# Lab: Graphs Bellman-Ford, Longest Path in DAG

This document defines the lab for the ["Algorithms – Advanced (C#)" course @ Software University](https://softuni.bg/trainings/3810/algorithms-advanced-with-c-sharp-september-2022).

Please submit your solutions (source code) to all below-described problems in [Judge](https://judge.softuni.org/Contests/2575/Graphs-Bellman-Ford-Longest-Path-in-DAG-Lab).

## Bellman-Ford

Find the **shortest** **path in a graph** and print it as a sequence **from S vertex to D vertex**.

Implement the Bellman-Ford algorithm.

### Input

The input comes from the console.

* On the first line, you will receive an integer – n – the number of **nodes**.
* On the next line, you will receive an integer – e – the number of edges.
* On the next e lines, you will receive an edge in the following format:"{from} {to} {weight}".
* On the last two lines, you will receive source and destination nodes.

### Output

* Print "Negative Cycle Detected"if you detect a negative cycle.
* Otherwise, print the **shortest path** separated by a space and the total **distance**.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 6  8  1 2 8  1 3 10  2 4 1  3 6 2  4 3 -4  4 6 -1  6 5 -2  5 3 1  1  6 | 1 2 4 3 6  7 |
| 4  4  1 2 1  2 3 -1  3 4 -1  4 1 -1  1  4 | Negative Cycle Detected |

## Longest Path

Find the **longest** **path from S to D in a graph** and print the **total distance** of that path.

### Input

The input comes from the console.

* On the first line, you will receive an integer – n – number of nodes.
* On the second line, you will receive an integer – e – number of edges.
* On the next e lines, you will receive edges in the following format: **"{source} {destination} {weight}"**.
* On the next line, you will receive an integer – s – the start of the path.
* On the last line, you will receive an integer – d – the destination of the path.

### Output

* Print the **total distance** of the longest path.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  4  1 2 5  1 3 3  3 4 6  4 2 4  1  2 | 13 |
| 6  10  1 2 5  1 3 3  2 4 6  2 3 2  3 5 4  3 6 2  3 4 7  4 6 1  4 5 3  5 6 4  1  6 | 21 |