MODULE-3 & 4

GRAPHS and TREES

Discrete mathematics

For Computer Science

Graph Theory (Introduction)

→ introduced by Euler.

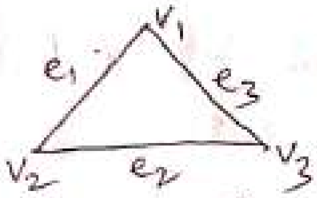
→ A graph G in a plane or in 3-dimensional space consists of a set of points (V) and set of line segments (E)

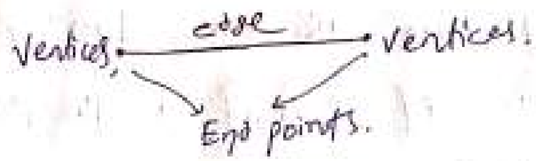
So, $G = (V, E)$

where $V = \{v_1, v_2 \dots v_n\}$, v_i are called vertices.
$E = \{e_1, e_2 \dots e_n\}$, e_i are called edges.

We can also write the graph $(V(G), E(G))$

→ Each edge has one or two vertices associated with it called its End points.



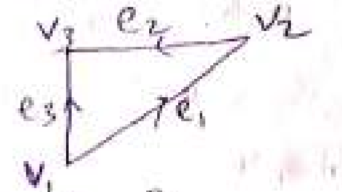


$V = \{v_1, v_2, v_3\}$
$E = \{e_1, e_2, e_3\}$; $G = (V, E)$

Basic Terminology -

1) Directed graph/oriented graph

There is direction in the edge.

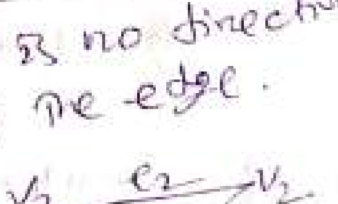


$e_1 = (v_1, v_2)$
$e_2 = (v_2, v_3)$
$e_3 = (v_1, v_3)$

order pair of edges

Undirected/unoriented graph

There is no direction in the edge.



$e_1 \to (v_1, v_2)$
$e_1 \to (v_2, v_1)$

unordered pair.

Scanned with CamScanner

(1) ~~Simple~~ graph:

① **Isolated vertex:-**
A vertex without any edge connecting it.

② **Trivial graph:-** It is a graph consisting of only a single isolated vertex is called Trivial graph.

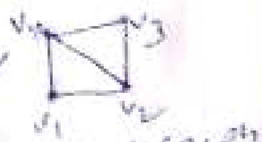
③ **Null graph:-** A Null graph is a graph whose vertex set and edge set are empty.
Ex- A trivial graph is a Null graph.

④ **Loop:-** A Loop is an edge connecting a vertex to itself. It is not a proper edge.

⑤ **Proper edge:-** it is an edge that is not a loop is called a proper edge.
· It has two vertex and one edge.

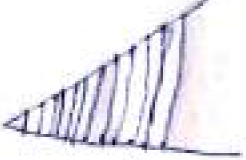
⑥ **Multiedge (Parallel edge):-**
The collection of two or more edge having identical end points.

⑦ **Finite graph:-**
A graph having both vertex set and edge set is finite. is called finite graph. otherwise infinite graph.



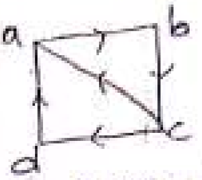
Ex- $\begin{cases} V(G) = Z \\ E(G) = e_{ij} : |i-j| = 1 \end{cases}$

Infinite graph

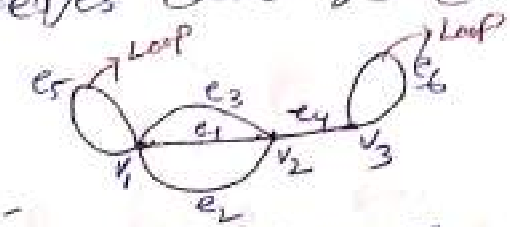


Scanned with CamScanner

(8) Simple graph:- A graph That has no self loops or Parallel edges.



(9) Pseudographs:- These are The graphs That include Loops and ~~Possibility~~ Possibly multiple edges Connecting some pair of vertices.

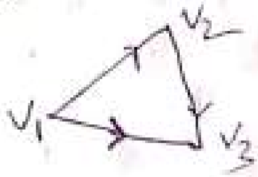


(10) Order and Size of The graph:-

- Number of vertices is called order of graph.
  i.e. $|V(G)|$ = order of graph.
- Number of edges is called ~~order~~ size of graph.
  $|E(G)|$ = Size of graph.

(11) Adjacent Vertices:- two vertices have a common edges Then it is called adjacent vertices.

(12) Adjacent edges:- If two edges have a common vertex, Then it is called adjacent edges.



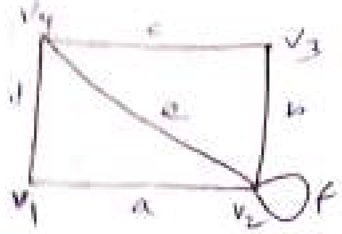
(13) The degree of a Vertex:-

It is The number of edges including self loops incident at vertex v.

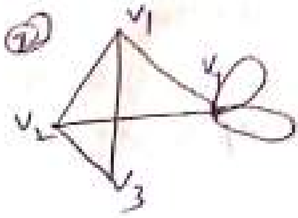
$$\boxed{deg(v) = n_e + 2n_L}$$

where
$n_e \rightarrow$ no. of edges incident at vertex v.
$n_L \rightarrow$ no. of self Loops incident at vertex v.

Example ①



②

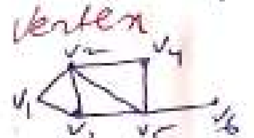


$d(v_1)=3$
$d(v_2)=3$
$d(v_3)=2$
$d(v_4)=2+2(2)$
$=6$

$d(v_1)=(a,d)=2$ $(\because n_e=2, n_L=0)$
$d(v_2)=3+2(1)=5$ $(\because d(v)=n_e+n_L)$
$d(v_3)=2$ n_e = no. of edges incident at v.
$d(v_4)=3$ n_L = no. of ~~edges~~ Loops incident at v.

Note ① Vertex of degree zero → Isolated Vertex →

② Vertex of degree one → Pendant Vertex



v_6 → Pendant vertex.

Defn

e: directed edge.
u is said to adjacent to v.
and v is said to adjacent from u.

So, u : initial vertex } of (u,v).
v : Terminal vertex }

The initial and Terminal vertex of a loop are same.

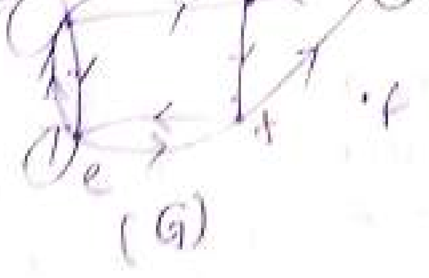
A graph with directed edges -

Defn In-degree of a vertex v, denoted by $deg^-(v)$ is The number of edges with v as their terminal vertex.

out-degree of v denoted by $deg^+(v)$ is The no. of edges with as their initial vertex.

Note a loop of a vertex contributes 1 to both The in-degree and out-degree of this vertex.

Ex



(G)

In degree	⟶ out degree
$deg^-(a)=2$ (Loop=1, incident=1)	$deg^+(a)=4$ (Loop=1 +3)
$deg^-(b)=2$	$deg^+(b)=1$
$deg^-(c)=3$	$deg^+(c)=2$
$deg^-(d)=2$	$deg^+(d)=2$
$deg^-(e)=3$ (Loop=1, incident =2)	$deg^+(e)=3$ (Loop=1 +2)
$deg^-(f)=0$	$deg^+(f)=0$

Scanned with CamScanner

Thm(1) The Handshaking Thm

Let $G=(V,E)$ be an undirected graph with 'e' edges then $\boxed{2e = \sum_{u \in V} deg(u)}$

i.e. sum of the degree of the vertices of an undirected graph is even.

Ex- How many edges are there in a graph with 10 vertices each of degree 6.

Soln. Sum of degree of the vertices $= 6 \cdot 10 = 60$

$\therefore$ we have sum of degree of vertices $= 2e$

$\Rightarrow 60 = 2e$

$\Rightarrow \boxed{e = 30}$

Thm(2) An undirected graph has an even number of vertices of odd degree.

(or) The number of vertices of odd degree in a graph is always even.

pf- Let $G=(V,E)$ be any graph with 'n' number of vertices and 'e' number of edges.

Let $v_1 v_2 \ldots v_k$ be the odd degree vertices and $v_1', v_2' \ldots \ldots v_m'$ be the even degree vertices.

By Handshaking Thm

$$\sum_{i=1}^{n} deg(v_i) = 2e$$

$$\Rightarrow \sum_{i=1}^{k} deg(v_i) + \sum_{j=1}^{m} deg(v_j') = 2e$$

Scanned with CamScanner

$$\sum_{i=1}^{k} deg(v_i) + \text{even number} = \text{even number}$$

$$\Rightarrow \sum_{i=1}^{k} deg(v_i) = \text{even number} - \text{even number}$$

$$\Rightarrow \sum_{i=1}^{k} deg(v_i) = \text{even number}$$

$\Rightarrow$ ~~Sum of odd degree vertices = even no.~~

$$\Rightarrow \boxed{k = \text{even number}}$$

$\therefore$ The no. of vertices of odd degree is always even.

Hence Proved.

Thm (3) Let $G=(V,E)$ be a graph with directed edges. Then

$$\boxed{\sum_{u \in V} deg^-(u) = \sum_{u \in V} deg^+(u) = |E|}$$

or,

$$\boxed{\sum_{i=1}^{n} deg^-(v_i) = \sum_{i=1}^{n} deg^+(v_i) = |E|}$$

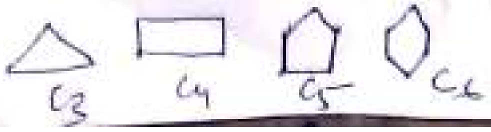
Complete graph -

A simple graph in which edge between every pair of vertices.

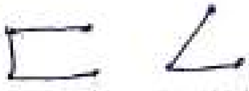
- It is denoted by K_n

Cycle graph

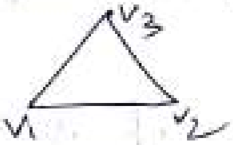
A cyclic graph of order n is a connected graph whose edge from a cycle of length n, and denoted by C_n.



Acyclic graph:- A graph which doesnot have any cycles is called acyclic graph.

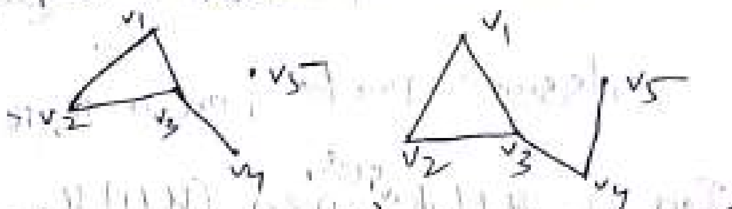


Regular graph:- Degree of each vertex is same.



$d(v_1)=2$
$d(v_2)=2$
$d(v_3)=2$

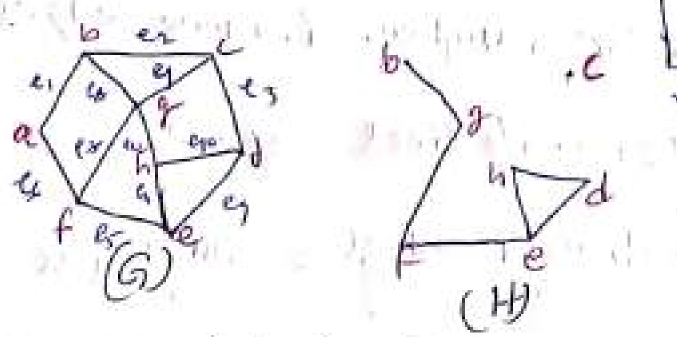
Connected graph:- A connected graph means at least one path between every pair of vertices.



(Not connected graph) (Connected graph)

Mixed graph- If some edges are directed in a graph and some are undirected is a mixed graph.

