Chapter 6: Graph Theory

Graph theory deals with routing and network problems and if it is possible to find a "best" route, whether that means the least expensive, least amount of time or the least distance. Some examples of routing problems are routes covered by postal workers, UPS drivers, police officers, garbage disposal personnel, water meter readers, census takers, tour buses, etc. Some examples of network problems are telephone networks, railway systems, canals, roads, pipelines, and computer chips.

Section 6.1: Graph Theory

There are several definitions that are important to understand before delving into Graph Theory. They are:

- A graph is a picture of dots called **vertices** and lines called **edges**.
- An edge that starts and ends at the same vertex is called a **loop.**
- If there are two or more edges directly connecting the same two vertices, then these edges are called **multiple edges**.
- If there is a way to get from one vertex of a graph to all the other vertices of the graph, then the graph is **connected.**
- If there is even one vertex of a graph that cannot be reached from every other vertex, then the graph is **disconnected.**

Example 6.1.1: Graph Example 1

Figure 6.1.1: Graph 1
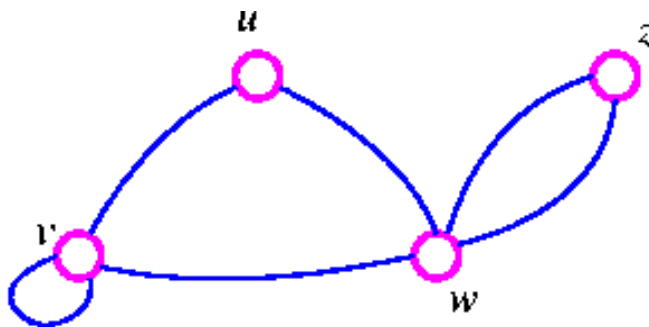
In the above graph, the **vertices** are U, V, W, and Z and the **edges** are UV, VV, VW, UW, WZ_1, and WZ_2.

This is a **connected** graph. VV is a loop. WZ_1, and WZ_2 are **multiple edges**.

Example 6.1.2: Graph Example 2

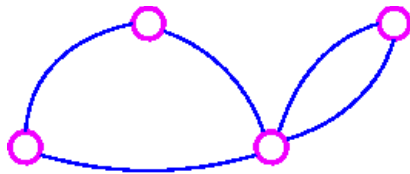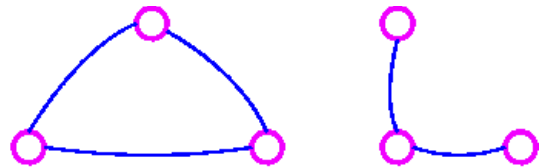**Figure 6.1.2: Graph 2**   **Figure 6.1.3: Graph 3**

The graph in Figure 6.1.2 is connected while the graph in Figure 6.1.3 is disconnected.

Graph Concepts and Terminology:

Order of a Network: the number of vertices in the entire network or graph
Adjacent Vertices: two vertices that are connected by an edge
Adjacent Edges: two edges that share a common vertex
Degree of a Vertex: the number of edges at that vertex
Path: a sequence of vertices with each vertex adjacent to the next one that starts and ends at different vertices and travels over any edge only once
Circuit: a path that starts and ends at the same vertex
Bridge: an edge such that if it were removed from a connected graph, the graph would become disconnected

Example 6.1.3: Graph Terminology

Figure 6.1.4: Graph 4
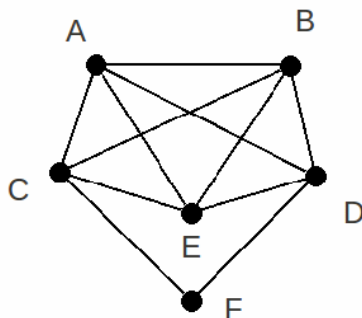
In the above graph the following is true:

Vertex A is adjacent to vertex B, vertex C, vertex D, and vertex E.

Vertex F is adjacent to vertex C, and vertex D.

Edge DF is adjacent to edge BD, edge AD, edge CF, and edge DE.

The degrees of the vertices:

A	4
B	4
C	4
D	4
E	4
F	2

Here are some paths in the above graph: (there are many more than listed)

A,B,D

A,B,C,E

F,D,E,B,C

Here are some circuits in the above graph: (there are many more than listed)

B,A,D,B

B,C,F,D,B

F,C, E, D, F

The above graph does not have any bridges.

Section 6.2: Networks

A network is a connection of vertices through edges. The internet is an example of a network with computers as the vertices and the connections between these computers as edges.

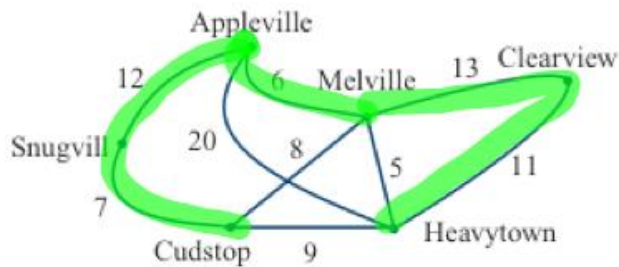**Spanning Subgraph:** a graph that joins all of the vertices of a more complex graph, but does not create a circuit

Example 6.2.1: Spanning Subgraph
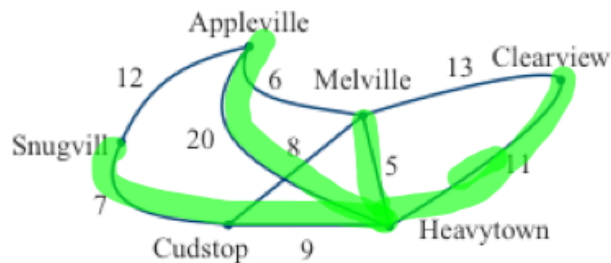
Figure 6.2.1: Map of Connecting Towns
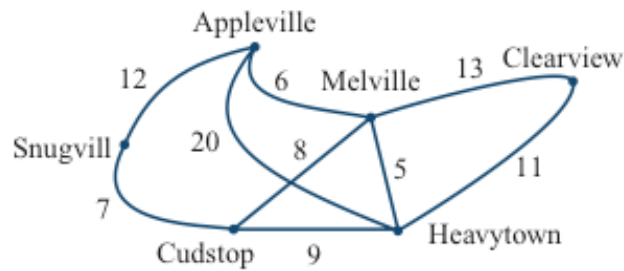
This is a graph showing how six cities are linked by roads. This graph has many spanning subgraphs but two examples are shown below.

Figure 6.2.2: Spanning Subgraph 1

This graph spans all of the cities (vertices) of the original graph, but does not contain any circuits.

Figure 6.2.3: Spanning Subgraph 2

This graph spans all of the cities (vertices) of the original graph, but does not contain any circuits.


Tree: A tree is a graph that is connected and has no circuits. Therefore, a spanning subgraph is a tree and the examples of spanning subgraphs in Example 6.2.1 above are also trees.

Properties of Trees:

1. If a graph is a tree, there is one and only one path joining any two vertices. Conversely, if there is one and only one path joining any two vertices of a graph, the graph must be a tree.
2. In a tree, every edge is a bridge. Conversely, if every edge of a connected graph is a bridge, then the graph must be a tree.
3. A tree with N vertices must have N-1 edges.
4. A connected graph with N vertices and N-1 edges must be a tree.

Example 6.2.2: Tree Properties

Figure 6.2.2: Spanning Subgraph 1
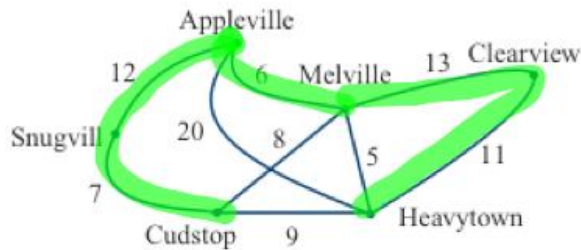
Consider the spanning subgraph highlighted in green shown in Figure 6.2.2.

a. Tree Property 1
   Look at the vertices Appleville and Heavytown. Since the graph is a tree, there is only one path joining these two cities. Also, since there is only one path between any two cities on the whole graph, then the graph must be a tree.

b. Tree Property 2
   Since the graph is a tree, notice that every edge of the graph is a bridge, which is an edge such that if it were removed the graph would become disconnected.

