This project asks you to investigate a Traveling Salesman Problem using various algorithms.
The second page contains some useful reference information that will help you complete this project.

As part of a new job, you must travel to the eight cities shown in the following mileage chart.

	St. Louis	Pittsburgh	Nashville	Louisville	Houston	Fargo	Dallas
Boston	1141	561	1088	941	1804	1652	1748
Dallas	630	1204	660	819	243	1091	
Fargo	850	1129	1112	940	1304		
Houston	779	1313	769	928			
Louisville	263	388	168				
Nashville	299	553					
Pittsburgh	588						

(a) How many distinct Hamilton circuits are there for the weighted graph associated with this mileage chart? Note: A circuit traveled backward is not to be considered distinct from the original circuit.

(b) If it takes you three minutes to check each Hamilton circuit, how long will it take you to check all of the distinct Hamiliton circuits in this problem? (Use a unit of time appropriate to your answer.)

(c) Apply the cheapest-link algorithm to find a Hamilton circuit in the graph. Give three things: the list of the edges in the order in which they were added, the circuit as it would be traveled starting in Fargo, and the total mileage for the trip.

(d) Apply the nearest-neighbor algorithm with Fargo as the starting vertex. Give two things: the circuit as it would be traveled starting at Fargo, and the total mileage for the trip.

(e) Apply the nearest-neighbor algorithm with Nashville as the starting vertex. Give three things: the circuit as created starting with Nashville, the same circuit rotated to start at Fargo, and the total mileage for the trip.

Property: The complete graph with n vertices has $(n-1)!$ Hamilton circuits. Of these, half are repeats of the other half, but traveled backward.

Brute Force Algorithm for Solving TSPs

- List all possible Hamilton circuits for the weighted graph.
- For each Hamilton circuit, find the weight of the circuit.
- Of all of the circuits, the one(s) with the least weight is optimal, and is therefore a solution to the problem.

Cheapest Link Algorithm for Solving TSPs

- Pick the edge with the smallest weight first (in the case of a tie pick one at random). Mark the edge.
- Pick the next cheapest unmarked edge and mark it unless it closes a smaller circuit or results in three marked edges coming out of a single vertex. If there are ties, break them arbitrarily.
- Repeat until the Hamilton circuit is complete.

Nearest Neighbor Algorithm for Solving TSPs

- Choose a starting vertex.
- Whenever you are at a vertex, pick the next vertex to visit from among the ones you have not yet visited that is nearest (has the smallest weight on the edge) to the vertex that you are currently at. In the case of a tie, choose at random.
- Repeat until you have visited all of the vertices.
- Complete the circuit by returning to your initial vertex.