Kruskal's algorithm

Kruskal's algorithm is a greedy algorithm used to find a minimum spanning tree for a connected, weighted graph.

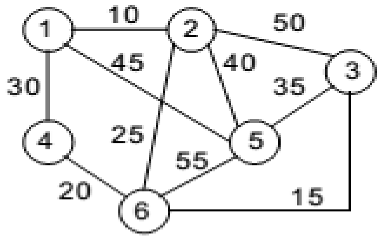
```
          MST-KRUSKAL(G, w)
O(1)        A = ∅
O(V)        for each vertex v ∈ G.V
                MAKE-SET(v)
O(E logE)   sort the edges of G.E into nondecreasing order by weight w
            for each edge (u, v) ∈ G.E, taken in nondecreasing order by weight
                if FIND-SET(u) ≠ FIND-SET(v)
O(V logV)           A = A ∪ {(u, v)}
                    UNION(u, v)
            return A
```

Sort Edges by Weight: Arrange all the edges in the graph in non-decreasing order of their weights.

Iterate through Edges: For each edge in the sorted order: If adding the edge to the set of selected edges doesn't create a cycle, add the edge to the set (ensuring it doesn't create a cycle in the current set).

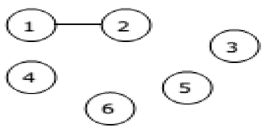
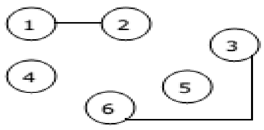
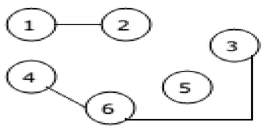
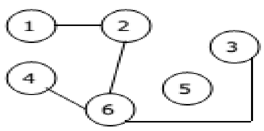
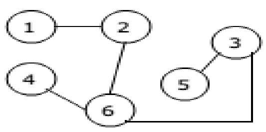
Cycle Detection: To check if adding an edge creates a cycle, you can use a disjoint-set data structure (e.g., Union-Find). Keep track of the disjoint sets (connected components) and merge them as you add edges to ensure no cycles are formed.

Example:



Arrange all the edges in the increasing order of their costs:

Cost	***10***	***15***	***20***	***25***	***30***	***35***	***40***	***45***	***50***	***55***
Edge	(1, 2)	(3, 6)	(4, 6)	(2, 6)	(1, 4)	(3, 5)	(2, 5)	(1, 5)	(2, 3)	(5, 6)

EDGE	COST	STAGES IN KRUSKAL'S ALGORITHM	REMARKS
(1, 2)	10		The edge between vertices 1 and 2 is the first edge selected. It is included in the spanning tree.
(3, 6)	15		Next, the edge between vertices 3 and 6 is selected and included in the tree.
(4, 6)	20		The edge between vertices 4 and 6 is next included in the tree.
(2, 6)	25		The edge between vertices 2 and 6 is considered next and included in the tree.
(1, 4)	30	Reject	The edge between the vertices 1 and 4 is discarded as its inclusion creates a cycle.
(3, 5)	35		Finally, the edge between vertices 3 and 5 is considered and included in the tree built. This completes the tree. The cost of the minimal spanning tree is 105.

Time Complexity

- Considering the sorting and disjoint-set operations, the overall time complexity of Kruskal's algorithm is O(E log E)