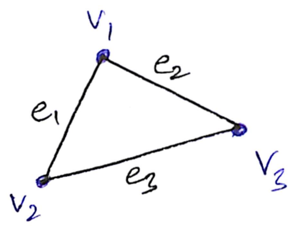
UNIT-4

Graph theory

* Graph:

Graph G is a pair of vertices and edges i.e., G(V, E)



- The no. of vertices in a graph is called order of the graph. $|V(G)|$
- The no. of edges in a graph is called size of the graph. $|E(G)|$

So, In above diagram,

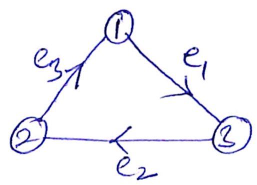
Order is 3, size is 3.

* types of graphs:

(i) Directed graph:

Each edge of the graph has a direction, such edge is called as directed edge & that graph is called as directed graph.

Eg.

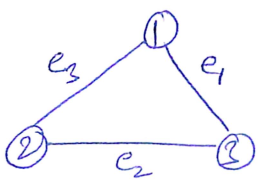


$e_1 = (1,3)$
$e_2 = (3,2)$
$e_3 = (2,1)$

(ii) Undirected graph:

It a graph 'G' consists of set of V vertices & E edges such that each edge is associated with an unordered pair of vertices.

Eg:

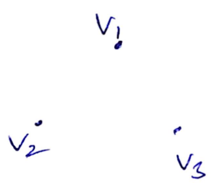


$e_1 = \{1,3\}$ or $\{3,1\}$
$e_2 = \{2,3\}$ or $\{3,2\}$
$e_3 = \{2,1\}$ (or) $\{1,2\}$

(iii) Null graph:

- A graph in which number of edges is zero

Eg:

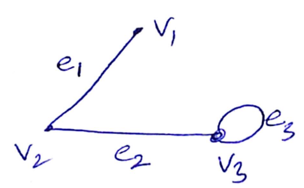


Here, we have 3 vertices with '0' edge. So It is null graph.

(iv) Self loop:

- An edge joining a vertex to itself is called self loop.

Eg:



Here, e_3 edge is joining with same vertex v_3.

(v) Finite & Infinite graph:

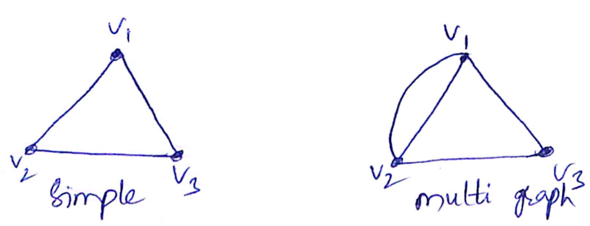
- A graph is finite, if both vertex set & edge set are finite.

Otherwise, that graph is infinite graph.

(vi) Simple graph:

- A graph which has no self loops & parallel edges is called as a Simple graph. Otherwise, it is a multigraph.

Eg:

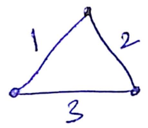


- Parallel edge means, In a graph, a single pair of vertices may have more than one edge.

(vii) Weighted graph:

- A graph in which weights are assigned to each edge or each vertex.

Eg:



1, 2, 3 are weights for the edges

*Basics of graph theory:

Initial vertex & terminal vertex:

Initial: A vertex which doesn't have incoming edge

Terminal: A vertex which doesn't have outgoing edge

Degree: is the no. of edges incident on the vertex.

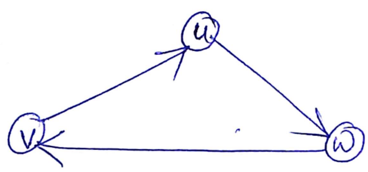
Indegree: ($d^-(v)$) or $id(v)$

The no. of edges for which v is the terminal vertex.

Out degree: ($d^+(v)$) or $od(v)$

The no of edges for which v is the initial vertex.

