CS340 Project 6

Background: This assignment compares two approaches to the travelling salesman problem. The first approach is using a brute-force method to compute the exact solution. The second approach is to use a MST-based approximation algorithm.

Program specifications: All programs should be well-structured, meaning that you should follow the principles of good design: a modular set of classes and functions, no function should be too long, each function/class should correspond to a well-defined and cohesive task/concept, etc.

You will be supplied with 10 graphs (with 13 nodes each) in the format described in Project 4. Each vertex should be considered a “city” in the salesman’s route. The graphs are planar, so you will also be supplied with a file giving geographic coordinates for each vertex in the graph.

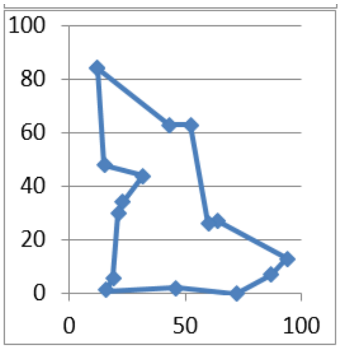
The output for each algorithm is a "tour", i.e., a permutation of the city numbers, and the total distance traveled by such a tour (don't forget to include the distance from the last city back to the first one). The distance between two consecutive cities in the tour is the standard Euclidean distance. For example, a possible output is

```
tour:  1 2 4 3 12 10 5 6 11 8 7 9
distance:  74.62
```

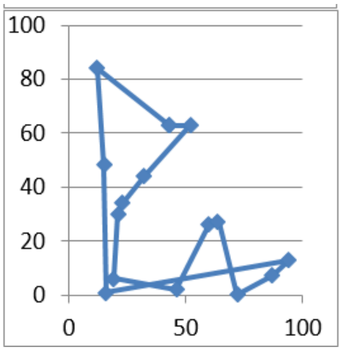
The brute force algorithm should generate all permutations of the cities. For each permutation, calculate the total distance traveled by this tour (don't forget to return to the starting vertex). Keep track of the tour with the smallest total distance as you test all the permutations. The output is the tour with the smallest total distance, as well as the corresponding distance.

The approximation algorithm is from Section 35.2.1 in your textbook. You may reuse your MST code from Project 4.

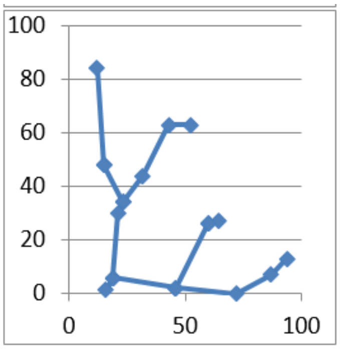
Experiments: After writing the code, you are to run both algorithms on all 10 supplied graphs, keeping track of running times. Create a plot that shows the running time for each graph for both the brute-force and approximation algorithms. Second, create a table that demonstrates the error in the answer returned by the approximation algorithm for each input. Do this by comparing the (exact) answer returned by the brute force algorithm to the answer returned by the approximation algorithm. Give the ratio of the two numbers. For the two graphs with the worst ratios, show results like this:



True shortest tour



MST Approximation Tour



Minimum Spanning Tree

Report: You should submit a report that includes the plot and table, as well as the analysis of the two worst approximations mentioned above. Answer the following questions:

1. How well does the second algorithm approximate the exact answer? Attempt to understand and explain any discrepancies you note.
2. Can you detect properties of the MST approximations that make some worse than others? What are properties of a good tour, and what are properties of a bad tour?
3. Can you think of any techniques to speed up your brute force algorithm?

The project will be graded according to the following rubric:

Code: Overall	0pts No code supplied	5pts Code sloppy and unprofessional	10pts Code adequate	15pts Code of professional quality
Code: Brute force method	0 pts No brute force method	10 pts Brute force method with major errors	15pts Brute force method with minor errors	20pts Brute force method with no errors
Code: MST-based approximation	0 pts No MST-based approximation	10 pts MST-based approximation with major errors	15pts MST-based approximation with minor errors	20pts MST-based approximation with no errors
Report: Plot of running times	0pts No running times	5 pts Plot with errors	10pts Plot with no errors	
Report: Illustration of 2 graphs with worst ratios	0pts No illustration	10pts Illustration with errors	15pts Illustration with no errors	
Report: Analysis	0pts No analysis	10pts Analysis with major errors	15pts Analysis with minor errors	20pts Analysis with no errors