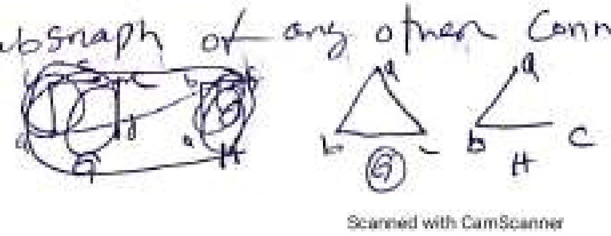
Subgraph:- If G is a graph, then H is a subgraph of G. $\boxed{\text{If } V(H) \subseteq V(G) \text{ and } E(H) \subseteq E(G)}$



$V(G) = \{a, b, c, d, e, f, g, h\}$
$V(H) = \{b, c, g, f, h, e, d\}$
$E(H) = \{e_8, e_2, e_5, e_7, e_{10}, e_{11}\}$

~~Connected graph~~ **Component of a graph:-**

Let $G=(V,E)$ be a graph. A subgraph H of G is called a component of G. If H is connected and it is not a subgraph of any other connected subgraph. Ex



Theo(4) A simple graph with n vertices have maximum $\frac{n(n-1)}{2}$ edges.

Pf:- we proof by using mathematical Induction.

Suppose $n=1$, ~~vertices~~ vertices,

Then, $\frac{1(1-1)}{2}=0$ edges.

It is true

Suppose $n=k$ (where k is +ve integer) vertices.

Then $\frac{k(k-1)}{2}$ edges.

Assume that for $n=k$, it is true,

For $n=k+1$ vertices, then $\frac{(k+1)k}{2}$ edges.

we have to show that it is true for $n=k+1$ vertices.

For the Graph G with k vertices, we have $\frac{k(k-1)}{2}$ edges

The last vertex is the $(k+1)$th vertex will be connected to maximum of 'k' no. of vertices giving rise to 'k' number of more edges.

~~So, that Total, no. vertices~~ $\cancel{= \frac{k(k-1)}{2}}$

So. total number of edges $= \frac{k(k-1)}{2}+k$

$= k\left(\frac{k-1}{2}+1\right)$

∴ For $n=k+1$ vertices, have maximum $\frac{k(k+1)}{2}$ edges. $= k\left(\frac{k-1+2}{2}\right)$

$= k\frac{(k+1)}{2}$

∴ Hence the Theo is true $\forall n$. (n is +ve integer)

Thm (5) A simple graph with n-vertices and k-components can't have more than $\frac{(n-k)(n-k+1)}{2}$ edges.

Pf Let the number of vertices in each of the k components of a graph G are $n_1, n_2, \dots n_k$.

Total no. of vertex in $G = n$

$\Rightarrow n_1 + n_2 + \dots + n_k = n \quad \forall n_i \geq 1$

The proof of the thm depends on an algebric inequality.

$$\boxed{\sum_{i=1}^{k} n_i^2 \leq n^2 - (k-1)(2n-k)} \text{ --- } (1)$$

The maximum no. of edges in a graph G is

$$\sum_{i=1}^{k} \frac{n_i(n_i-1)}{2}$$

($\because$ The maximum no. of edges in the ith component of graph G is $\frac{n_i(n_i-1)}{2}$)

$$= \frac{1}{2} \sum_{i=1}^{k} n_i(n_i-1)$$

$$= \frac{1}{2} \sum_{i=1}^{k} (n_i^2 - n_i)$$

$$= \frac{1}{2}\left(\sum_{i=1}^{k} n_i^2 - \sum_{i=1}^{k} n_i \right)$$

$$= \frac{1}{2}\left(\sum_{i=1}^{k} n_i^2 \right) - \frac{n}{2} \quad \left(\because \sum_{i=1}^{n} n_i = n_1 + n_2 + \dots + n_k = n \right)$$

$$\leq \frac{1}{2}\left(n^2 - (k-1)(2n-k) \right) - \frac{n}{2} \quad \left(\text{from eqn (1)} \right)$$

$$\leq \frac{1}{2}\left[n^2 - 2kn + k^2 + 2n - k \right] - \frac{n}{2}$$

$$\leq \frac{1}{2}\left[n^2 - 2nk + k^2 + 2n - k - n\right]$$

$$\leq \frac{1}{2}\left[(n-k)^2 + (n-k)\right]$$

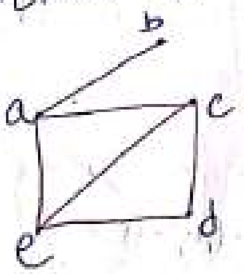
$$\leq \frac{1}{2}(n-k)(n-k+1)$$

Hence proved.

Graph Representation:-

① Adjacency Lists:- It is a way to represent a graph without multiple edges. It specifies all the vertices that are adjacent to each vertex of a graph.

Ex-

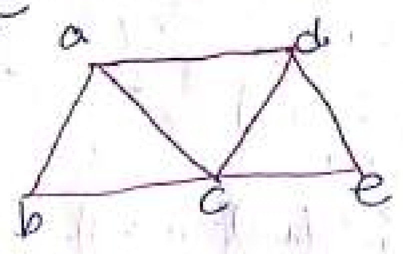


vertex	adjacent vertices
a	b, c, e
b	a
c	a, d, e
d	c, e
e	a, c, d

② Adjacency Matrices:- $A(G)$ is a $n \times n$ zero-one matrix with 1 as if (i,j)th entry when i and j are adjacent and 0 otherwise.

$$A = [a_{ij}] = \begin{cases} 1, & \text{If } (i,j) \text{ is edge of } G \\ 0, & \text{otherwise} \end{cases}$$

Ex-



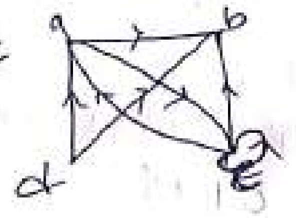
$$\begin{array}{c c} & \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \\ e \end{array} & \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Diagraphs and Relations

Diagraph:- A diagraph is a pictorial representation of a relation.

• Diagraph is also called Directed Graphs

Ex-



what is relation is represented by the diagraph.

$R = \{(a,b)\ (a,c)\ (c,a)\ (c,b)\ (c,c)\ (d,a), (d,b)\}$

Matrix

$$\begin{array}{c|cccc} & a & b & c & d \\ \hline a & 0 & 1 & 1 & 0 \\ b & 0 & 0 & 0 & 0 \\ c & 1 & 1 & 1 & 0 \\ d & 1 & 1 & 0 & 0 \end{array}$$

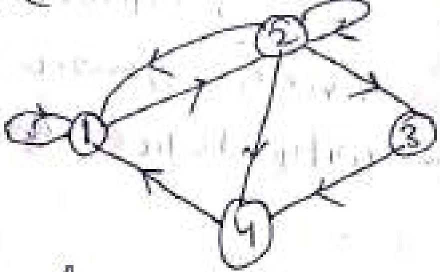
Ex- $A = \{1, 2, 3, 4\}$

R is a relation from A to A

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

Find Diagraph of R and matrix M(R)?

Diagraph



and Matrix

$$M(R) = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 0 & 0 & 0 & 1 \\ 4 & 1 & 0 & 0 & 0 \end{array}$$

(1) Diagraph Representing a Reflexive, Symmetric, & Transitive relation & Equivalence Relation.

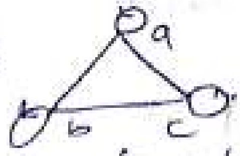
Reflexive Diagraph - A diagraph of a reflexive relation will have a self loop at each ~~vertices~~ of its vertices.

If diagraph does not contain a self loop, Then it is called Irreflexive Diagraph.

~~Sym~~ Symmetric Diagraph - A diagraph representing a Symmetric relation.

Transitive - A diagraph representing Transitive Relation

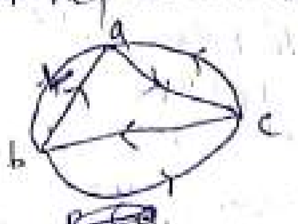
Ex-



Relation $R = \{(a,a)\ (b,b)\ (c,c)\}$

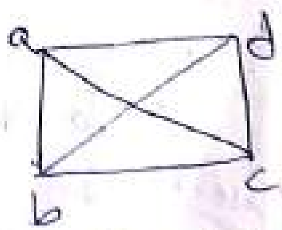
Reflexive diagraph

or, Graph representing Reflexive relation



Symmetric relation

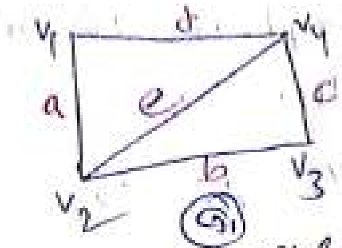
$R = \{(a,c)\ (c,b), (c,a), (b,a)\ (b,c)\ (a,b)\}$



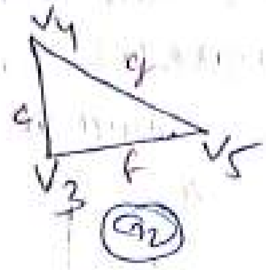
Transitive relation

Operations on Graphs :-

(1)



(G_1)



(G_2)

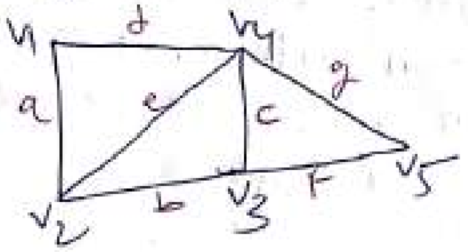
(1). Union of two graphs.

If $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ are two graphs.

Then $G_3 = G_1 \cup G_2$ where $V_3 = V_1 \cup V_2$

& $E_3 = E_1 \cup E_2$

(it contains edges either in G_1 or G_2 or in both)

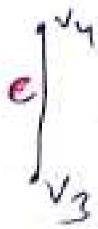


$(G_1 \cup G_2)$

(2) Intersection :- If $G_1=(V_1,E_1)$ and $G_2=(V_2,E_2)$

Then $G_3 = G_1 \cap G_2$ where $V_3 = V_1 \cap V_2$ & $E_3 = E_1 \cap E_2$

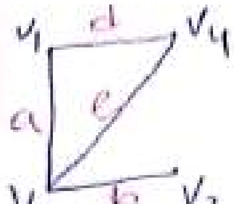
(it contains all the edges which are in both G_1 and G_2)



$(G_1 \cap G_2)$

(3) Difference of two graph

$G = G_1 - G_2$ (contains all the edge in G_1 but not in G_2)



$(G_1 - G_2)$

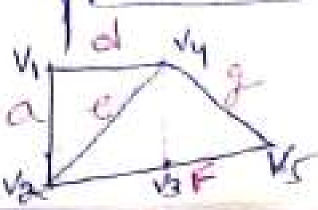
(4) Addition of two graph

Ring Sum $\oplus$

$G = G_1 \oplus G_2$

includes all the edges either in G_1 or G_2 but not in both.

$\boxed{G_1 \oplus G_2 = (G_1 \cup G_2) - (G_1 \cap G_2)}$

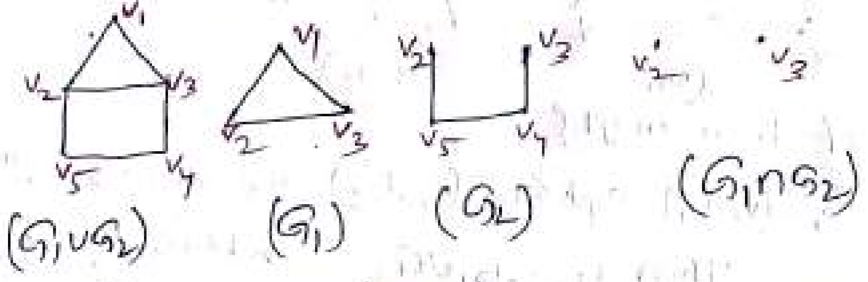


$(G_1 \oplus G_2)$

(5) Decomposition

A graph G is said to having decomposed into two subgraphs G_1 and G_2 If. $G_1 \cup G_2 = G$ but $G_1 \cap G_2$ = Null graph.

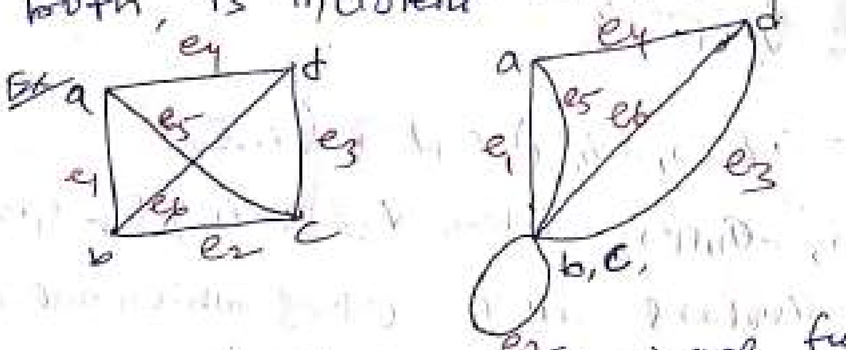
Ex:



(6) Fusion:- (Fusion of vertex)

A pair of vertices a & b in a graph are said to be fused If The two vertices are replaced by a single new vertex such That every edge That was incident on either a or b or in both, is incident on The new vertex.

