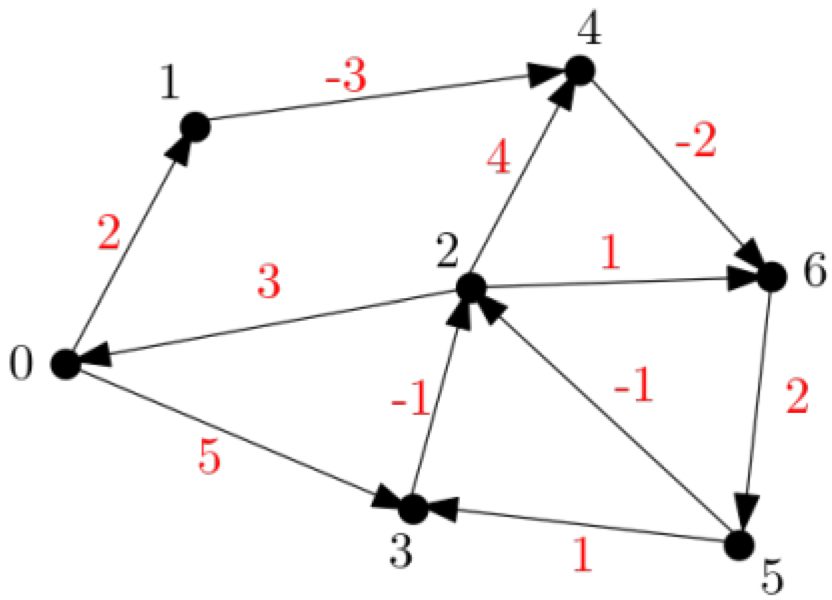
Bellman Ford Algorithm

Implement the Bellman-Ford algorithm to compute single source shortest paths in a graph that can have edge weights that are either positive or negative.

We will assume that the graph is a directed graph.

You should implement the graph with an adjaceny list representation.

There are two input files for this project. The first is graphInput.txt which contains all of the information you need to construct the graph. The provided file gives the input for the following graph.



In this figure, the black numbers are the IDs of the vertices, and the red numbers are the weights on the edges.

The graphInput.txt will first contain n, the number of vertices in the graph. You should assume that the vertices are labeled 0, 1, …, n-1. Then on each line there will be the information of a single edge of the graph in the following format: startingVertex endingVertex edgeWeight For example, the edge connecting vertex 0 to vertex 1 with a weight of 2 is given by “0 1 2”.

The second input file is shortestPaths.txt which contains the shortest paths that we want to compute in the following format: startingVertex destinationVertex You then want to run the Bellman-Ford algorithm starting from startingVertex and output the length of the shortest path to destinationVertex.

You only need to report the length of a shortest path, not the actual path itself.

You then output in the following format: A shortest path from startingVertex to destinationVertex has length pathLength.

So for example, if shortestPaths.txt has “0 4”, you would want to run Bellman-Ford from 0 until we compute a shortest path to vertex 4. This path has length -1. So you would output: A shortest path from 0 to 4 has length -1.

You can assume that a shortest path will always exist for the vertices we ask you to compute the paths for. Namely, you can assume there will not be a negative weight cycle in the graph, and that there will always be at least one path from the startingVertex to the destinationVertex.