CS180 Midterm Exam

1. (25 pt) Consider the generalization of stable matching problem where a certain man-woman pairs are forbidden. The set of forbidden pairs is denoted as F. Each man m ranks all the woman w for which $(m, w) \notin F$ and each woman w ranks all the man m for which $(m, w) \notin F$. Consider the following algorithm for finding a stable matching that consists of only unforbidden pairs:

```
Initially all m ∈ M and w ∈ W are free; S = ∅
While there is a man m who is free and hasn't proposed to every woman w for which (m, w) ∉ F
    Choose such a man m
    Let w be the highest-ranked woman in m's preference list to which m has not yet proposed
    If w is free:
        Add (m, w) to solution S
    Else (w is current matched with m′):
        If w prefers m′ to m
            m remains free
        Else
            Replace (m′, w) by (m, w) in S
Return solution S
```

Answer **true or false** for the following questions:

(a) Any woman w remains engaged from the point at which she receives her first proposal, and the sequence of partners to which she is engaged get better and better.

(b) If a man m is free at the end of the algorithm, then he must have proposed to every non-forbidden woman.

(c) If a woman w is free at the end of the algorithm, then it must be that no man ever proposed to w.

(d) At the end of the algorithm, there can be a man m and a woman w, such that $(m, w) \notin F$, but neither of which is part of any pair in the matching S.

(e) At the end of the algorithm, there can be a pair $(m, w) \in S$ and a man m' that is free, $(m', w) \notin F$, but such that w prefers m' to m.

2. (25 pt) We are given an undirected graph in a very different format. We still know a number n, denoting the number of nodes in the graph. However, instead of being given the adjacency list of the graph, we are given a sorted "non-adjacency list." The non-adjacency list means there is no edge between the node and the nodes on the list, and if a node pair is not on the list, then there exists an edge. For example, if the sorted linked-list of node 3 is 1 -> 2 -> 10 -> NULL, it means that there are no edges between pairs: (3,1),(3,2), and (3,10), but all the other (3, j) pairs exist. Assume that there are totally m elements in the non-adjacency list and $m > n$, design an algorithm to check whether the given graph is connected in $O(m)$ time. Explain why your algorithm is correct and why the time complexity is $O(m)$.

3. (25 pt) Given a Directed Acyclic Graph (DAG) $G = (V, E)$, design an algorithm to determine whether there exists a path that can visit every node. The algorithm should have time complexity of $O(|E| + |V|)$. Prove why your algorithm is correct.

4. (25 pt) There is an array with n integers, but the values are hidden to us. Our goal is to partition the elements into groups based on their values — elements in the same group should have the same value, while elements in different groups have different values. The values are hidden to us so you cannot directly call for a value, like Arr[i]. However, we can probe the array by calling the function **query()**. The function takes a subset of elements as inputs and returns the number of unique integers in this subset. Design an algorithm to partition these n elements in $O(n \log n)$ queries. (**query()** is called $O(n \log n)$ times)