Ex

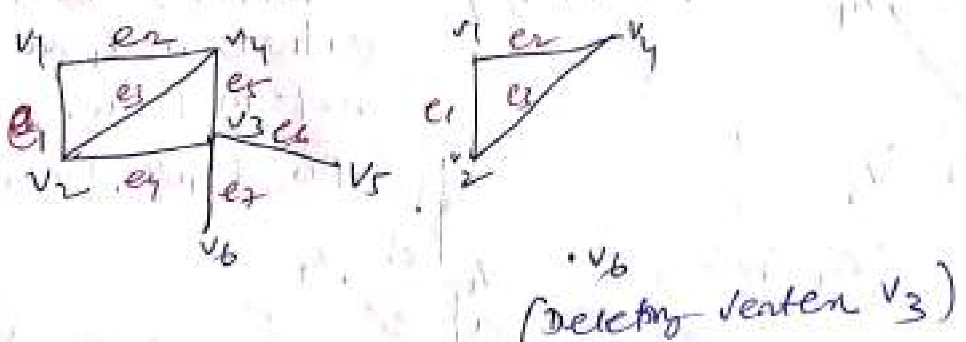


Here (b,c) are fusion.

(7) Deletion (Deletion of a vertex):-

If V_i is a vertex in G, Then $G - V_i$ is a Subgraph of G obtained by removing The vertex V_i and The edges incident on it.

(Deletion of vertex means The deletion of all edge incident on That vertex).



Path and Circuit

Walk on a Graph:-

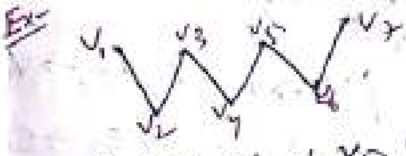
A walk is nothing but alternative sequence of vertices and edges.

is $v_1, e_{12}, v_2, e_{23}, v_3 - - - v_n$

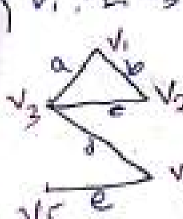
Open walk / path

Initial and End vertex are different.

Ex:-



$\{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$



edges (a, b, c, d, e)

v_3 appears twice

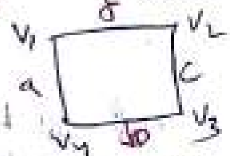
(so, not a path)

(b, c, d, e) (Path)
no, repeated

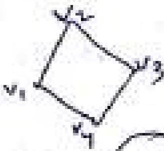
Closed walk / Circuit

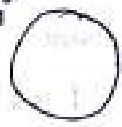
Initial and End vertex are same.

Ex:-



$(v_1 \to v_2 \to v_3 \to v_4 \to v_1)$





→ also called circular path or cycle

cycle:- (closed path)
(circular path)

Euler Graphs :-

Euler Path :- It is a path that traverse each edge exactly once and only once.
or (Eulerian path)

A Graph That contains an Euler path is called as Euler Graph!

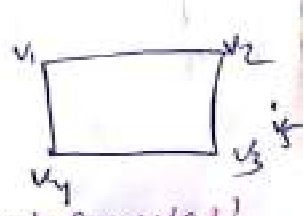
(Vertex can be repeated but edge can't)

Ex:-

$e_1 \to e_2 \to e_3 \to e_4$

Each edge is visited only once.

- Euler Graph is always connected, because Euler path contains all the edges of the Graph.

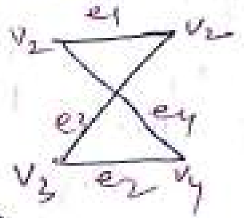


(not connected)

(Connected)

Euler Circuit: First and Last vertex are same it (or) is a circuit that traverses each edge exactly once and only once.

(Eulerian circuit)



1st and Last vertex same ($v_1 = v_1$) each edge is visited only once.

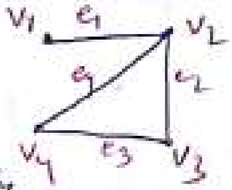
$\begin{pmatrix} v_1 \to v_2 \to v_3 \to v_4 \to v_1 \\ e_1 \to e_2 \to e_3 \to e_4 \end{pmatrix}$

Note ① A connected Graph is Euler Graph iff it has atmost 2 odd degree vertices.

Euler circuit → Each ~~vertex~~ vertex is of even degree.

Euler path → maximum two vertices odd degree.

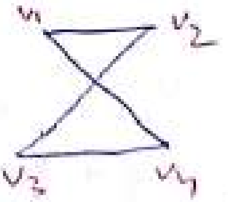
Ex-



$e_1 \to e_2 \to e_3 \to e_4$
$(v_1, v_2, v_3, v_4, v_2)$

$deg(v_1) = 1$ (odd)
$deg(v_2) = 3$
$deg(v_3) = 2$
$deg(v_4) = 2$

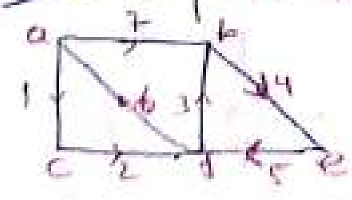
(Euler path)



$deg(v_1) = 2$ (even)
$deg(v_2) = 2$
$deg(v_3) = 2$
$deg(v_4) = 2$

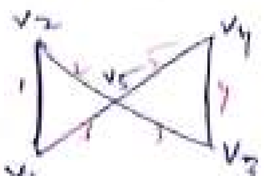
(Euler circuit)

Ex- Find Euler path for Graph below.



$a \to c \to d \to b \to e \to d \to a \to b$

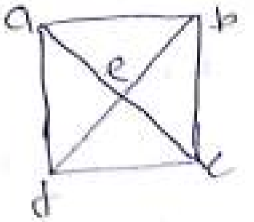
degree
$deg(a) = 3$
$b = 3$ } 2 vertex odd degree
$c = 2$
$d = 4$
$e = 2$ so, it is a Euler path.



initial $= v_1$ & final $= v_1$

$v_1 \to v_2 \to v_5 \to v_3 \to v_4 \to v_5 \to v_1$

$deg(v_1) = 2$ $\quad deg(v_5) = 4$
$v_2 = 2$
$v_3 = 2$ (even) ∴ it is a Euler circuit
$v_4 = 2$



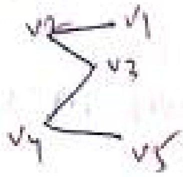
deg(a) = 3, deg(e) = 4
deg(b) = 3
deg(c) = 3
deg(d) = 3 ∴

(4 vertices with odd degree)
does not have any Euler path.

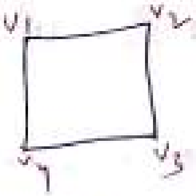
Hamilton path and circuit or (Hamiltonian Path and circuit)

Hamilton path :- Contains each vertex exactly once.

Hamilton circuit :- First and Last vertex are same.



H. path - $v_1 \to v_2 \to v_3 \to v_4 \to v_5$



H. circuit - $v_1 \to v_2 \to v_3 \to v_4 \to v_1$

Note- ① (On removing any one edge from a Hamilton ~~path~~ circuit, then we are left with Hamilton path)

Ex-

H. Circuit (H. Path)

(removing one edge 'e_1')

so, we get $v_1 \to v_4 \to v_3 \to v_2$ (H. Path)

② Length of Ham. path is a Connected Graph of 'n' vertices is $(n-1)$ edges. (not necessary, not sufficient)

Theorem Let G be a graph of 'n' vertices then 'G' has a Hamilton path if ~~and~~ for only two vertices of G (let two vertices u & v)

Then $\boxed{deg(u) + deg(v) \geq n}$

Ex-

no. of vertices (n) = 3

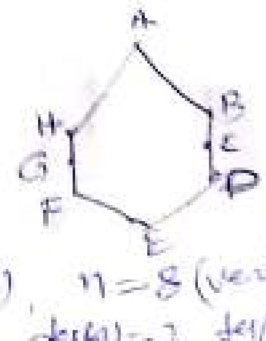
Consider two vertex A & B.

deg(A) = 2, deg(B) = 2

deg(A) + deg(B) = 4 ≥ n

= 4 ≥ 3 (True)

∴ (H. Path)



n = 8 (vertices)

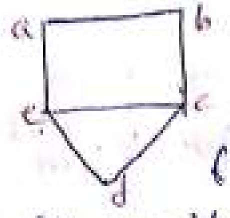
deg(A) = 2, deg(D) = 2

deg(A) + deg(D) = 4 ≥ n

= 4 ≥ 8

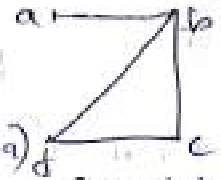
(So not H. path) (False)

Ex- Which of the following Graphs have a H. circuit. If not a H. path

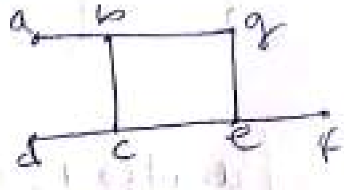


(a→b→c→d→e→a)

(H. circuit)



(a→b→d→c or a→b→c→d) H. path



a→b→c

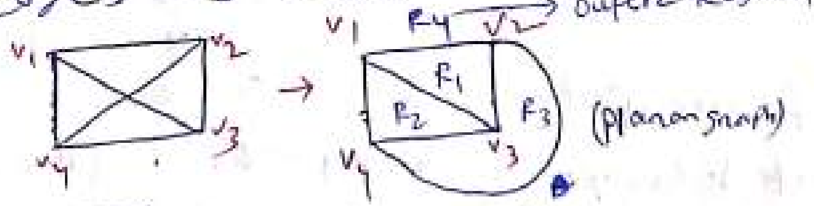
(not contain either path or circuit)

PLANAR GRAPH:-

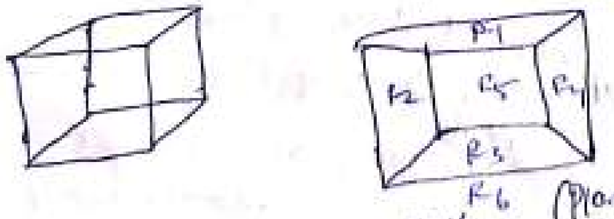
A Graph is said to be the planar if it can be drawn in the plane without any edge crossing.

and a Graph that cannot be drawn on a plane without a cross over between its edges is called Non-planar Graph.

Ex- (Complete graph K_4)



(planar graph) total - 4 regions. (R_1 R_2 R_3 R_4)

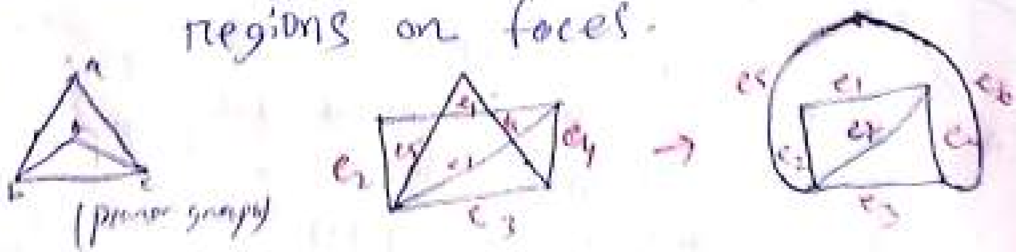


(planar graph) (outer region)

total - 6 regions are panel (R_1 R_2 R_3 R_4 R_5 R_6)

Regions:- A plane graph divides the plane in to regions or faces.

Example



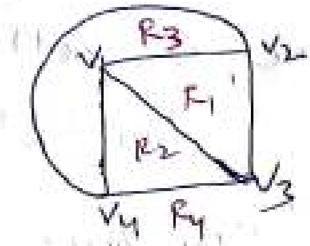
(planar graph)

Euler's Formula

Let G be a connected planar simple graph with 'e' edges and vertices 'v'. Let 'r' be the no. of regions in planar representation of G, Then

$$\boxed{r = e - v + 2}$$

Ex:-



$e = 6, v = 4.$

$r = e - v + 2$

$= 6 - 4 + 2 = 4$ (R_1, R_2, R_3, R_4)

Proof:-

Prove the result by mathematical Induction method.

(i) Basis step P(1) is true.

(ii) Inductive step (assume that above result is true for 'k')

(iii) Verify by proving the result for k+1.

