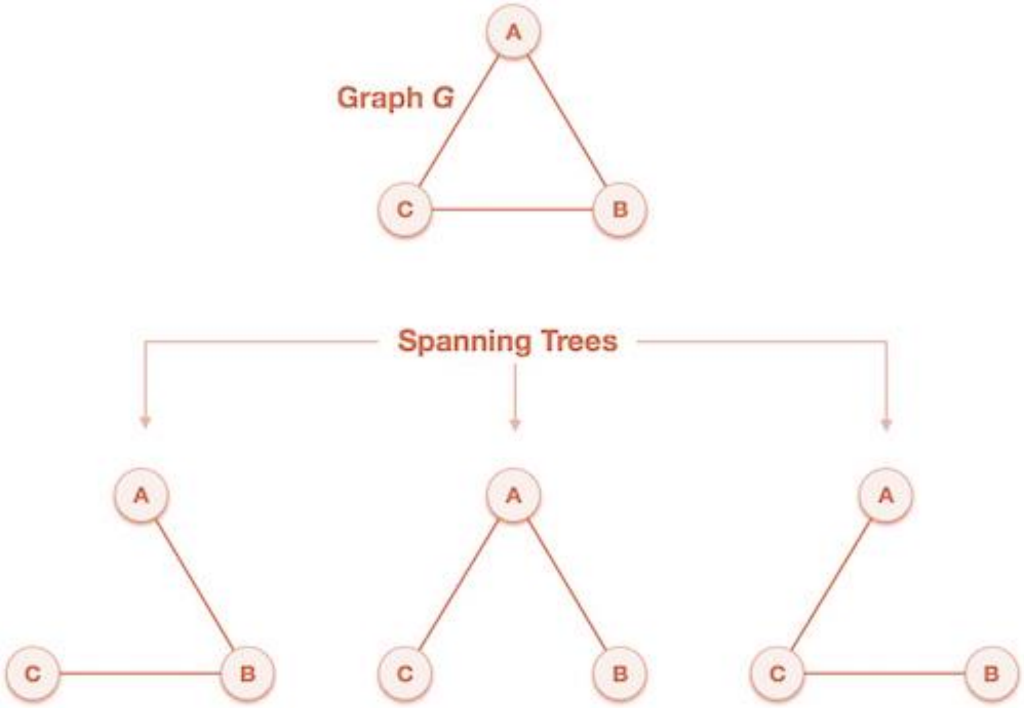
DATA STRUCTURES & ALGORITHMS

Spanning Tree

Instructor: Engr. Laraib Siddiqui

Spanning Trees

A spanning tree is a subset of Graph G, which has all the vertices covered with minimum possible number of edges.



Minimum Spanning Tree (MST)

In a weighted graph, a minimum spanning tree is a spanning tree that has minimum weight than all other spanning trees of the same graph.

In real-world situations, this weight can be measured as distance, congestion, traffic load or any arbitrary value denoted to the edges.

