

Kruskal's algorithm is another popular algorithm for finding the minimum spanning tree (MST) of a weighted undirected graph.

It is based on sorting the edges of the graph in non-decreasing order of their weights.

Outline of Kruskal's algorithm:

1. Initialize an empty set to store the MST.
2. Create a disjoint-set data structure to keep track of the connected components. Initially, each vertex is in its own set.
3. Sort the edges of the graph in non-decreasing order of their weights. This can be done using any sorting algorithm.
4. Iterate through each edge in the sorted order:
 - a. Check if adding the current edge to the MST creates a cycle. This can be done by checking if the vertices of the edge belong to different sets in the disjoint-set data structure.
 - b. If the edge does not create a cycle, add it to the MST and merge the sets of the vertices using the disjoint-set data structure.
5. Return the MST.

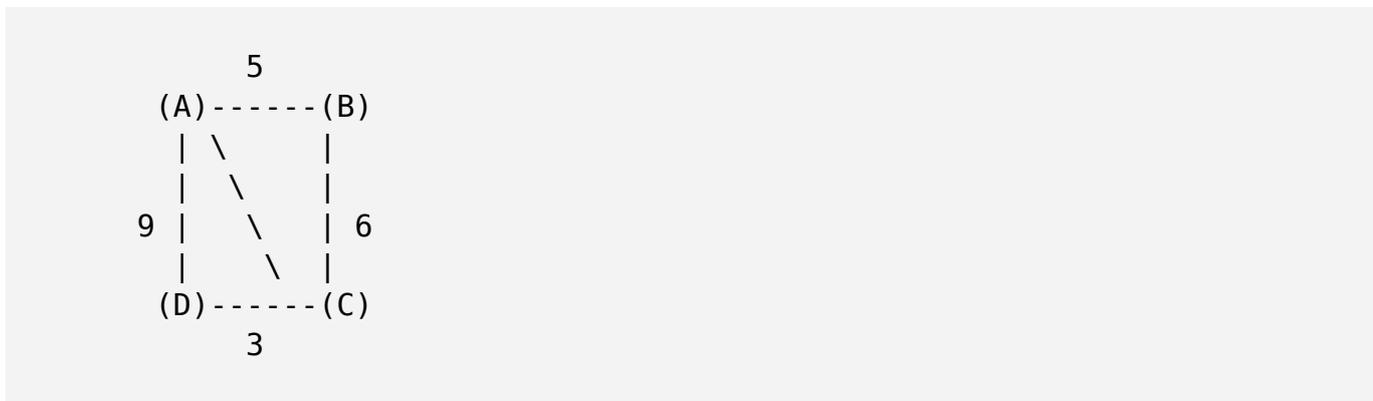
The pseudocode for Kruskal's algorithm:

```
Kruskal's Algorithm:  
Input: Graph G with vertices V and edges E, weights assigned to each edge
```

```
1. Initialize an empty set to store the MST: MST = {}
2. Create a disjoint-set data structure to keep track of the connected components.
3. Sort the edges of G in non-decreasing order of their weights.
4. Iterate through each edge (u, v) in the sorted order:
  a. If adding (u, v) to MST does not create a cycle:
    - Add (u, v) to MST.
    - Merge the connected components of u and v using the disjoint-set data structure.
5. Return MST.
```

Example:

Undirected graph



Resulting minimum spanning tree

