~~Mi~~ Minimum cost spanning Tree.

Graph (Non-linear DS)

$\hookrightarrow G = (V, E)$

- $\hookrightarrow$ set of vertices
- $\hookrightarrow$ set of Edge.

$|V| = n$ (vertices)

$|E| = e$ (Edges)

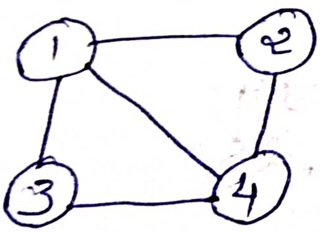
Undirected / Directed

Weighted / unweighted.

Graph Representation

Adjacency Matrix

Adjacency List.



Undirected graph.

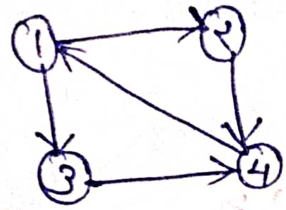
$n = 4$ = Adjancey Matrix.

$n \times n$ matrix

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \left[\begin{array}{cccc} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{array}\right]_{n \times n} \end{array} \quad O(n^2)$$

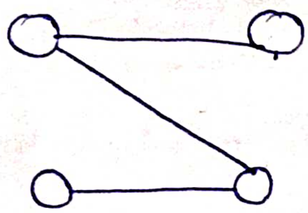
$\overset{5}{2}|E| = 10 \hookrightarrow 1's.$

$\overset{\downarrow}{e}$

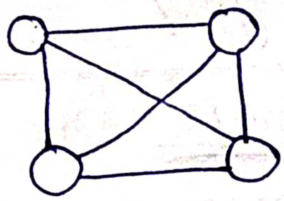


directed edge Graph.

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 1 & 0 & 1 \\ 2 & 0 & 0 & 1 & 0 \\ 3 & 1 & 0 & 0 & 0 \\ 4 & 0 & 0 & 1 & 0 \end{array}$$

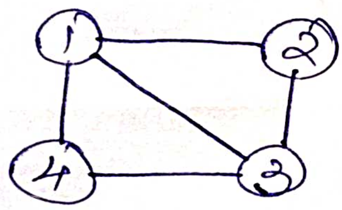


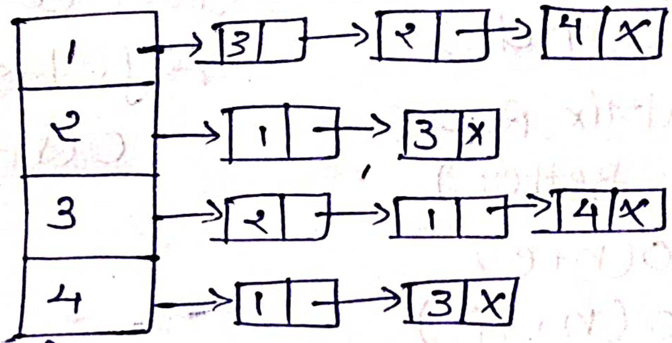
sparse



Dense

(2) Adjacency List :-





A

undirected graph size. (n+2E)

Directed (n+E)

O(n+2)

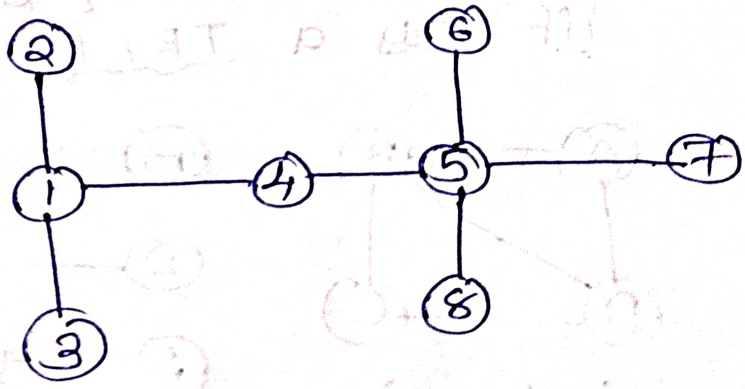
Graph Traversal

BFS

DFS

(1) Visiting Node

(2) Exploring Node.



1 2 3 4 5 6 7 8

1 2 3 4 5

1 dist.

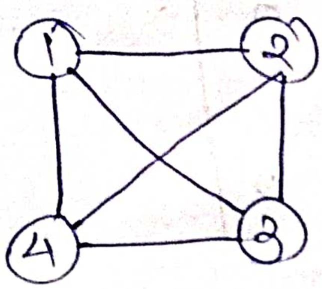
2 dist.