---

c. Tree Property 3

Since the graph is a tree and it has six vertices, it must have N – 1 or six – 1 = five edges.
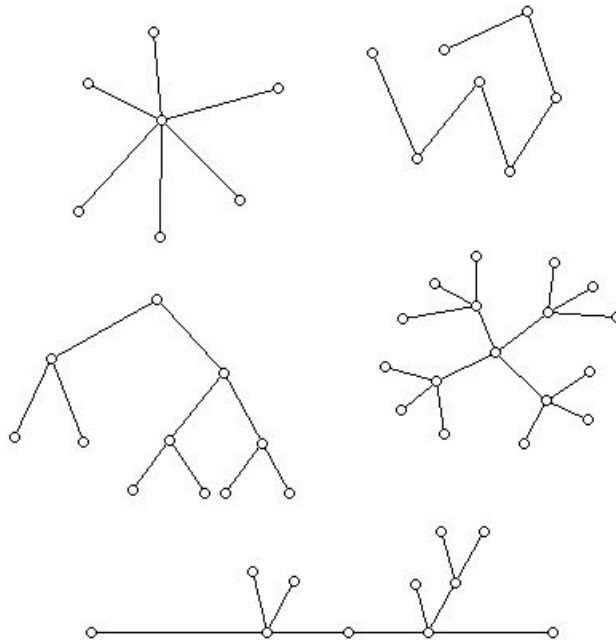
d. Tree Property 4

Since the graph is connected and has six vertices and five edges, it must be a tree.
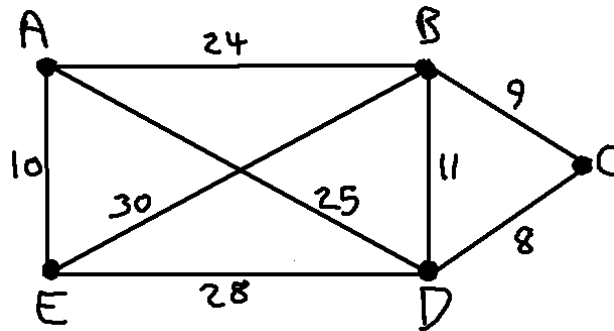
Example 6.2.3: More Examples of Trees:

All of the graphs shown below are trees and they all satisfy the tree properties.

Figure 6.2.4: More Examples of Trees

Minimum Spanning Tree: A minimum spanning tree is the tree that spans all of the vertices in a problem with the least cost (or time, or distance).

Example 6.2.4: Minimum Spanning Tree

Figure 6.2.5: Weighted Graph 1

The above is a weighted graph where the numbers on each edge represent the cost of each edge. We want to find the minimum spanning tree of this graph so that we can find a network that will reach all vertices for the least total cost.

**Figure 6.2.6: Minimum Spanning Tree for Weighted Graph 1**
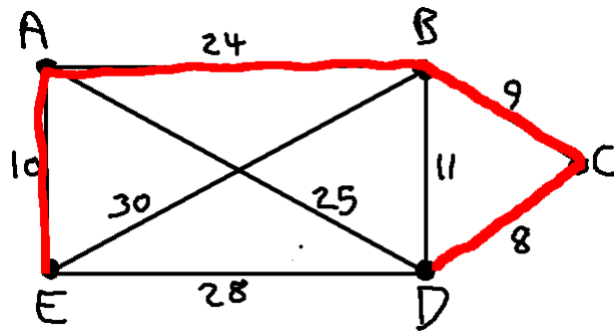
This is the minimum spanning tree for the graph with a total cost of 51.

Kruskal's Algorithm: Since some graphs are much more complicated than the previous example, we can use Kruskal's Algorithm to always be able to find the minimum spanning tree for any graph.

1. Find the cheapest link in the graph. If there is more than one, pick one at random. Mark it in red.
2. Find the next cheapest link in the graph. If there is more than one, pick one at random. Mark it in red.
3. Continue doing this as long as the next cheapest link does not create a red circuit.
4. You are done when the red edges span every vertex of the graph without any circuits. The red edges are the MST (minimum spanning tree).

**Example 6.2.5: Using Kruskal's Algorithm**

Figure 6.2.7: Weighted Graph 2
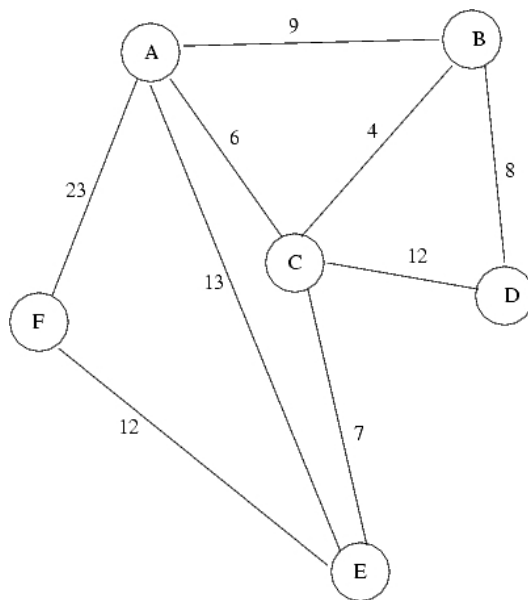
Suppose that it is desired to install a new fiber optic cable network between the six cities (A, B, C, D, E, and F) shown above for the least total cost. Also, suppose that the fiber optic cable can only be installed along the roadways shown above. The weighted graph above shows the cost (in millions of dollars) of installing the fiber optic cable along each roadway. We want to find the minimum spanning tree for this graph using Kruskal's Algorithm.
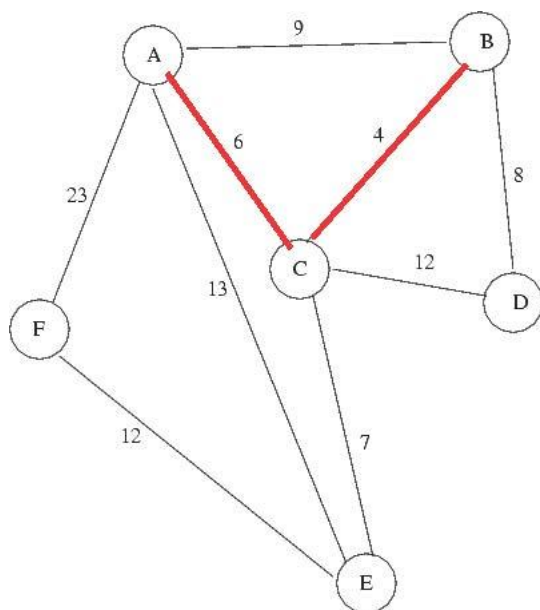

Step 1: Find the cheapest link of the whole graph and mark it in red. The cheapest link is between B and C with a cost of four million dollars.

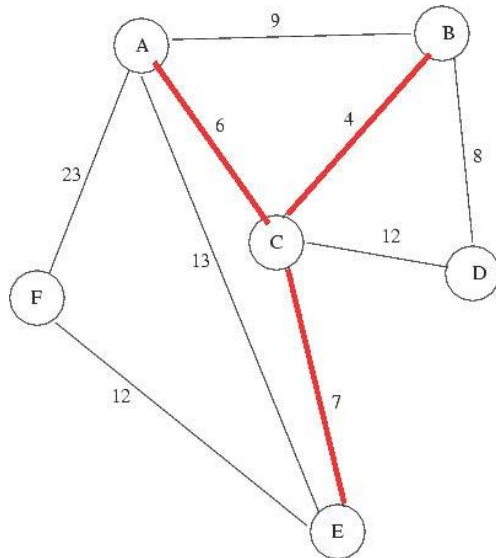**Figure 6.2.8: Kruskal's Algorithm Step 1**
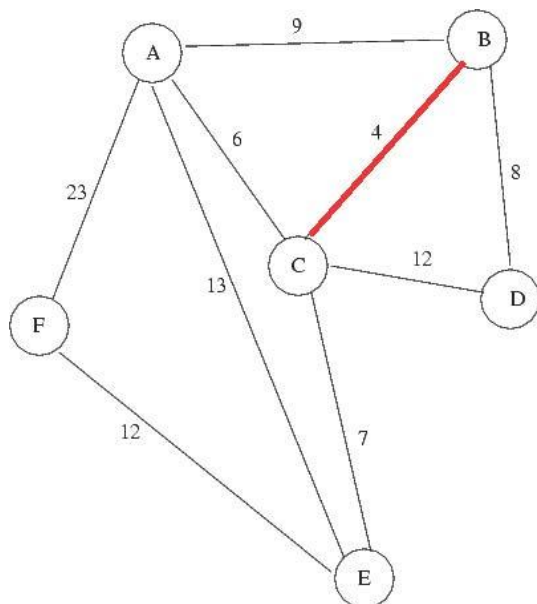
Step 2: Find the next cheapest link of the whole graph and mark it in red. The next cheapest link is between A and C with a cost of six million dollars.
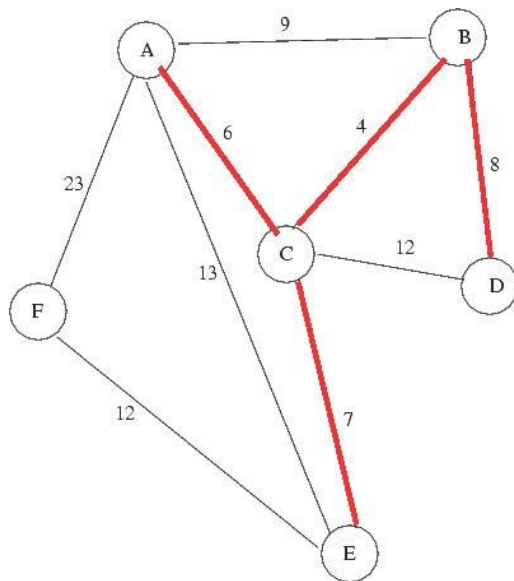
**Figure 6.2.9: Kruskal's Algorithm Step 2**

Step 3: Find the next cheapest link of the whole graph and mark it in red as long as it does not create a red circuit. The next cheapest link is between C and E with a cost of seven million dollars.

