# **Prim's Algorithm**

Prim's algorithm is a [minimum spanning tree](https://www.programiz.com/dsa/spanning-tree-and-minimum-spanning-tree#minimum-spanning) algorithm that takes a graph as input and finds the subset of the edges of that graph which

* form a tree that includes every vertex
* has the minimum sum of weights among all the trees that can be formed from the graph

## **How Prim's algorithm works**

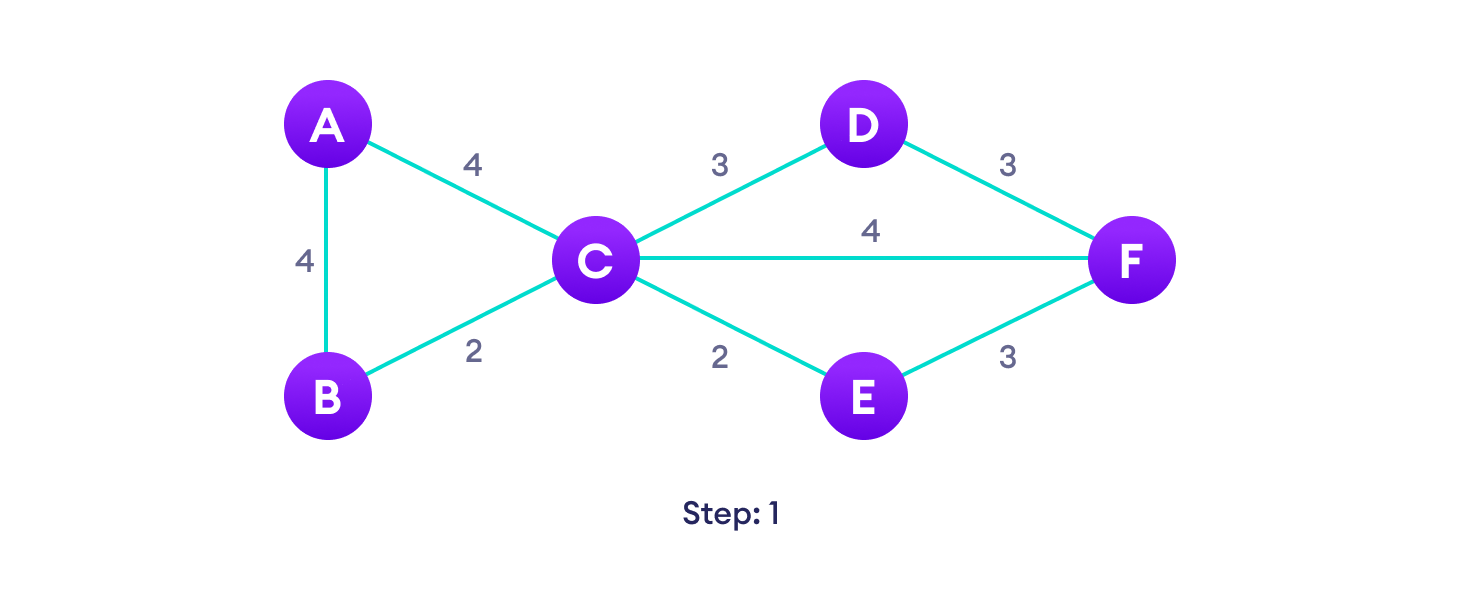
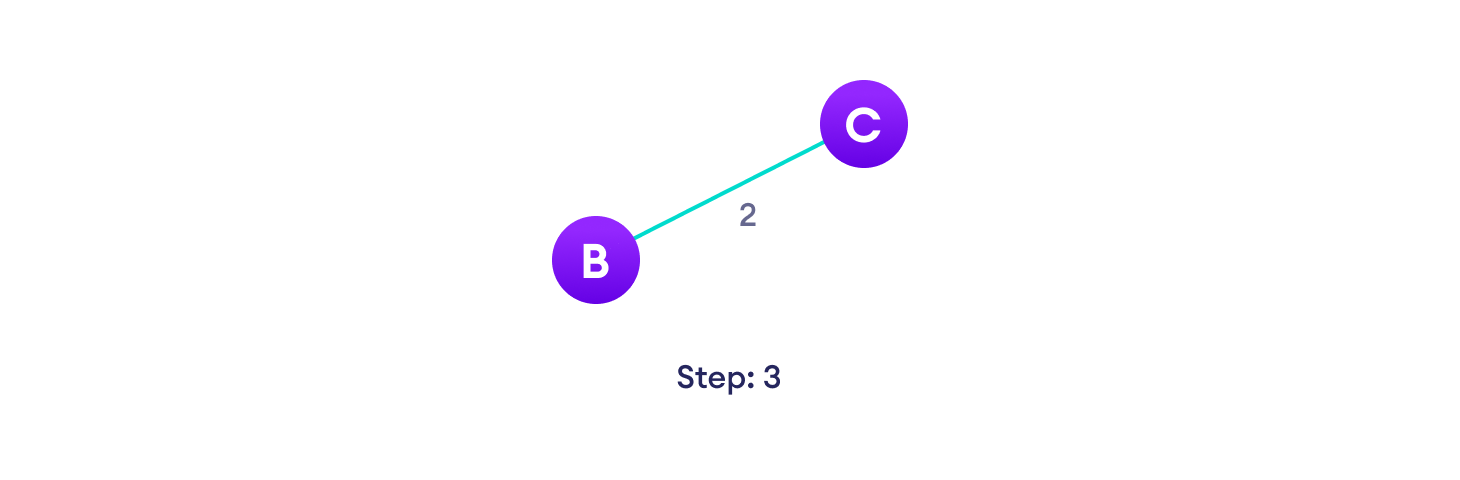
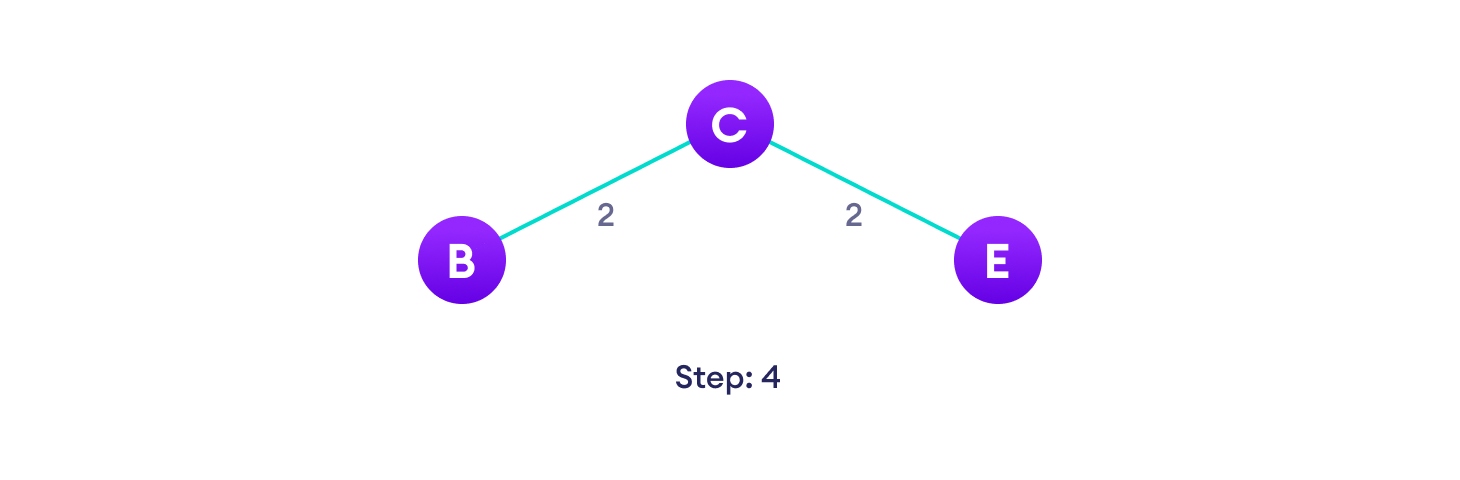
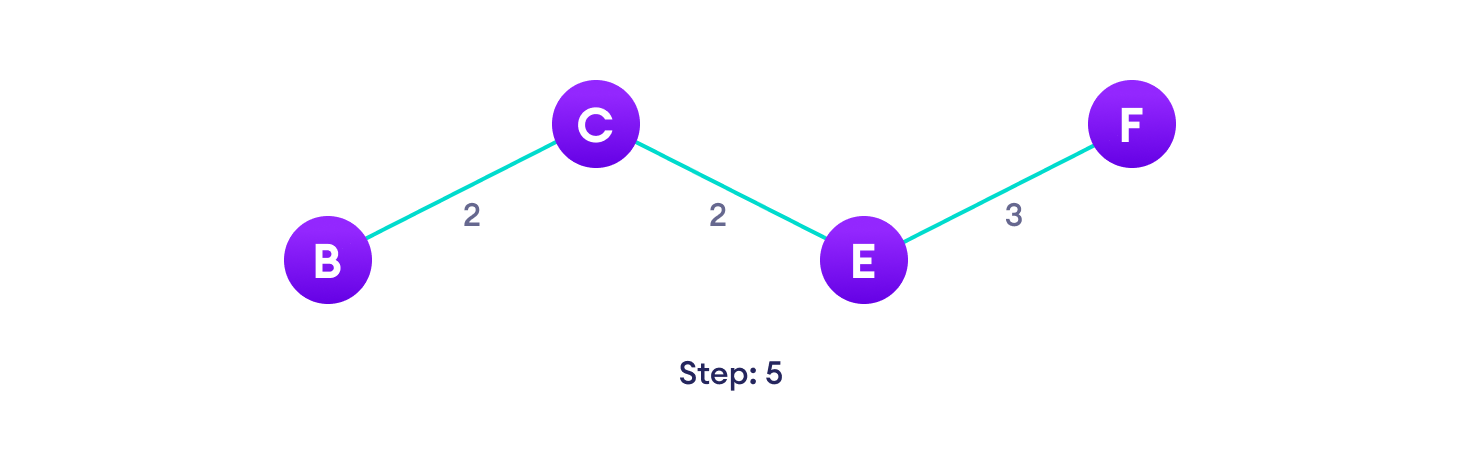
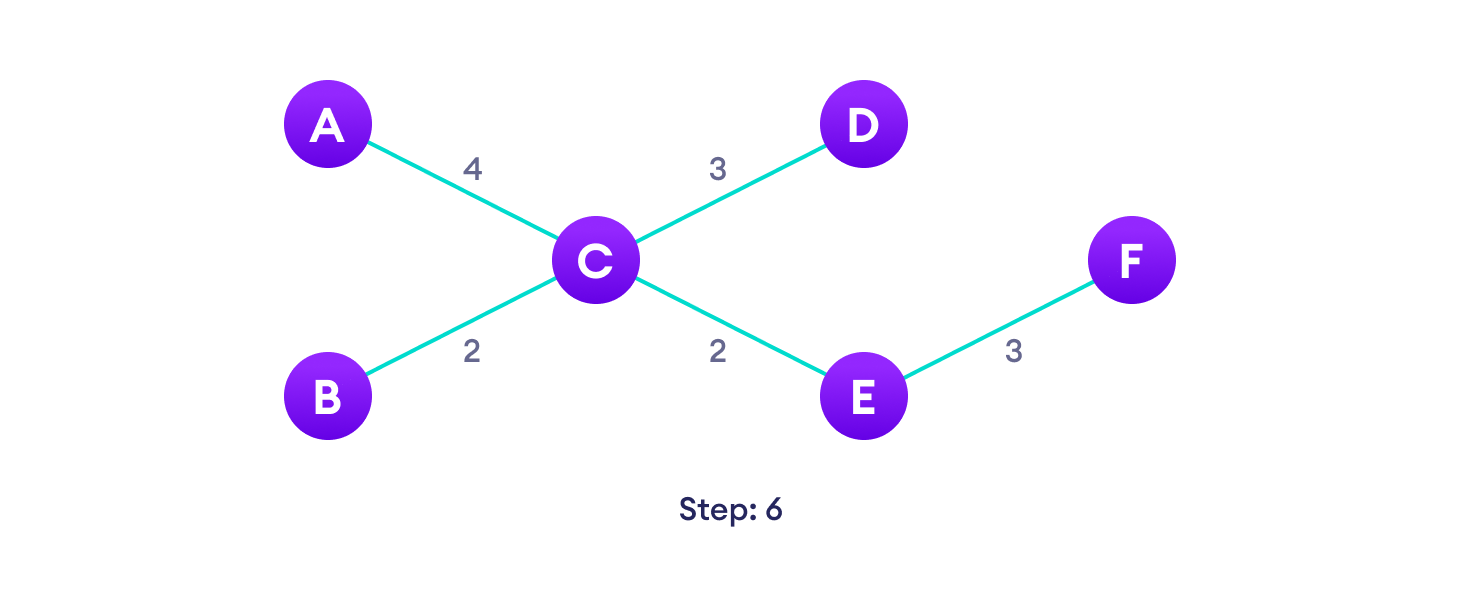
It falls under a class of algorithms called [greedy algorithms](https://www.programiz.com/dsa/greedy-algorithm) that find the local optimum in the hopes of finding a global optimum.

