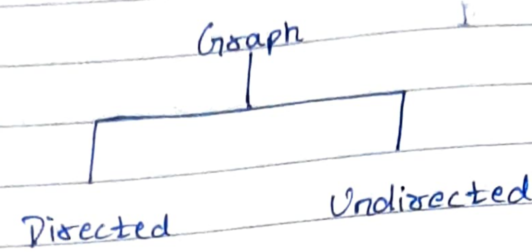
/2022

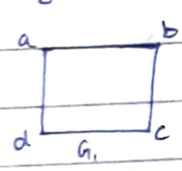
Graphs And Trees

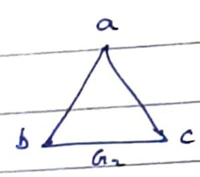
★ 

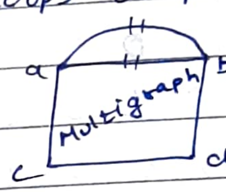
★ **Types of Graphs:**

1. Undirected Graphs (Diagraph)
2. Simple Graph

→ A graph that has no self loops or parallel edges.

Ex. 





→ If there exists more than one edge between a pair of vertices, the resulting graph is called multigraph.

→ A simple graph with n vertices has a maximum of $n(n-1)/2$ edges.

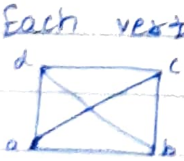
→ G_1 : 4 vertices → $4(3)/2 = \underline{6}$ = maximum vertices.

3. Spanning Subgraph $(V = V')$
4. Complete Graph

→ The undirected complete graph of n ~~vertices~~ vertices, denoted as K_n, is a graph with n vertices in which there is an edge between each pair of distinct vertices.

→ Degree of each vertex is $n-1$.

→ Each vertex is multually adjacent to every other vertex.

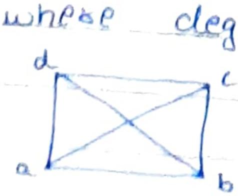
Ex. 

K_4

Deg(each vertex) = 3

5. Regular Graph

→ G on n vertices, where degree of each vertex is k.

→ 3-regular Graph: 

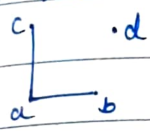
→ 2-regular Graph :

→ 1-regular Graph :

★ Every complete graph is a regular graph but every regular graph need not be a complete graph.

★ <u>Isolated Vertex</u>

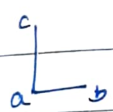
→ A vertex with degree 0 is called isolated vertex.

Ex: 

d is an isolated vertex.

★ <u>Pendent Vertex</u>

→ A vertex with degree 1 is called pendent vertex.

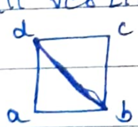
Ex: 

b and c are pendent vertices.

★ <u>Sum of Degree Theorem.</u>

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

→ In a simple undirected graph, the sum of degree of all vertices is twice ~~of~~ the number of edges.

Ex. 

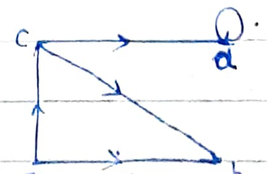
$2+3+2+3 = 10 = 2(5)$

★ <u>Degree of Vertex</u>

→ Undirected Graph

G_1

→ Directed Graph



G_2

→ Degree: Number of edges passing through a vertex.

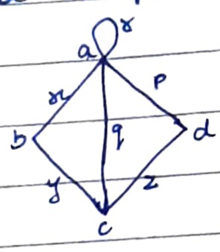
→ G_1: deg(a) = 2 ; deg(b) = 4 ; deg(c) = 0 ; deg(d) = 2.
Loop is counted as two edges.

→ G_2: Indeg(a) = 0   Outdeg(a) = 2
Indeg(b) = 2   Outdeg(b) = 0
Indeg(c) = 1   Outdeg(c) = 2
Indeg(d) = 2   Outdeg(d) = 1
Loop is counted as one edge.

☆ Representation of Graph

1. Incidence Matrix

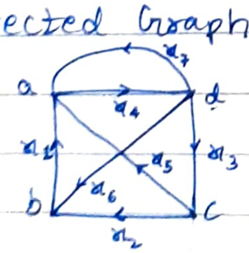
→ Undirected Graph:



→
$$
\begin{array}{c|cccccc|l}
 & x & y & z & p & q & r & \\
\hline
a & 1 & 0 & 0 & 1 & 1 & 2 & \rightarrow 5 \text{ (Degree of a)} \\
b & 1 & 1 & 0 & 0 & 0 & 0 & \rightarrow 2 \text{ (Degree of b)} \\
c & 0 & 1 & 1 & 0 & 1 & 0 & \rightarrow 3 \text{ (Degree of c)} \\
d & 0 & 0 & 1 & 1 & 0 & 0 & \rightarrow 2 \text{ (Degree of d)} \\
\hline
 & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & n \times e \\
 & 2 & 2 & 2 & 2 & 2 & 2 & \text{(Always 2)}
\end{array}
$$