Figure 6.2.10: Kruskal's Algorithm Step 3

Step 4: Find the next cheapest link of the whole graph and mark it in red as long as it does not create a red circuit. The next cheapest link is between B and D with a cost of eight million dollars.

Figure 6.2.11: Kruskal's Algorithm Step 4

Step 5: Find the next cheapest link of the whole graph and mark it in red as long as it does not create a red circuit. The next cheapest link is between A and B with a cost of nine million dollars, but that would create a red circuit so we cannot use it. Therefore, the next cheapest link after that is between E and F with a cost of 12 million dollars, which we are able to use. We cannot use the link between C and D which also has a cost of 12 million dollars because it would create a red circuit.

Figure 6.2.12: Kruskal's Algorithm Step 5
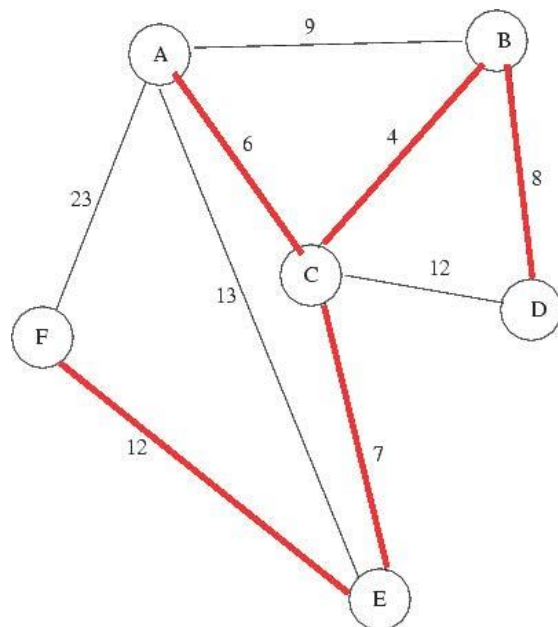
This was the last step and we now have the minimum spanning tree for the weighted graph with a total cost of $37,000,000.

Section 6.3: Euler Circuits

Leonhard Euler first discussed and used Euler paths and circuits in 1736. Rather than finding a minimum spanning tree that visits every vertex of a graph, an Euler path or circuit can be used to find a way to visit every edge of a graph once and only once. This would be useful for checking parking meters along the streets of a city, patrolling the streets of a city, or delivering mail.

Euler Path: a path that travels through *every* edge of a connected graph once and only once and starts and ends at different vertices
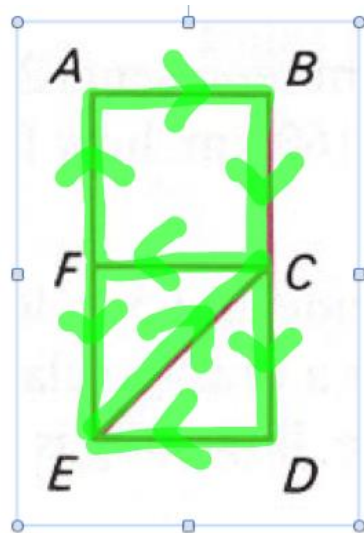
Example 6.3.1: Euler Path

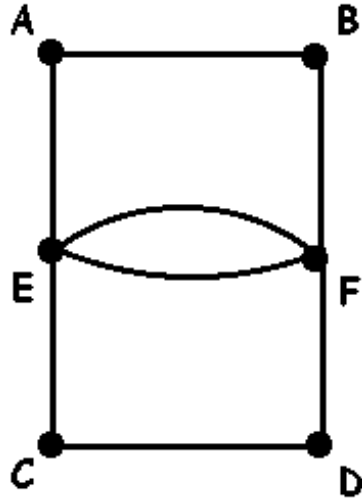**Figure 6.3.1: Euler Path Example**
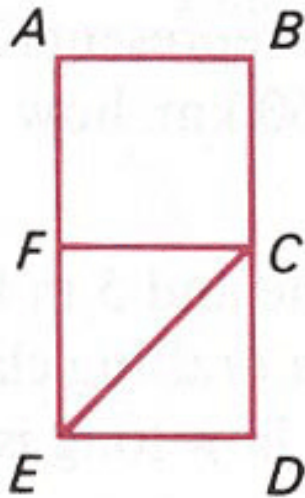
One Euler path for the above graph is F, A, B, C, F, E, C, D, E as shown below.
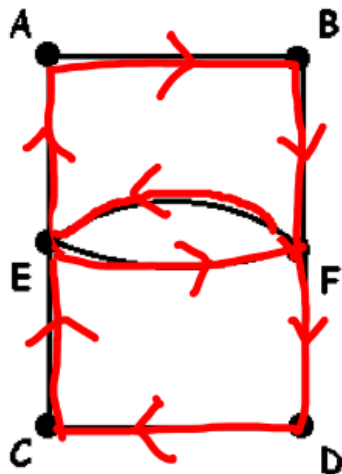
Figure 6.3.2: Euler Path

This Euler path travels every edge once and only once and starts and ends at different vertices. This graph cannot have an Euler circuit since no Euler path can start and end at the same vertex without crossing over at least one edge more than once.

Euler Circuit: an Euler path that starts and ends at the same vertex

Example 6.3.2: Euler Circuit

Figure 6.3.3: Euler Circuit Example

One Euler circuit for the above graph is E, A, B, F, E, F, D, C, E as shown below.

Figure 6.3.4: Euler Circuit

This Euler path travels every edge once and only once and starts and ends at the same vertex. Therefore, it is also an Euler circuit.

Euler's Theorems:

> **Euler's Theorem 1:** If a graph has any vertices of odd degree, then it cannot have an Euler circuit.
>
> If a graph is connected and every vertex has an even degree, then it has at least one Euler circuit (usually more).

> **Euler's Theorem 2**: If a graph has more than two vertices of odd degree, then it cannot have an Euler path.
>
> If a graph is connected and has exactly two vertices of odd degree, then it has at least one Euler path (usually more). Any such path must start at one of the odd-degree vertices and end at the other one.

> **Euler's Theorem 3**: The sum of the degrees of all the vertices of a graph equals twice the number of edges (and therefore must be an even number).
>
> Therefore, the number of vertices of odd degree must be even.

Finding Euler Circuits:

1. Be sure that every vertex in the network has even degree.
2. Begin the Euler circuit at any vertex in the network.
3. As you choose edges, never use an edge that is the only connection to a part of the network that you have not already visited.
4. Label the edges in the order that you travel them and continue this until you have travelled along every edge exactly once and you end up at the starting vertex.

Example 6.3.3: Finding an Euler Circuit

Figure 6.3.5: Graph for Finding an Euler Circuit
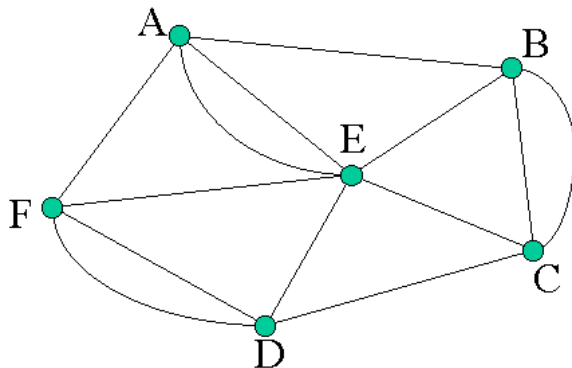
The graph shown above has an Euler circuit since each vertex in the entire graph is even degree. Thus, start at one even vertex, travel over each vertex once and only once, and end at the starting point. One example of an Euler circuit for this graph is A, E, A, B, C, B, E, C, D, E, F, D, F, A. This is a circuit that travels over every edge once and only once and starts and ends in the same place. There are other Euler circuits for this graph. This is just one example.

