# Spanning Tree

## Definition

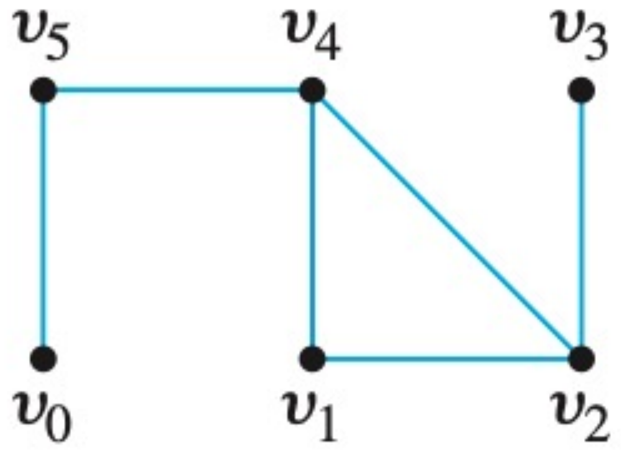
A **spanning tree** for a graph *G* is a subgraph of *G* that contains every vertex of *G* and is a tree.

## Theorem:

1. Every connected graph has a spanning tree.
2. Any two spanning trees for a graph have the same number of edges.

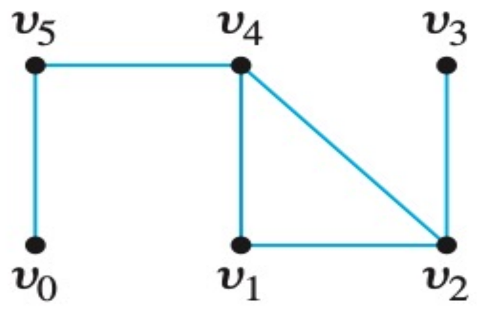
# Example:

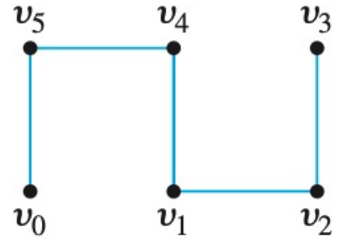
Find all spanning trees for the graph *G* pictured below.

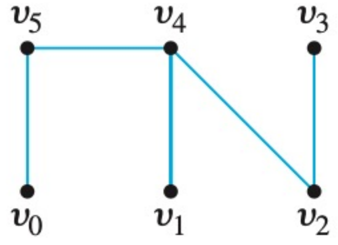


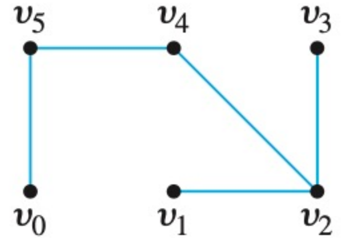
## Example:

Find all spanning trees for the graph *G* pictured below.