(Sum)

→ Directed Graph:

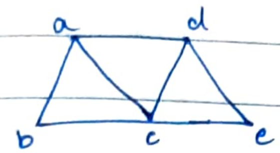


→ Incoming : −1
Outgoing : +1
Self-loops cannot be represented in ~~indic~~ incident matrix for directed graphs.

$$
\begin{array}{c} \\ a \\ b \\ c \\ d \end{array}
\begin{array}{c}
\begin{array}{ccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \end{array} \\
\begin{bmatrix}
-1 & 0 & 0 & 1 & -1 & 0 & -1 \\
1 & -1 & 0 & 0 & 0 & -1 & 0 \\
0 & 1 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 1 & 1
\end{bmatrix}_{n \times e}
\end{array}
$$

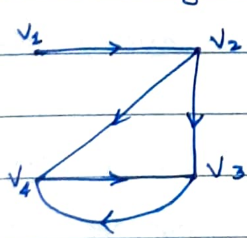
2. **Adjacency Matrix**

→ Undirected graph:



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

→ ~~Ex~~ Directed graph:



→
$$
\begin{array}{c} \\ V_1 \\ V_2 \\ V_3 \\ V_4 \end{array}
\begin{array}{c}
\begin{array}{cccc} V_1 & V_2 & V_3 & V_4 \end{array} \\
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0
\end{bmatrix}
\end{array}
$$

Incoming : 1 ; Outgoing = 0

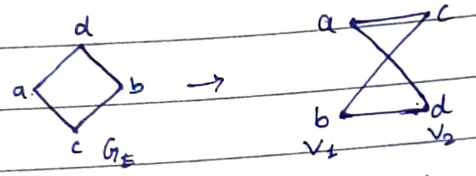
Q. How many nodes/vertices are necessary to construct a graph with exactly 6 edges in which each node is of degree 2.

$$\sum_{i=1}^{n} deg(v_i) = 2e$$
$\Rightarrow 2n = 2(6)$
$\therefore \Rightarrow n = 6$

☆ Biphartite and Complete Biphartite Graph ($K_{m,n}$)

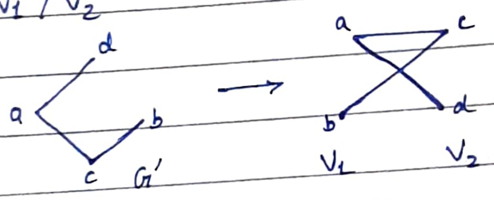
1. Biphartite Graph
→ A graph $G(V,E)$ with vertex partition $V = \{V_1, V_2\}$ is called a biphartite graph if every edge of a graph joins a vertex in V_1 to a vertex in V_2.

Ex.



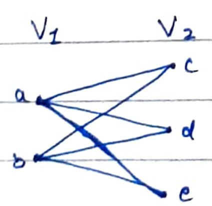
→ There should be no edge between any two vertices of V_1 / V_2.

Ex.



2. Complete Biphartite Graph
→ Every vertex in V_1 is adjacent to every other vertex in V_2.
→ G is a complete biphartite graph (Above 1st example).
→ It is denoted by $K_{m,n}$ [m = Number of vertices in V_1, n = Number of vertices in V_2)
→ $G = K_{2,2}$.

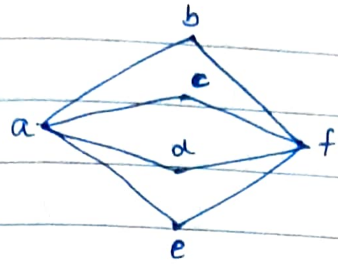
Ex. $K_{2,3}$:



→ Total number of vertices = m+n
Total number of edges = m×n

Q. Determine whether the following graphs are biphartite ex not:

a

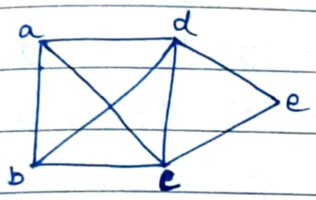


G

→ $V_1 = \{a, f\}$; $V_2 = \{b, c, d, e\}$

∴ Complete biphartite graph.

b.



G

→ Not a biphartite graph.

Given graph is not a biphartite graph as we cannot divide the vertices of G into two sets, V_1 and V_2, such that ~~ev~~ every edge of the graph joins vertex of V_1 to vertex of V_2.