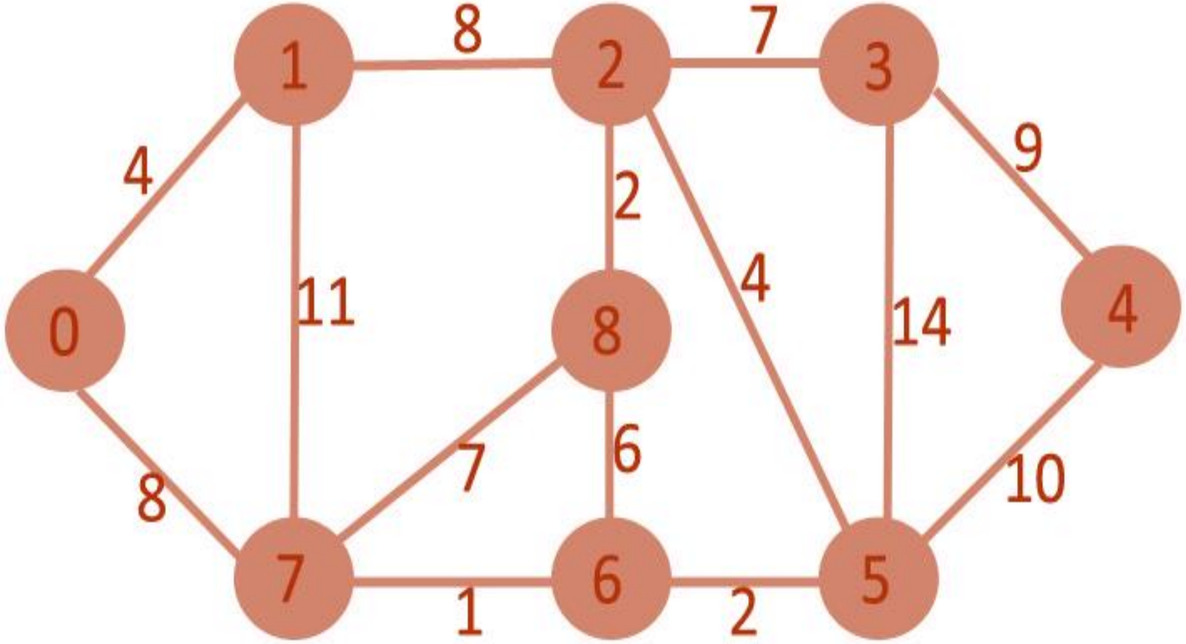
- Kruskal Algorithm
- Prim's Algorithm

Kruskal's Algorithm

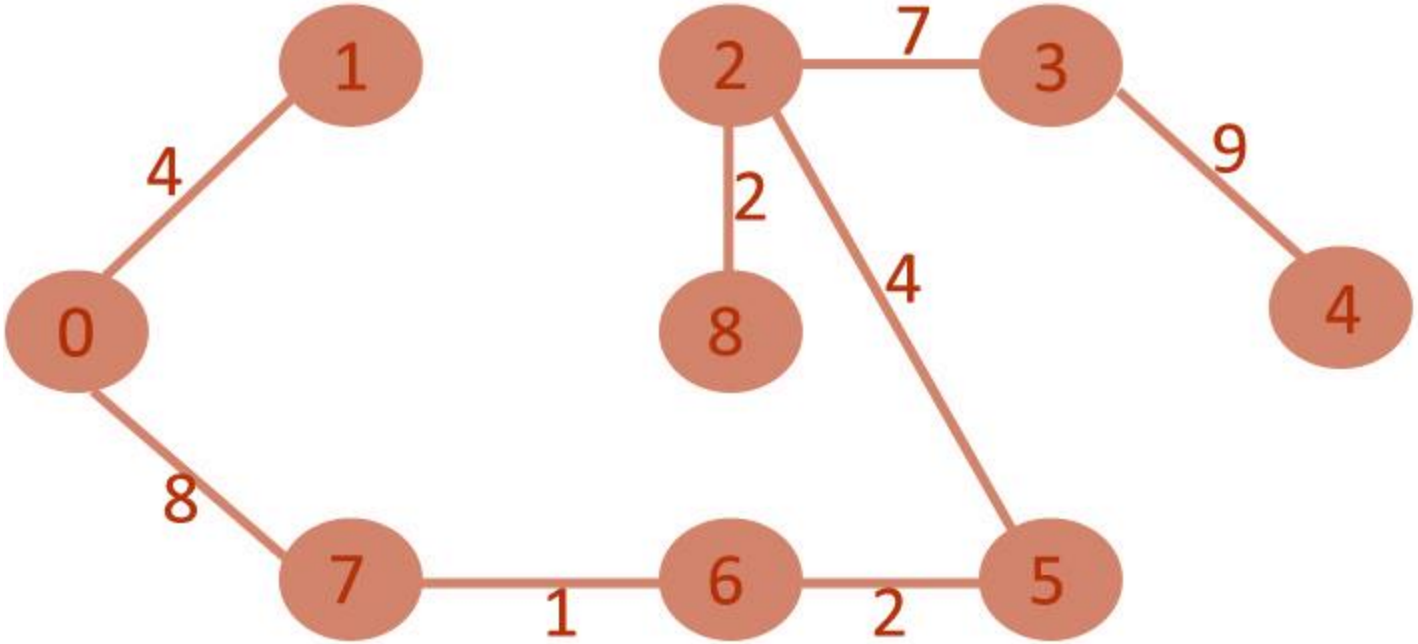
Kruskal's algorithm to find the minimum cost spanning tree uses the greedy approach. This algorithm treats the graph as a forest and every node it has as an individual tree. A tree connects to another only and only if, it has the least cost among all available options and does not violate MST properties.

Algorithm

- Arrange the edge of G in order of increasing weight.
- Starting only with the vertices of G and proceeding sequentially add each edge which does not result in a cycle, until (n - 1) edges are used.
- EXIT.



