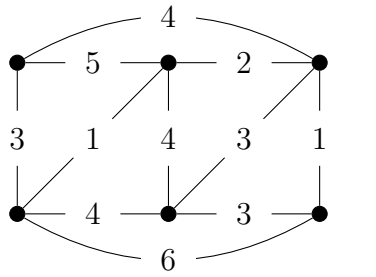
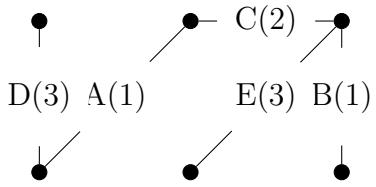
Frank Cline
Math 307
HW 8

6.6

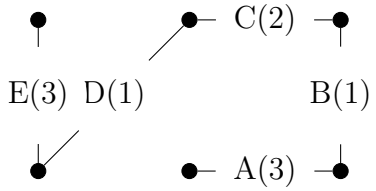
10. (a) Use Kruskal's algorithm to find a minimum spanning tree of the graph below. Label the edges in alphabetical alphabetical order as you choose them. Give the weight of the minimum spanning tree.





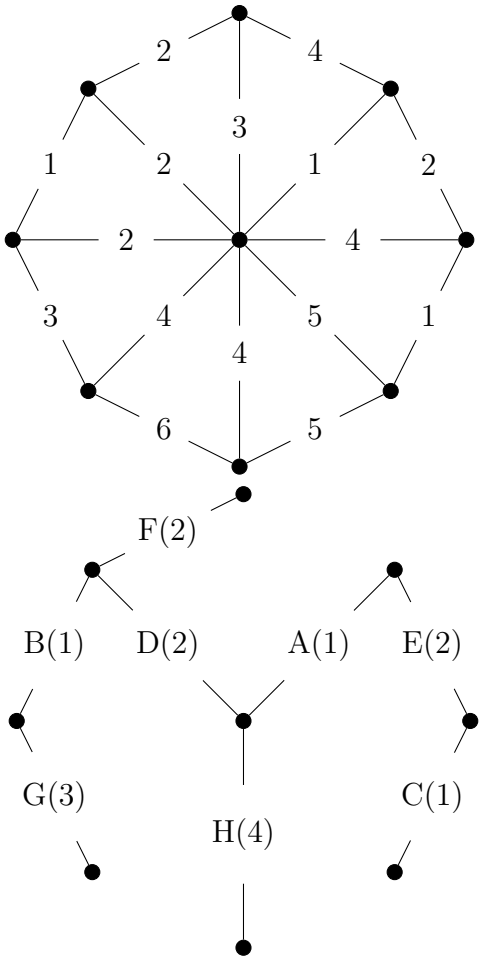
The weight of the spanning tree is $1 + 1 + 2 + 3 + 3 = 10$.

(b) Repeat part (a) with Prim's algorithm, starting at the lower middle vertex.



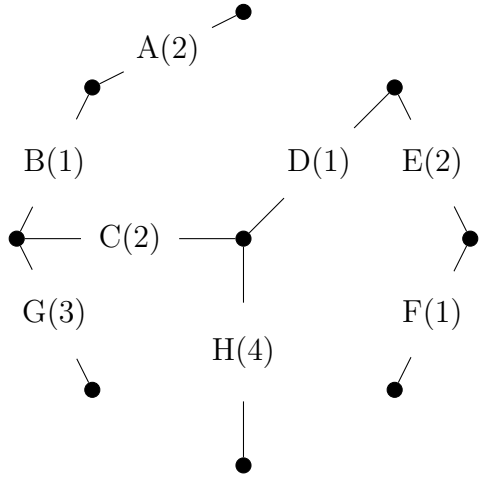
The weight of the spanning tree is $1 + 1 + 2 + 3 + 3 = 10$.

11. (a) Use Kruskal's algorithm to find a minimum spanning tree of the graph below. Label the edges in alphabetical alphabetical order as you choose them. Give the weight of the minimum spanning tree.



The weight of the spanning tree is $1 + 1 + 1 + 2 + 2 + 2 + 3 + 4 = 16$.