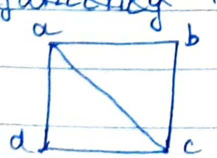
☆ Isomorphic Graph

→ Two graphs G_1 and G_2 are said to be isomorphic if there exists one-to-one correspondance between vertices and edges, such that incidences are preserved.

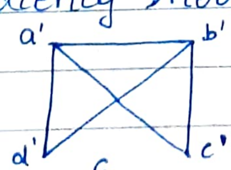
→ Conditions to check if two graphs are isomorphic:

- Number of vertices should be equal in both graphs G_1 and G_2.
- Number of edges should be equal in both graphs G_1 and G_2.
- Degree sequence of both the graphs G_1 and G_2 must be same.
- ~~Adjacency~~ Adjacency should be preserved.

Ex.



G_1



G_2

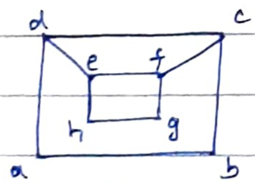
- Number of edges: ~~$G_1 = 4$, $G_2 = 4$~~ $G_1 = 5$, $G_2 = 5$
- Number of vertices: $G_1 = 4$, $G_2 = 4$.
- Degree sequence: $G_1 = (d, b = 2)(a, c = 3)$ 2, 2, 3, 3.
  $G_2 = (a', b' = 3)(c', d' = 2)$ 2, 2, 3, 3.
- Adjacency: G_1: a is connected to b (2), c (3), d (2) = 2, 2, 3
  b is connected to a (3), c (3) = 3, 3
  c is connected to a (3), b (2), d (2) = 2, 2, 3.
  d is connected to a (3), c (3) = 3, 3
  G_2: a' is connected to b' (3), c' (2), d' (2) = 2, 2, 3
  b' is connected to a' (3), c' (2), d' (2) = 2, 2, 3
  c' is connected to b' (3), a' (3), ~~d' (2)~~ = 3, 3
  d' is connected to a' (3), b' (3) = 3, 3

$a \to a'$ (2, 2, 3)
$b \to d'$ (3, 3)
$c \to b'$ (2, 2, 3)
$d \to c'$ (3, 3)

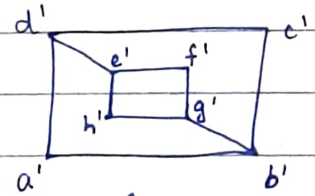
G_1 G_2

$\therefore$ G_1 and G_2 are isomorphic.

Q.



G_1



G_2

Are they isomorphic?

→ ~~E~~ Vertices: $G_1 = 8$, $G_2 = 8$

Edges: $G_1 = 6$, $G_2 = 6$

Degree sequence: $G_1 = (c, d, e, f = 3)(a, b, h, g = 2) = 2, 2, 2, 2, 3, 3, 3, 3$
$G_2 = (a', c', f', h' = 2)(b', d', e', g' = 3) = 2, 2, 2, 2, 3, 3, 3, 3$

Adjacency: G_1: a = 2, 3 ; b = 2, 3 ; c = 2, 3, 3 ; d = 2, 3, 3
e = 2, 3, 3 ; f = 2, 3, 3 ; g = 2, 3 ; h = 2, 3
G_2: a' = 3, 3 ; b' = 2, 2, 3 ; c' = 3, 3 ; d' = 2, 2, 3 ;
e' = 2, 2, 3 ; f' = 3, 3 ; g' = 2, 2, 3 ; h' = 3, 3

Consider a vertex of degree 3 in G_1 (d). Its adjacent vertices are having degree 2, 3, 3. Consider the same vertex of degree 3 in G_2 (d') and its adjacency vertices are having degree 2, 2, 3.

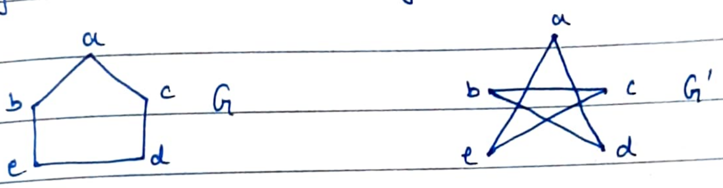
Hence, G_1 is not ~~isome~~ isomorphic to G_2.

$\therefore$ $G_1 \not\cong G_2$.

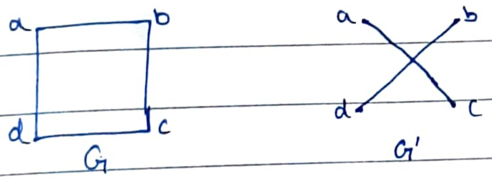
★ Complement of a Graph

→ The complement of a graph G is a graph G' on the same set of vertices as of G, such that there will be an edge between two vertices (v, e) in G', if and only if there is no edge in between (v, e) in G.

Ex.

