Assignment 1: Congestion with Shortest-Path Routing

***Due date:** Friday, March 27, 2020 at 22:00*

This is an individual assignment. You may discuss it with others, but your code and documentation must be written on your own.

Write a program called `shortest_paths` that, given a network and a set of flows computes the total utilization of each link in the network when the given flows are routed using a basic shortest-path routing scheme. The network is given as a directed graph where each arc is associated with a cost or distance. Each flow is defined by a source node, a target node, and a flow demand. The total utilization of a link is the sum of the demands of all flows routed through that link.

Input

Normally, the program takes two command-line parameters. The first parameter is the name of the file containing the network graph. The second parameter, which is optional, is the name of the file containing the flows. If the second parameter is not given, then the program must use a default set of flows, one for each pair of nodes in the network, and all of them with traffic demand 1.

The network graph is given in a text file containing one arcs per line. Each arc is defined by a source node, a target node, and a distance (cost), separated by spaces. Distances are floating-point numbers. For example, the following input defines a very simple graph consisting of three nodes, labeled “Lugano”, “Zurich”, and “Basel”, respectively, with a bi-directional between Lugano and Zurich with distance or cost 206.5, and a bi-directional between Zurich and Basel with distance 83.9.

```
Lugano Zurich 206.5
Zurich Lugano 206.5
Zurich Basel 83.9
Basel Zurich 83.9
```

The flows are given in a text file containing one flow per line. Each flow is defined by a source node, a target node, and a flow demand, separated by spaces. For example, the following input defines two flows, one from Lugano to Basel with a demand of 10.2 units of traffic, and one from Lugano to Zurich with a demand of 35.0.

```
Lugano Basel 10.2
Lugano Zurich 35.0
```

Output

The program must output, the total traffic crossing each link with the given flows. Each link must be written on the output with one link per line. The format. For example, using the network and flows of the examples above, for example running `./shortest_paths network flows`, assuming the network and the flows are stored in files called `network` and `flows`, respectively, the program must output the following link utilizations, although not necessarily in this order:

```
Basel Zurich 0
Lugano Zurich 45.2
Zurich Basel 10.2
Zurich Lugano 0
```

Running instead `./shortest_paths network`, and therefore using the default all-pairs unit flows, the output should be (in any order):

```
Basel Zurich 2
Lugano Zurich 2
Zurich Basel 2
Zurich Lugano 2
```

Submission Instructions

You may write your solution in C, C++, Java, or Python. Package all the source files plus a README file in a single zip or tar archive. Make sure that you include all the necessary components to build and run your solution on a standard installation of a C, C++, Java, or Python environment. In particular, make sure your solution works with the most basic command-line tool, outside of any integrated development environment.

Add comments to your code to explain sections of the code that might not be clear. Use the README file to add general comments to properly acknowledge any and all external sources of information you may have used, including code, suggestions, and comments from other students. If your implementation has limitations and errors you are aware of (and were unable to fix), then list those as well in the README file.

Submit your solution package through the iCorsi system.