}

The algorithm works as follows. Each college makes offers in order of decreasing order of preference to students. College c_i has k_i seats to offer. If a college has a seat to offer, then it makes an offer to the highest ranking student in its preference list to whom it has not yet made an offer. A student's action is as follows. A student keeps at most one offer *in hand*. If a student does not yet have an offer in hand, then he/she takes the first position offered to him/her. Thereafter, if an offer is

presented to the student, then the student compares the new offer with the offer in hand. If the current offer is more preferable to the offer in hand, then, the student takes the current offer in hand and rejects the earlier one. Otherwise, the current offer is rejected. The algorithm terminates when there are no more positions to offer by any of the colleges.

The following properties are direct extensions of the Gale-Shapely pairing algorithm.

1. A student s always has an offer in hand from the time he/she is first offered. Moreover, the college offers that the student has in hand progressively becomes better (or stays static, but does not get worse) over time in terms of his/her preference list. This is true since a student only rejects the current offer on hand for a better offer.

2. The sequence of students who which a college proposes gets worse in terms of the college's preference list.

In each step, a college may give an offer to a student who has not been offered by the college earlier. Thus each college may make at most n offers. There are m colleges, so at most mn offers may be made. The number of iterations of the algorithm is bounded by mn.

Suppose the algorithm terminates with some assignment. Consider instability 1, let s be assigned to c, c prefers s' to s and s' does not have any offer. However, this cannot occur, since c would have made an offer to s' before making an offer to s, and so, s' will have an offer from c or some better college until the end. Hence, s' remains assigned to a college.

Consider instability 2. Let s be assigned to c, s' be assigned to c', but s prefers c' to c and c' prefers s to s'. Since, c' prefers s to s', it would have made an offer to s before it made an offer to s'. Since, at some point then or therafter, the offer of c' was rejected by s, it means that s had a better offer than c', and therefore at the end, continues to have a better offer than c' (property 1 above). Since s is paired with c, this means that s prefers c to c', which is a contradiction to the assumption that s prefers c' to c. In other words, such an instability cannot happen.