## Definition:

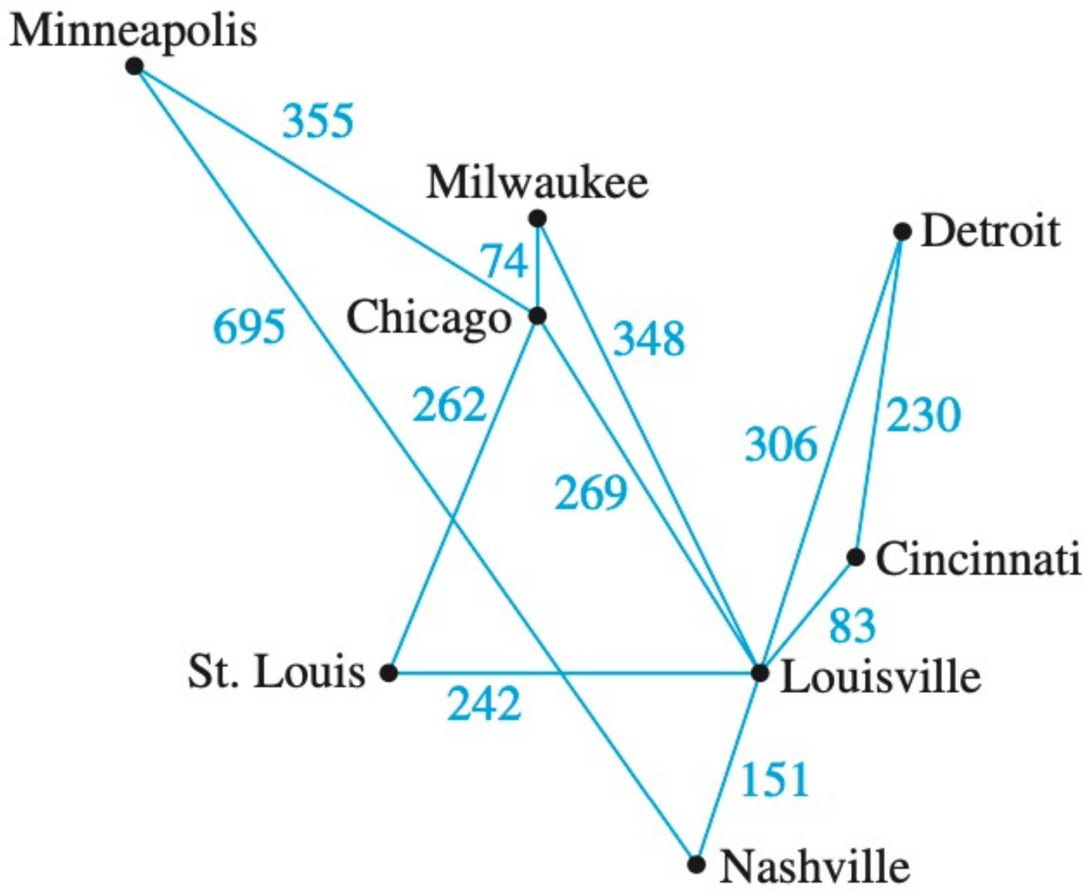
- A **weighted graph** is a graph for which each edge has an associated positive real number **weight**.
- The sum of the weights of all the edges is the **total weight** of the graph.
- A **minimum spanning tree** for a connected, weighted graph is a spanning tree that has the least possible total weight compared to all other spanning trees for the graph.
- If $G$ is a weighed graph and $e$ is an edge of $G$, then $\boldsymbol{w(e)}$ denotes the weight of $e$ and $\boldsymbol{w(G)}$ denotes the total weight of $G$.

## *Kruskal’s Algorithm*

In Kruskal’s algorithm, the edges of a connected, weighted graph are examined one by one in order of increasing weight. At each stage the edge being examined is added to what will become the minimum spanning tree, provided that this addition does not create a circuit. After $n$-1 edges have been added (where $n$ is the number of vertices of the graph), these edges, together with the vertices of the graph, form a minimum spanning tree for the graph.

## Example:

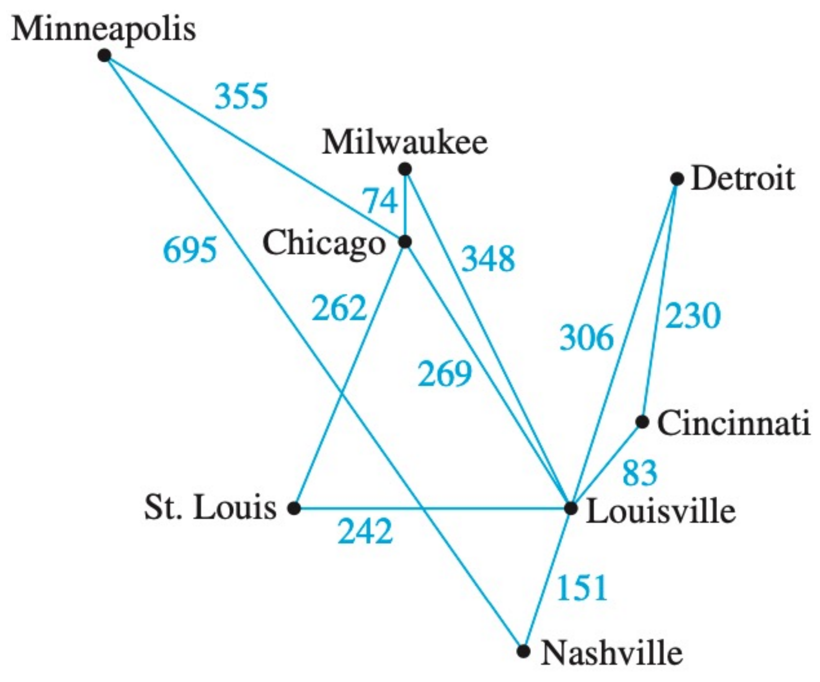
Describe the action of Kruskal's algorithm on the graph shown in the figure, where *n*=8.