(i) Basis step For $n=1$,

$r_1 = e_1 - v_1 + 2$

$= 1 - 2 + 2$

$\boxed{r_1 = 1}$ is true.

$r = r_1 = 1$

$v = v_1 = 2$

$e = e_1 = 1$

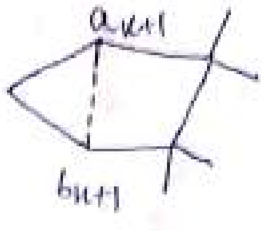
(ii) Inductive step Assume that the eqn is true for $n = k$.

i.e. $r_k = e_k - v_k + 2$ is true for G_k (graph)

(iii) Verification for $n = k+1$,

Let (a_{k+1}, b_{k+1}) be the edge that is added to Graph G_k.

Case-I



both the vertices a_{k+1}, b_{k+1} are in G_k.

So, $r_{k+1} = r_k + 1$

$e_{k+1} = e_k + 1$

$v_{k+1} = v_k$

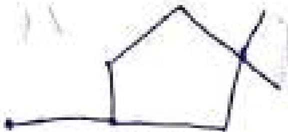
(both vertices are in G_k)

Now $r_{k+1} = e_{k+1} - v_{k+1} + 2$

$\Rightarrow r_k + 1 = e_k + 1 - v_k + 2$

$\Rightarrow \boxed{r_k = e_k - v_k + 2}$ which is true.

Case II



b_{k+1} a_{k+1}

(new edge)

$r_{k+1} = r_k$, $v_{k+1} = v_k + 1$

$e_{k+1} = e_k + 1$

Now

$r_{k+1} = e_{k+1} - v_{k+1} + 2$

$\Rightarrow r_k = e_k + 1 - (v_k + 1) + 2$

$\Rightarrow \boxed{r_k = e_k - v_k + 2}$ (True)

$\therefore$ By Induction method Euler formula is proved.

Problem (1) If there are 20 vertices each of degree 3. Then how many regions does a of this planar graph splits the plane.

Soln
$V = 20$, we have sum of degree of vertices $= 2e$

i.e. $\sum \deg(v) = 2e$

$\Rightarrow 20 \times 3 = 2e$

Now we have $\Rightarrow 60 = 2e \Rightarrow e = 30$

$r = e - v + 2$

$= 30 - 20 + 2 = 10 + 2 = 12$

$\boxed{r = 12}$

Corollary:
If G is a simple Graph with atleast 3 vertices, then

(1) $e \le 3v - 6$

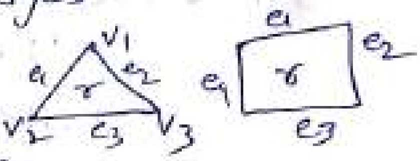
(2) If also G is triangle free, Then $\cancel{e \le 2v}$ $e \le 2v - 4$

Scanned with CamScanner

(3) There is a Vertex v of G such that $\deg(v) \le 5$.

Pf ① Since every region is bounded by minimum ~~[illegible]~~ 3 no. of edges.

i.e $r \geq 3$



So, $\sum_{v \in r(G)} \deg(v) \geq 3r$ —①

But we know that

$$\sum_{v \in r(G)} \deg(v) = 2e$$

So, eqn ① becomes.

$$2e \geq 3r$$

$$\Rightarrow \boxed{r \le \frac{2}{3}e}$$

By Euler's Formula $v - e + r = 2$

$$\Rightarrow r = e - v + 2$$

$$\Rightarrow \frac{2}{3}e \geq e - v + 2$$

$$\Rightarrow \frac{2}{3}e - e \geq -v + 2$$

$$\Rightarrow -\frac{e}{3} \geq -v + 2$$

$$\Rightarrow \frac{e}{3} \le v - 2$$

i.e $\boxed{e \le 3v - 6}$ (Proved)

Pf ②

Since G is a triangle free,

Thus, degree of each region ≥ 4.

i.e $\boxed{r \geq 4}$

Scanned with CamScanner

So, $\sum_{v \in (G')} deg(v) \geq 4r$ — ①

But we know that $\sum_{v \in (G)} deg(v) = 2e$

So, eqn ① becomes.

$2e \geq 4r$

$\Rightarrow \boxed{r \leq \frac{1}{2}e}$

By Euler's Formula

$r = e - v + 2$

$\Rightarrow e - v + 2 = r$

$\Rightarrow e - v + 2 \leq \frac{1}{2}e$

$\Rightarrow \frac{1}{2}e - v + 2 \leq 0$

$\Rightarrow e - 2v + 4 \leq 0 \Rightarrow \boxed{e \leq 2v - 4}$

(proved)

pt ③ Assume the contradiction that all the vertices are of degree ≥ 6.

So, $\sum deg(v_i) \geq 6v$ — ①

But we know! $\sum deg(v_i) = 2e$

$\therefore$ eqn ① becomes $2e \geq 6v$

$\Rightarrow \boxed{v \leq \frac{1}{3}e}$

Also, we have $\boxed{r \leq \frac{2}{3}e}$

From Euler's Formula -

$r = e - v + 2$

$\Rightarrow e = r + v - 2$

$\leq \frac{2}{3}e + \frac{1}{3}e - 2 = e - 2$

$\Rightarrow e \leq e - 2 \Rightarrow \boxed{0 \leq -2}$ False.

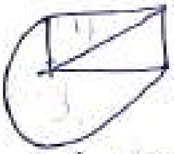
Which is a contradiction due to our wrong assumption. Thus, all the vertices are of degree ≤ 5. Proved

Thm- A complete graph K_n is planar iff $n \leq 4$.

Pf- Consider $n=1$, K_1 • (Planar)

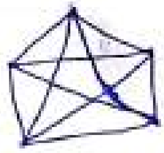
$n=2$, K_2 —— (Planar)

$n=3$, K_3  (This is Planar)

$n=4$, K_4,  (This is Planar)

$\Rightarrow K_n$ is Planar if $n \leq 4$.

$n=5$, Assume the contradiction that, K_5 is Planar.

So, if $v=5$, $e=10$ 

Now, $r = e-v+2 = 10-5+2 = 7$

But as K_5 is simple and loop free.

$\boxed{r \leq \frac{2e}{3}}$

i.e $3r \leq 2e$

$\Rightarrow 3 \times 7 \leq 2 \times 10 \Rightarrow 21 \leq 20$ (False)

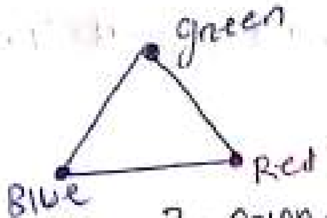
which is contradiction due to our wrong assumption that K_5 is Planar.

Hence K_5 is not Planar.

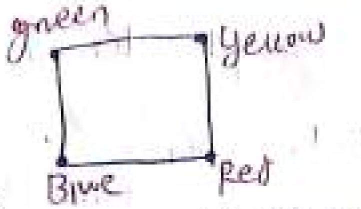
Graph Colouring

- A Coloring of a Simple Graph is the assignment of a color to each vertex of the graph
- So that no two adjacent vertices are assigned the same ~~color~~ color.

(Always try to color with minimum colors)



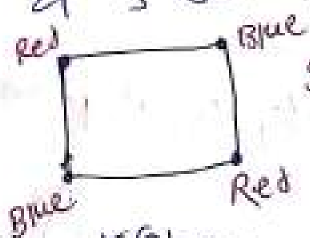
3- Color graph
(i)



4- color graph.
(ii)

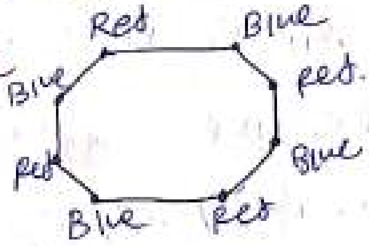
Chromatic Number

- It is defined as the least no. of colors needed for coloring the graph.
- It is denoted by $\chi(G)$ or K-chromatic Graph.



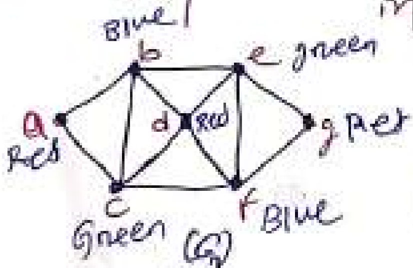
$\chi(G)=2$

$\chi(G)=3$



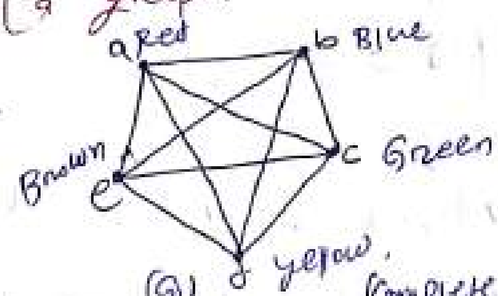
$\chi(G)=2$

Ex- Find chromatic number of a graph?



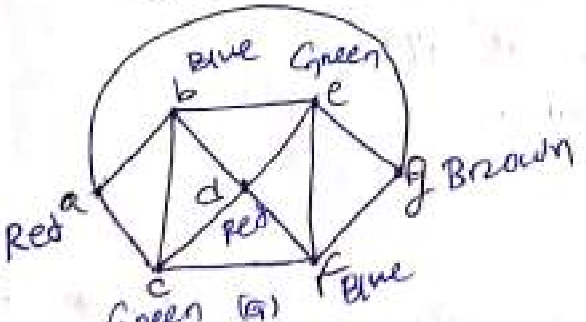
(G_1)

$\chi(G_1)=3$



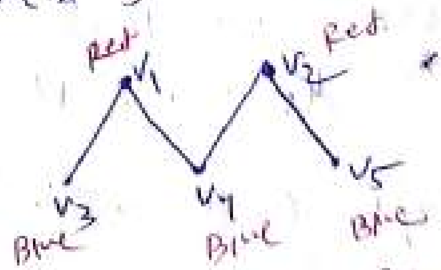
(G_2)

$\chi(G_2)=5$ (Complete graph K_5)



(G)

$\chi(G)=4$



$\chi(G)=2$ (Bipartite Graph)

Note every bipartite graph is 2-colorable

Theory 5 - Color Graph

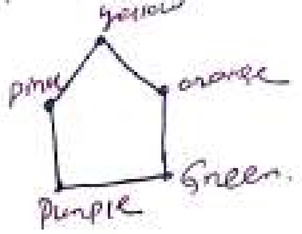
Every Planar Graph with n Vertices can be colored using atmost 5 colors.

Pf - We Prove Them by mathematical Induction method.

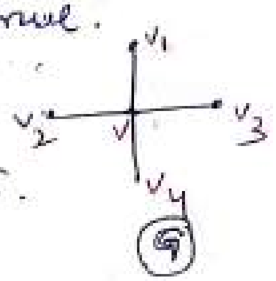
Basis Step (i) $P(n \le 5) \rightarrow$ Graph can be colored using 5 colors.

We have a Lemma, Every planar Graph contains a vertex with degree

$$\boxed{\deg(v) \le 5}.$$



(ii) Inductive step Let $K=4$, $\deg(v) \le 4$ It will be true.



(iii) Inductive Hypothesis we verify for $K=1$, $\deg(v) \le 5$.

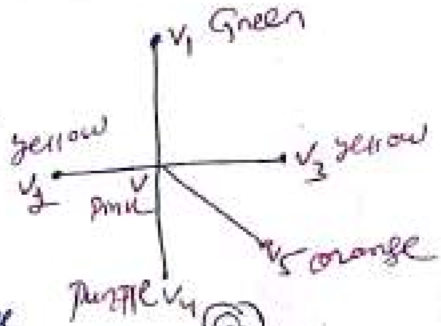
Suppose, a one degree add.

i.e. an edge/vertex v_5 add in G.

Now we have used five colors.

Case-I No Direct edge between v_1 & v_3, v_2 & v_4, v_1 & v_5, v_2 & v_3 - - - - -

i.e. No direct edge in G_1.

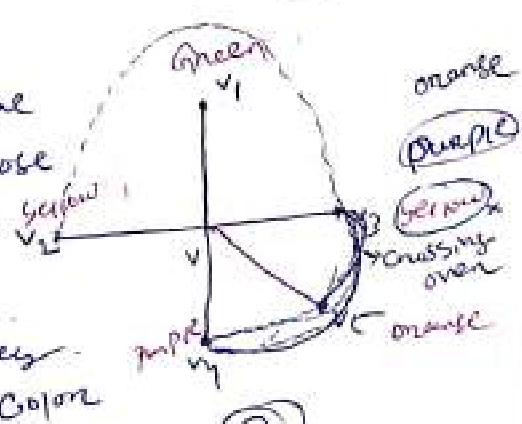


Case-II Now assume that direct edge between v_2 & v_3.

but v_2 & v_3 vertex having same color, so color changes suppose v_3 yellow to purple.

again, suppose the direct edge between v_3 & v_4, but they having same color purple, so, color changes suppose v_3, purple to orange.