Figure 6.3.6: Euler Circuit
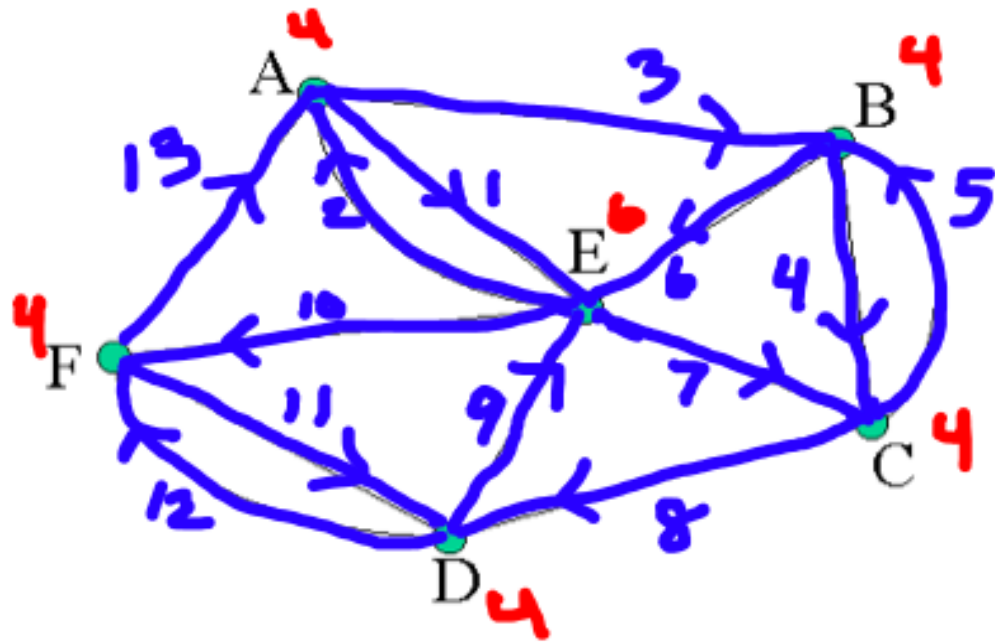
The degree of each vertex is labeled in red. The ordering of the edges of the circuit is labeled in blue and the direction of the circuit is shown with the blue arrows.

Section 6.4 Hamiltonian Circuits

The Traveling Salesman Problem (TSP) is any problem where you must visit every vertex of a weighted graph once and only once, and then end up back at the starting vertex. Examples of TSP situations are package deliveries, fabricating circuit boards, scheduling jobs on a machine and running errands around town.

> **Hamilton Circuit**: a circuit that must pass through each vertex of a graph once and only once

> **Hamilton Path**: a path that must pass through each vertex of a graph once and only once

Example 6.4.1: Hamilton Path:

Figure 6.4.1: Examples of Hamilton Paths
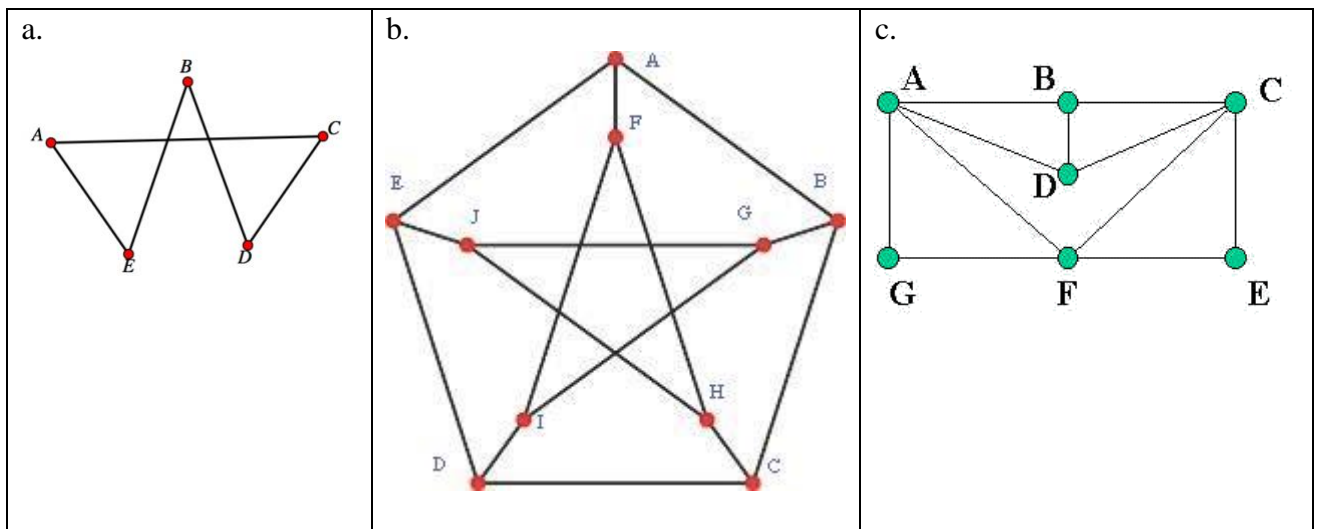
Not all graphs have a Hamilton circuit or path. There is no way to tell just by looking at a graph if it has a Hamilton circuit or path like you can with an Euler circuit or path. You must do trial and error to determine this. By the way if a graph has a Hamilton circuit then it has a Hamilton path. Just do not go back to home.

Graph a. has a Hamilton circuit (one example is ACDBEA)
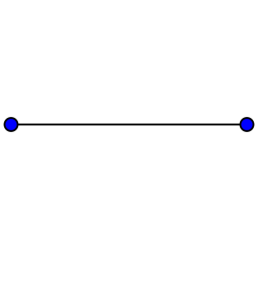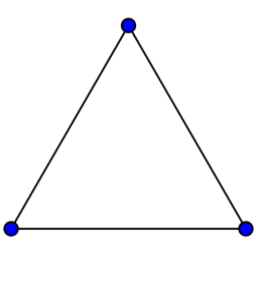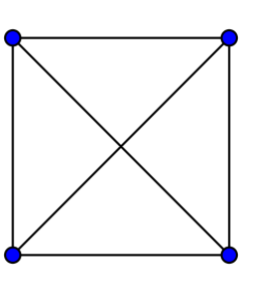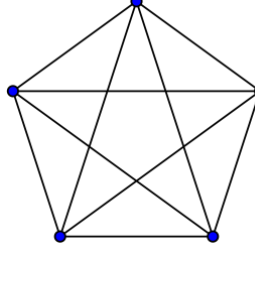
Graph b. has no Hamilton circuits, though it has a Hamilton path (one example is ABCDEJGIFH)

Graph c. has a Hamilton circuit (one example is AGFECDBA)

> **Complete Graph**: A complete graph is a graph with N vertices in which every pair of vertices is joined by exactly one edge. The symbol used to denote a complete graph is K_N.

Example 6.4.2: Complete Graphs

Figure 6.4.2: Complete Graphs for N = 2, 3, 4, and 5

| a. K_2	b. K_3	c. K_4	d. K_5
two vertices and one edge	three vertices and three edges	four vertices and six edges	five vertices and ten edges

In each complete graph shown above, there is exactly one edge connecting each pair of vertices. There are no loops or multiple edges in complete graphs. Complete graphs do have Hamilton circuits.

Reference Point: the starting point of a Hamilton circuit

Example 6.4.3: Reference Point in a Complete Graph

Many Hamilton circuits in a complete graph are the same circuit with different starting points. For example, in the graph K_3, shown below in Figure 6.4.3, ABCA is the same circuit as BCAB, just with a different starting point (reference point). We will typically assume that the reference point is A.

Figure 6.4.3: K₃
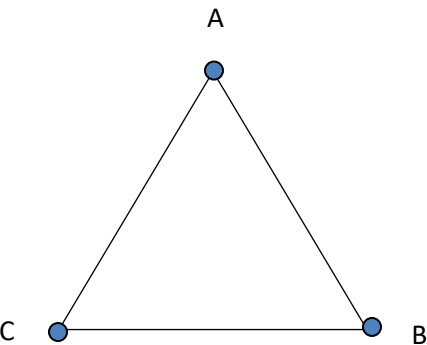
> **Number of Hamilton Circuits:** A complete graph with N vertices is (N-1)! Hamilton circuits. Since half of the circuits are mirror images of the other half, there are actually only half this many unique circuits.

Example 6.4.4: Number of Hamilton Circuits

How many Hamilton circuits does a graph with five vertices have?

$(N - 1)! = (5 - 1)! = 4! = 4*3*2*1 = 24$ Hamilton circuits.

How to solve a Traveling Salesman Problem (TSP):

A traveling salesman problem is a problem where you imagine that a traveling salesman goes on a business trip. He starts in his home city (A) and then needs to travel to several different cities to sell his wares (the other cities are B, C, D, etc.). To solve a TSP, you need to find the cheapest way for the traveling salesman to start at home, A, travel to the other cities, and then return home to A at the end of the trip. This is simply finding the Hamilton circuit in a complete graph that has the smallest overall weight. There are several different algorithms that can be used to solve this type of problem.

A. Brute Force Algorithm

1. List all possible Hamilton circuits of the graph.
2. For each circuit find its total weight.
3. The circuit with the least total weight is the optimal Hamilton circuit.

Example 6.4.5: Brute Force Algorithm:

Figure 6.4.4: Complete Graph for Brute Force Algorithm
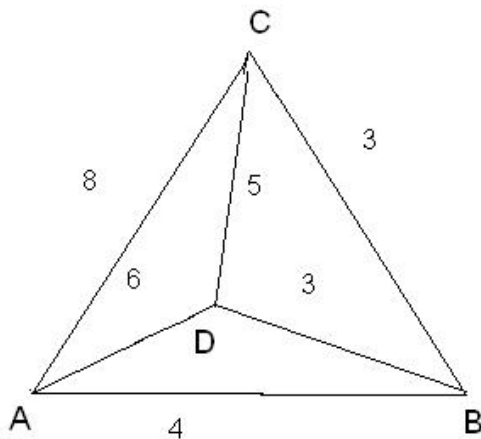
Suppose a delivery person needs to deliver packages to three locations and return to the home office A. Using the graph shown above in Figure 6.4.4, find the shortest route if the weights on the graph represent distance in miles.

