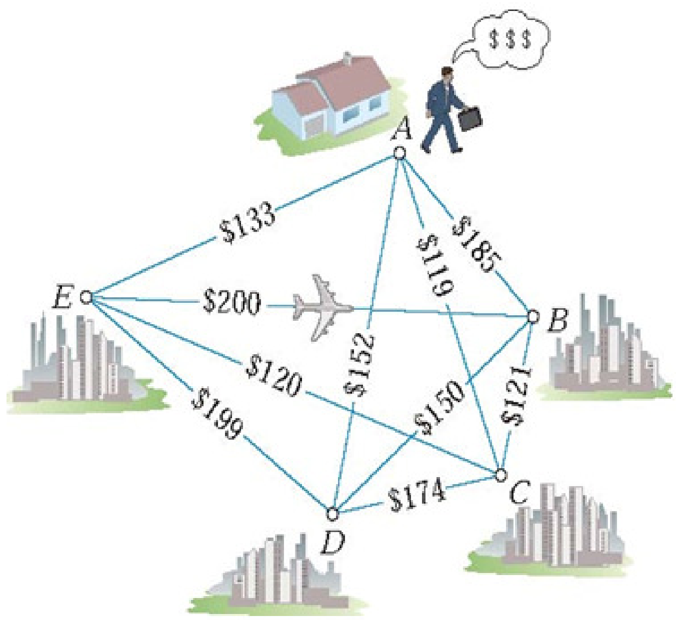
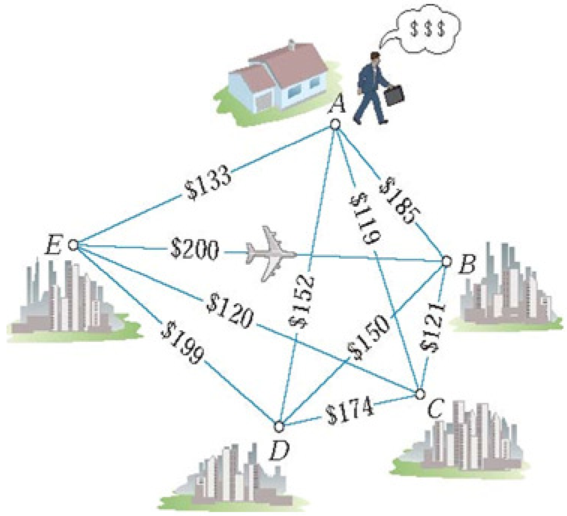
A Sample Traveling Salesman Problem (TSP)



Willy needs to schedule a trip that will start and end at his home in city A and visit all five cities A, B, C, D, and E. The graph above shows the cost of a one-way airline ticket between each pair of cities. How should Willy schedule his trip to minimize his air travel expenses? (That is, what is the optimal Hamilton circuit for the graph?)

Willy's Possible Routes and Their Costs			
	Hamilton Circuit	Total Cost	Mirror-image Circuit
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			

Identify all optimal circuits found by this *"brute force"* algorithm.



We will now approach Willy's TSP with an algorithm called the *"nearest-neighbor"* algorithm.

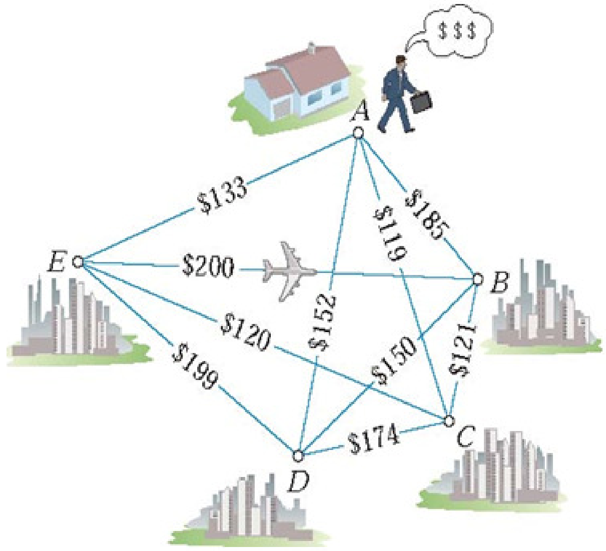
- **Start at the designated starting vertex.**
- **From the staring vertex go to its *nearest neighbor* (ie., e one associated with the lowest cost).**
- **From this new vertex go to its nearest neighbor, choosing only among the vertices that haven't been visited. Continue this process until all the vertices have been visited.**
- **From the last vertex return to the starting vertex.**

Describe Willy's nearest neighbor-circuit and give its cost.

We will now consider a third algorithm called the *"repetitive nearest-neighbor"* algorithm.

- **Start at any vertex X. Find the nearest-neighbor circuit starting at X and calculate its cost.**
- **Repeat this process with each of the other vertices as the starting vertex.**
- **Of the nearest-neighbor circuits, keep the best one. If there is a designated starting vertex, rewrite the circuit using that vertex as the starting vertex.**

Apply the *"repetitive nearest-neighbor"* algorithm to Willy's TSP and find the optimal nearest-neighbor circuit and give its cost.



The fourth algorithm we will consider is called the *"cheapest-link"* algorithm.

- **Pick the *cheapest link* (i.e. edge with smallest weight) available. Mark it *(perhaps in red).***
- **Pick the next cheapest link available and mark it.**
- **Continue this process marking the cheapest unmarked link available that does not**
  (a) close a circuit, or
  (b) create three edges out of a single vertex.
- **Connect the last two vertices to close the marked circuit.**

Apply the *"cheapest-link"* algorithm to Willy's TSP and find the optimal nearest-neighbor circuit and give its cost.