# Example:

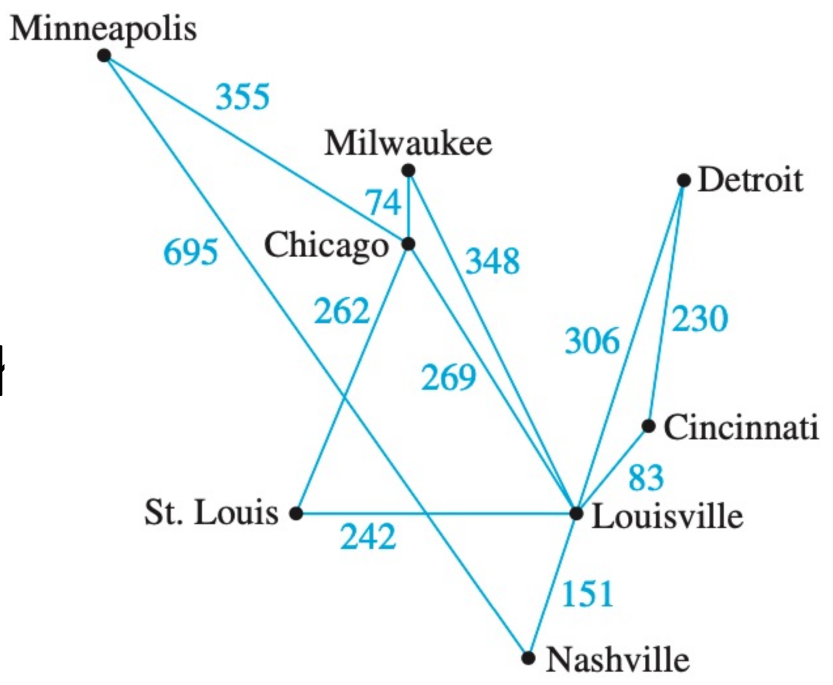
Describe the action of Kruskal’s algorithm on the graph shown in the figure, where *n*=8.

