Programming HW # 2

Graphs are used extensively to model networks of various kinds. In this assignment, you will see their application to the field of peer-to-peer mobile networks.

Suppose you were at ACL, and in the process of trying to decide which concert to attend you experienced significant trouble in reaching your friends through cellphones. Such a phenomenon is common during large events because of the heavy load on cellular towers in the area. But unlike many others who were complaining about the terrible cellular coverage, you saw an opportunity to strike gold. You came up — or, at least you thought you did — with this next big idea in mobile telephony: use peer-to-peer technology over mobile devices rather than routing text messages through cellular towers! Specifically, you proposed that a text message be forwarded to every mobile device that is nearby (for now ignore how this is done, just assume that it is; using BlueTooth/IR or some other means). And then, each of those mobile devices forward the message again to their nearby neighbors and so on. Thus, given sufficient time, the message should hopefully be delivered to its intended recipient.

Before dropping out of school and investing all of your college fund in this idea though, you want to do a preliminary analysis to evaluate its practicality. Your first step is to analyze some data generated by a reliable simulator that generates *communication time-traces* to model the temporal vicinity of mobile devices. Based on this data, you have to predict 'timely' delivery of the text messages. Formally, you have to determine if your approach ensures that a message sent by device s at time t_s can be delivered to device t by time t_u.

The formal specification of this analysis problem is as follows:

There are n devices: U_0 to U_{n-1}, such that each device U_i has a unique id i, $0 \leq i \leq n-1$. Every time-trace generated by the simulation represents a time instance at which a pair of devices was in contact with one another. A trace is represented by three integers separated by a single white-space. The first two integers i and j represent the unique numbers (ids) of the devices U_i and U_j, and the third integer denotes the time of communication between these two devices. Hence, a trace 'i j t_k' means that U_i and U_j were in contact at time t_k.

Assuming that the transfer of message data is instantaneous, and each device can store infinite messages, the rule for message propagation is: if there is a trace 'i j t_k', then if either of U_i or U_j holds some message(s) at t_k then the message(s) are forwarded to each other. That is, if U_i has message x, and U_j has a message y and they communicate at time t_k, then instantaneously both U_i and U_j shall end up with messages x and y in their local storage at t_k.

Given a set of traces, we define a query in the form of four non-negative integers separated by a single white-space: 'i j t_s t_u'. This query encodes the question: can a message sent by U_i at time t_s be delivered to device U_j by time t_u. The following two examples further illustrate the scenario:

Example 1: Consider the following four traces.

0 1 0
1 2 1
2 3 1
3 4 2

Then, for the query '0 4 0 4' the answer is yes. This can be easily verified as follows. U_0 and U_1 communicate at time 0, so U_1 receives the message at time 0. U_1 and U_2 communicate at time 1, so U_2 receives the message by time 1. U_2 and U_3 communicate at time 1, so U_3 receives the message by time 1. U_3 and U_4 communicate at time 2, so U_4 receives the message by time 2.

Example 2: Consider the following two traces.
0 1 2
1 2 1

If the query is: '0 2 0 2', then the answer is no. Even though U_0 communicated with U_1 and U_1 communicated with U_2, the message will not be delivered to U_2 by time 2; or for that matter ever if these are the only two traces.

Your task: Given that the traces are guaranteed to be provided in non-decreasing order of time, design and implement an algorithm that does the following: for a set of traces, determine if a message sent by device s at time t_s can be delivered to device t by time t_u. Your algorithm should take $\mathcal{O}(m+n)$ time where m is the number of traces, and n is the number of cellphones/devices.

Your program will read from an input file that will be the only argument to your program, and write to standard output. The formats of the input and output are defined ahead.

Input File Format: Your program will read its input from a plain-text file. The first line of the input contains two positive integers, n and m, separated by a single space, where n represents the number of devices, and m denotes the number of trace entries. Each of the next m lines contains a trace (one trace per line) in the format defined above. The last line of the input contains the query whose format we defined above.

Output Block Format: The output of your program must follow the format described next. If the answer to the query (encoded in the last line of the input file) is no, then the output should be 0.

If the answer to the query is yes, then the first line of output should print the non-negative integer k that denotes the number of exchanges that were required for delivering the message at the destination, starting from the source node in the query. The subsequent k lines should print the trace entries involved in the message delivery sorted in the non-decreasing order of time.

Note: It is possible that there is more than one possible sequences that lead to timely message delivery. Your program needs to only output one such sequence.

For your submission, you need to:

Part 1: Provide pseudo code for an algorithm that runs in $\mathcal{O}(m+n)$. Justify the runtime complexity of your algorithm. (15 points)
Part 2: Prove the correctness of your algorithm. (5 points)
Part 3: Implement the algorithm that reads a given input file of the format described above, and produces an output as per the required format. (80 points)

Following the general programming homework guidelines regarding the runnable file names, submission zip filename, etc.