$\therefore$ v_3 & v_5 having same color orange so, there is no direct edge between v_3 & v_5.



suppose, a direct edge v_4 to v_3.

$\therefore$ again a direct edge between v_5 to v_3.

Then, a direct edge v_5 to v_3 is a crossing over the edge v_4 to v_3. (i.e not a planar graph)

So, there is no direct edge in G_2. (Hence Them Proved).

GRAPH TRAVERSALS

Traversing a Graph:- Traversing a graph means visiting all the vertices of the graph exactly once.

Graph Traversal Technique

- DFS (Depth-First search)
- BFS (Breadth-First search)

DFS (Depth-First search)

① First visit the vertex V, then visit all the vertices that lie along a path which begins at V.

That means first visit the vertex V, then visit the vertex which is immediate adjacent to V. (say V_x).

Next, If V_x has any adjacent vertex (say V_y), then visit it and continue till there is a dead end. (ie a vertex which does not have immediate adjacent vertex or immediate adjacent vertex already been visited)

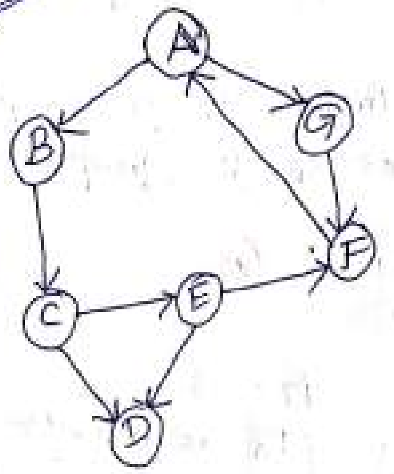
② DFS is also called backtracking. (since the algorithm returns to vertices previously visited to add paths)

③ A data structure 'Stack' used in DFS.

a Stack is an ordered collection of homogeneous data elements where the insertion and deletion operation takes place at one end only.

- The insertion operation is called PUSH and Deletion operation is called POP.

Ex DFS



List visited

A — Start vertex A.

AG — visit A to G (adjacent of A)

AGF — From G visit to vertex F

AGFB → back track to A because there is no path from F to any vertex other than A. So from A visit to B (adjacent vertex).

AGFBC

AGFBCE

$\boxed{\text{AGFBCED}}$

Went from B to C, and C to E, went E to D.

As there is no path from D to any other vertex which is not visited, the algorithm stops at this point.

② BFS (Breadth-First Search)

① Find choose a starting vertex

② Find all the vertices which are connected to the starting vertex

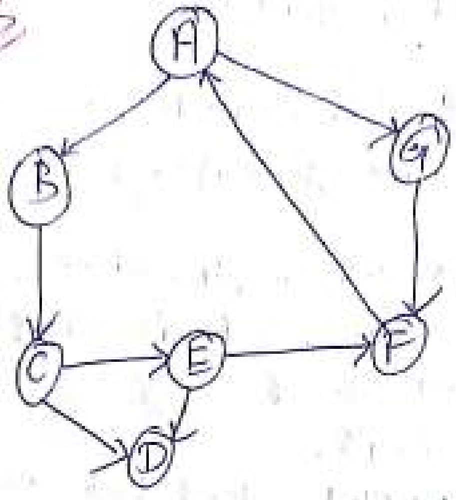
③ Then choose one of the connected vertices and find all the vertices that are connected to this vertex.

④ Continue this process until all the vertices are visited.

Note - ① A data structure QUEUE is used in BFS.

① a ~~queue~~ queue is an ordered collection of homogeneous data elements, but in contrast to the stack, in case insertion and deletion operation takes place at two extreme ends.

Ex- BFS

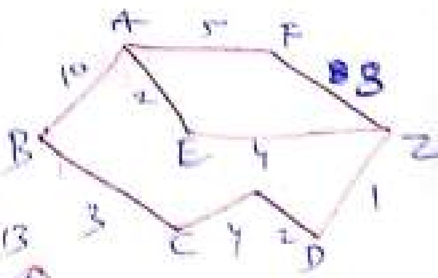


	List visited
① Start with vertex A.	A
② Visit the vertices in the next level i.e B and G	ABG
③ Visit B's adjacent vertex C	ABGC
and G's adjacent vertex F.	ABGCF
④ Visit C's adjacent vertex D & E.	$\boxed{ABGCFDE}$

SHORTEST PATHS IN WEIGHTED GRAPH

The concept of shortest path between two vertex. It means to find the smallest length or distance between two given vertex.

- There are several well known procedures for solving shortest path between the two vertices.
- Here we discuss the well known method. Dijkstra's Algorithm.



There are 3 paths. ~~A-F-Z~~
Reached

A-F-Z = 5+8 = 13
A-E-Z = 2+4 = 6
A-B-C-D-Z = 10+3+4+2+1 = 20
∴ Shortest path = 6

Dijkstra's Algorithm

Dijkstra's Alg. is use to find the shortest path between a (source vertex) and b (destination vertex)

Suppose $G=(V,E,W)$ be any graph, where 'V' is the set of vertices, E is the set of edges and 'W' is the set of weighted of edges.

Step-I Remove all the loops.

Step-II Remove all parallel edges between two vertices except the one with least weight.

Step-III Create the weighted matrix table.

(i) Set '0' (zero) to the source vertex (starting vertex) and infinite (∞) to the remaining vertices.

For all vertices repeat (ii) and (iii)

(ii) Mark the smallest unmarked value and mark that vertex.

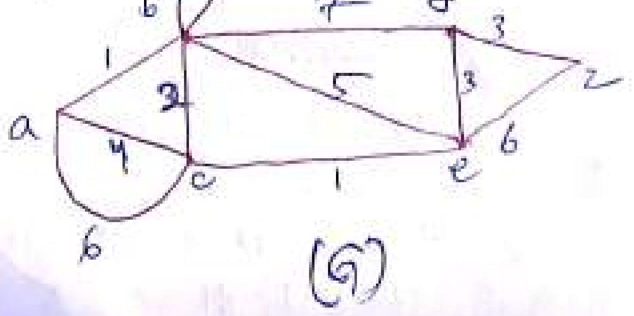
(iii) Find those vertices which are directly connected with marked vertex and uploaded all.

Uploaded value formula

$$\boxed{\text{New destination value} = \text{minimum}(\text{Old destination value}, \text{Marked value} + \text{Edge weight})}$$

Ex-

Using Dijkstra's Alg. Find the shortest path

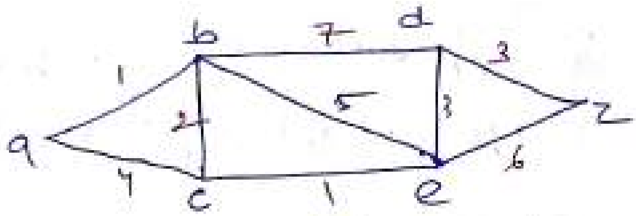


(G)

Soln - Source vertex = a
Destination vertex = z

Step-I Remove all the loops. (In a graph G' Remove the loop at vertex 'b')

Step-II Remove all parallel edges between the except one ~~with~~ with least weight.
(In a Graph G, parallel edges a-c & a-c removed but except least ~~vertex~~ weight a-c = 4



G

Step-3

Weighted matrix Table

Set '0' to the Source vertex 'a'. and Remaining all the vertices are ∞.

a	b	c	d	e	z	Marked vertex
0	∞	∞	∞	∞	∞	a
	1	4	∞	∞	∞	b
		3	8	6	∞	c
			8	4	∞	e
			7		10	d
					10	z

∴ Minimum value $= \min(0, \infty) = 0$

So, the marked vertex = a
& marked value = 0.

Now, 'a' has two adjacent/destination vertex. b & c.

We have Formula: New Destination value $= \min(\text{old destination value}, \text{marked value} + \text{Edge weight})$

∴ Destination value at vertex 'b' $= D(b) = \min(\infty, 0+1) = 1$

$D(c) = \min(\infty, 0+4) = 4$

(Putting in a table of matrix table)

∴ min. value = 1

So the marked vertex = b
& marked value = 1

Now, 'b' has 4 adjacents vertices a, c, d, e but vertex a already marked.

So, $D(c) = \min(4, 1+2) = \min(4,3) = 3$.

$D(d) = \min(\infty, 1+7) = \min(\infty, 8) = 8$

$D(e) = \min(\infty, 1+5) = \min(\infty, 6) = 6$.

(Putting the value in 3rd row of weighted matrix table)

$\therefore$ min. value $= 3$

i.e marked value $= 3$

So, marked vertex $= c$

($\because$ '3' is destination value of the vertex c so, marked vertex $= c$)

Now, 'c' has Three adjacent vertices. a, b, e

but. vertex a, b already marked

So, $D(e) = \min(6, 3+1) = \min(6,4) = 4$.

$\therefore$ (Putting 4th row of the matrix table)

$\therefore$ min. value $= 4$ ($\because \min(8, 4, \infty) = 4$)

i.e marked value $= 4$,

So, marked vertex $= e$

Now, 'e' has ~~three~~ four adjacent vertices. c, b, d, z.

but. c, b already marked.

So, $D(d) = \min(8, 4+3) = 7$

$D(z) = \min(\infty, 4+6) = 10$

(Putting 5th row of the matrix table)

$\therefore$ min. value $= 7$

i.e marked value $= 7$, So marked vertex $= d$

Now. 'd' has one destination vertex z. ($\because$ 7 is the destination value of d)

So, $D(z) = \min(10, 7+3) = 10$ (Putting the 6th row of the matrix table)

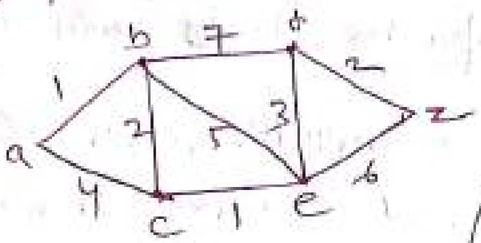
Hence Shortest path is (z, e, c, b, a) or (a, b, c, e, z)

i.e a–b–c–e–z

$\therefore$ The length of path $= 1+2+1+6 = 10$

Assignment

1. Find the Shortest Path. (using Dijkstra's Alg.)



(Ans - (a, b, c, e, d, z)
Shortest distance = 9)

2. Euler path and Circuit.

3. Hamiltonian Path and Circuit.

4. A simple Graph with n vertices have maximum $\frac{n(n-1)}{2}$ edges.

5. A simple Graph with n vertices and k-component Can have more than $\frac{(n-k)(n-k+1)}{2}$ edges.

6. Let G be a connected graph. (planar Graph) Then $\boxed{r = e - v + 2}$ where e - edges
or proof Euler formula. v - vertices
r : no. of regions in a G.

7. If G is a simple Graph with atleast 3 vertices Then (i) $e \leq 3v - 6$

(ii) $e \leq 2v - 4$ (If G is a Triangle free)

8. Prove 5-Color Then

9. Write DFS and BFS Algorithm. in a Graph Traversal.

10. Travelling Sales Person Problem.

11. $A = \{1, 2, 3, 4, 5\}$ and relation $R = \{(1,1) (1,2) (1,4) (3,2)\}$
Represent the relation R as a matrix and draw its digraph.

12. Given a matrix $M(R) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$, $A = \{a, b, c, d, e\}$
Write relation R. and draw the associated Digraph.

Travelling salesperson Problem (TSP)

- TSP is just similar to Hamiltonian circuit/Graph.
- Let $G = (V, E, W)$ be a complete graph of n vertices where W is a function from E to the set of the real numbers.

Such that for any 3 vertices, i, j, k in V

$$\boxed{w(i,j) + w(j,k) \geq w(i,k)}$$

Where $w(i,j)$ means length of an edge $\{i, j\}$

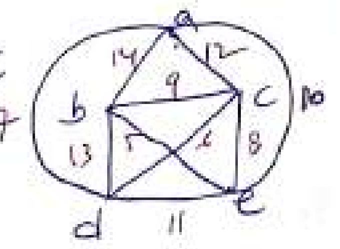
- The Travelling salesperson asks for a Hamiltonian circuit of a minimum length.