Recall the way to find out how many Hamilton circuits this complete graph has. The complete graph above has four vertices, so the number of Hamilton circuits is:

$(N - 1)! = (4 - 1)! = 3! = 3*2*1 = 6$ Hamilton circuits.

However, three of those Hamilton circuits are the same circuit going the opposite direction (the mirror image).

Table 6.4.5: Hamilton Circuits

Hamilton Circuit	Mirror Image	Total Weight (Miles)
ABCDA	ADCBA	18
ABDCA	ACDBA	20
ACBDA	ADBCA	20

The solution is ABCDA (or ADCBA) with total weight of 18 mi. This is the optimal solution.

B. Nearest-Neighbor Algorithm:

1. Pick a vertex as the starting point.
2. From the starting point go to the vertex with an edge with the smallest weight. If there is more than one choice, choose at random.
3. Continue building the circuit, one vertex at a time from among the vertices that have not been visited yet.
4. From the last vertex, return to the starting point.

Example 6.4.6: Nearest-Neighbor Algorithm

A delivery person needs to deliver packages to four locations and return to the home office A as shown in Figure 6.4.5 below. Find the shortest route if the weights represent distances in miles.

Starting at A, E is the nearest neighbor since it has the least weight, so go to E. From E, B is the nearest neighbor so go to B. From B, C is the nearest neighbor so go to C. From C, the first nearest neighbor is B, but you just came from there. The next nearest neighbor is A, but you do not want to go there yet because that is the

starting point. The next nearest neighbor is E, but you already went there. So go to D. From D, go to A since all other vertices have been visited.

**Figure 6.4.6: Complete Graph for Nearest-Neighbor Algorithm**
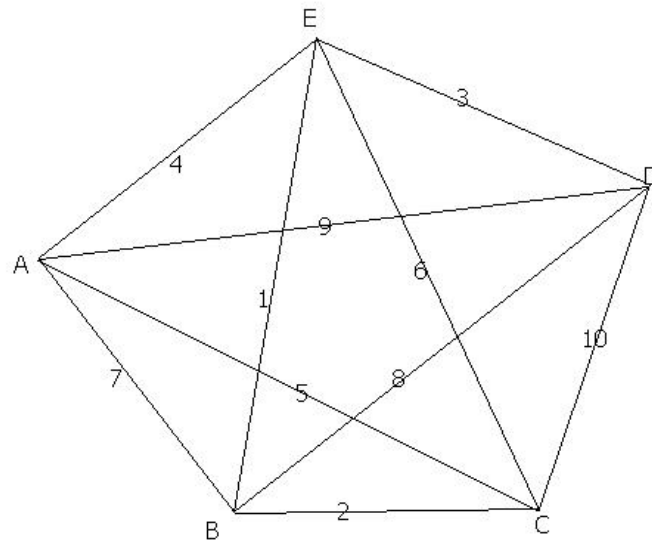
The solution is AEBCDA with a total weight of 26 miles. This is not the optimal solution, but it is close and it was a very efficient method.

C. Repetitive Nearest-Neighbor Algorithm:

1. Let X be any vertex. Apply the Nearest-Neighbor Algorithm using X as the starting vertex and calculate the total cost of the circuit obtained.
2. Repeat the process using each of the other vertices of the graph as the starting vertex.
3. Of the Hamilton circuits obtained, keep the best one. If there is a designated starting vertex, rewrite this circuit with that vertex as the reference point.

Example 6.4.7: Repetitive Nearest-Neighbor Algorithm

Suppose a delivery person needs to deliver packages to four locations and return to the home office A. Find the shortest route if the weights on the graph represent distances in kilometers.

Figure 6.4.7: Complete Graph for Repetitive Nearest-Neighbor Algorithm
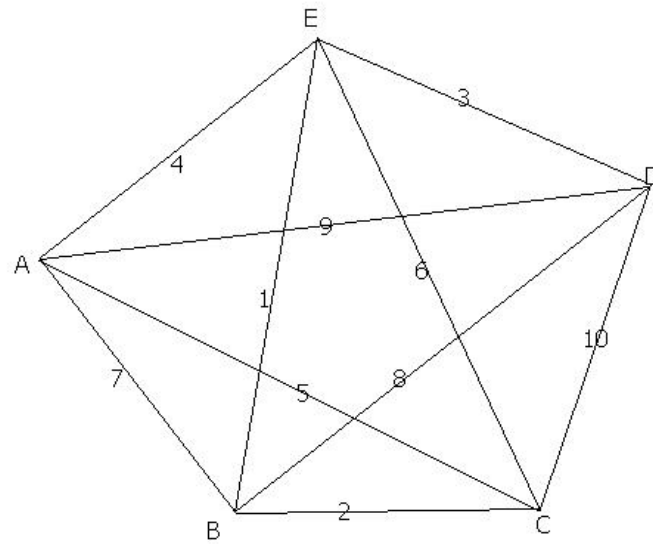
Starting at A, the solution is AEBCDA with total weight of 26 miles as we found in Example 6.4.6. See this solution below in Figure 6.4.7.

Figure 6.4.8: Starting at A
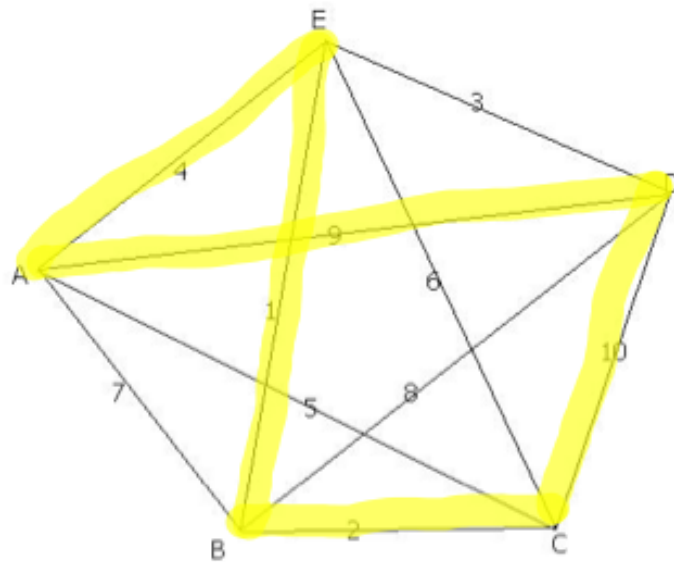
Starting at B, the solution is BEDACB with total weight of 20 miles.

Figure 6.4.9: Starting at B
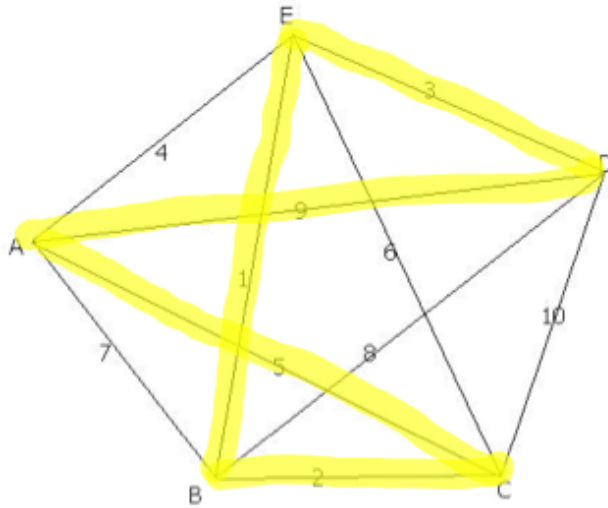
Starting at C, the solution is CBEDAC with total weight of 20 miles.
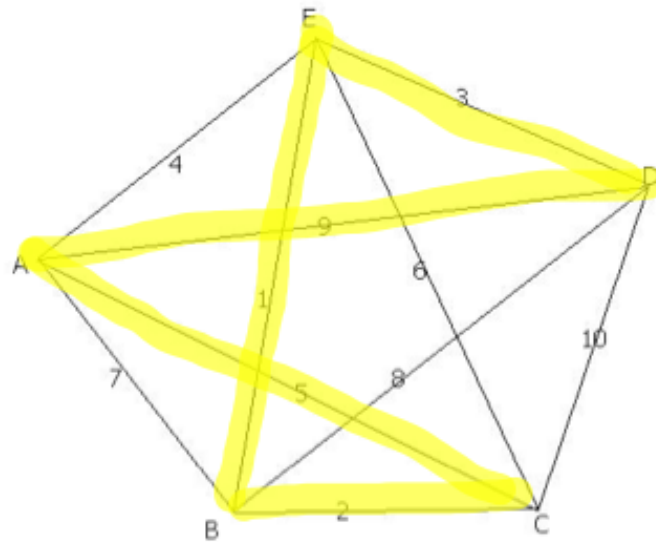
Figure 6.4.10: Starting at C
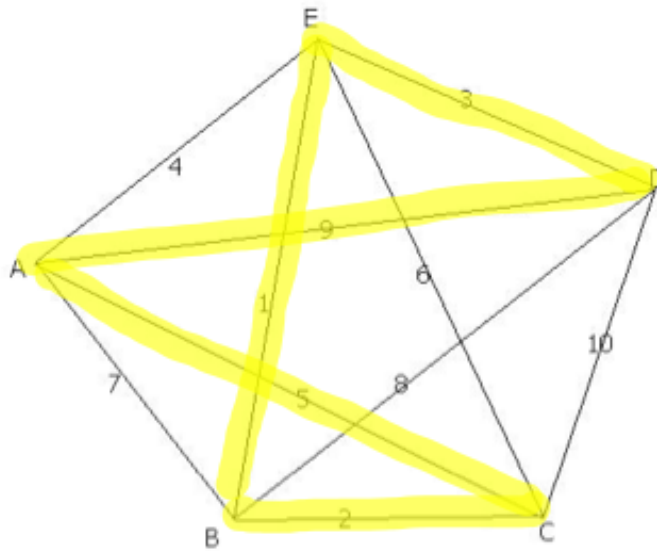
Starting at D, the solution is DEBCAD with total weight of 20 miles.