6 7 8

3 dist.

Bfs

" $e \propto n$ " (in worst case)



Undirected graph. $e = \frac{(n-1)n}{2}$

Directed graph $e = n(n-1)$

$O(n^2)$

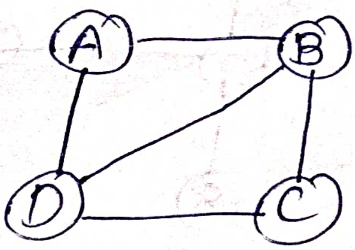
Graph

Dense Graph. <almost complete> (Matrix Repr. is Better) $O(n+e)$ $O(n+n^2)$

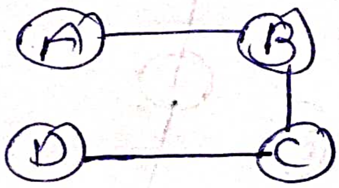
Sparse Graph. <Not Complete> Adj. List. $O(n+E)$

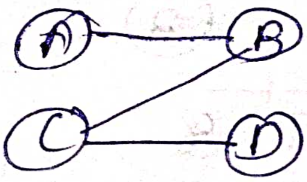
⇒ Defination of spanning Tree :-

⇒ A subgraph $T(V, E')$ of Given graph $G(V, E)$, where $E' \subset E$ is spanning Tree, iff T is a TREE.

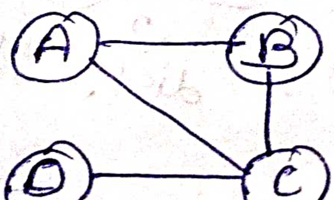


$G = (V, E)$





Spanning Tree
A Tree with n - vertices will have $(n-1)$ edges.



X

Q) Given Graph having n-vertices & e edges then No of edges that must be Remove from the graph to get Spanning Tree is $\underline{e-n+1}$.

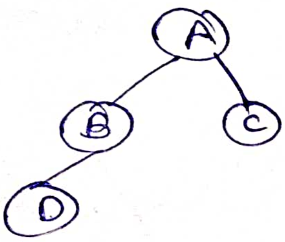
⇒ total edges in graph : $\boxed{e}$

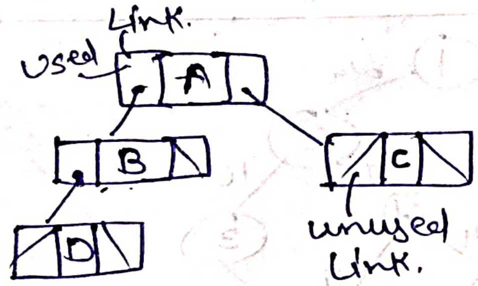
⇒ Spanning Trees with 'n' vertices will have $\boxed{(n-1) \text{ edge } \langle \text{used} \rangle}$

Removed edge : $e-(n-1)$

$\boxed{e-n+1}$

Q) A Tree with n-vertices will have $\underline{(n+1)}$ nil fields.





total links : $\boxed{2n}$

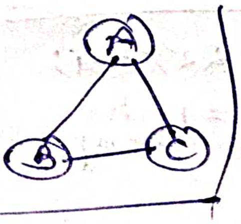
Edges : $\boxed{(n-1)}$

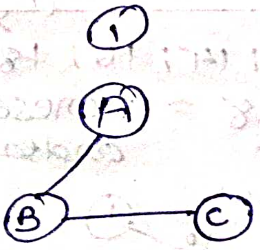
Unused Link : $2n-(n-1)$

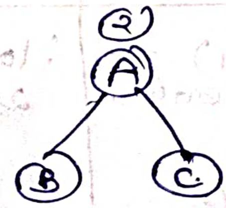
$= \boxed{n+1}$

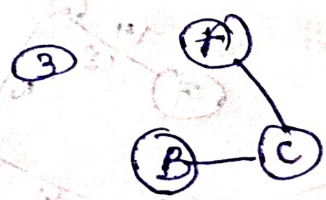
⇒ Solution Space for a given graph with n-vertices & 'e' edges the max. # of sp. Trees.

$= \dfrac{(n^{n-2})}{\text{for complete Graph } K_n}$.









$= 3^{3-2} = \boxed{3}$

$\boxed{n^{n-2}}$ Possible max. SP. Trees.

Applications of Spanning Tree :-

i) Multicasting & Broadcasting in both Wired & wireless Networks.

ii) Construct / implement circuits (Electronic/ ~~~~ in communication Network.