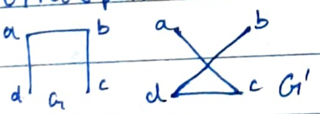


Ex.

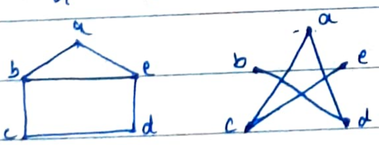


→ Self-complementary graph is a graph which is isomorphic to its complement.

Ex.



Ex.

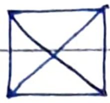
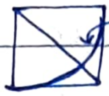
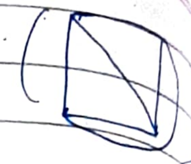


(Not self-complementary)

★ Planar Graphs

→ A graph is said to be planar if it can be drawn on a plane in such a way that no two edges cross one another, except at common vertices.

→ Region: A region of a planar graph is defined to be an area of the plane that is bounded by edges and is not further divided into subareas.

Ex.  →  Passing over the other diagonal. 

(Planar graph example)

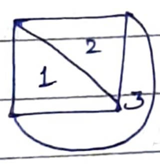
→ Theorem: For any connected planar graph:

- $v-e+r=2$ [Proof by M.I. (Euler's formula)] (r = region)
- $e <= 3v-6$ provided no self-loops and $e >= 2$.

Ex. $K_{2,3}$:

$e=6$; $v=5$ ∴ Planar.

Ex. (Region)

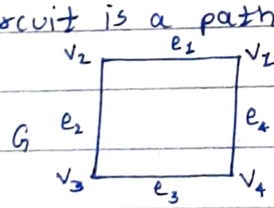


$v-e+r = 4-6+4$
$= 2$

☆ Path and Circuit

→ Path is a route along edges that start at a vertex and end at a vertex.

→ Circuit is a path that begins and ends at the same vertex.

Ex. 

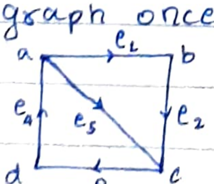
Path: $v_1 v_2 v_3 v_4$ OR $e_1 e_2 e_3 e_4$

Circuit: $v_1 v_2 v_3 v_4 v_1$ OR $e_1 e_2 e_3 e_4$

1. Eulerian Path and Circuit.

→ Eulerian Path is a path that traverses each edge in the graph once and only once.

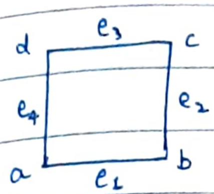
→ Eulerian Circuit is a circuit that traverses each edge in the graph once and only one.

Ex. 

Eulerian Path: $e_1 e_2 e_3 e_4 e_5$

Eulerian Circuit: $\underset{e_1}{a}\underset{e_2}{b}\underset{e_3}{c}\underset{e_4}{d}\underset{e_5}{a}\underset{e_3}{c}\underset{e_4}{d}a$ (Hence, EC is not possible as edges are being repeated).

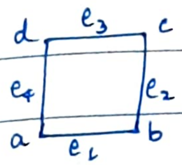
Date.

Ex. 

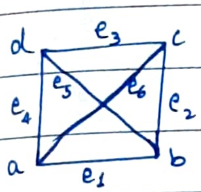
Eulerian path: $e_1 e_2 e_3 e_4$

Eulerian circuit: abcda

→ Theorem 1: An undirected graph possesses Eulerian path if and only if it is connected and has either zero or two vertices of odd degree.

Ex. G 

EP: e_1, e_2, e_3, e_4 (Possible)

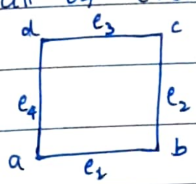
Ex. G' 

4 vertices have a degree of 3.

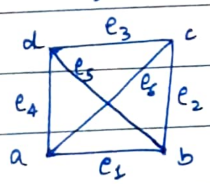
EP is not possible (edges repeats)

($e_1 e_2 \underline{e_3} e_4 e_6 \underline{e_3} e_5$)

→ ~~Vertise~~ Theorem 2: Undirected graph possesses Eulerian circuit if and only if it is connected and its vertices are all of even degree.

Ex. G 

EC: abcda (Possible)

Ex. G' 

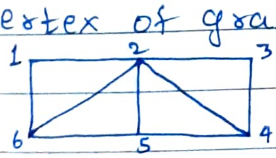
All vertices are of odd degree (3).

EC is not possible.

2. Hamiltonian Path and Circuit:

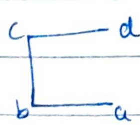
→ Hamiltonian path is a path that passes through each of the vertices in a graph once and only once.

→ Hamiltonian circuit is a circuit that passes through each vertex of graph once and only once.