Weight	Source	Destination	
1	7	6	√
2	8	2	√
2	6	5	√
4	0	1	√
4	5	2	√
6	6	8	X
7	2	3	√
7	7	8	X
8	0	7	√
8	1	2	X
9	3	4	√
10	5	4	X
11	1	7	X
14	5	3	X



1
2
7
3
4
2
9
4
0
8
4
8
7
1
6
2
5

Prim’s Algorithm

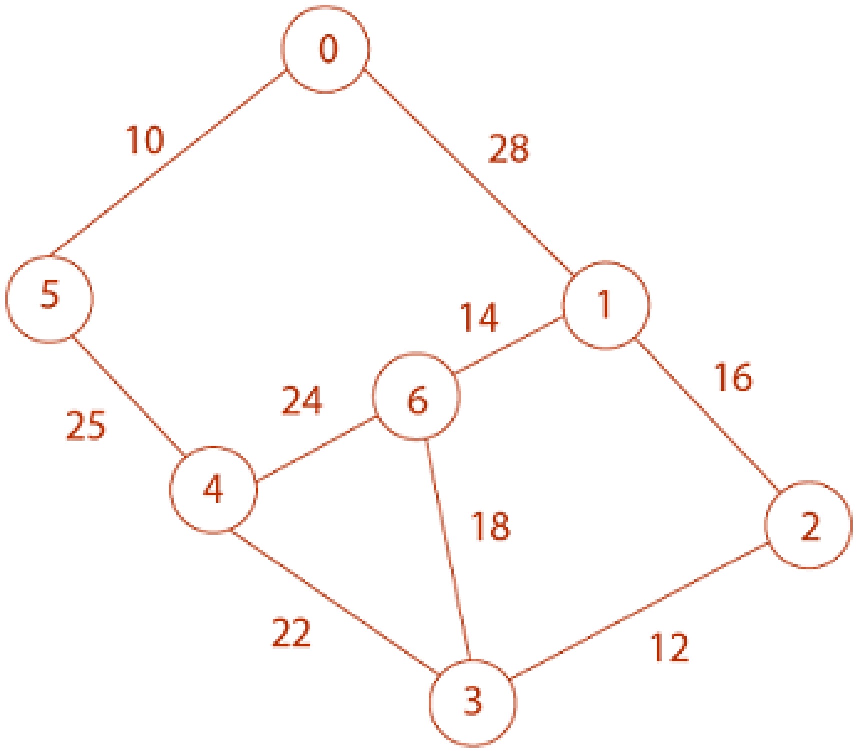
It is a greedy algorithm. It starts with an empty spanning tree. The idea is to maintain two sets of vertices:

- Contain vertices already included in MST.
- Contain vertices not yet included.

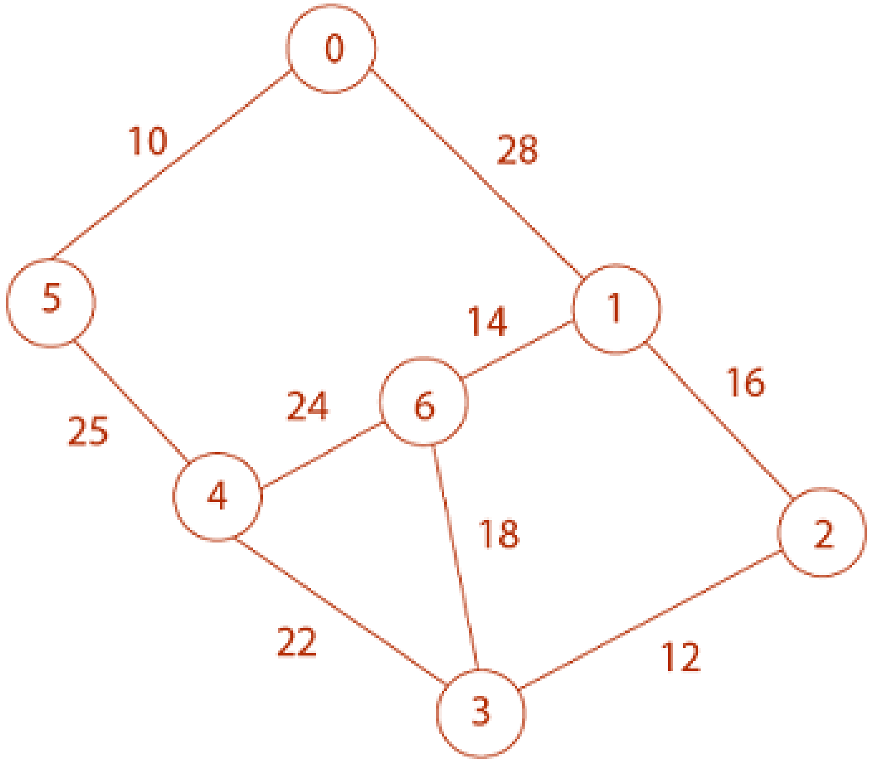
- At every step, it considers all the edges and picks the minimum weight edge. After picking the edge, it moves the other endpoint of edge to set containing MST.

