Prim's Algorithm

In this tutorial, you will learn how Prim's Algorithm works. Also, you will find working examples of Prim's Algorithm in C, C++, Java and Python.

Prim's algorithm is a minimum spanning tree 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

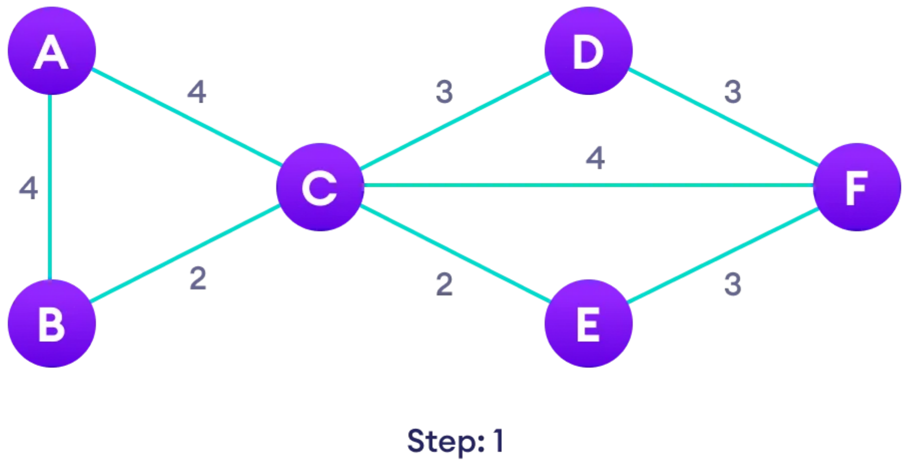
It falls under a class of algorithms called greedy algorithms that find the local optimum in the hopes of finding a global optimum.

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

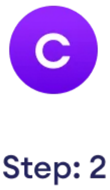
The steps for implementing Prim's algorithm are as follows:

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

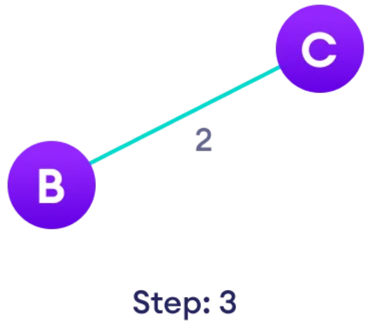
Example of Prim's algorithm



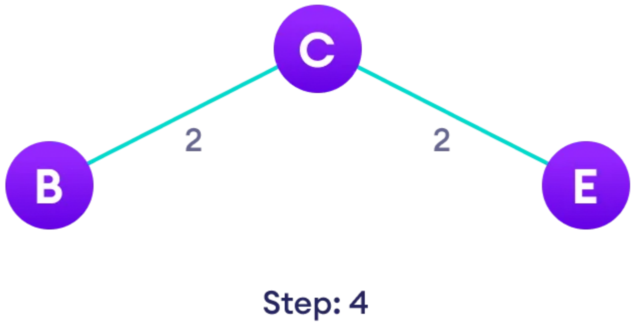
Start with a weighted graph



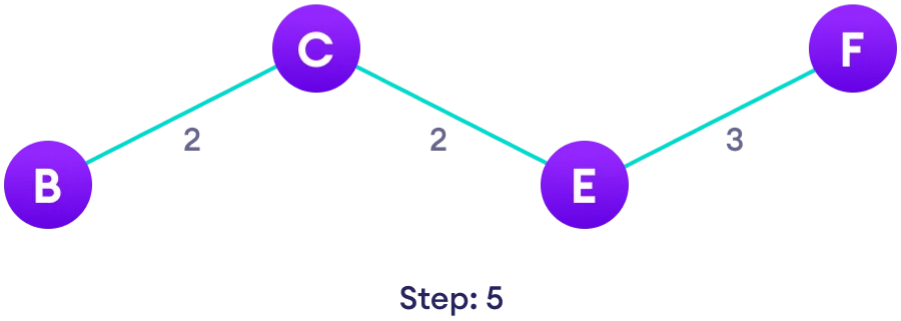
Choose a vertex



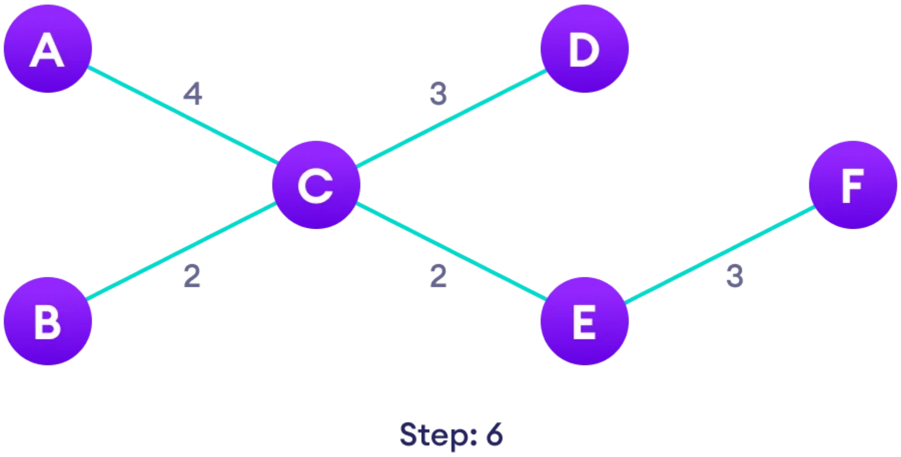
Choose the shortest edge from this vertex and add it



Choose the nearest vertex not yet in the solution



Choose the nearest edge not yet in the solution, if there are multiple choices, choose one at random



Repeat 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}
```

Python, Java and C/C++ Examples

Although adjacency matrix representation of graphs is used, this algorithm can also be implemented using Adjacency List to improve its efficiency.

Python Java C C++

```
// Prim's Algorithm in C

#include<stdio.h>
#include<stdbool.h>

#define INF 9999999

// number of vertices in graph
#define V 5

// create a 2d array of size 5x5
//for adjacency matrix to represent graph
int G[V][V] = {
  {0, 9, 75, 0, 0},
  {9, 0, 95, 19, 42},
  {75, 95, 0, 51, 66},
  {0, 19, 51, 0, 31},
  {0, 42, 66, 31, 0}};

int main() {
  int no_edge;  // number of edge

  // create a array to track selected vertex
  // selected will become true otherwise false
  int selected[V];

  // set selected false initially
  memset(selected, false, sizeof(selected));
```

Prim's vs Kruskal's Algorithm

Kruskal's 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