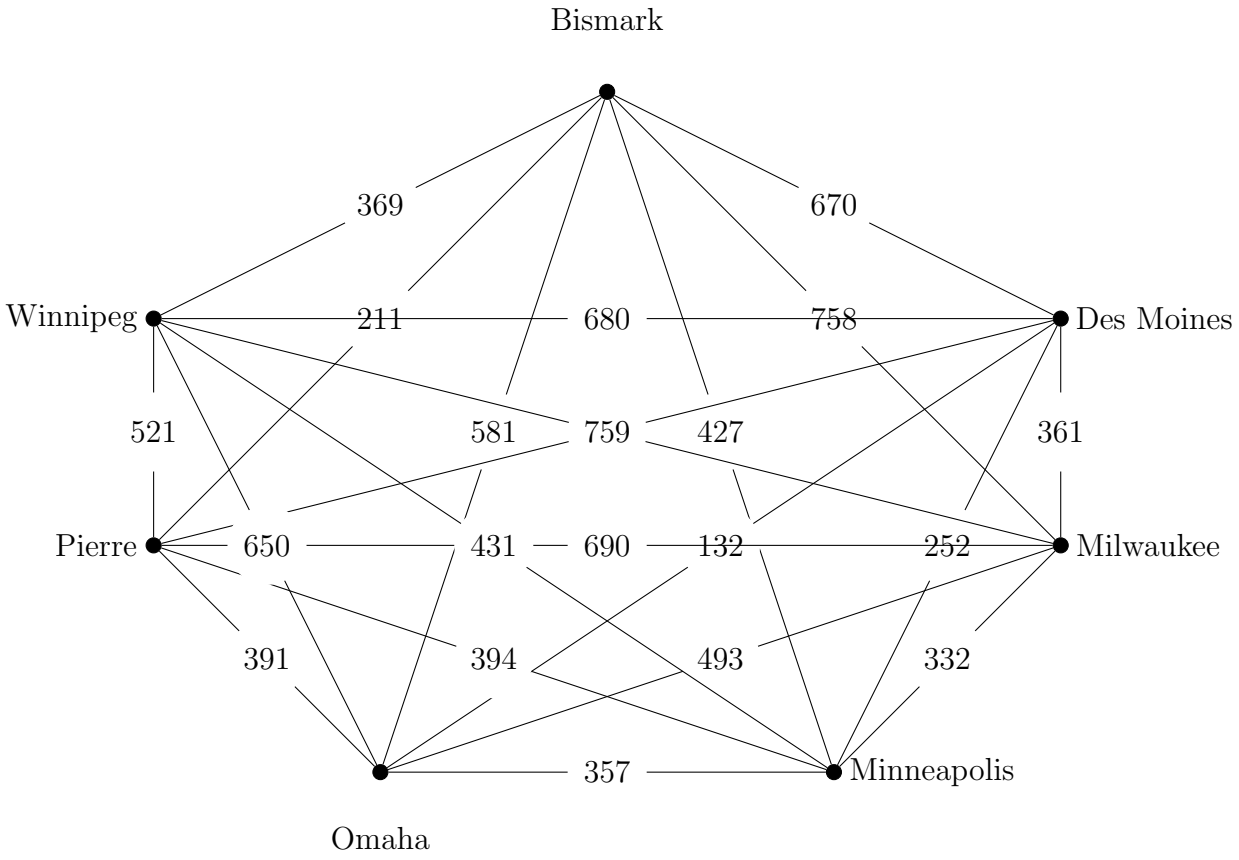
(b) Repeat part (a) with Prim's algorithm, starting at the top vertex.