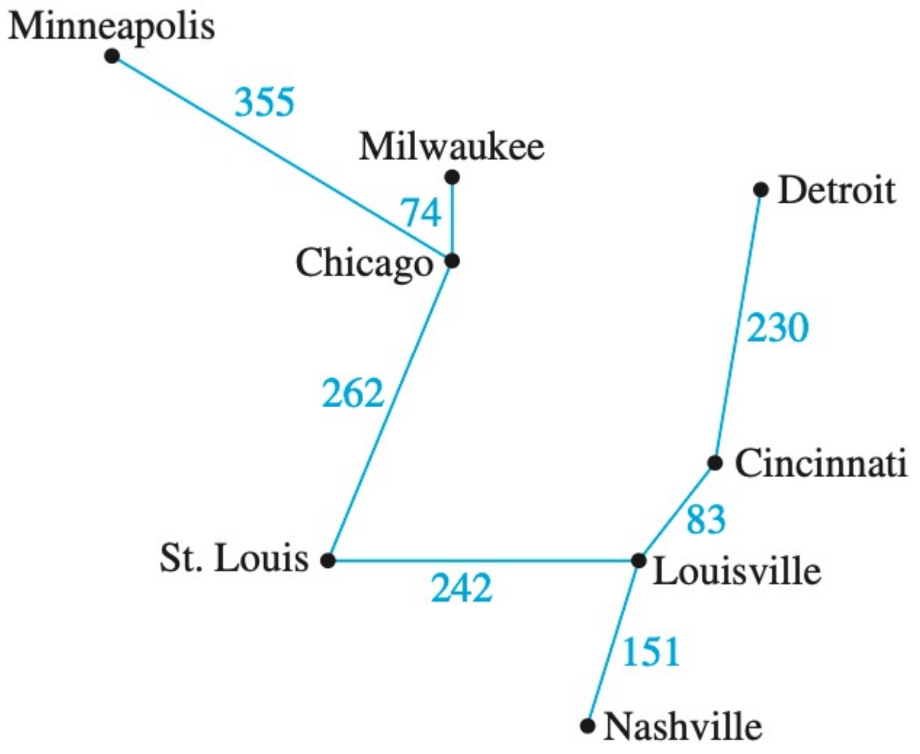
| Iteration Number | Edge Considered | Weight | Action Taken |
|---|---|---|---|
| 1 | Chicago–Milwaukee | 74 | added |
| 2 | Louisville–Cincinnati | 83 | added |
| 3 | Louisville–Nashville | 151 | added |
| 4 | Cincinnati–Detroit | 230 | added |
| 5 | St. Louis–Louisville | 242 | added |
| 6 | St. Louis–Chicago | 262 | added |
| 7 | Chicago–Louisville | 269 | not added |
| 8 | Louisville–Detroit | 306 | not added |
| 9 | Louisville–Milwaukee | 348 | not added |
| 10 | Minneapolis–Chicago | 355 | added |

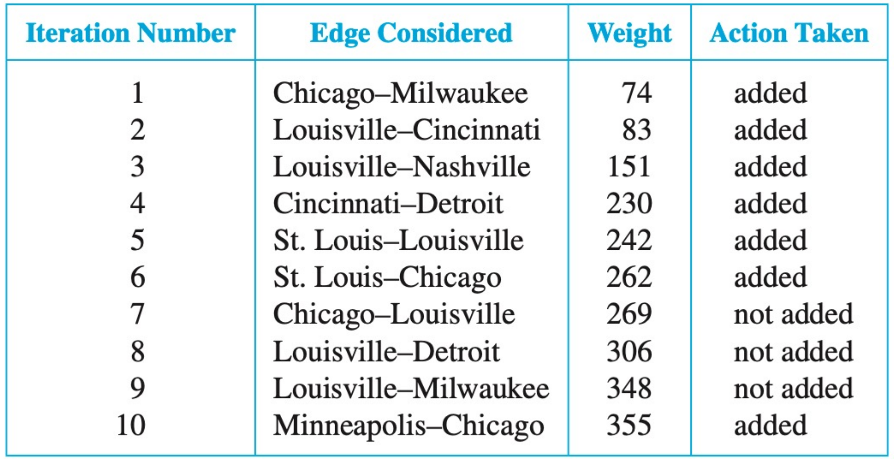


## Example:

Describe the action of Kruskal's algorithm on th
figure, where *n*=8.







| Iteration Number | Edge Considered | Weight | Action Taken |
|---|---|---|---|
| 1 | Chicago–Milwaukee | 74 | added |
| 2 | Louisville–Cincinnati | 83 | added |
| 3 | Louisville–Nashville | 151 | added |
| 4 | Cincinnati–Detroit | 230 | added |
| 5 | St. Louis–Louisville | 242 | added |
| 6 | St. Louis–Chicago | 262 | added |
| 7 | Chicago–Louisville | 269 | not added |
| 8 | Louisville–Detroit | 306 | not added |
| 9 | Louisville–Milwaukee | 348 | not added |
| 10 | Minneapolis–Chicago | 355 | added |