A physical interpretation means

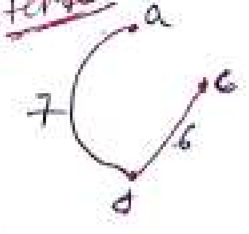
Consider a Graph G as a map of 'n' cities where $w(i,j)$ is the distance between cities i & j.

- A salesperson wants to have a tour of the same city and includes visiting each of the remaining $(n-1)$ cities once and only once. Moreover, an itinerary that has a minimum total distance desired.

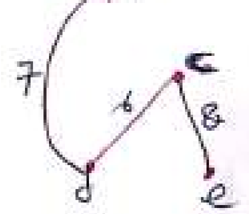
Ex



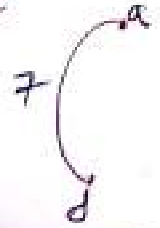
STEP-II



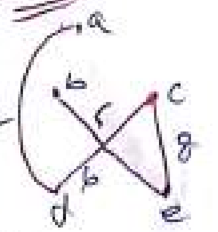
STEP-III



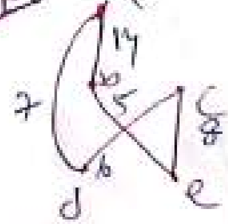
STEP-I



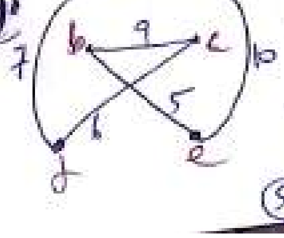
STEP-IV



STEP-V



STEP-(VI)



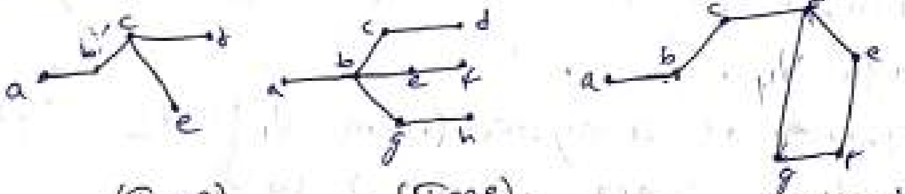
∴ The total distance of the circuit $= 40$

and the total distance (Step-I) of a minimum H. circuit $= 7+9+10+6+5 = 37$

(STEP VI)

TREES

- A Tree is a Connected graph That contains no simple circuit.
- Tree must have at least one vertex.
- Tree has to be a simple graph having neither a self loop nor parallel edges.



(Tree) (Tree) (not a tree) (deg of circuit)

Note 1) ① A vertex of degree 1 in a tree is called a leaf or a terminal node.

② A vertex of degree larger than 1 is called branch node or an Internal node.

Properties of Trees:-

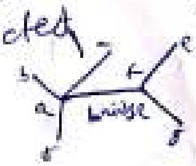
① A Tree has unique path between every pair of vertices.

② A Tree with n vertices has n-1 edges.

③ Any connected Graph with n vertices and n-1 edges is also a Tree.

④ Every edge in a tree is a bridge.

(bridge is an edge If remove the edge from the Tree then the Tree become disconnected)



Thm ① A simple non directed Graph G is a tree iff G is connected and contains no cycle.

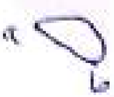
Pf. Suppose G is a tree.
Since There is a unique path between every pair of vertices, G is connected.
Assume the contradiction that it contains a cycle between v_1 to v_2 through v_3 and also there is another path between v_1 to v_2 through v_4.

$\Rightarrow$ There is a cycle from v_1 to v_2 and another cycle from v_2 to v_1
$\Rightarrow$ There exists two pair of vertices between v_1 & v_2
which contradiction to the def^n of Tree.
Hence G is connected without any cycle.

Conversly. As G is connected and contains no cycle. Thus it contains a unique path between every pair of vertices.
So, G is a Tree.

Them ② There is a unique path between every pair of vertices in a Tree T.

Pf- Since T is a connected graph, There must exist at least one path between every pair of vertices in T.
Now, suppose that between two vertices 'a' & 'b' of Tree T.
There are two distinct paths



The union of these two paths will contain a circuit and T cannot be a Tree.

Thm (3)
In a Graph G There is a unique path between every pair of vertices, G is a Tree.

(OR) An undirected Graph is a Tree $\Leftrightarrow$ There exists a unique path between any two of its vertices.

Pf. We have a graph in which there is a path between every pair of vertices is connected.

Moreover, The Graph Cannot contain a Circuit If These paths are unique, Since The existence of a Circuit implies The existence of two distinct Paths between a Certain pair of vertices.

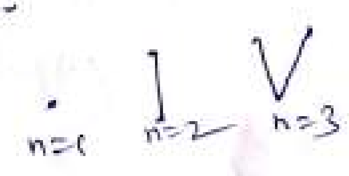
Thus, we Can Conclude That a Graph in which There is a unique path between every pair of vertices is a tree. (Proved)

Thm (4)
A Tree with n-vertices has (n-1) edges.

Pf. we will Prove The Thm by Induction method on the number of vertices.

For n=1, vertex, then, 0 edges.

n=2, Then 1 edges.



True.

Suppose For n=K where 'K' is the integer.

Assume That The Thm is true.

For a Tree having K-vertices has (K-1) edges.

For n=K+1, Then (K+1-1) = K edges.

We will Prove That The Thm is true for K+1 vertices, has K edges.

As every tree has atleast one vertex of degree one remove that vertex along with the edge connecting it.

Then the rest tree has k no. of vertices and $(k-1)$ edges.

Connect with that the removed vertex and the removed edge,

So, we have a tree with $(k+1)$ no. of vertices and 'k' no. of edges.

$\therefore$ The Thm is true for $n = k+1$, vertices has k edges.

Hence A Tree with n vertices has $n-1$ edges is true.

Thm ② A Graph G is a Tree iff G has no circuits and $e = v-1$

(or) A Graph with $e = v-1$ that has no circuit is a Tree.

Pf= Suppose G is a Tree.

Then, as it is connected and has a unique path between every pair of vertices.

Thus, it has no circuits.

we have, a Tree with n vertices has $(n-1)$ edges.

i.e $v = n$

$e = n-1$

$\therefore e = n-1$

$\Rightarrow \boxed{e = v-1}$

Conversely, assume that G has no circuits and $e = v-1$

To show that G is a Tree.

i.e we have to prove that it contains only 1 components.

Assume a contradiction that G has ~~k components~~ 'k' no. of components.

where $K > 1$, i.e $G_1, G_2 \cdots G_K$ components.

Each component being connected is a Tree.

Let $V_1, V_2 \cdots V_K$ are the no. of vertices in $G_1\ G_2 \cdots G_K$.

and $e_1, e_2 \cdots e_K$ are the no. of edges in $G_1, G_2 \cdots G_K$.

In a graph G -

$$v_1 + v_2 + \cdots + v_K = v$$

$$e_1 + e_2 + \cdots + e_K = e$$

Now, since $\boxed{e = v-1}$

Similarly, $e_1 = v_1 - 1$

$e_2 = v_2 - 1 \cdots \cdots e_K = v_K - 1$

Adding $e_1 + e_2 + \cdots + e_K = v_1 + v_2 + \cdots + v_K \underbrace{-1-1-1-1\cdots}_{K \text{ times}}$.

$$\Rightarrow e = v - K$$

$$\Rightarrow v - 1 = v - K \quad (\because e = v-1)$$

$$\Rightarrow \boxed{K = 1}$$

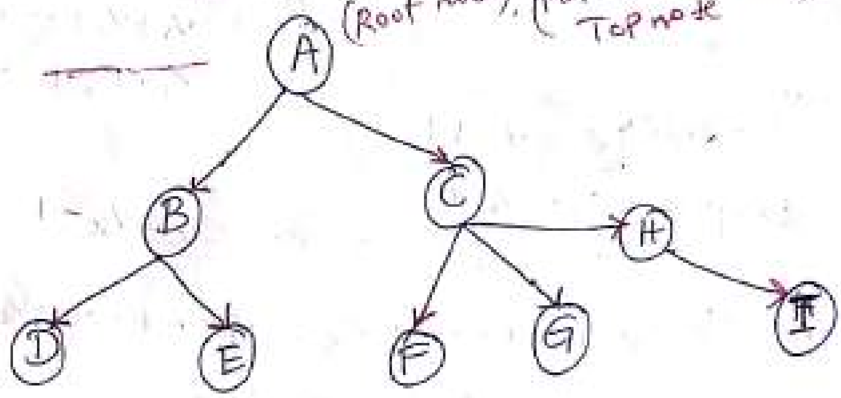
Thus, the Graph G has only one component.

Hence It is a Tree. (Proved)

Rooted Tree

A Rooted tree is a tree in which one vertex has been designated as the root and every edge is directed away from the root. or (It is a tree in which some node is distinguish as a root)

(Root node), (Top of the Tree Top node)



A ⇒ Root vertex. (Top of the Tree Top node/vertex)

B ⇒ parent of D & E. , B has two children D & E.

C ⇒ parent of F, G, H. C has Three children F, G, H. H has one child I.

Ancester - (Any node higher Than parent node)

A → Ancester of D & E.

D & E → descendant of A.

Left child

Right Child

B & C → Sibling (having Same Parent A)

E & F → are not Sibling (not Same parent)

Leaf node/vertex → (Those nodes donot have any child)

or (Terminal node)

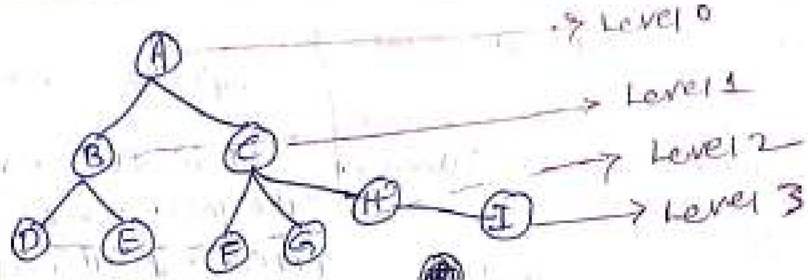
D, E, F, G, I

(In a rooted Tree a vertex whose outgoing degree is zero)

Branch Node (Internal Node) → (a vertex whose outgoing degree is non zero)

B, C, H

Level of This Tree → we always put the level from The root node start from 0 level.



Height of The Root node

It is the length of longest Path between root to any leaf node. Total no. of edges across while traverses.

Height $A \to I = 3$ $(A \to C \to H \to I)$
$A \to E = 2$ $(A \to B \to E)$

Height of a Tree :- It is the hight of the root.
In above Tree, Height of Tree = 3. $(A \to I = 3)$

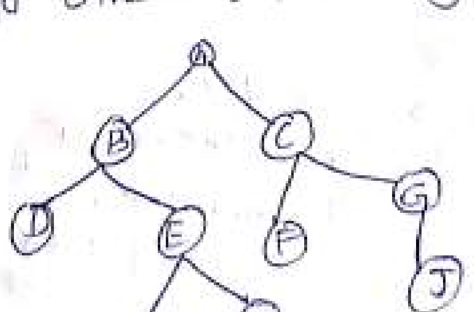
Degree of a Tree :- It is the maximum degree of any vertex of that tree.

In above Tree, Degree of vertex C = 3 (maximum)
So, Degree of Tree = 3

Binary Tree

A Tree is said to binary Tree whose each vertex at most two children.
(i.e either no child, or one child, or two children).

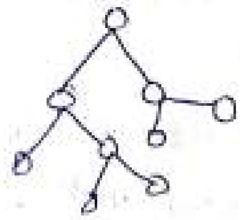
- If a vertex has one child Then it is either left or Right child.
- If a vertex has two child, Then one is Left child and other will be right child.



D, F, J, H, I have zero child.
G has one child.
E has two child.
B, C, A, " ".

Strict / proper / Full Binary Tree

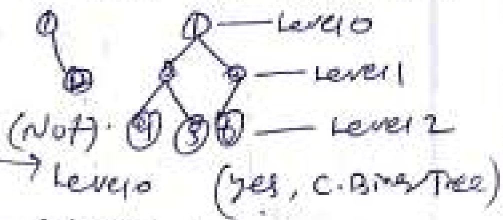
each node can have either zero or two children.

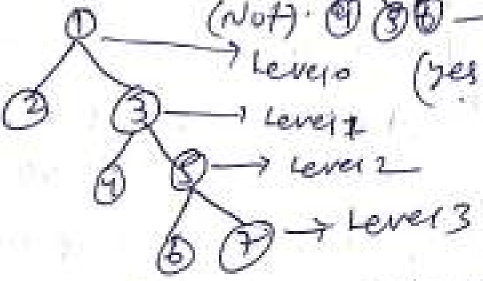


(~~one child~~ / 2 child)

Complete Binary Tree

all level except possibly the last are completely filled and all nodes/vertices as left as possible.