Figure 6.4.11: Starting at D
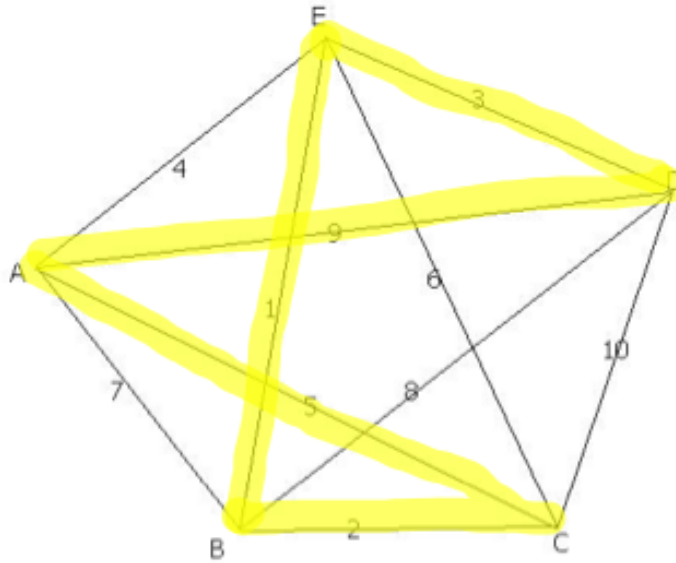
Starting at E, solution is EBCADE with total weight of 20 miles.

Figure 6.4.12: Starting at E

Now, you can compare all of the solutions to see which one has the lowest overall weight. The solution is any of the circuits starting at B, C, D, or E since they all have the same weight of 20 miles. Now that you know the best solution using this method, you can rewrite the circuit starting with any vertex. Since the home office in this example is A, let's rewrite the solutions starting with A. Thus, the solution is ACBEDA or ADEBCA.

D. Cheapest-Link Algorithm

1. Pick the link with the smallest weight first (if there is a tie, randomly pick one). Mark the corresponding edge in red.
2. Pick the next cheapest link and mark the corresponding edge in red.
3. Continue picking the cheapest link available. Mark the corresponding edge in red except when a) it closes a circuit or b) it results in three edges coming out of a single vertex.
4. When there are no more vertices to link, close the red circuit.

Example 6.4.8: Cheapest-Link Algorithm

Figure 6.4.13: Complete Graph for Cheapest-Link Algorithm
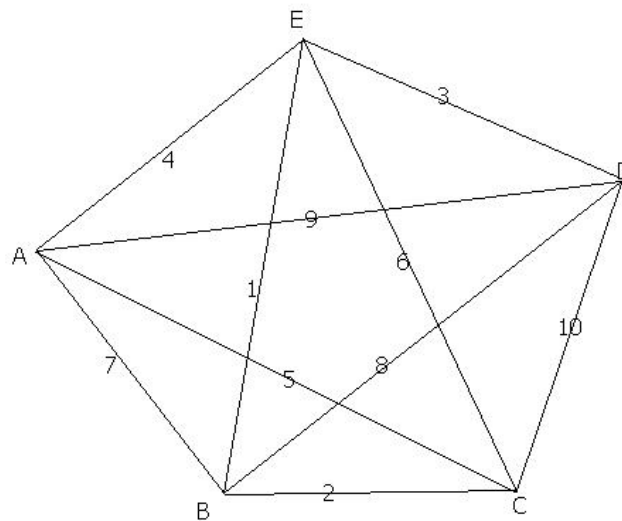
Suppose a delivery person needs to deliver packages to four locations and return to the home office A. Find the shortest route if the weights represent distances in miles.

Step 1: Find the cheapest link of the graph and mark it in blue. The cheapest link is between B and E with a weight of one mile.

Figure 6.4.14: Step 1
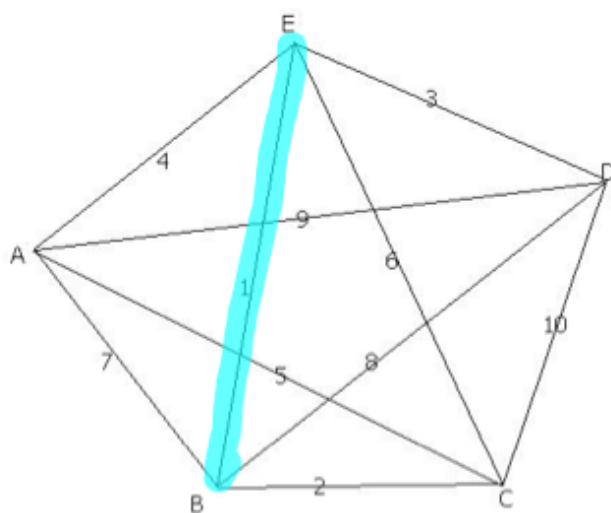
Step 2: Find the next cheapest link of the graph and mark it in blue. The next cheapest link is between B and C with a weight of two miles.

Figure 6.4.15: Step 2
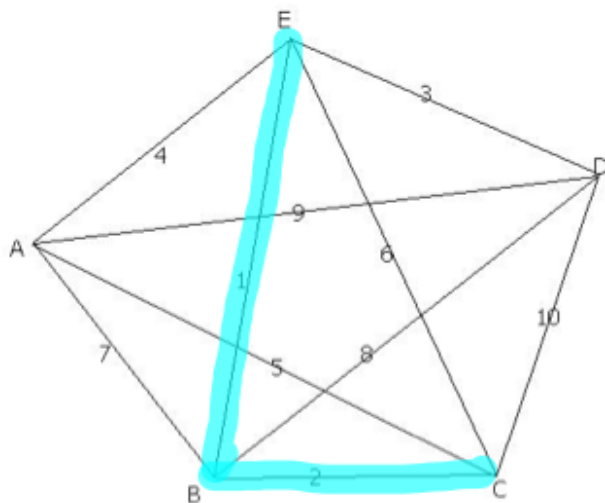
Step 3: Find the next cheapest link of the graph and mark it in blue provided it does not make a circuit or it is not a third edge coming out of a single vertex. The next cheapest link is between D and E with a weight of three miles.

Figure 6.4.16: Step 3
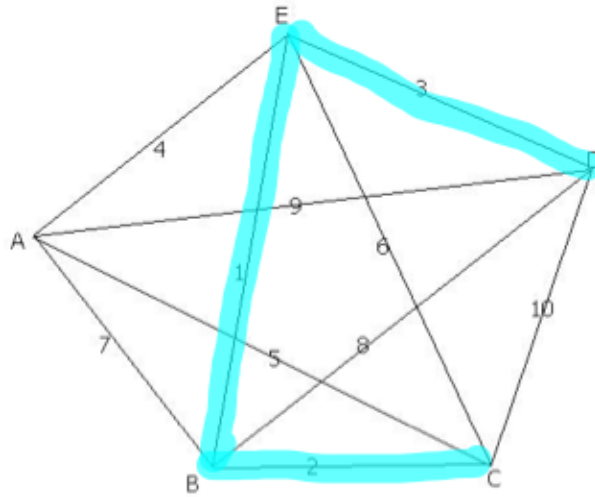
Step 4: Find the next cheapest link of the graph and mark it in blue provided it does not make a circuit or it is not a third edge coming out of a single vertex. The next cheapest link is between A and E with a weight of four miles, but it would be a third edge coming out of a single vertex. The next cheapest link is between A and C with a weight of five miles. Mark it in blue.