## Theorem: Correctness of Kruskal’s Algorithm

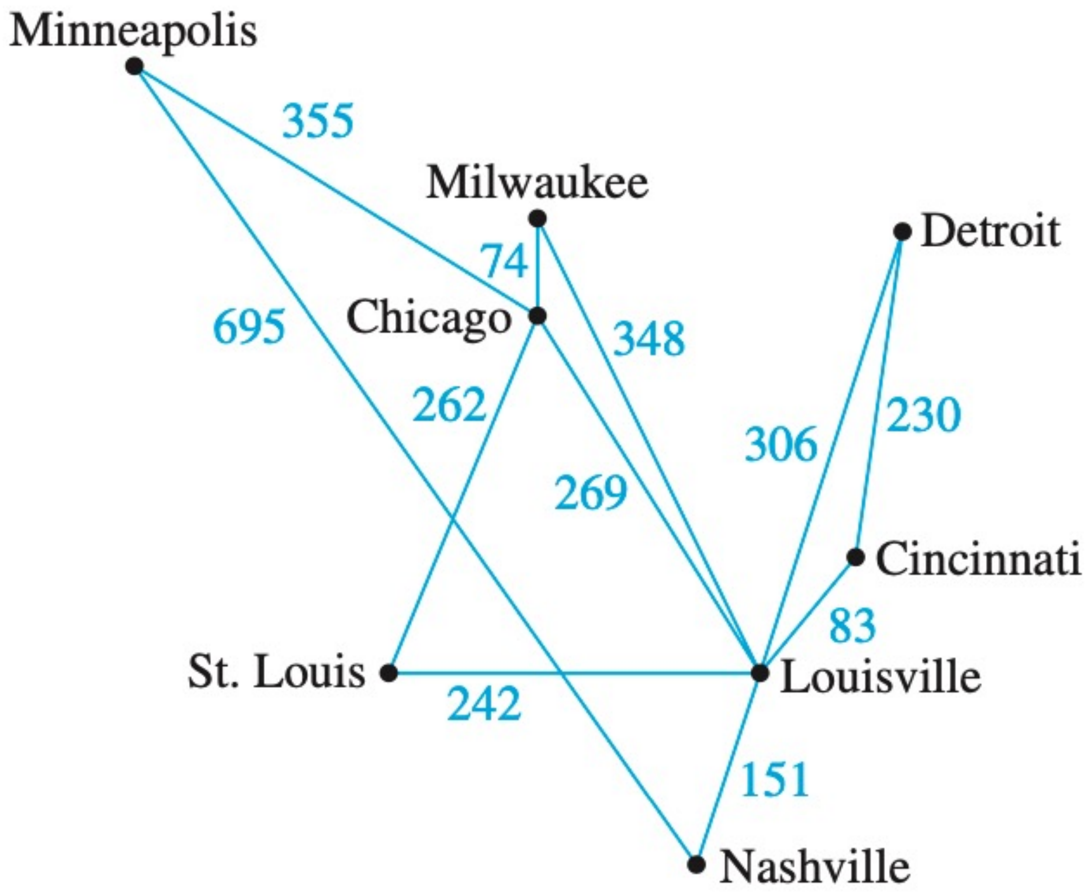
When a connected, weighted graph is input to Kruskal’s algorithm, the output is a minimum spanning tree.

## Prim’s Algorithm:

Prim’s algorithm works differently from Kruskal’s. It builds a minimum spanning tree *T* by expanding outward in connected links from some vertex. One edge and one vertex are added at each stage. The edge added is the one of least weight that connects the vertices already in *T* with those not in *T*, and the vertex is the endpoint of this edge that is not already in *T*.