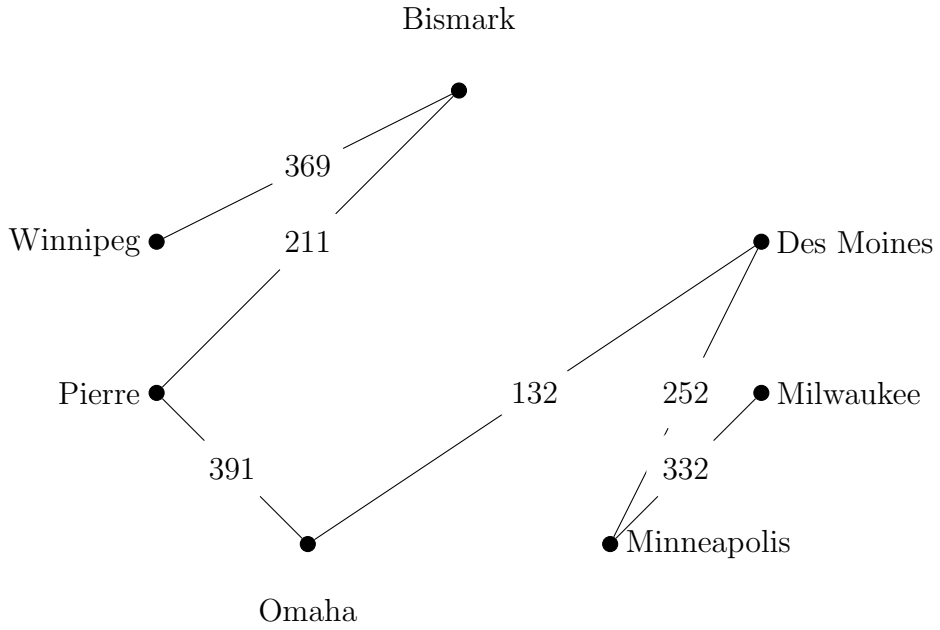


The weight of the spanning tree is $2 + 1 + 2 + 1 + 2 + 1 + 3 + 4 = 16$.

13. An oil company wants to connect the cities in the mileage chart below by pipelines going directly between cities. What is the minimum number of miles of pipeline needed?

	Des Moines	Milwaukee	Minneapolis	Omaha	Pierre	Winnipeg
Bismark	670	758	427	581	211	369
Des Moines		361	252	132	492	680
Milwaukee			332	493	690	759
Minneapolis				357	394	431
Omaha					391	650
Pierre						521





The minimum number of miles of pipe needed would be
$132 + 211 + 252 + 332 + 369 + 391 = 1687$ miles.

Additional Problems

Figures:

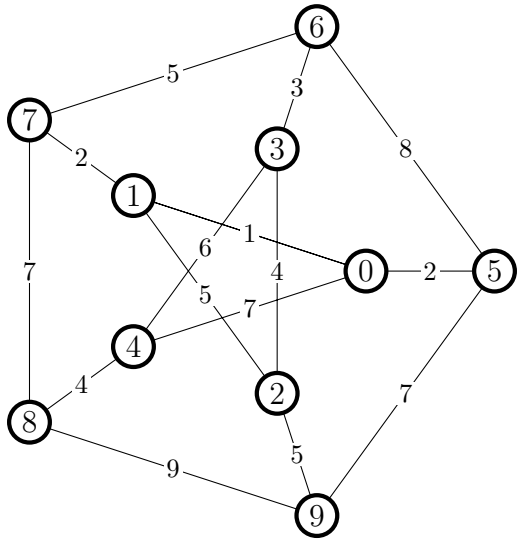


Figure 1: Graph for Kruskal, Prim, and Dijkstra's Algorithm

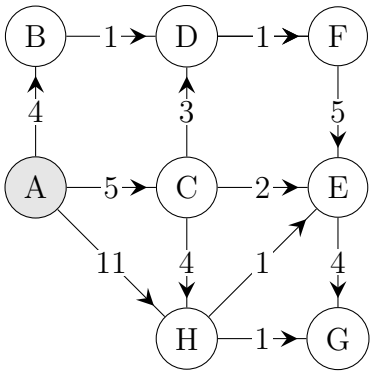


Figure 2: A digraph

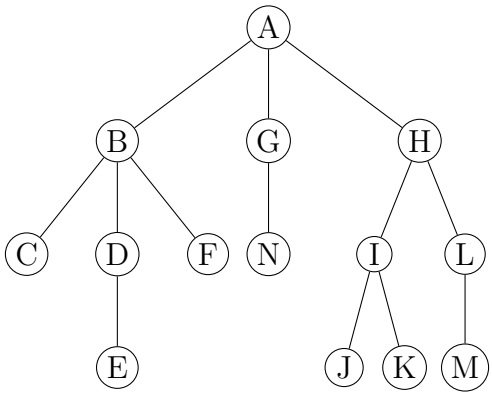
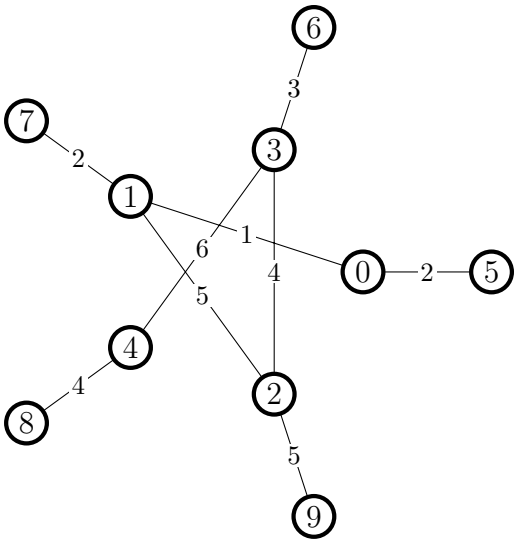


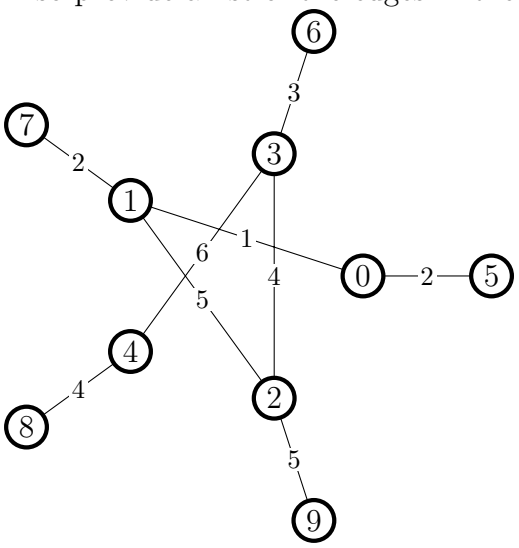
Figure 3: A tree

1. Use Kruskal's Algorithm to find a minimal distance spanning tree for the graph shown in Figure 1. Draw in the edges of the spanning tree on a new copy of the vertices. Break ties by choosing edges with a smaller-labelled endpoint. Also provide a list of the edges in the order you added them.



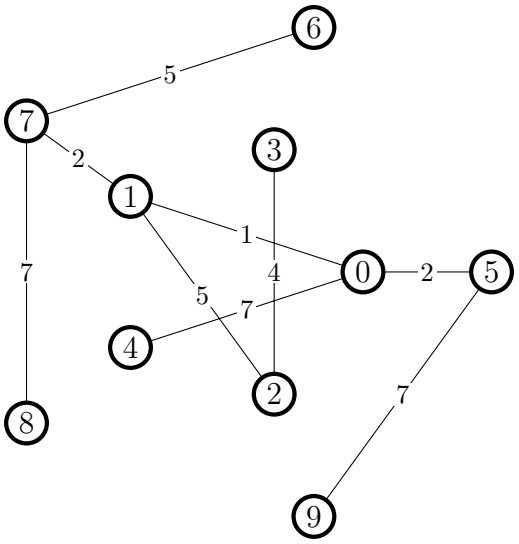
Edges in order: 1-0, 0-5, 1-7, 3-6, 3-2, 4-8, 1-2, 2-9, 4-3

2. Use Prim's Algorithm to find a minimal distance spanning tree, **starting at vertex 6** for the graph shown in Figure 1. Draw in the edges of the spanning tree on a new copy of the vertices. Break ties by choosing edges with a smaller-labelled endpoint. Also provide a list of the edges in the order you added them.



Edges in order: 6-3, 3-2, 2-1, 1-0, 0-5, 1-7, 2-9, 3-4, 4-8

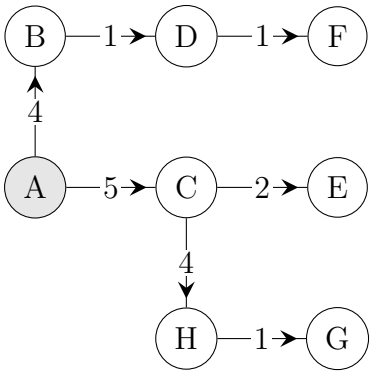
3. Use Dijkstra's Algorithm to find a minimal distance spanning tree for the graph shown in Figure 1, **starting at vertex 0**. List all the vertices along with their distance from vertex 0. Also provide a list of edges in the order you added them.



