9. return EXTRACT-MIN(Q) // return the root of the tree

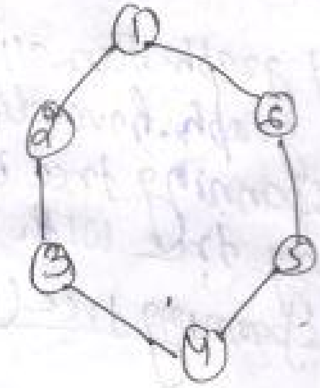
TIME COMPLEXITY:

$$\boxed{O(n \log n)}$$

MINIMUM COST SPANNING TREE

→ let $G=(V,E)$ be the graph where V is the set of vertices, E is the set of edges and $|V|=n$. The spanning tree $G'=(V,E')$ is a sub graph of G in which all the vertices of the graph G are connected with number of edges. The minimum number of edges required to connect all the vertices of graph G is $n-1$. ~~spanning is~~

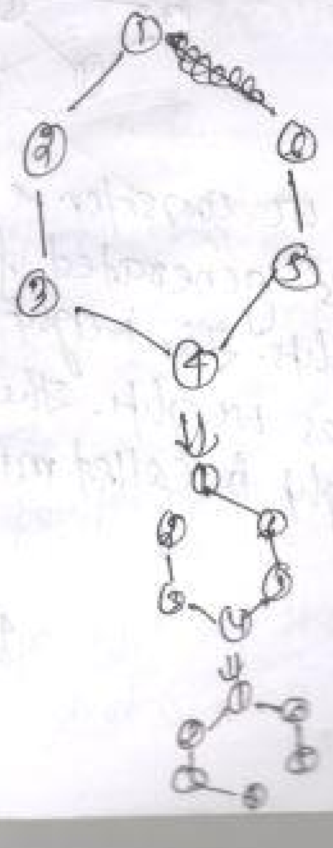
Ex



$G=(V,E)$

$V=\{1,2,3,4,5,6\}$

$E=\{(1,2),(2,3),(3,4)\ldots\}$



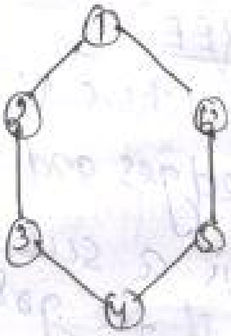
$|E|=|V|-1$

$=6-1=5$

⇒ From a given graph we can have ${}^{|E|}C_{|V|-1}$

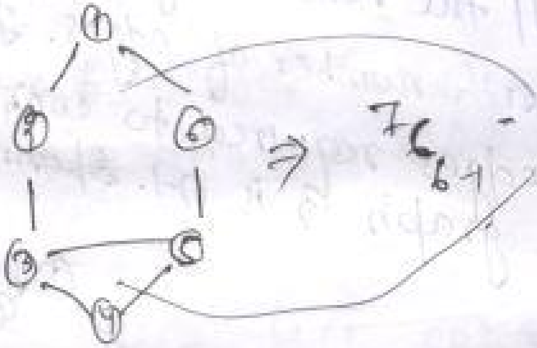
No. of possible spanning tree = ${}^{|E|}C_{|V|-1}$ − no. of cycles with less than

so for.



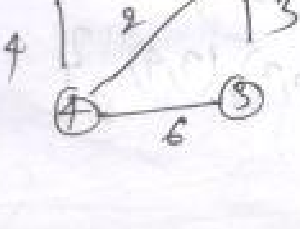
$\Rightarrow {}^{6}C_{6-1} - 0 = {}^{6}C_{5} = 6$

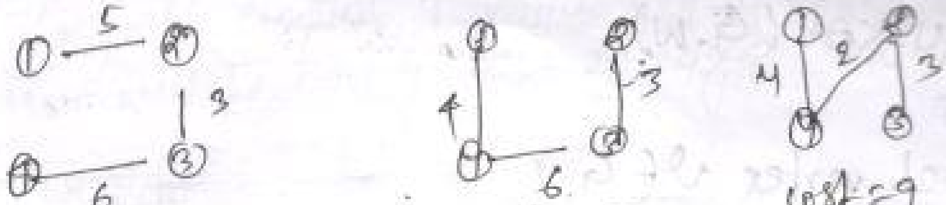
so for



$\Rightarrow {}^{7}C_{6-1} - 2 = {}^{7}C_{5} - 2$

⇒ If we consider a weighted graph then all the spanning trees generated from the graph have different weights. The weight of the spanning tree is sum of its edges weights. The spanning tree with minimum weight is called minimum spanning tree (MST).





cost of the tree = 14 cost = 13 cost = 9

KRUSKAL'S ALGORETHM

↳ This algorithm starts with list of edges sorted in non decreasing order of weights. It repeatedly adds the smallest edge to the spanning tree that does not create cycle. Initially each vertex in its own tree in the forest. Then algorithm considers each edge ordered by increasing weights. of the edge (u,v) connects two different trees, then (u,v) is added to the set of edges of the MST and two trees connected by an edge (u,v) are merged into a single tree. of ~~the~~ an edge (u,v) connects two vertices in the same tree, then edge(u,v) is discarded.