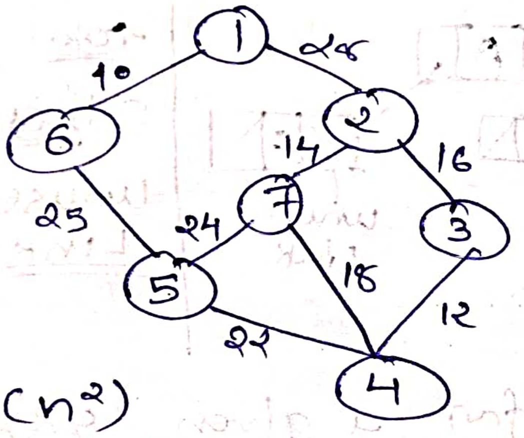
*) Algorithms for Constructing of min. cost spanning Trees.

① Prims Jarnik Algorithm

② Kruskal's Algorithm.

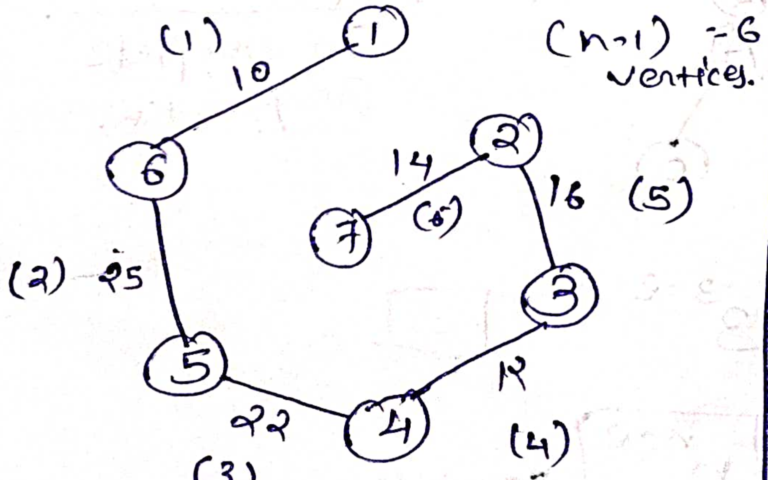
③ Dijkstra's Algorithms.

Question graph

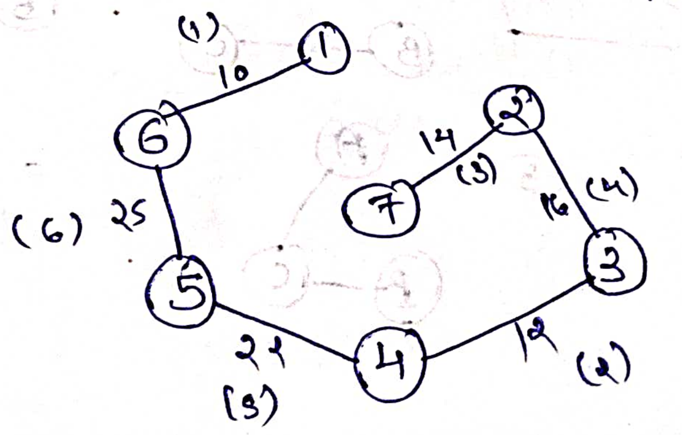


Time $O(n^2)$ | $O(e\log e)$

Prims Algorithm	Kruskal Algorithm
	~~10~~; ~~12~~; 14; ~~16~~; ~~18~~; 22; ~~24~~; ~~25~~; ~~28~~ = Increse order = min heap.



No lead cycle + min. cost.



*) Prim's method Always maintain Tree structure Properties at each step.

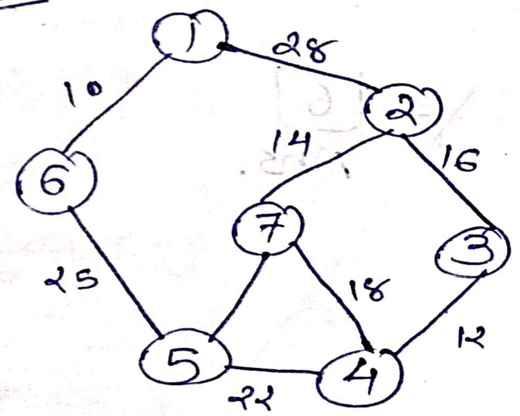
*) Cost of spanning trees with both Approa
will always same.

*) Final Tree structure may/may NOT sa

Tree	Cost
may be diff.	Always Same.