Ex. G 

Hamiltonian path: 1-2-3-4-5-6

Hamiltonian circuit: 1-2-3-4-5-6-1

Ex. 

G'

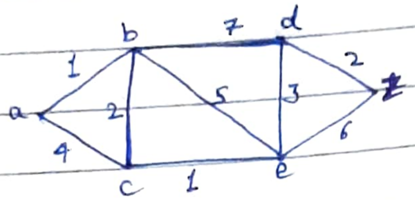
Hamiltonian path: abcd

Hamiltonian circuit: abcdcba (Hence, not possible as edges/vertices repeat).

✯ Dijktra's shortest Path Algorithm (Gives shortest path value)

Q. Find the shortest path from a to z.

a:



→ Step 1: $P = \{\phi\}$, $T = \{a, b, c, d, e, z\}$.

$l(a) = 0$, $l(x) = \infty$ $\forall x \in T$, $x \neq a$.

Step 2: $P = \{a\}$, $T = \{b, c, d, e, z\}$.

$l(b) = \min\{\text{old } l(b), l(a) + w(a,b)\}$

$= \min\{\infty, 0+1\} = \min\{\infty, 1\}$

$l(b) = 1$

$l(c) = \min\{\text{old } l(c), l(c) + w(a,c)\}$

$= \min\{\infty, 0+4\} = \min\{\infty, 4\}$

$l(c) = 4$

$l(d) = \min\{\text{old } l(d), l(d) + w(a,d)\}$

$= \min\{\infty, 0+\infty\}$

$l(d) = \infty$

$l(e) = \min\{\text{old } l(e), l(e) + w(a,e)\}$

$= \min\{\infty, 0+\infty\}$

$l(e) = \infty$

$l(z) = \min\{\text{old } l(z), l(z) + w(a,z)\}$

$= \min\{\infty, 0+\infty\}$

$l(z) = \infty$

Step 3: $P = \{a, b\}$, $T = \{c, d, e, z\}$

$l(c) = \min\{\text{old } l(c), l(b) + w(b,c)\}$ (refers (previous) old values)

$= \min\{4, 1+2\} = \min\{4, 3\}$

$l(c) = \cancel{\infty}\ 3$

$l(d) = \min\{\text{old } l(d), l(b) + w(b,d)\}$

$= \min\{\infty, 1+7\} = \min\{\infty, 8\}$

$l(d) = 8$

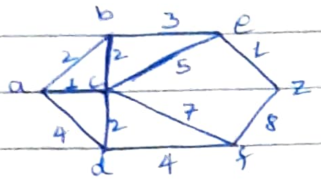
$l(e) = \min\{\text{old } l(e),\ l(b) + w(b,e)\}$
$= \min\{\infty, 1+5\} = \min\{\infty, 6\}$
$l(e) = 6$
$l(z) = \min\{\text{old } l(z),\ l(b) + w(b,z)\}$
$= \min\{\infty, 1+\infty\}$
$l(z) = \infty$

Step 4 : $P = \{a, b, c\}$, $T = \{d, e, z\}$
$l(d) = \min\{\text{old } l(d),\ l(c) + w(c,d)\}$
$= \min\{8, 3+\infty\}$
$l(d) = 8$
$l(e) = \min\{\text{old } l(e),\ l(c) + w(c,e)\}$
$= \min\{6, 1+3\} = \min\{6, 4\}$
$l(e) = 4$
$l(z) = \min\{\text{old } l(z),\ l(c) + w(c,z)\}$
$= \min\{\infty, 3+\infty\}$
$l(z) = \infty$

Step 5: $P = \{a, b, c, e\}$, $T = \{d, z\}$
$l(d) = \min\{\text{old } l(d),\ l(e) + w(e,d)\}$
$= \min\{8, 4+3\} = \min\{8, 7\}$
$l(d) = 7$
$l(z) = \min\{\text{old } l(z),\ l(e) + w(e,z)\}$
$= \min\{\infty, 4+6\} = \min\{\infty, 10\}$
$l(z) = 10$

Step 6: $P = \{a, b, c, e, d\}$, $T = \{z\}$
$l(z) = \min\{\text{old } l(z),\ l(d) + w(d,z)\}$
$= \min\{10, 7+2\} = \min\{10, 9\}$
$l(z) = 9$
$P = \{a, b, c, e, d, z\}$.

b.