Eg:



$id(u) = d^-(u) = 1$ & $od(u) = d^+(u) = 1$

$id(v) = d^-(v) = 1$ $\quad d^+(v) = 1$

$d^-(w) = 1$ $\quad d^+(w) = 1$

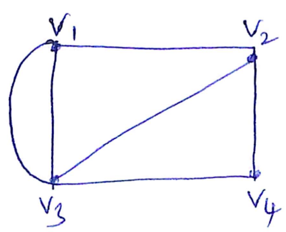
$\sum d^-(v_i) = 3$ $\quad \sum d^+(v_i) = 3$

This graph contains 3 vertices & 3 edges.

Sum of indegree & out degree is same. This is applicable for all digraphs.

Degree sequence: Degree of the vertices of a graph arranged in a non decreasing order is called degree sequence of the graph.

Eg:



$d(v_1) = 3$

$d(v_2) = 3$

$d(v_3) = 4$

$d(v_4) = 2$

degree of the graph is $= 2$ {i.e., each vertex has 2 min}

degree sequence is 2, 3, 3, 4

* A vertex of degree 1 is called "pendant vertex" & of degree '0' – "Isolated vertex"

* Hand shaking property:

The sum of the degree of all the vertices in a graph is an even number & that is equal to twice the no. of edges in the graph.

For $G(V,E)$, $\sum deg(V) = 2|E|$

Eg: In the previous graph, no. of edges $= 6$

sum of degrees $= 3+3+4+2 = 12$

$= 2 \times |E| = 2 \times 6$

* Theorem:

For any undirected graph or multigraph, the no of vertices of odd degree must be even.

(or)

An undirected graph has even number of vertices with odd degree.

Proof:

Consider a graph with 'n' vertices.

Suppose 'k' vertices are odd degree in these 'n' vertices. So, $\{v_1, v_2, v_3 --- v_k\}$

So, 'n-k' " " even degree $\{v_{k+1}, v_{k+2}, --- v_n\}$

Now, sum of the degrees of the vertices is,

$$\sum_{i=1}^{n} d(v_i) = \sum_{i=1}^{k} d(v_i) + \sum_{i=k+1}^{n} d(v_i) \quad \text{——} ①$$

According to handshaking theorem,

sum of degree of all vertices is even & equal to twice the no of edges

So, Left hand side is even

In right hand side second part is also sum of even degree

So, the first part of the right hand side also even

$$\sum_{i=1}^{k} d(v_i) = d(v_1) + d(v_2) + --- d(v_k) = \text{even} \quad (2)$$

but $d(v_1), d(v_2) -- d(v_k)$ is odd

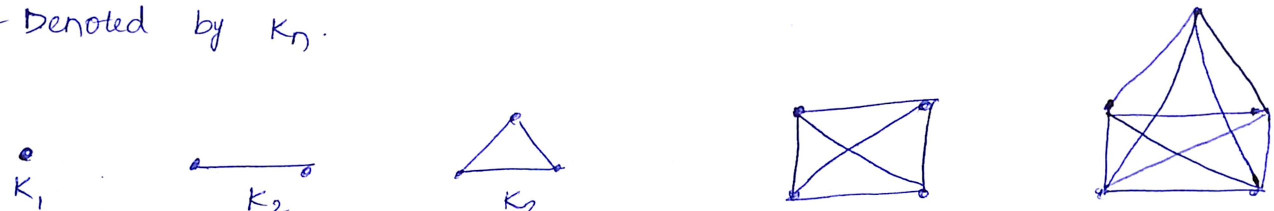
∴ ~~d(v)~~

∴ In eq (2), the no of terms in the left hand side must be even

i.e., k is even.

* Complete graph:

- A simple graph of (order ≥ 2) that contains exactly one edge between each pair of distinct vertices.
- Denoted by K_n.



K_1 K_2 K_3 K_4 K_5

- In Complete graph (K_n), we have $\boxed{\frac{n(n-1)}{2}}$ no of edges.

In K_6 we will have, $\frac{6(6-1)}{2} = 3(5) = 15$ edges.