## Example:

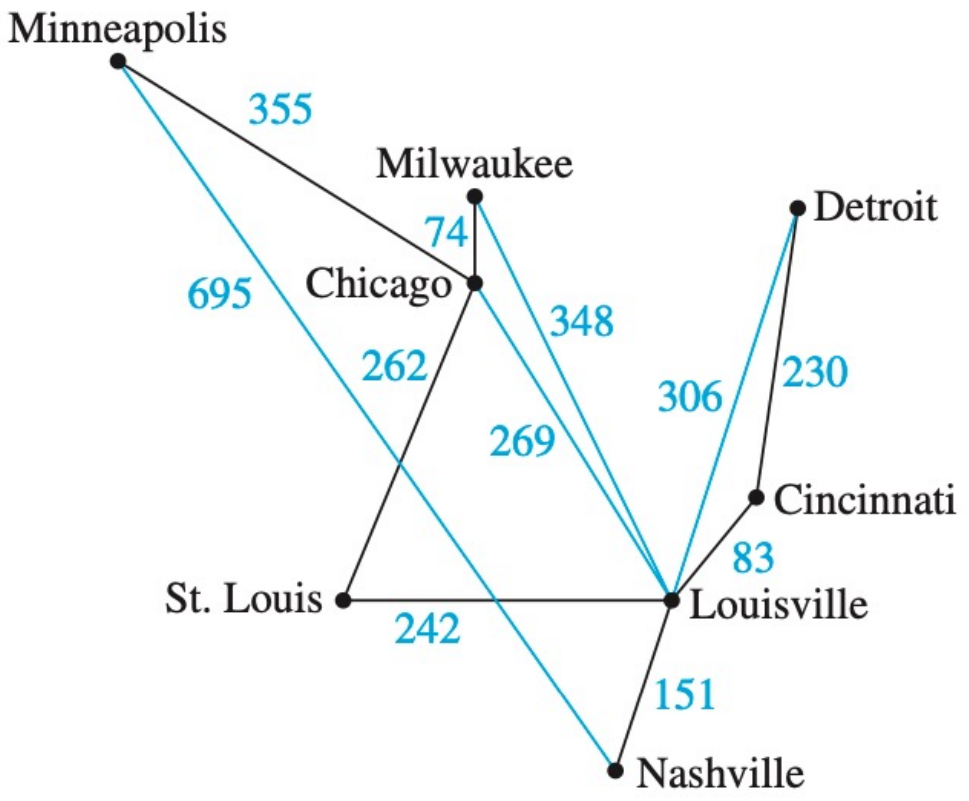
Describe the action of Prim’s algorithm on the graph shown in the figure, where *n*=8.



## Example:

Describe the action of Prim's algorithm on the graph shown in the figure, where *n*=8.

| Iteration Number | Vertex Added | Edge Added | Weight |
|---|---|---|---|
| 0 | Minneapolis | | |
| 1 | Chicago | Minneapolis–Chicago | 355 |
| 2 | Milwaukee | Chicago–Milwaukee | 74 |
| 3 | St. Louis | Chicago–St. Louis | 262 |
| 4 | Louisville | St. Louis–Louisville | 242 |
| 5 | Cincinnati | Louisville–Cincinnati | 83 |
| 6 | Nashville | Louisville–Nashville | 151 |
| 7 | Detroit | Cincinnati–Detroit | 230 |