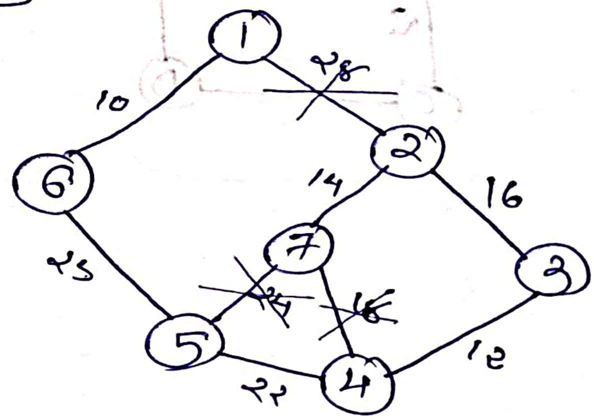
*) Tree structure will also be same iff all edge have unique cost. (logical point)

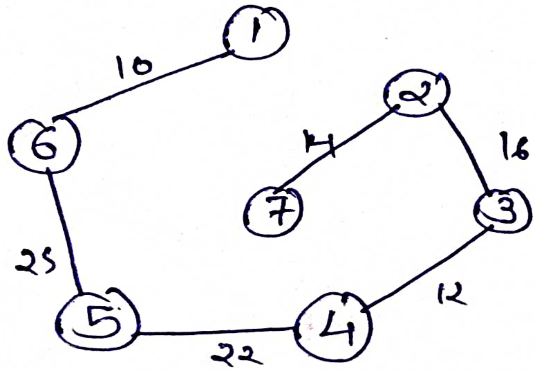
Ex



$\boxed{O(n \cdot e)}$

Dijkstra's Algorithm.



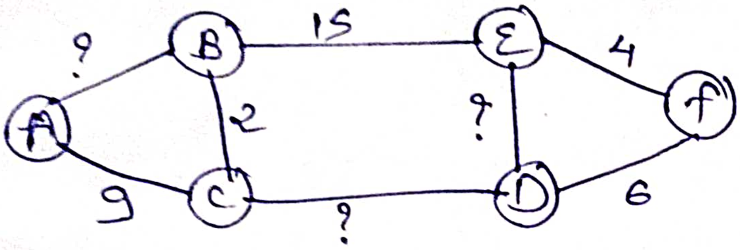


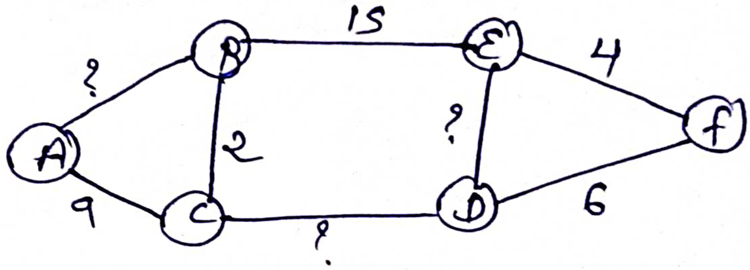
i) connect start from anywhere edges.

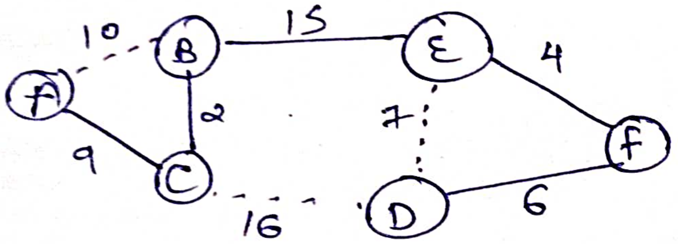
ii) Avoid cycle

iii) when cycle create Remove heighest element.

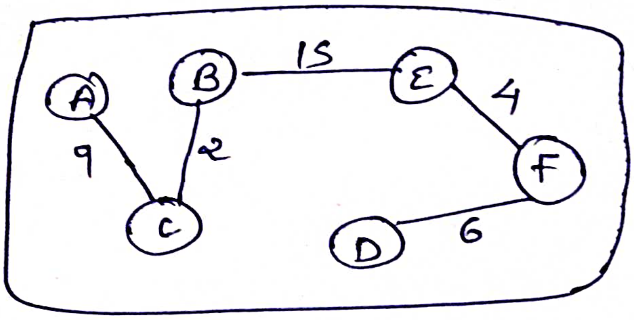
Q) Consider following Graph whose min. Cost spanning Tree marked with edge Values has weight of 36. minimum Possible sum of all edge of Graph G is.







Missing because it will create cycle.



spanning Tree form.

$= 10 + 15 + 4 + 6 + 7 + 16 + 9$

$= \boxed{69}$ Ans.