⇒ The operations and disjoint set for Kruskal's algorithm

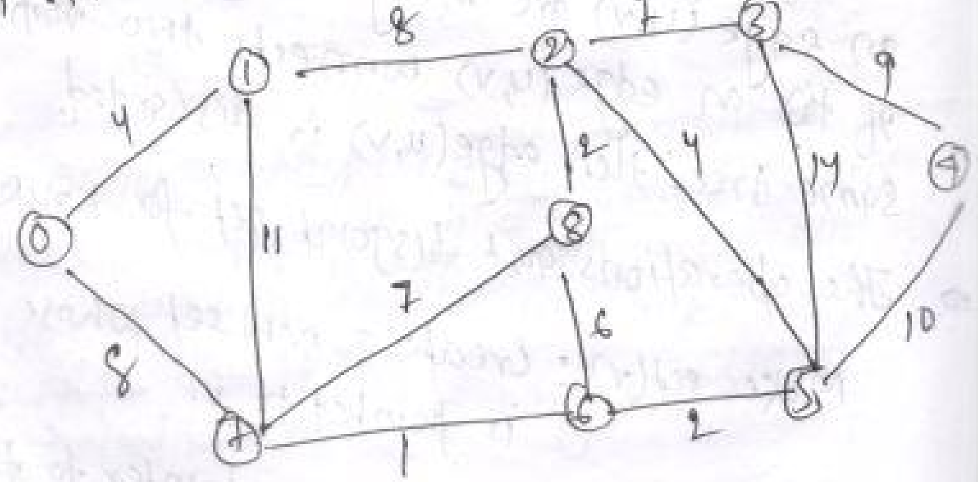
Make-set(v) : Create a new set whose only member is pointed to v.

Find-set (v) : returns a pointer to the set contaning v.

Union(u,v) : unites the dynamic set that contain u and v into a new set that is union of these set

MST-KRUSKAL(G,W)

1. A = ∅
2. for each vertex v ∈ G.V
3. MAKE-SET(v)
4. sort the edges of G.E into non-decreasing order by weight w.
5. for each edge (u,v) ∈ G.E, taken in non-decreasing order by weight
6. if ~~FIND(~~ FIND-SET(u) ≠ FIND-SET(v)
7. A = A ∪ {(u,v)}
8. UNION(u,v)
9. return A.



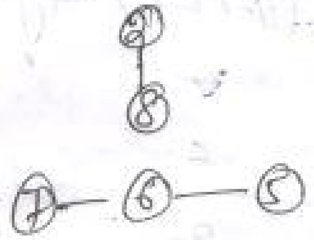
→ The graph contains 9 vertices and 14 edges, so the MST will be having (9-1 = 8) edges

weight	u	v
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

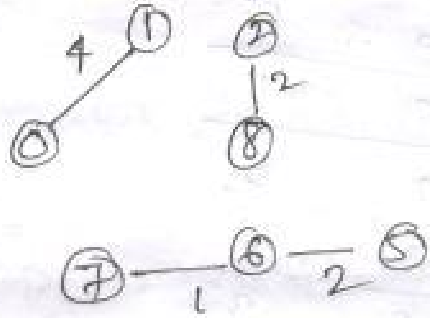
① pick 7-6, no cycle.

② pick 8-2

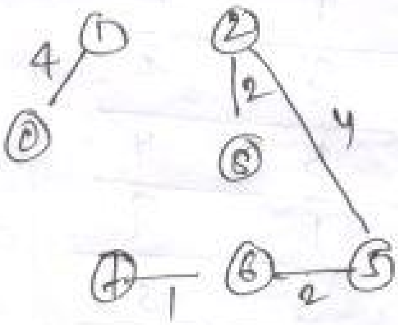
③ pick 6-5, NC



④ pick 0-1.

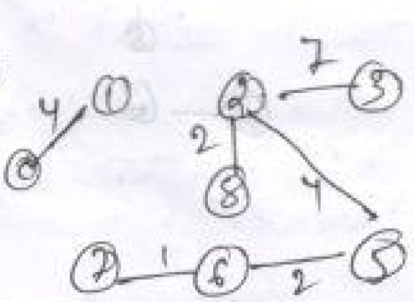


⑤ pick 2-5



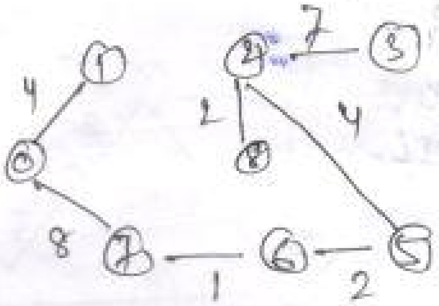
⑥ pick 8-6, including make the cycle hence discard it.

⑦ pick 2-3



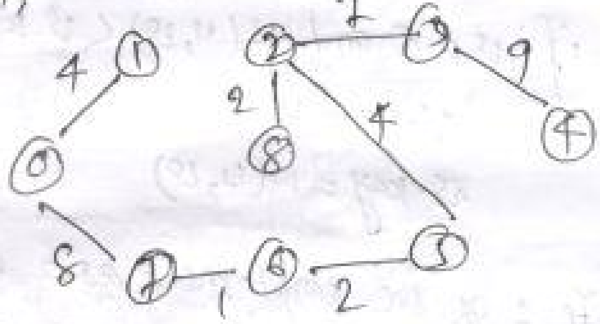
⑧ pick 7-8, forms cycle, discard it.

⑨ pick edge, 0-7 NC



⑩ pick 1-2, cycle, discard

⑪ pick 3-4,



⑫ since no. of edges is (v-1) hence algorithm stops here.

Complexity :→ The running time of Kruskal's algorithm for Graph $G = (V, E)$ depends on the implementation of disjoint set data structure.

$\boxed{O(E \log E)}$