Figure 6.4.17: Step 4
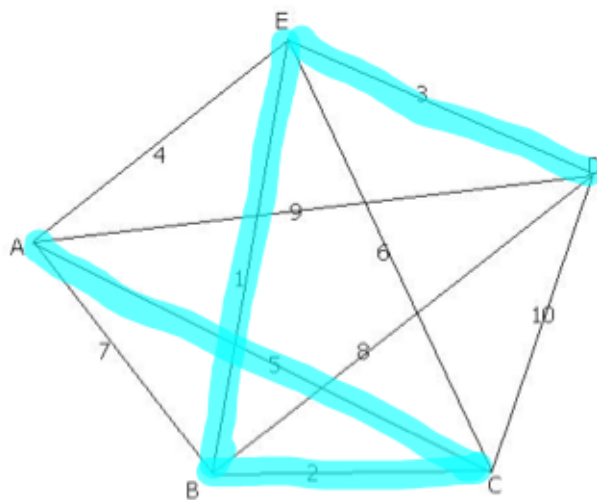
Step 5: Since all vertices have been visited, close the circuit with edge DA to get back to the home office, A. This is the only edge we could close the circuit with because AB creates three edges coming out of vertex B and BD also created three edges coming out of vertex B.

Figure 6.4.18: Step 5
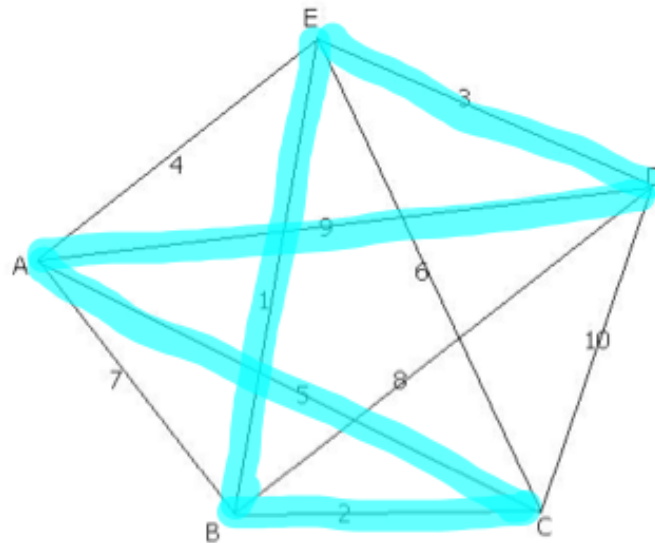
The solution is ACBEDA or ADEBCA with total weight of 20 miles.

Efficient Algorithm: an algorithm for which the number of steps needed to carry it out grows in proportion to the size of the input to the problem.

Approximate Algorithm: any algorithm for which the number of steps needed to carry it out grows in proportion to the size of the input to the problem. There is no known algorithm that is efficient and produces the optimal solution.

Chapter 6 Homework

1. Answer the following questions based on the graph below.
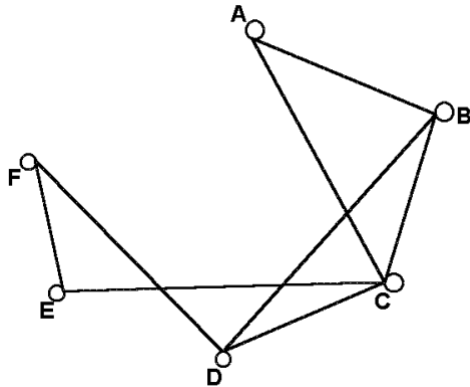
    a. What are the vertices?
    b. Is this graph connected?
    c. What is the degree of vertex C?
    d. Edge FE is adjacent to which edges?
    e. Does this graph have any bridges?

2. Answer the following questions based on the graph below.
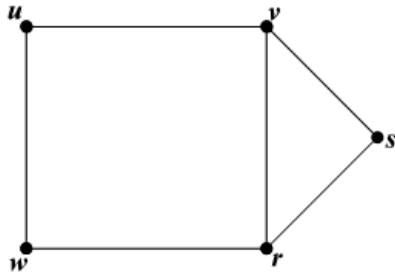
    a. What are the vertices?
    b. What is the degree of vertex u?
    c. What is the degree of vertex s?
    d. What is one circuit in the graph?

3. Draw a spanning subgraph in the graph below.
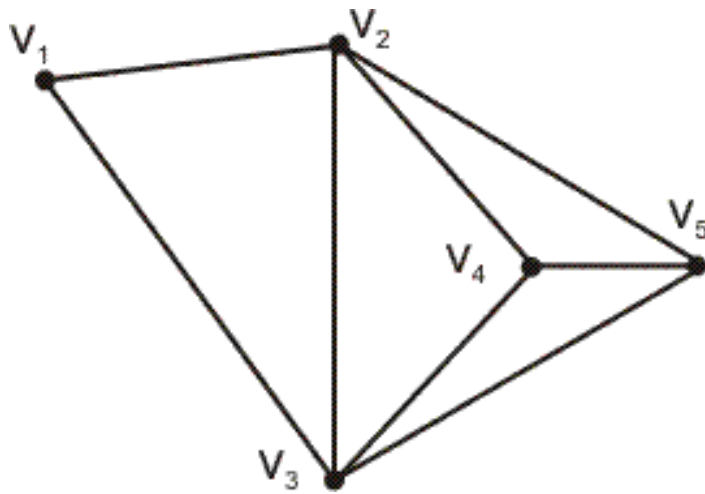
4. Find the minimum spanning tree in the graph below using Kruskal's Algorithm.
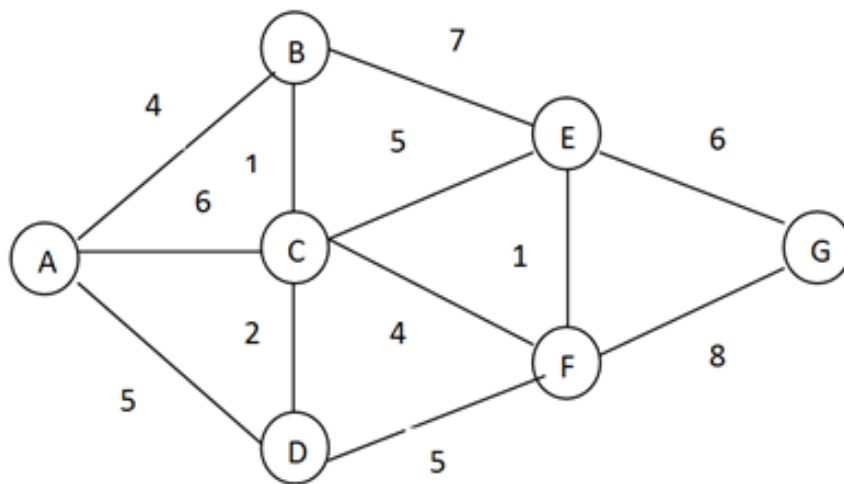
5. Find the minimum spanning tree in the graph below using Kruskal's Algorithm.
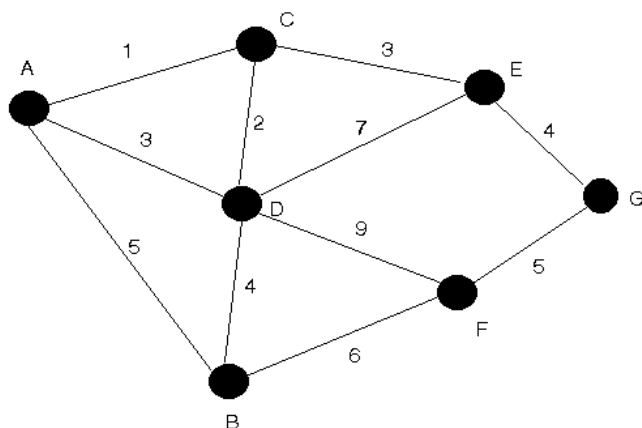
6. Find an Euler Path in the graph below.
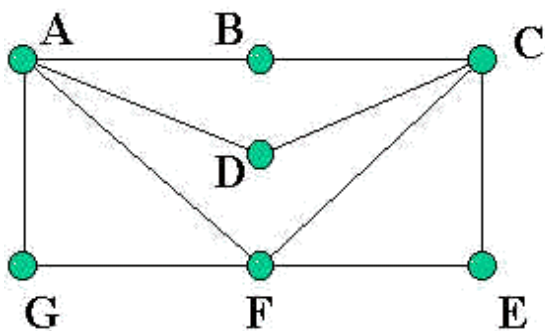
7. Find an Euler circuit in the graph below.
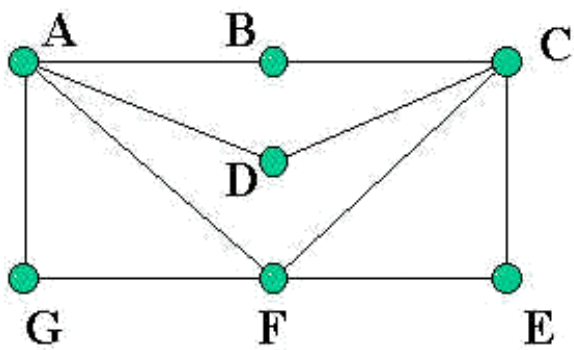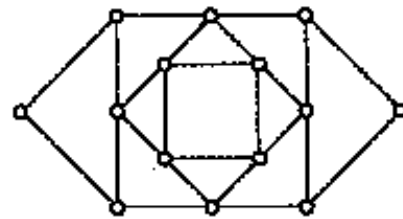
8. Which graphs below have Euler Paths? Which graphs have Euler circuits?
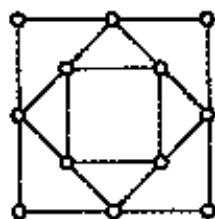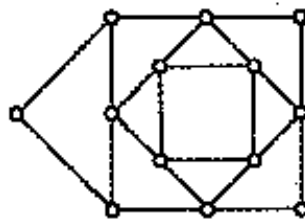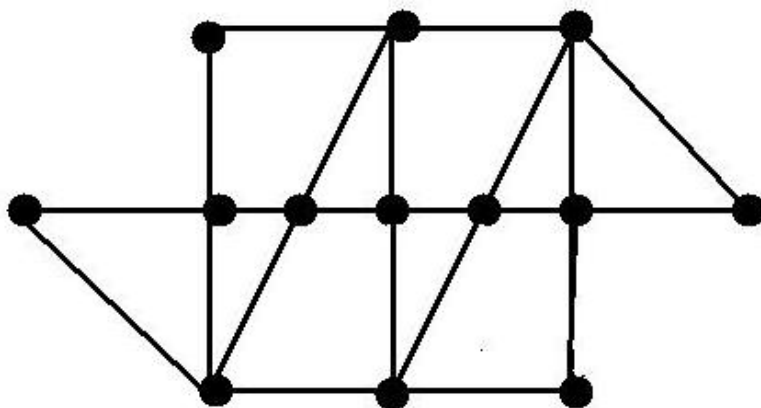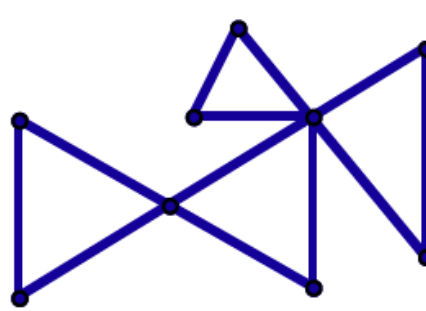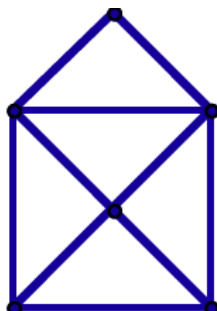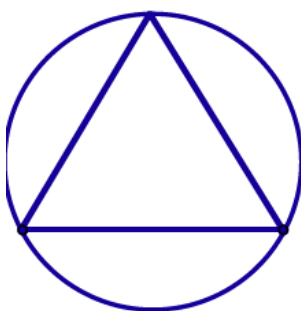