→ Step 1: $P = \{\phi\}$, $T = \{a, b, c, d, e, f, z\}$

$l(a) = 0$, $l(x) = \infty \ \forall x \in T, x \neq a$

Step 2: $P = \{a\}$, $T = \{b, c, d, e, f, z\}$

$l(b) = \min\{\text{old } l(b), l(a) + w(a,b)\}$

$= \min\{\infty, 0+2\} = \min\{\infty, 2\}$

$l(b) = 2$

$l(c) = \min\{\text{old } l(c), l(a) + w(a,c)\}$

$= \min\{\infty, 0+1\} = \min\{\infty, 1\}$

$l(c) = 1$

$l(d) = \min\{\text{old } l(d), l(a) + w(a,d)\}$

$= \min\{\infty, 0+4\} = \min\{\infty, 4\}$

$l(d) = 4$

$l(e) = \min\{\text{old } l(e), l(a) + w(a,e)\}$

$= \min\{\infty, 0+\infty\}$

$l(e) = \infty$

$l(f) = \min\{\text{old } l(f), l(a) + w(a,f)\}$

$= \min\{\infty, \infty + 0\}$

$l(f) = \infty$

$l(z) = \min\{\text{old } l(z), l(a) + w(a,z)\}$

$= \min\{\infty, \infty\}$

$l(z) = \infty$

Step 3: $P = \{a, c\}$, $T = \{b, d, e, f, z\}$

$l(b) = \min\{2, 1+2\} = \min\{2, 3\}$

$l(b) = 2$

$l(d) = \min\{4, 1+2\} = \min\{4, 3\}$

$l(d) = 3$

$l(e) = \min\{\infty, 1+5\} = \min\{\infty, 6\}$

$l(e) = 6$

$l(f) = \min\{\infty, 1+7\} = \min\{\infty, 8\}$

$l(f) = 8$

$l(z) = \min\{\infty, 1+\infty\}$

$l(z) = \infty$

Step 4: $P = \{a, c, b\}$, $T = \{d, e, f, z\}$

$l(d) = \min\{3, 2+\infty\} = \min\{3, \infty\}$

$l(d) = 3$

$l(e) = \min\{6, 2+3\} = \min\{6, 5\}$

$l(e) = 5$

$l(f) = \min\{8, 2+\infty\} = \min\{8, \infty\}$

$l(f) = 8$

$l(z) = \min\{\infty, 2+\infty\}$

$l(z) = \infty$

Step 5: $P = \{a, c, b, d\}$, $T = \{e, f, z\}$

$l(e) = \min\{5, 3+\infty\} = \min\{5, \infty\}$

$l(e) = 5$

$l(f) = \min\{8, 3+4\} = \min\{8, 7\}$

$l(f) = 7$

$l(z) = \min\{\infty, 3+\infty\}$

$l(z) = \infty$

Step 6: $P = \{a, c, b, d, e\}$, $T = \{f, z\}$

$l(f) = \min\{7, 5+\infty\} = \min\{7, \infty\}$

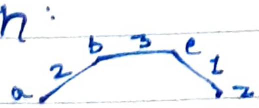
$l(f) = 7$

$l(z) = \min\{\infty, 5+1\} = \min\{\infty, 6\}$

$\therefore$ $l(z) = 6$

$\therefore$ $P = \{a, c, b, d, e, z\}$.

Path:



Date:

★ Trees

→ A connected (undirected) graph that ~~cir~~ contains no circuits

→ Collection of disjoint trees is a forest.

→ Properties:

1. There exists a unique path between any two vertices.
2. $v - e + r = 2 \quad r = 1$

   $\Rightarrow \boxed{v - 1 = e}$
3. A tree with two or more nodes has at least two leaves.

→ Rooted Tree: Tree starting from a root node (incoming degree = 0). Other vertices having incoming degree 1.

→ Path Length: Number of edges from the root node to the given vertex.

→ Height: Maximum of the path lengths in a tree.

m-ary tree of height h has m^h leaves.

→ M-ary Tree: Every internal node has at most 'm' sons.

→ Binary Tree: Every internal node has at most 2 sons.

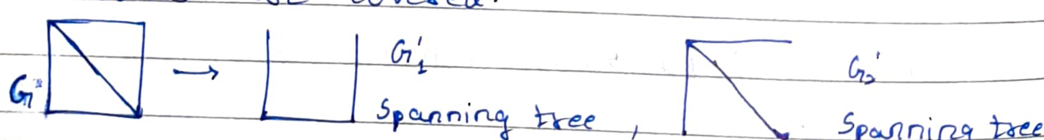
→ Full Binary Tree: Every internal node has exactly 2 sons

★ Spanning Tree and Minimum Spanning Tree

→ Spanning tree: A spanning subgraph of a connected graph G, which is a tree is called a spanning tree.

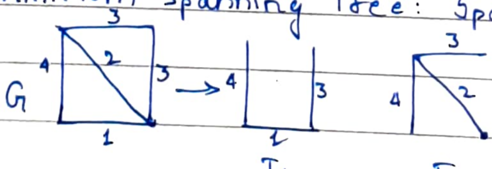
→ Spanning subgraph: Number of vertices are equal and all vertices are covered.

