Homework 2 – Algorithm

Part 1 Sorting

Introduction

In this part you need to implement 2 different sorting algorithms and try to do comparison between them.

Description

1. You need to implement 2 different sorting algorithms.
    - Quick Sort
    - Merge Sort
2. You need to compare the results including:
    - Execution Time
        - Average Time
        - Fastest Time
        - Slowest Time
    - Stability
        - https://en.wikipedia.org/wiki/Sorting_algorithm#Stability
3. Note
    - You can implement it with array, vector or linked-list.
    - You can measure the execution time using <ctime> (time.h) http://www.cplusplus.com/reference/ctime/

Constraint

- You are required to implement in C++.
- Using std::sort is not allowed.

Format Checking

- https://www.hackerrank.com/contests/2021-ds-and-oop-homework2/challenges/q1-quick-sort-algorithm
- https://www.hackerrank.com/contests/2021-ds-and-oop-homework2/challenges/q1-merge-sort-algorithm

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.

Part 2 Minimum Spanning Tree

Introduction

In this part you need to implement 2 different minimum spanning tree algorithms and try to do comparison between them.

Description

1. You need to implement 2 different shortest path algorithms.
    - **Prim Algorithm**
    - **Kruskal Algorithm**
2. Input / Output
    - **Input**

      The input command contains the **undirectional** graph information used for the cost of the minimum spanning tree. The first line contains two integers represent the number of nodes(m) and edges(n) in the graph. 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.

      $0 < m <= 1000$, $m < n <= 200000$, $0 <= a, b < m$. $0 < c < 10001$
    - **Output**

      The output should contain one integer that represents the cost of the minimum spanning tree.
3. Discussion
    - Compare the time complexity of the two algorithms.
    - Which is better algorithm in which condition

Constraint

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

Format Checking

- https://www.hackerrank.com/contests/2021-ds-and-oop-homework2/challenges/q2-prim-algorithm
- https://www.hackerrank.com/contests/2021-ds-and-oop-homework2/challenges/q2-kruskal-algorithm
- This judging page is for quickly checking the format and correctness of your minimum spanning tree algorithm. The judging result is not used for grading. However, the submitted code should fulfill the format of the judging page above.

Part 3 Shortest Path

Introduction

In this part you need to implement 2 different single source shortest path algorithms and try to do comparison between them.

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 <= 3000, 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/contests/2021-ds-and-oop-homework2/challenges/q3-dijkstras-algorithm
- https://www.hackerrank.com/contests/2021-ds-and-oop-homework2/challenges/q3-bellman-ford-algorithm
- This judging page is for quickly checking the format and correctness of your shortest path algorithm. The judging result is not used for grading. However, the submitted code should fulfill the format of the judging page above.

Grading

Correctness (50%)

- Quick sort (5%)
- Merge sort (5%)
- Prim Algorithm (10%)
- Kruskal Algorithm (10%)
- Dijkstra's Algorithm (10%)
- Bellman-Ford Algorithm (10%)

Paper Report (50%)

- Part 1 Sort Algorithm (10%)
- Part 2 Minimum Spanning Tree Algorithm (20%)
- Part 3 Shortest Path Algorithm (20%)

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. You must write down three parts of it. Each part must have introduction, implement details, results, discussion, conclusion. (The list below each bullet is just for recommendation for your content.)

- Introduction
  - Introduction of different method/ algorithm
  - Etc.
- Implement Details
  - Steps
  - Etc.
- Results
  - Time complexity
  - Etc.
- Discussion

    - Your discover
    - Which is better algorithm in which condition
    - Challenges you encountered
    - Etc.
- Conclusion
    - What did you learn from this homework
    - How many time did you spend on this homework
    - Feedback to TA
    - Etc.

Submission

1. A zip file named <HW2_studentID.zip> includes the following
    - A folder named <HW2_studentID> includes the following
        - Quicksort_studentID.cpp
        - Mergesort_studentID.cpp
        - Prim_studentID.cpp
        - Kruskal_studentID.cpp
        - Dijkstra_studentID.cpp
        - BellmanFord_studentID.cpp
    - A paper report named <report_studentID.pdf>
2. Uploaded onto new E3 platform before 06 / 19 (Sat) 11:55 pm.
3. The late penalty will be 10% per day. The last submission is allowed before 06 / 22 (Tue) 11:55 pm.
4. If there is any wrong format. You will get 5% penalty each.
5. Plagiarism is forbidden. You will get 0 point if we find it.