Algorithm

- Create MST set that keeps track of vertices already included in MST.
- Assign key values to all vertices in the input graph. Initialize all key values as INFINITE (∞). Assign key values like 0 for the first vertex so that it is picked first.
- While MST set doesn't include all vertices.
  - Pick vertex u which is not is MST set and has minimum key value. Include 'u'to MST set.
  - Update the key value of all adjacent vertices of u. To update, iterate through all adjacent vertices. For every adjacent vertex v, if the weight of edge u.v less than the previous key value of v, update key value as a weight of u.v



Vertex	0	1	2	3	4	5	6
Key	0	∞	∞	∞	∞	∞	∞
Parent	NIL	NIL	NIL	NIL	NIL	NIL	NIL

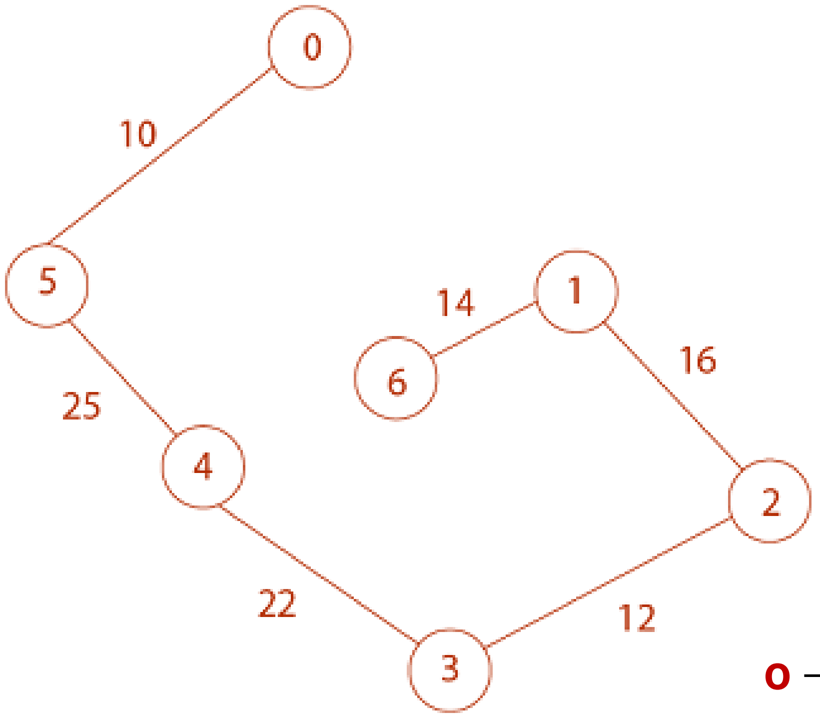


Vertex	0	1	2	3	4	5	6
Key	0	28	∞	∞	∞	10	∞
Parent	NIL	0	NIL	NIL	NIL	0	NIL

Vertex	0	1	2	3	4	5	6
Key	0	28	∞	∞	25	10	∞
Parent	NIL	0	NIL	NIL	5	0	NIL

Vertex	0	1	2	3	4	5	6
Key	0	28	∞	22	25	10	24
Parent	NIL	0	NIL	4	5	0	4

Vertex	0	1	2	3	4	5	6
Key	0	28	12	22	25	10	18
Parent	NIL	0	3	4	5	0	3



Vertex	0	1	2	3	4	5	6
Key	0	16	12	22	25	10	18
Parent	NIL	2	3	4	5	0	3

Vertex	0	1	2	3	4	5	6
Key	0	16	12	22	25	10	14
Parent	NIL	2	3	4	5	0	1

$0 \longrightarrow 5 \longrightarrow 4 \longrightarrow 3 \longrightarrow 2 \longrightarrow 1 \longrightarrow 6$

Total Cost = 10 + 25 + 22 + 12 + 16 + 14 = 99