Ex.



→ Minimum Spanning Tree: Spanning tree with the least cost.

Ex.



$T_1 = 4 + 3 + 1 = 8$; $T_2 = 4 + 3 + 2 = 9$

T_1 is the minimum spanning tree.

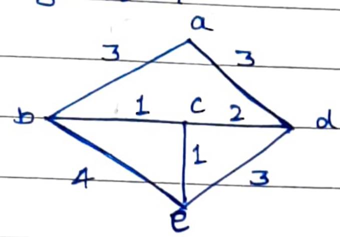
Prim's and Kruskal's Algorithm to find MST.

Q. Find the minimum cost spanning tree for the following graph using:

a. Prim's algorithm.

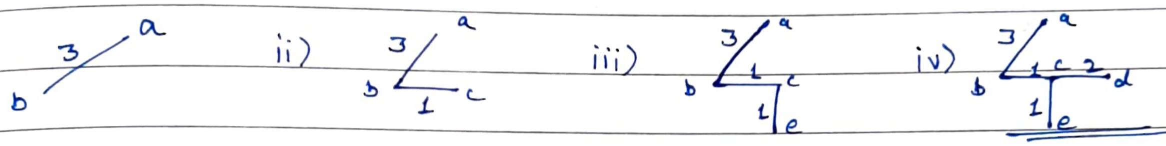
b. Kruskal's algorithm.

1.



a.

→ i) ii) iii) iv)



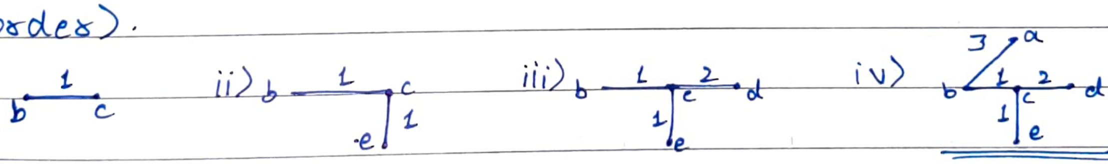
∴ Cost $= 3+1+1+2 = \underline{7}$

Edges $=$ Vertices $- 1 = 5-1 = \underline{4}$

b.

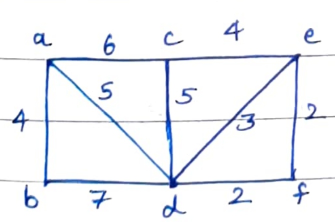
→ Select edge with the minimum cost and add to the spanning tree. When n vertices with (n−1) edges are covered, stop the procedure. (Cost in ascending order).

i) ii) iii) iv)



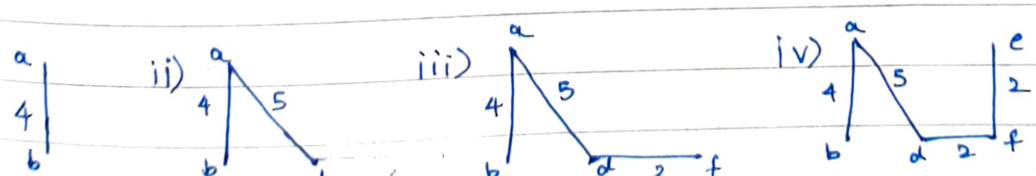
Cost $= 3+1+1+2 = \underline{7}$

2.

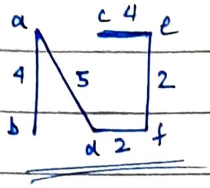


a.

→ i) ii) iii) iv)



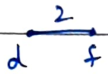
v)



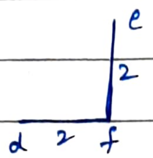
Cost = 4+5+4+2+2 = **17**.

b.

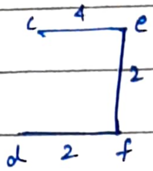
→ i)



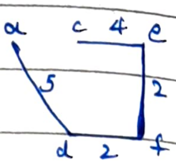
ii)



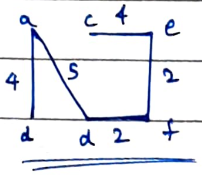
iii)



iv)



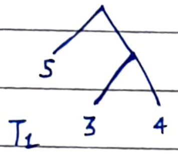
v)

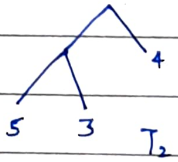


Cost = ~~4+~~ 2+2+4+5+4 = **17**

✶ Optimal Tree

→





$W(T_1) = 5\times1 + 3\times2 + 4\times2 = 5+6+8$

$W(T_1) = 19$

$W(T_2) = 5\times2 + 3\times2 + 4\times1 = 10+6+4$

$W(T_2) = 20$

$W(T_1) < W(T_2)$

∴ T_1 is the optimal tree.