Edges added in order: 0-1, 0-5, 1-7, 1-2, 0-4, 7-6, 5-9, 3-2, 7-8

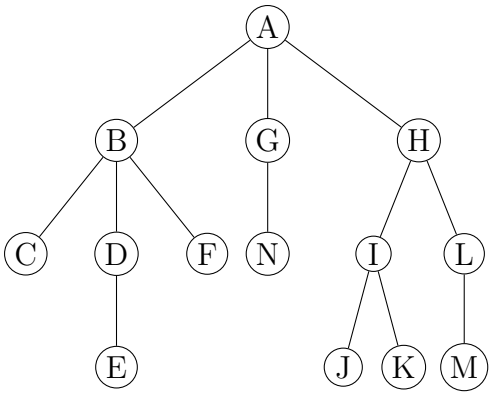
Vertices	1	5	7	2	4	6	9	3	8
d(0,vtx)	1	2	3	6	7	8	9	10	10

4. Use Dijkstra's Algorithm to find a minimal distance directed spanning tree for the digraph shown in Figure 2, **starting at vertex A**. List all the vertices along with their distance from vertex A. Are there any unreachable vertices (from A)? Which ones?



Edges added in order: A-B, A-C, B-D, D-F, C-E, C-H, H-G
There are no unreachable vertices from A.

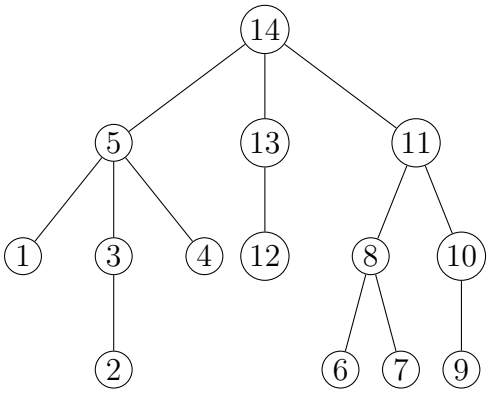
Vertices	A	B	C	D	F	E	H	G
d(A,vtx)	0	4	5	5	6	7	9	10



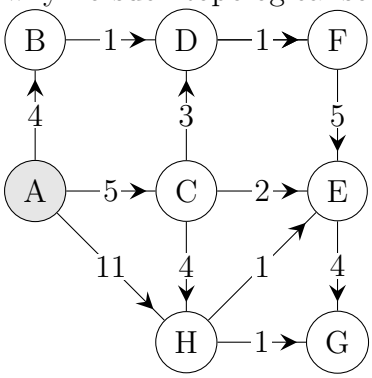
5. List the vertices of the tree shown in Figure 3 using the preorder traversal.
   Preorder: A B C D E F G N H I J K L M

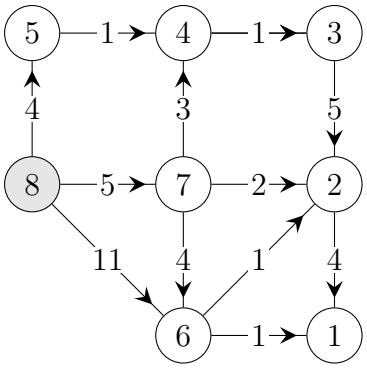
6. List the vertices of the tree shown in Figure 3 using breadth first search.
   BFS: A B G H C D F I L N E J K M (Ties broken alphabetically)

7. Label the vertices of the tree shown in Figure 3 using a topological ordering.



8. Find a topological ordering of the vertices of the digraph shown in Figure 2, or explain why no such topological sorting exists.





9. The "while" statement of the topological ordering digraph algorithm includes the statement that we can find a vertex v with no incoming edges. Explain why, if every vertex of a digraph has at least one incoming edge, then the digraph must have a cycle.

   If there isn't a vertex with no incoming edges, then there is no source in the graph. If there is no source, then there is no root in the graph, so there must be a cycle.