We start from one vertex and keep adding edges with the lowest weight until we reach our goal.

The steps for implementing Prim's algorithm are as follows:

1. Initialize the minimum spanning tree with a vertex chosen at random.
2. Find all the edges that connect the tree to new vertices, find the minimum and add it to the tree
3. Keep repeating step 2 until we get a minimum spanning tree

## **Example of Prim's algorithm**

****Start with a weighted graphChoose a vertexChoose the shortest edge from this vertex and add itChoose the nearest vertex not yet in the solutionChoose the nearest edge not yet in the solution, if there are multiple choices, choose one at randomRepeat until you have a spanning tree

## **Prim's Algorithm pseudocode**

The pseudocode for prim's algorithm shows how we create two sets of vertices U and V-U. U contains the list of vertices that have been visited and V-U the list of vertices that haven't. One by one, we move vertices from set V-U to set U by connecting the least weight edge.

T = ∅;

U = { 1 };

while (U ≠ V)

let (u, v) be the lowest cost edge such that u ∈ U and v ∈ V - U;

T = T ∪ {(u, v)}

U = U ∪ {v}

## **Prim's vs Kruskal's Algorithm**

[Kruskal's algorithm](https://www.programiz.com/dsa/kruskal-algorithm) is another popular minimum spanning tree algorithm that uses a different logic to find the MST of a graph. Instead of starting from a vertex, Kruskal's algorithm sorts all the edges from low weight to high and keeps adding the lowest edges, ignoring those edges that create a cycle.

## **Prim's Algorithm Complexity**

The time complexity of Prim's algorithm is O(E log V).

## **Prim's Algorithm Application**

* Laying cables of electrical wiring
* In network designed
* To make protocols in network cycles