→ $W(T) = \sum_{i=1}^{n} W_i P_i$ (W_i = Weight, P_i = Path length)

Q. Construct an optimal tree for the following weights using Huffman algorithm: 8, 9, 10, 11, 13, 15, 22.

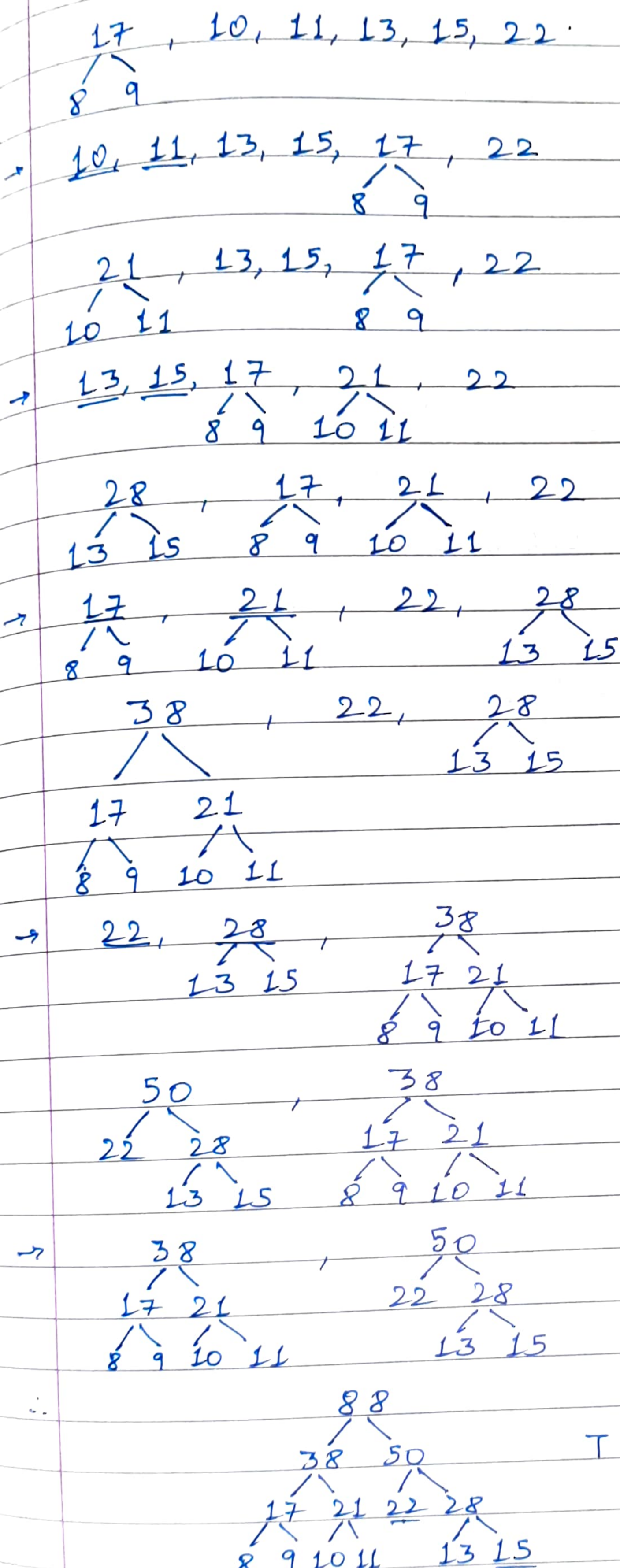
→ ✶ **Huffman Algorithm**

1. Arrange the given weights in increasing order.
2. Select two minimum weights and add them.
3. Repeat steps 1. and 2. until you get a root node.

→ <u>8, 9</u>, 10, 11, 13, 15, 22

8+9 = 17

Date:



$W(T) = (8\times3)+(9\times3)+(10\times3)+(11\times3)+(22\times2)+(13\times3)+(15\times3)$

$\therefore\ W(T) = 242$

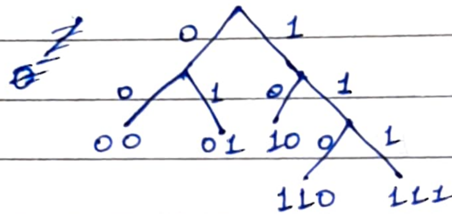
☆ Prefix Code

→ ~~Set 1~~ No code in the sequence should be prefix of another code in the set.

Ex. Set : {00, 01, 11, 101}

→ Left = 0 ; Right = 1.

Ex.



Prefix code: {00, 01, 10, 110, 111}

Ex. set : {00, 001, 110, 111}

—— x ——