CSE 591: Foundations of Algorithms
Programming Assignment# 1
Due: Submit all code, report, README file on Blackboard by 4:30 pm on Feb. 12, 2016

Submission Instructions:

- This is a group assignment. You can form groups of 3.
- You need to submit a report and all your code with all instructions on how to run your code in a README file by the due date on Blackboard.
- The report must cover the following points - Algorithm to solve Problem 1, proof of optimality of your algorithm, programming language used and reason for doing so, sample output on 3 small examples of your own.
- The README file should contain detailed instructions on how to run your code, all dependencies and settings. Please upload your complete code. For e.g., if you are using Eclipse, make sure to submit the entire Eclipse project.

Problem 1: Suppose that you are given a set of cities and there are some pre-existing roads connecting some of the cities. All the roads are such that they can carry traffic only in one direction - hence all the roads are unidirectional. You wish to find the shortest path from a specific city s to a specific city t. Here, length of a path from s to t is measured by the sum of the lengths of each road in the path. It turns out that the current shortest $s-t$ path is just too long to be acceptable. So, you wish to build a new uni-directional road connecting two cities which did not already have a road connecting them in that direction. There are k possible options. But, you have only enough money to build at most one new road. For e.g., $k = 2$ and you have the option of building a new road going from city 0 to city 1 with road-length 5, OR, a new road going from city 2 to city 3 with road length 6. But, beware - none of the new roads might reduce the length of the shortest $s-t$ path. Find out which are the new roads, if any, whose construction does reduce the length of the shortest $s-t$ path. If such roads do exist, find out construction of which road gives the maximum reduction in the length of the shortest $s-t$ path.

(a) Give an algorithm which runs in time $O(m + n \log n + k)$ where m is the number of pre-existing edges and n is the number of cities.

(b) Prove that your algorithm always gives optimal solution.

Coding Instructions:

- Your code should be able to take input both from the console as well as from an input file. The input file will be of the following format: first line containing the number n of cities in the city graph (enumeration begins at 0), second line containing the number m of uni-directional pre-existing roads m of the city graph, this is followed by m lines each containing a road of the form of $< city1 >:< city2 >:< road - length >$ (for e.g., 3:4:10 specifying a road going from city 3 to city 4 with road-length 10 units). The next line contains a number k, this is followed by k lines each containing a new optional uni-directional road of the form $< city1 >:< city2 >:< road - length >$ (for e.g., 3:5:7 specifying a new optional uni-directional road between city 3 and city 5 with road-length 7 units). Finally, the last two lines of the input will contain s and t, i.e., the source and the target cities.

- You can choose your output format. But you need to output the following information.

  1. List of the new roads, if any, whose construction does reduce the length of the shortest $s - t$ path.
  2. If such roads do exist, output the road whose construction gives the maximum reduction in the length of the shortest $s - t$ path.

- Your option menu must contain option to select the input type. If the input type is a file, take the name of the file as the next input. If the input type is from console, your code must be prepared to take input from the user in the same sequence and format as the input file.