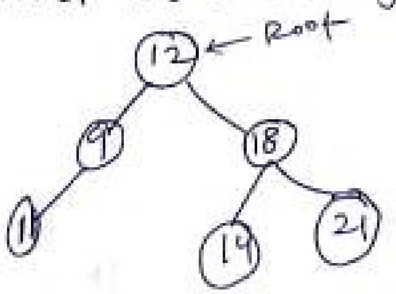
Perfect Binary Tree

all levels are completely filled.

(any vertex having two child)

Binary Search Tree (BST)

- In (BST) nodes/vertices left child must have value less than its parent value and nodes right child must have value greater than its parent value.



Left child (value) < parent value

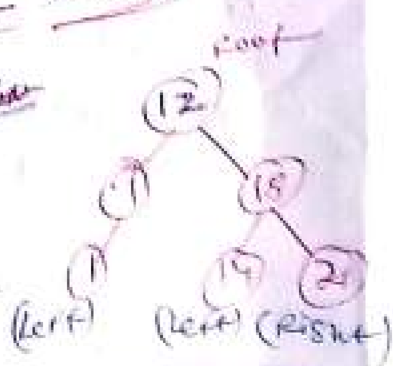
Parent (value) < Right child (value)

- BST are used mainly for searching purpose.

Ex— ~~Deletion~~
Basic operation of BST

1. Inserting a element.
2. Deletion of element
3. Searching
4. Sorting → can be done by Traversing a BST in inorder. (left data right)

Ex— Sorting

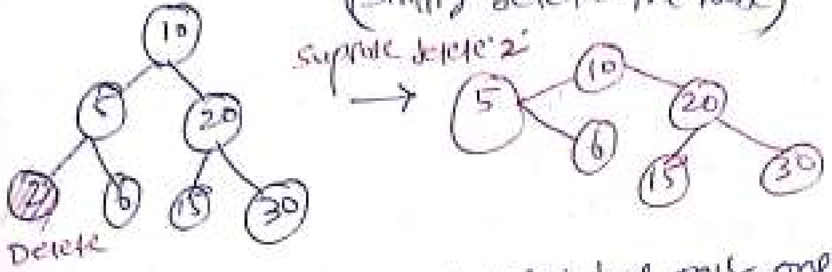


1	9	12	14	18	21

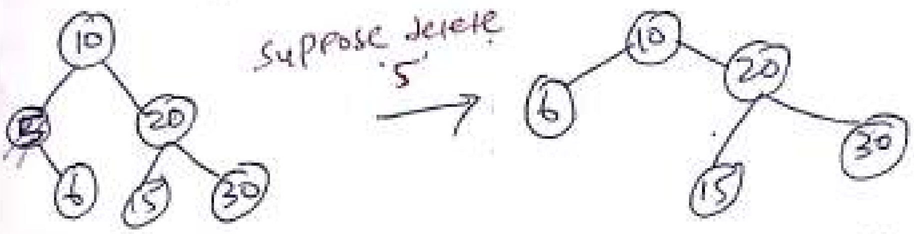
(sorting in increasing order)

Deletion in Binary Search Tree (BST)

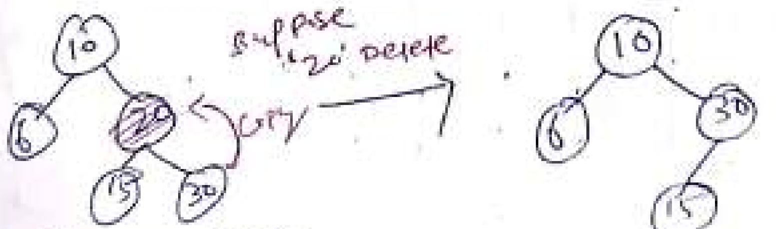
Case-1 If the node to be deleted is a leaf node → (no child)
(Simply delete the node)



Case-2 If the node to be deleted has only one child
(copy child to node and delete child)



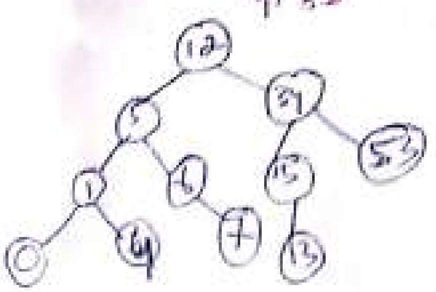
Case-3 If the node to be deleted has two children nodes.
[find Inorder successor of node, then copy content of inorder successor node and delete Inorder successor]



LDR → 6 10 15 20 30

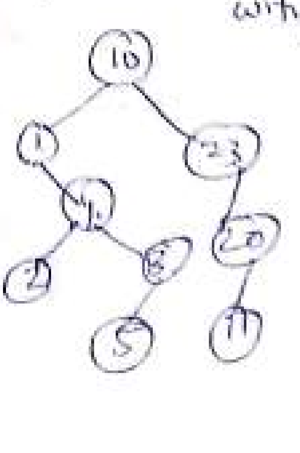
Insertion in BST

Ex-1 12, 34, 5, 1, 15, 6, 7, 0, 13, 4, 53



Ex-2 Draw a Binary Search Tree whose level order indices are {1, 4, 7, 8, 5, 23, 20, 11, 10} with 10 as a root

Soln

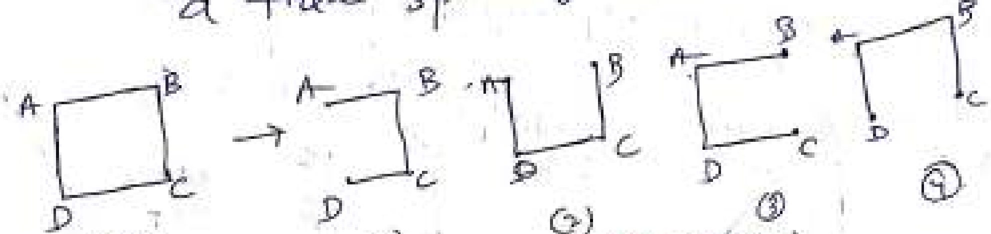


Spanning Tree

It is a connected graph using all vertices in which there is no cycle.

(or) It is composed of all the vertices and some edges of Graph G.

- Selection of edges in a Graph G that forms a tree spanning every vertex.



(G) → (1) (2) (3) (4)

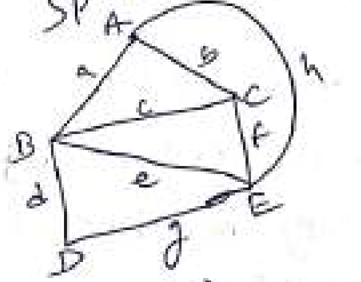
(Spanning tree)

two terms

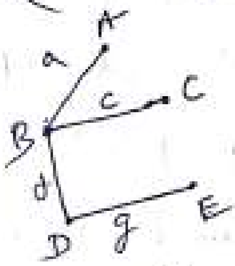
Branch (an edge in a spanning tree)

Chord (an edge of G that is not given in spanning tree)

Ex-



(G)



(T) spanning tree

Branch → {a, c, d, g}

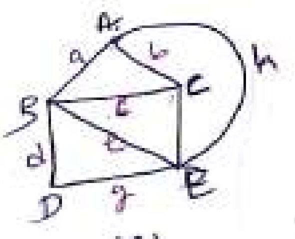
Chord → {b, f, e, h}

Fundamental Circuits:-

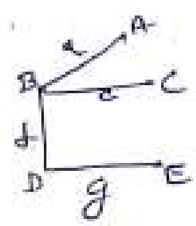
G: Graph (connected)

T: Spanning Tree

A circuit formed by adding a chord to spanning Tree (T) is called a Fundamental circuit.



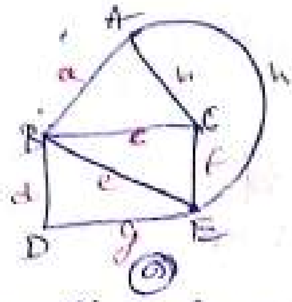
(G)

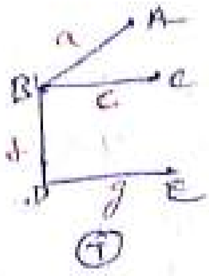


(T)

Branch = {a, c, d, g}

Chord = {b, e, f, h}

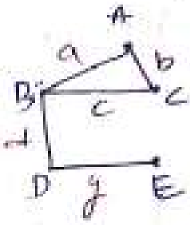




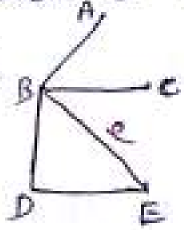
Branch = {a, c, d, g}
Chord = {b, e, f, h}

Fundamental circuits are → (adding a chord)

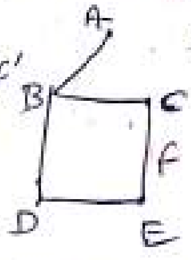
1st add 'b' in a spanning Tree (T)



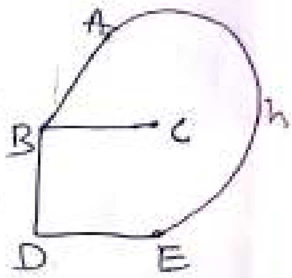
add 'e' in a (T)



add 'f' in a (T)



add 'h' in a (T)

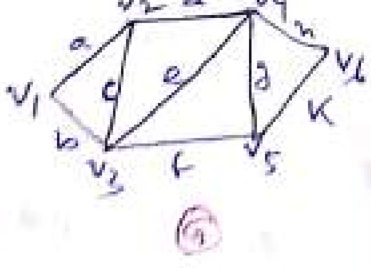


There are 4 fundamental circuit.

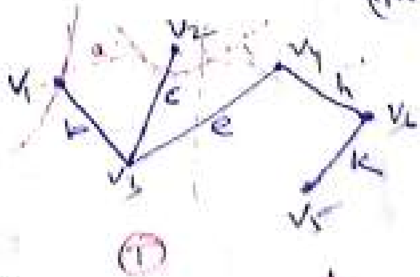
Note Graph G having 'e' edges, and n vertices.
Then Spanning Tree (T) having (n-1) branches.
Then there are exactly (e-n+1) Chord.
and (e-n+1) circuit Fundamental.

Fundamental cut set (Basic cut set)

- Partition of all vertices in to two disjoint sets.
- always contains only one branch and rest of the edges are cuts.
- such a cut set is called Fundamental cut set.



(Arbitrary spanning Tree)



Branch :- {b, c, e, h, k}
Chord :- {a, d, f, g}

1. {a, b}
2. {a, c, d}
3. {d, e, f}
4. {h, g, f}
5. {f, g, k}

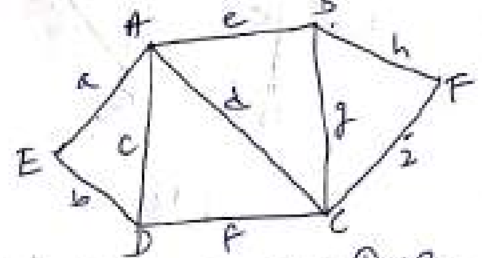
→ Fundamental cut set or Basic cut set.

All Spanning Tree of a graph:

1. Select from an arbitary spanning Tree. Calculate Branch Set, chord set.
2. Remove one edge from the chord set.
3. It generates a fundamental circuit. Now remove any Branch from the circuit.
4. Now Repeat this step-2 until chord set is empty.

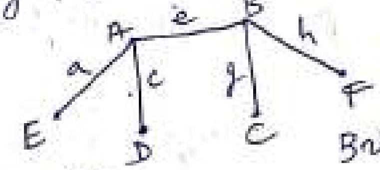
Ex Find all spanning Tree of the following Graph.

1.



