Homework 3 - Shortest Path

Introduction

The purpose of this homework is to learn and implement 2 different single source shortest path algorithms.

Description

1. You need to implement 2 different shortest path algorithms.
   - **Dijkstra's Algorithm**
   - **Bellman-Ford Algorithm**
     - You have to detect if there is any negative loop in the graph when implementing this algorithm.
2. Input / Output
   - **Input**
     The input command contains the **directional** graph information and the start node and the target node used for the shorted path calculation.
     The first line contains two integers represent the number of nodes(*m*) and edges(*n*) in the graph.
     The second line contains two integers for the shorted path calculation: the start node(*s*), and the target node(*t*).
     After that, the following *n* lines represent the edge information. Each line contains three integers: start node(*a*), end node(*b*), and the cost(*c*).
     In each line, all the integers will be separated by space.
     $0 < m <= 1000$, $0 < n <= 2000$, $0 <= s, t, a, b < m$.
     For Dijkstra's Algorithm, $0 < c < 10000$
     For the Bellman-Ford Algorithm, $-10000 < c < 10000$.
   - **Output**
     The output should contain the cost if there is no negative loop in the graph.

     If there is no negative loop in the graph:
     The output should contain one integer that represents the

cost of the shortest path.

If there is a negative loop in the graph, please output the string "Negative loop detected!"

3. Discussion
    - Describe how you detect the negative loop in the Bellman-Ford Algorithm.
    - If you need to print out the path of the shortest path, describe how it can be done?
    - Compare the time complexity of the two algorithms.

Constraint

- You are required to implement it in C++.
- You are free to use the C++ standard library.

Format Checking

https://www.hackerrank.com/ds-and-oop-homework-3
This judging page is for quickly checking the format and correctness of your sorting algorithm. The judging result is not used for grading. However, the submitted code should fulfill the format of the judging page above.

Grading

Correctness (60%)
- Dijkstra's Algorithm (30%)
- Bellman-Ford Algorithm (30%)

Paper Report (40%)

Paper Report Guideline

The paper report should be detailed, clear, and well-organized.

You have to describe how you implemented the algorithm and show the result. If you encountered any challenges during the implementation, it is encouraged to describe it in the Discussion as well.

Please write the paper report with the following bullets.

- Introduction
- Implement Details
- Results
- Discussion
- Conclusion

Submission

1. A zip file named <HW3_studentID.zip> includes the following
   - A folder named <ShortesPath_studentID> includes the following
     - Dijkstra_studentID.cpp
     - BellmanFord_studentID.cpp
   - A paper report named <report_studentID.pdf>
2. Uploaded onto new E3 platform before 07 / 03 (Fri) 11:55 pm
3. The late penalty will be 10% per day.