CS 4820, Spring 2019	**Homework 2, Problem 2**
Name: Yanling Wu	
NetID: yw996	
Collaborators: Zitao Zheng (zz632), Yunting Li (yl2989)	

(2) *(15 points)* Consider the following simplified model of how a law enforcement organization such as the FBI apprehends the members of an organized crime ring. The crime ring has n members, denoted by $x_1, \ldots, x_n$. A *law enforcement plan* is a sequence of n actions, each of which is either:

- apprehending a member x_j directly: this succeeds with probability q_j; or
- apprehending a member x_j using another member, x_i, as a decoy: this action can only be taken if x_i was already apprehended in a previous step. The probability of success is p_{ij}.

Let's assume that, if an attempt to apprehend x_j fails, then x_j will go into hiding in a country that doesn't allow extradition, and hence x_j can never be apprehended after a failed attempt. Therefore, a law enforcement plan is only considered *valid* if for each of the crime ring's n members, the plan contains only one attempt to apprehend him or her.

Assume we are given an input that specifies the number of members in the crime ring, n, and the probabilities q_j and p_{ij} for each $i \neq j$. You can assume these numbers are strictly positive and that $p_{ij} = p_{ji}$ for all $i \neq j$.

(2a) *(5 points)* Design an algorithm to compute a valid law enforcement plan that maximizes the probability of apprehending all of the crime ring's members, *in the order* $x_1, x_2, \ldots, x_n$. In other words, for this part of the problem you should assume that the j^{th} action in the sequence should be an attempt to apprehend x_j, and the only thing your algorithm needs to decide is whether to apprehend x_j directly or to use one of the earlier members as a decoy, and if so, which decoy to use.

(2b) *(10 points)* Design an algorithm to compute a valid law enforcement plan that maximizes the probability of apprehending all of the crime ring's members, *in any order*. In other words, for this part of the question, your algorithm must decide on the order in which to apprehend the crime ring's members *and* the sequence of operations to use to apprehend them in that order.

Solution:

(2a) Algorithm:

Create the set D to store the x that apprehend directly;
Create the set U to store the x that use the earlier member as decoy;
If j == 1:
    Should apprehend x_1 directly, store it into D;
For j in n:
    For($i = 1; i < j; i++$):
        Find the maximum value between q_j and the p_{ij};
    EndFor

If maximum value is q_j:
Should apprehend x_j directly, store it into D;
Else:
should use the earlier member as decoy and store it into U;
EndFor

Time complexity Running time should be $O(n^2)$, because the statement of "for every j in n:" will take $O(n)$ time and finding the the maximum p_{ij} for every j will take $O(n)$, so overall the algorithm takes $O(n^2)$ time.

Proof
For this question, the apprehending order is given and the apprehended chance of one member only depends on the probability that arrest him directly and the probabilities that use the arrested members as decoy to apprehend him. So if we compare these two kinds of probabilities' maximum value, the maximum probability will decide the way to arrest.

(2b)
Algorithm:
Create the set D to store the x that has been apprehend;
Sort the set of probability Q in descending order;
For i in n:
Sort the set $p_{ij}, j \in (1, n)$;
Store the sorted p_{ij} as the i_{th} row of the array P;
EndFor
Find the x_{max} with maximum probability to arrest directly in Q as the first one to arrest and store (max,0) to D and remove the q_{max} from Q;
For i in n-1:
Select x_{max} with the maximum probability to arrest directly q_{max};
For every member in D:
Find their maximum probability as decoy to arrest another member who has not been arrested from P separately;
EndFor
Compare current all maximum probability, find the maximum one, max_p;
If $max_p == q_{max}$:
Then arrest max directly and store (max,0) into D and remove the q_{max} from Q and remove p_{ij} from P;
Else (donate the maximum probability as p_{ij}):
compare the p_{ij} with q_j;
If $p_{ij} > q_j$:
Use member x_i as decoy to arrest x_j, store (x_j, x_i) to D and remove q_j from Q;
Else:
arrest directly and store (x_j,0) to D and remove the q_j from Q;
EndFor
Return D

Note that: there will be many pairs in set D, the first number represents the member that

arrest, the second number represents how to arrest him. If the second number equals to zero, it means this member is arrested directly, or he is arrested by the second member as decoy.

Time Complexity Running time of the worst case should be $O(n^3)$, the average running time is $O(n^2 logn)$

Proof
Actually, I am not sure how to give the correct proof. It is a little bit obvious. Because every time we decide who and how to arrest a member, we choose the one who has not been apprehended but with the highest probability to arrest successfully. So in the end, this algorithm will generate the sequence with the maximum probability of apprehending.