(a)   (b)   (c)

9. Highlight an Euler circuit in the graph below.
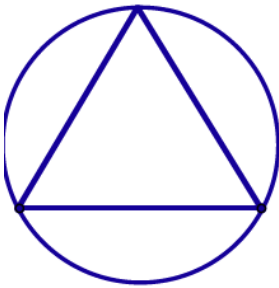

10. For each of the graphs below, write the degree of each vertex next to each vertex.
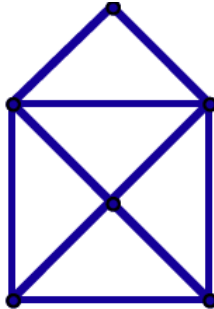
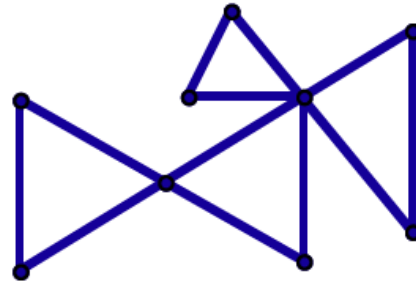11. Circle whether each of the graphs below has an Euler circuit, an Euler path, or neither.

|		
a. Euler circuit	a. Euler circuit	a. Euler circuit
b. Euler path	b. Euler path	b. Euler path
c. Neither	c. Neither	c. Neither

12. How many Hamilton circuits does a complete graph with 6 vertices have?

13. Suppose you need to start at A, visit all three vertices, and return to the starting point A. Using the Brute Force Algorithm, find the shortest route if the weights represent distances in miles.
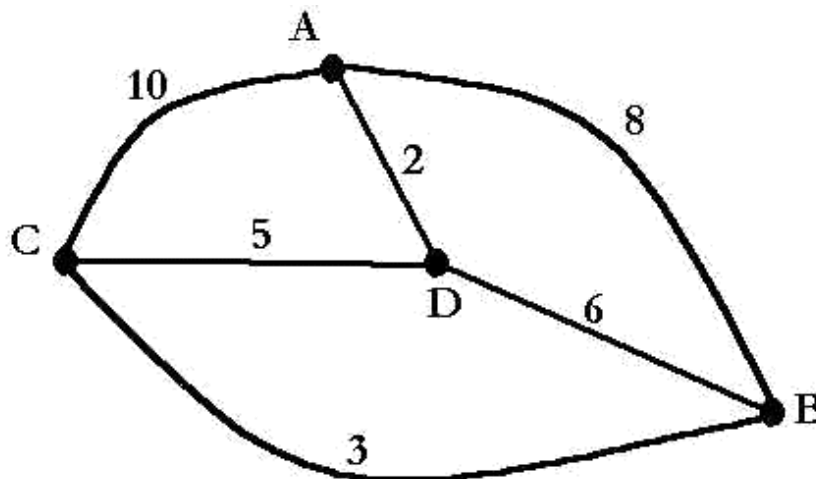
14. If a graph is connected and _____, the graph will have an Euler circuit.
    a. the graph has an even number of vertices
    b. the graph has an even number of edges
    c. the graph has all vertices of even degree
    d. the graph has only two odd vertices

15. Starting at vertex A, use the Nearest-Neighbor Algorithm to find the shortest route if the weights represent distances in miles.
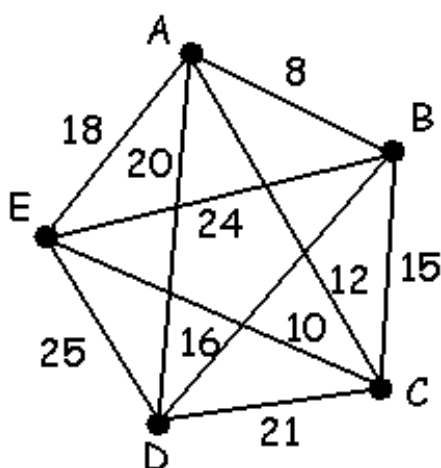
16. Find a Hamilton circuit using the Repetitive Nearest-Neighbor Algorithm.
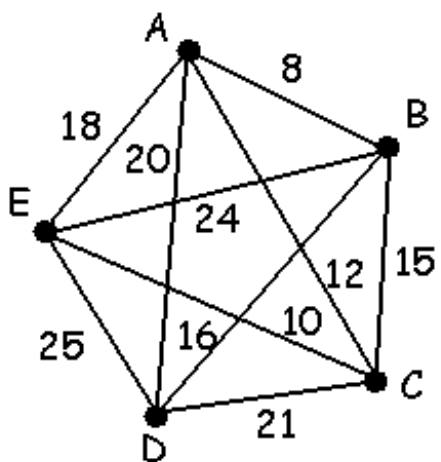
17. Find a Hamilton circuit using the Cheapest-Link Algorithm.
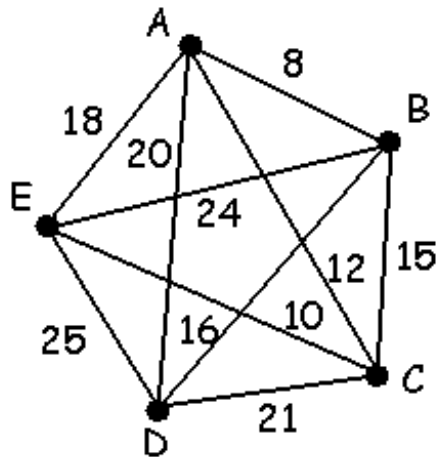
18. Which is a circuit that traverses each edge of the graph exactly once?
    A. Euler circuit   b. Hamilton circuit   c. Minimum Spanning Tree

19. Which is a circuit that traverses each vertex of the graph exactly once?
    A. Euler circuit   b. Hamilton circuit   c. Minimum Spanning Tree

20. For each situation, would you find an Euler circuit or a Hamilton Circuit?
    a. The department of Public Works must inspect all streets in the city to remove dangerous debris.
    b. Relief food supplies must be delivered to eight emergency shelters located at different sites in a large city.
    c. The Department of Public Works must inspect traffic lights at intersections in the city to determine which are still working.
    d. An insurance claims adjuster must visit 11 homes in various neighborhoods to write reports.