Sol

1. Let arbitary spanning Tree.

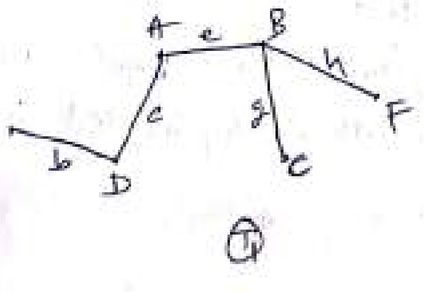


T

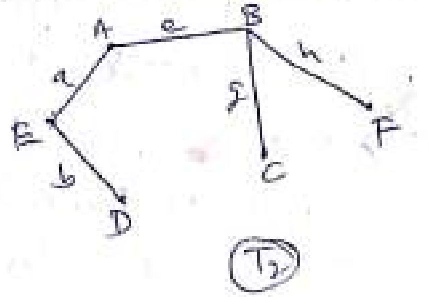
Branch set $= \{a, c, e, g, h\}$

Chord set $= \{b, d, f, i\}$

2. Add 1st chord 'b' and it generate circuit (Fundamental) abc.

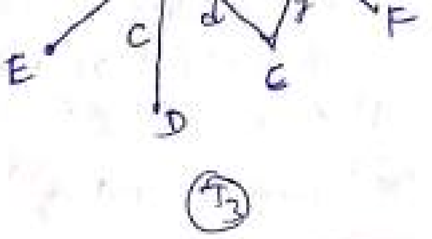


T_1



T_2

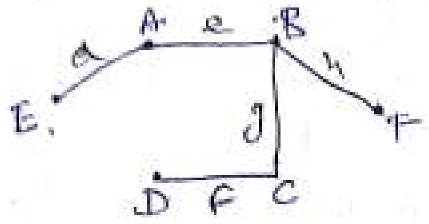
3. Add chord 'd' and it generate Fundamental circuit is deg.



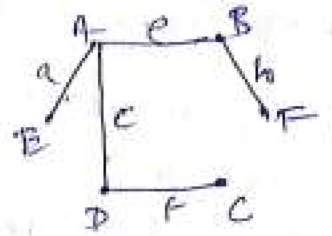
T_3

T_4

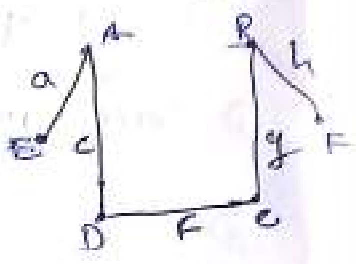
④ Add chord 'f' and so, it generates F.C is cfge



(T5)

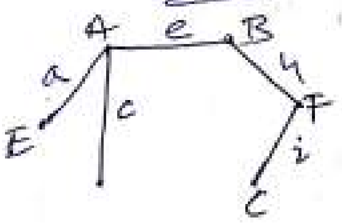


(T6)

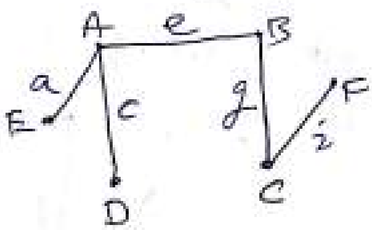


(T7)

⑤ Add chord 'i' so, and it generates F.C is igh.



(T8)



(T9)

∴ Total spanning Trees = 9+1=10 (T9+1=10 one B arbitrary spanning Tree)

Minimal spanning Tree (MST) :-

Using two algorithm process. to find a minimal spanning Tree (MST).

① Prims algorithm.
② Kruskal's algorithm.
→ are a greedy algorithm that finds a MST for a connected weighted undirected Graph.

Difference between Prims & Kruskal's algorithm

Prims A-G.	Kruskal's A-G.
① always initiate with a node.	① Initiate with an edge.
② span from one node to another.	② select an edge in such a way that the position of the edge is not based on the last step.
③ Graph must be connected Graph.	③ connected as well as disconnected.
④ Time complexity of O(V²)	④ Time complexity is O(log V).

Scanned with CamScanner

Kruskal's Algorithm

(Find the minimum spanning Tree with help of Kruskal Alg.)

Step-I

List all the edges of the Graph G in the increasing order of weights.

Step-II Select the smallest edge (having minimum weight) from the list / of the Graph G.

and (That edge is the branch of spanning Tree (T)).

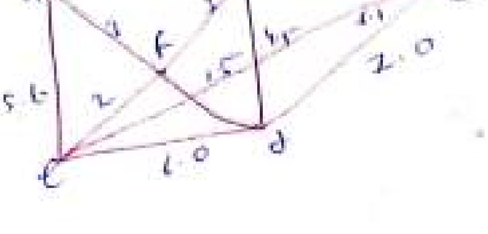
Then select another smallest edge that don't makes any circuit.

If the selected edge makes a circuit then remove it from the list.

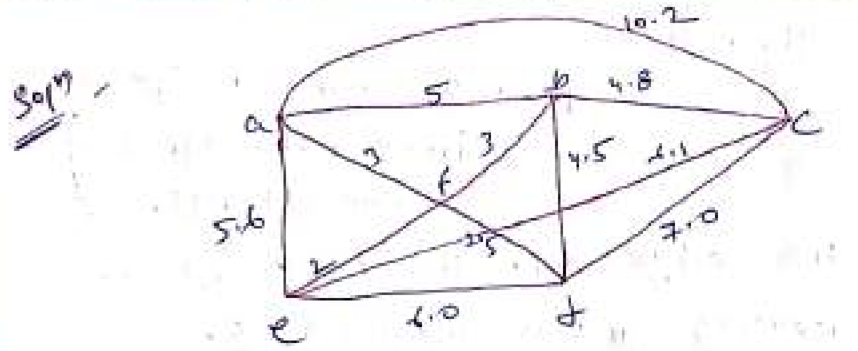
Step-III Continue in this process until (n-1) edges (where n is the no. of vertices) have been selected, or the list is empty.

Step-IV Now If the Tree contains less than n-1 edges and the list happened to be empty then no spanning Tree is possible, else it gives the minimum spanning Tree.

Ex- Find out minimum spanning Tree, using Kruskal Alg.



Sol^n -

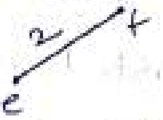


Step-1

Edges in increasing order of their weight	Weight of edges
e-f	2
f-d	2.5
a-f	3
f-b	3
b-d	4.5
b-c	4.8
a-b	5
a-e	5.6
e-d	6.0
e-c	6.1
d-c	7.0
a-c	10.2

Step-2 Select the smallest edge

(i) e-f = 2



(ii) select another smallest edge. (don't make any circuit)

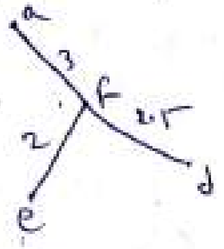
f-d = 2.5

(v) b-d = 4.5 X

not selected an edge because it makes a circuit formed.

Then remove it.

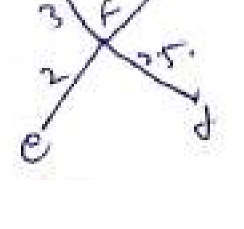
(iii) a-f = 3



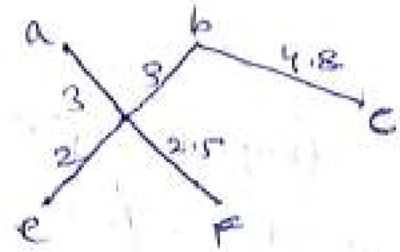
(vi) a-b = 5 x

not selected an edge because it makes a circuit formed. Then remove it.

(iv) f-b = 3



(vi) b - c = 4.8



(vii) a - b = 5 ×

not selected an edge
because. It makes a circuit formed.
Then Remove it.

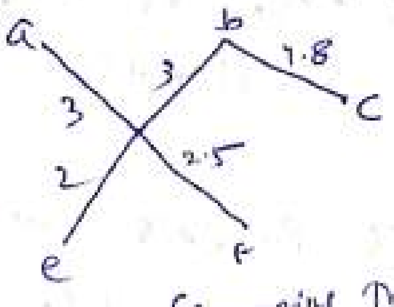
(viii) a - e = 5.6 not selected an edge
because It makes a circuit formed.
Then Remove it.

(ix) e - d = 6.0 × not selected an edge
as It makes a circuit formed.
So. Remove it.

(x) d - c = 7.0 × not selected an edge.
as circuit formed.
So, Remove it.

(xi) a - c = 10.2 × Remove it (as circuit formed)

∴



(Spanning Tree)
(T).

So, the weight of the
Tree = minimum of spanning Tree
= 2 + 2.5 + 3 + 3 + 4.8
= 15.3

Final
Spanning Tree (T) contains
6 vertices and 5 edges.

Prims Algorithm

(Find the MST using Prims Alg.)

Step-I Draw 'n' isolated vertices and label them as $v_1, v_2, \ldots, v_n$, where n is the no. of vertices.

Step-II Represent the given weight of the edges of graph G in an n×n matrix.

$$\begin{array}{c|c} & v_1 \quad v_2 \; \cdots \; v_n \\ \begin{array}{c} v_1 \\ v_2 \\ \vdots \\ v_n \end{array} & \begin{bmatrix} w_{11} & w_{12} & \\ & \vdots \text{ given weight} & \\ & \vdots & \end{bmatrix} \end{array}$$

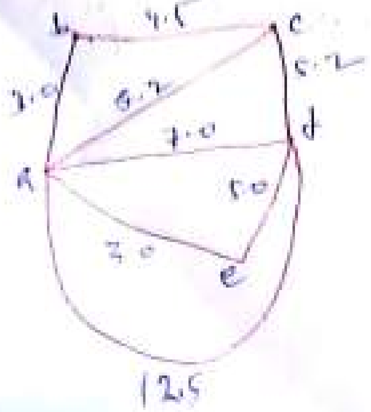
Step-III Start from vertex v_1 and connect it to its nearest neighbours by searching in row 1 of the matrix.

(If more than one smallest entry is there then arbitarily select one of them)

Step-4 Consider v_1 & v_2 as one subgraph and connect as step-III. don't form any circuit.

Step-5 Continue this process until we get spanning tree (T) having n vertices and (n-1) edges.

Ex- Find minimum spanning tree by Prims Algorithm.

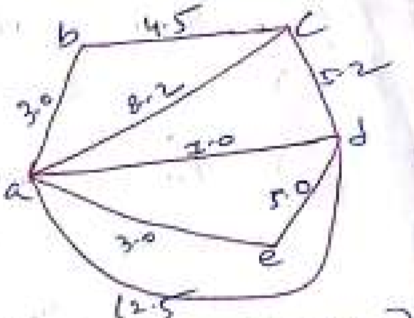


Soln

Step-I

Matrix form

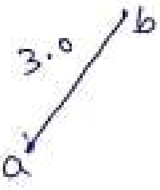
$$
\begin{array}{c|ccccc}
 & a & b & c & d & e \\
\hline
a & - & 3.0 & 8.2 & 7.0 & 3 \\
b & 3.0 & - & 4.5 & \infty & \infty \\
c & 8.2 & 4.5 & - & 5.2 & \infty \\
d & 7.0 & \infty & 5.2 & - & \infty \\
e & 3.0 & \infty & \infty & 5.0 & -
\end{array}
$$



(There is no search path $b \to d$, $b - e$ ... so, ∞)

Step-II Start from vertex a and connect to its nearest neighbour by searching Row 1.

(i) Suppose,
a—b

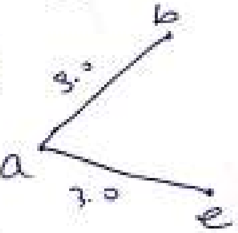


(one subgraph)
(a, b)

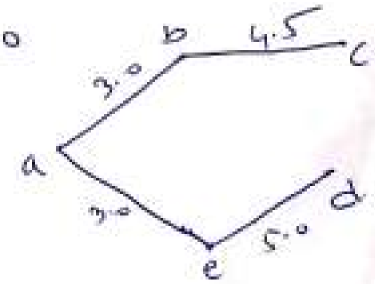
(a to b weight 3
a to e " 3
Select you one of them)

(ii) Next a–e = 3.0 , b–c = 4.5 ✗

(a, b, e)



(iii) Next adjacent of b : b–c = 4.5
adjacent of e : e–d = 5.0



a – c → 8.2
a – d → 7.0
c – d → 5.2
e – d → 5.0 lesser weight.

(a, b, c, e)

∴ The Spanning Tree 5 vertices 4 edges.

Now, Min. spanning Tree = 3+3.0+4.5+5.0
= 15.5

Assignment

1. Define Tree and Rooted Tree
2. A Tree with n-vertices has (n-1)-edges.
3. In a Graph G, there is a unique Path between every Pair of vertices, G is a tree.
4. A Graph with e=v-1 that has no Circuit is a Tree.
5. Write a note on Binary Search Tree.
6. Kruskal's Algorithm
7. Prim's Algorithm
8. Draw a binary search tree whose level order indices {17, 23, 4, 7, 19, 9, 45, 6, 2, 99, 37}

(i) with 19 as a root.

(ii) Delete 19

THE END