IT-081 - Pankhania Aanandi R.

DAA :-

Kruskal's Algorithm (Greedy method)

Problem Analysis:-

→ Kruskal's algo is used to find the minimum cost spanning tree (uses greedy method)

→ This Algo treats a graph as a forest & every node it has as an individual tree.

→ A tree connects another only and only if, it has the least cost among all available options and does not violate MST properties.

Development model:

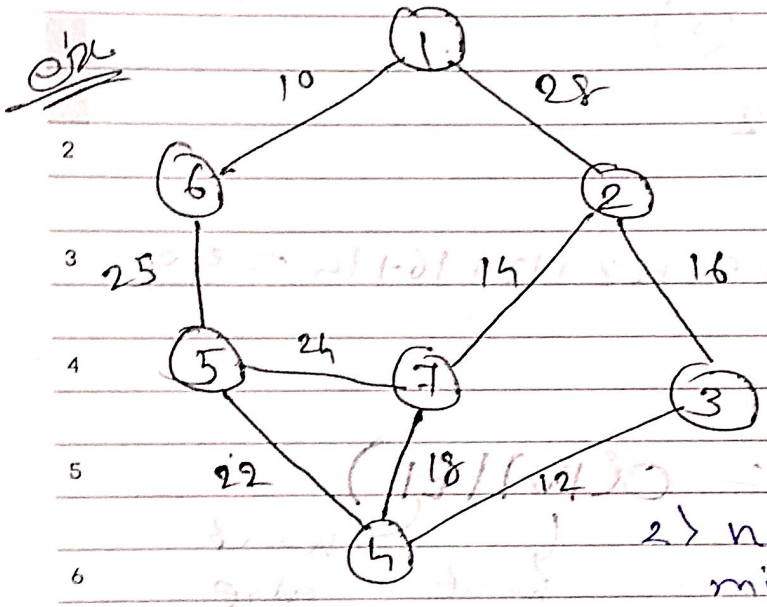
→ sort all the edges in non-decreasing order of their weight.

→ Pick the smallest edge. check if it forms a cycle with the spanning tree formed for so far. If cycle isn't formed, include this edge. Else discard it.

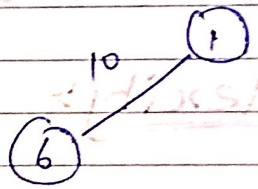
→ repeat above step until there are V-1 edges in the spanning tree.

→ Prim's Algo fails when there are negative weight edges in the given graph. When Kruskal's Algo can detect if there's any -ve edge present or not.

Analysis:-

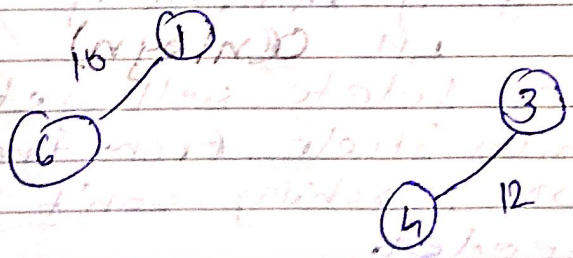


1.> select the min. cost edge first i.e. ① — ⑥ = 10

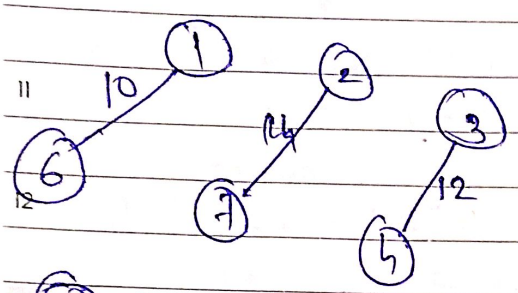


2> now select the second min. cost having edge i.e. ③ — ④ = 12

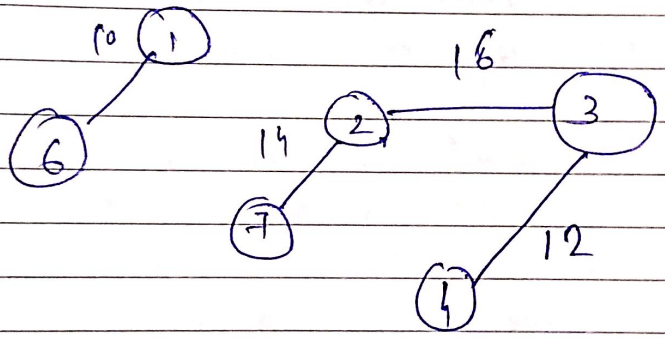
check if it is creating complete cycle in there? No. take it.



3) Now we should take 14 which isn't creating cycle so,..

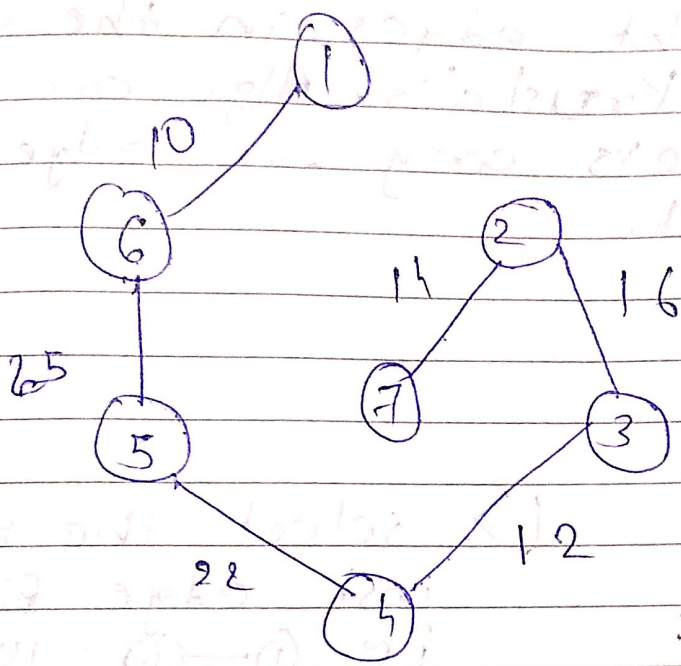


4) Now take 16...



5.) Now the next smallest cost / min cost is 18 but after taking it it'll make cycle ie 4-7-2-3-4 so don't take it

6.) 22 then 25 ... now if we see then edges are 6 in total so we've got our answer.



So, min. cost = 10 + 25 + 22 + 12 + 16 + 14 = 99

Algorithm:-

1) Begin
2) Create the edge list of given graph, with their weights. in ~~ascending order~~.
3) Sort the edge list according to their weights in ascending order.
4) Draw all the nodes to create skeleton for spanning tree.
5) Pick up the edge at the top of the edge list (ie. edge with min. weight).
6) Remove this edge from the edge list.
7) Connect the vertices in the skeleton with given edge. If by connecting the vertices, a cycle is created in the skeleton, then discard this edge.
8) Repeat step from 5) to 7), until n-1 edges are added or list of edges is over
9) return.

Complexity :-

Time complexity = $O(|V||E|)$

- $|V|$ → no. of vertices
- $|E|$ → no. of edge

(we can reduce by using min. Heap till $O(n \log n)$)

// when delete will get
// min itself from tree
// so searching won't be
// needed.