CS 3005: Programming in C++

The Traveling Salesman Problem (Part 1 - Any Cycle)

The Traveling Salesman Problem (TSP) is a classic NP-Complete problem. Solutions to the problem can be applied to many fields.

There are several variations of the problem. For this assignment we will use the following description. As you search for more information and solutions, be sure that they apply to this version.

Problem Definition

Given a complete graph \( G \), with \( n \) vertices, \( v_1, v_2, ..., v_n \), and edge weights \( w(u, v) \), such that \( w(u, v) = w(v, u) \) and the edge weights obey the triangle inequality, find the lowest weight cycle that visits all vertices.

The triangle inequality means that it is never more costly to go from vertex \( a \) to \( c \), than it is to go from \( a \) to \( b \) then to \( c \).

A cycle that visits all vertices must follow a path from the starting location, \( v_1 \), and visit every vertex exactly once, then return to the starting location \( v_1 \).

Assignment

In this step of the assignment, you will be required to read the description of graph \( G \) from standard input (\texttt{std::cin}), then find any cycle, and write a description of the cycle to standard output (\texttt{std::cout}). In the next step, you will be required to find a good cycle.

Input Format

The first line of input will be a single integer, \( n \). The rest of the input will consist of \( n*(n-1) \) lines, describing the weights of the edges in the graph. Each line will contain two integers and one floating point number. In this example line, there is an edge between vertex 2 and vertex 4 that has a weight of 3.14159.

```
2 4 3.14159
```

The input will describe every edge in both directions, even though the weights are the same in both directions. The vertices will be numbered 1, 2, ..., \( n \).

Output Format

The output should consist of a single line with the vertices in the order your cycle visits them, starting with vertex 1. Vertex 1 should not be repeated at the end. After the vertices, the total weight of the cycle will be printed, followed by the quality of your cycle.

The cycle quality is calculated by this formula:

```
1 - (weight-of-cycle - minimum-possible-weight)/(maximum-possible-weight - minimum-possible-weight)
```

\texttt{minimum-possible-weight} is the cost of the lightest edge multiplied by \( n \). \texttt{maximum-possible-weight} is the same, but for the heaviest edge.

Requirements

- Program must read content from \texttt{std::cin}.
- Program must expect nothing but the input described above.
- Program must write results to \texttt{std::cout}.
- Program must write nothing except the output described above.
- Program must give a single newline at the end of the output.
- Program must find a valid cycle.
- Program must correctly calculate the cost of the cycle.
- Program must correctly evaluate the quality of the cycle.
Program executable file must be called TSP.

**Hints**

- Consider creating a class Graph that supports all of the operations and storage necessary to complete the assignment.
- Consider using a vector of vector of double to store the edge weights.
- Consider the trivial cycle that visits the vertices in numerical order 1, 2, 3, ..., n.
- Sample outputs are provided with the sample files.

**Sample Files**

- zip
- tgz

**Show Off Your Work**

To receive credit for this assignment, you must upload the source code (.h and .cpp files) and the Makefile to the Canvas submission system.

Additionally, the program must build and run. Any incorrect performance or memory errors will be counted against the assignment score. (Hint: Use valgrind or other memory checkers.)