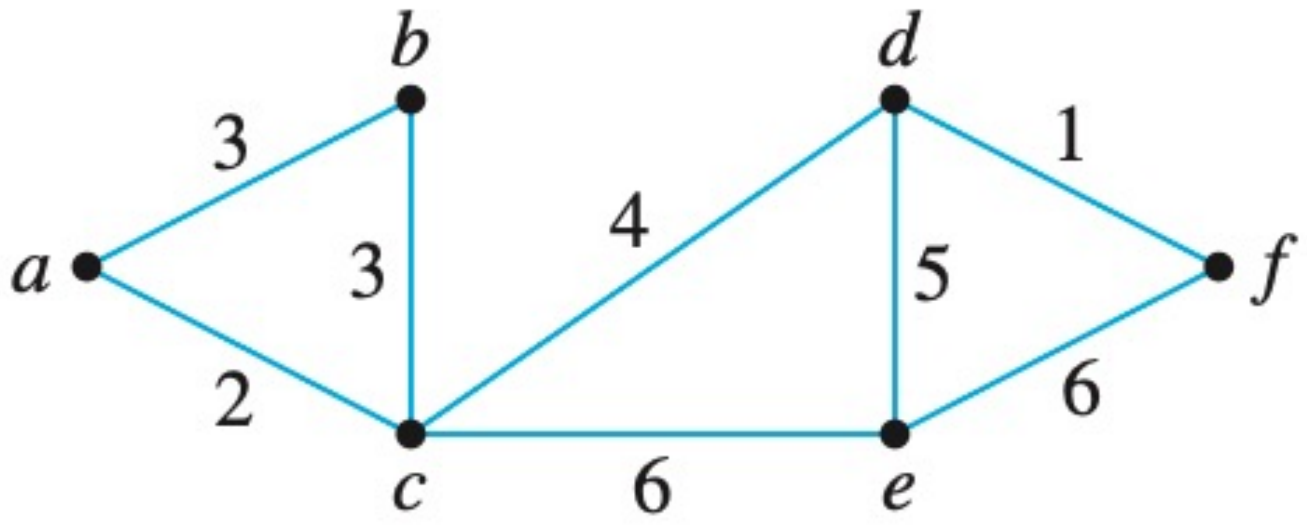


## Theorem: Correctness of Prim’s Algorithm

When a connected, weighted graph is input to Prim’s algorithm, the output is a minimum spanning tree.

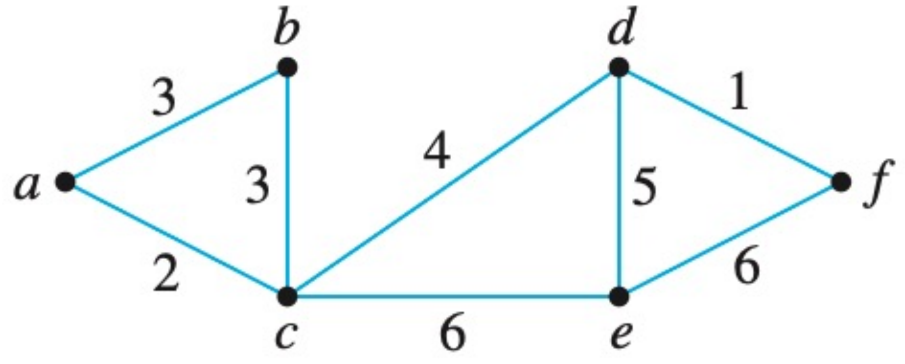
## Example:

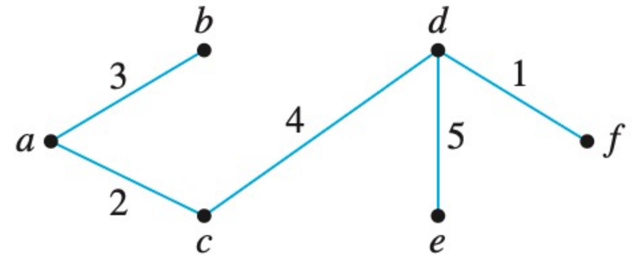
Find all minimum spanning trees for the following graph. Use Kruskal's algorithm and Prim's algorithm starting at vertex *a*. Indicate the order in which edges are added to form each tree.



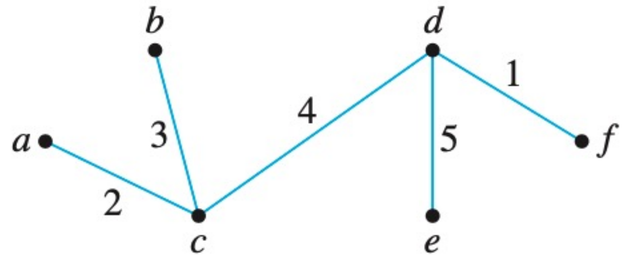
## Example:

Find all minimum spanning trees for the following graph. Use Kruskal’s algorithm and Prim’s algorithm starting at vertex *a*. Indicate the order in which edges are added to form each tree.





(a)



(b)