Assignment: Approximation Algorithms

In this assignment, you are going use approximation algorithms to solve NP-Hard problems. The problem you will be solving is “Traveling Salesman Problem”, TSP henceforth.

TSP asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?"

In this assignment, you will implement the exact algorithm to solve TSP, and also implement an approximation algorithm to solve TSP on a special kind of graph where the distance among vertices are Euclidean distance. Thus, the graph maintains the following metric properties and the resultant tour is called Metric TSP:

1. $d(x, y) \geq 0$ for all x, y.
2. $d(x, y) = 0$ if and only if $x = y$.
3. $d(x, y) = d(y, x)$.
4. (Triangle inequality) $d(x, y) \leq d(x, z) + d(z, y)$.

For your convenience, the assignment is divided into several parts. Write appropriate functions and classes to implement them.

1. Create a Graph class to store a weighted graph.

2. Implement a function $\boldsymbol{Create\_Random\_Graph(int\ V)}$ that will return a complete weighted graph with V vertices and each edge $e \in E$ having weight within $[50, 200]$. *Make sure the graph follows all the metric properties.*

3. Implement a function $\boldsymbol{Exact\_TSP(Graph\ G)}$ that will return the optimum tour. Please follow [Vazirani & Dasgupta, Page 178] for the algorithm. Note that, the algorithm will have a runtime of $O(n^2 2^n)$.

4. Implement another function $\boldsymbol{Metric\_Approximation\_TSP(Graph\ G)}$ that will return a tour resulted from running the $2 * OPT$ approximation algorithm mentioned in [Vazirani & Dasgupta, Page 279]. For your understanding, the algorithm is given below:

Algorithm (Metric TSP – factor of 2)

1. Find an MST (Minimum Spanning Tree), T, of G.
2. Double every edge of the MST to obtain a Eulerian graph.
3. Find an $Eulerian\ tour$, τ, on this graph.
4. Let C be the tour that visits vertices of G in the order of their first appearance in τ.
5. Return this tour C.

5. To calculate the cost of a given tour, implement a function $\boldsymbol{Calculate\_Tour\_Length(Tour\ T)}$.

6. In your main function, first take an integer X, the number of test cases, from user input.

7. For $x = 1\ to\ X$, create a graph calling $Create\_Random\_Graph\ (int\ V)$ function you implemented above. Provide V=20 in all the cases,

8. For each graph, find the optimum tour and tour found by running the approximation algorithm mentioned above. Calculate the lengths of both the tours. Store both the tour lengths in two separate arrays.

9. Calculate the ratio $Approximate\_Tour\_Length_x / Optimal\_Tour\_Length_x$, for each test case $\boldsymbol{x}$. Print the pair (x, ratio) in one line for each test case.

Submission Guideline: You have to use C++/C/Java to code this assignment. Create a folder named with your 7-digit student id. Put all your sources files into the directory. Zip the directory. Submit the zip file to moodle. Failure to comply to the guideline will result in 30% penalty, irrespective of the marks received.