

CS 350 Project 5: Naïve Minimum Weighted Hamilton Circuit

Your task is to develop and analyze an algorithm to find the shortest Hamilton circuit for a weighted path. Since initially we will not worry about ways the algorithm can be improved, your solution can be thought of as a naïve or non-optimal solution. We will talk about optimization and problems with this approach in class once the project has been completed.

You should begin by developing an ADT for undirected graphs. This is a simplification of the ADT for directed graphs found in your text, so this step should be fairly straightforward and should go very quickly.

You will use this ADT as a representation of a map—the nodes will be cities and the weighted edges will reflect the distances between pairs of cities. The distance information for cities will come from a supplied data file, *distances.txt*, available for download from the class web site. You should download and use this file. Your project should include code to read this file directly. This is a text file that you can open and examine to discover the layout and format. (The original data was collected using the website <http://www.geobytes.com/CityDistanceTool.htm>.)

With your program, you should be able to specify the number of cities to use. Your program will create a graph with the specified number of cities and will then exhaustively evaluate the possible circuits for just those cities. For example, if you specified four, you would investigate circuits among Boston, Chicago, London, and Moscow ignoring all the other cities in the data file. Your code should report the shortest cycle among these cities and the total distance traveled.

You should run your code varying the number of cities visited. Time these runs and use that information to estimate the complexity of your algorithm.

You may work on this project in pairs (teams of two) or individually (teams of one). However, as always, you should not discuss this project with anyone outside your team other than the instructor. Everyone on a team will receive the same grade. You should turn in one printed copy of the code and results for each team at the start of class on the due date, and you should submit as an email attachment one copy of the code per team before the start of that class. Each individual in the class, regardless of whether they are working on the team, should send email describing problems they encountered and estimations of how long they took for each phase of the project: designing the code, implementing the code, and debugging the code. In addition, each individual should write a brief description of how this algorithm might be improved and a brief comparison of the appropriateness of the graph ADT to other possible ADT for this particular problem.

As with other projects, this project counts as three routine homework assignments.

Due: Wednesday, December 3 by the start of class.