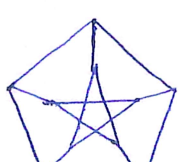
* Regular graph:

A graph in which all the vertices of the same degree k is called k-regular graph.

$$\text{No. of edges} = \frac{\text{No of vertices} * \text{degree}}{2}$$

* 3 regular graphs are called as cubic graphs

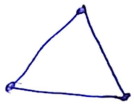
* In cubic graph, if it has 10 vertices & 15 edges is called "petersen graph".

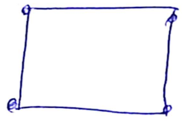


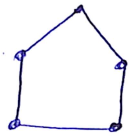
petersen graph

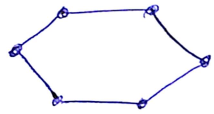
* **Cycle graph:** C_n, $n \geq 3$ consists of 'n' vertices $v_1, v_2, v_3 \ldots v_n$ & edges $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \ldots \{v_{n-1}, v_n\}, \{v_n, v_1\}$

Eg:


C_3

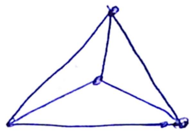

C_4

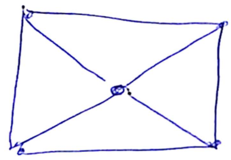

C_5


C_6

* **Wheel graph:** (W_n), obtained by adding additional vertex to C_n & connecting all vertices to this new vertex by new edges.

Eg:


W_3

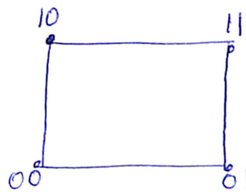

W_4

* **N-cubes:** (Q_n), vertices represented by 2^n bit strings of length n.

Two vertices are adjacent if and only if the bit strings that they represent differ by exactly one bit positions.

Eg:

Q_1

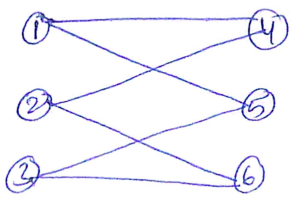


Q_2

* **Bipartite graph:**

A graph $G = (V, E)$ is called bipartite if $V = V_1 \cup V_2$ with $V_1 \cap V_2 = \emptyset$ & every edge of G is of the form $\{a, b\}$ with $a \in V_1$ & $b \in V_2$ & if each vertex in V_1 is joined with every vertex in V_2, then its complete bipartite &

Eg:

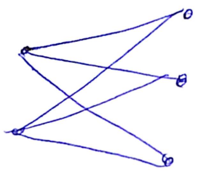


* Complete bipartite graph: $K_{m,n}$

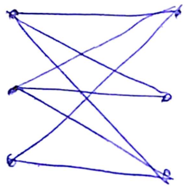
No. of vertices in $K_{m,n}$ is $m+n$

edges $= m * n$

Eg:



$K_{2,3}$



$K_{3,3}$

Eg: If the graph $K_{m,6}$ has 24 edges, what is m?

sol: $m * 6 = 24 \Rightarrow m = 4$

Eg: Can there be a graph contains $deg(A)=2$, $deg(B)=3$, $deg(C)=2$?

sol: NO, because sum of degrees of vertices should be even.

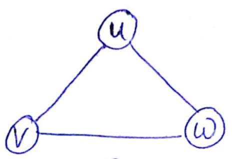
But here we got it as odd.

$deg(A)+deg(B)+deg(C) = 2+3+2 = 4+3 = 7.$ (odd)

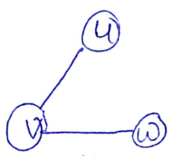
* Sub graph:

A Sub graph of a graph $G=(V,E)$ is a graph $H=(V',E')$ where V' is a subset of V & E' is a subset of E. & $V' \subseteq V \neq \emptyset$

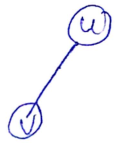
Eg:



G



H_1

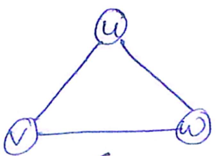


H_2

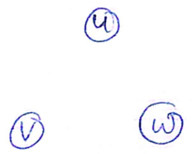
* Spanning sub graph:

In A graph $G=(V,E)$, Let $G_1=(V_1,E_1)$ be a sub graph of G, if $V_1 \in V$ & $E_1 \in E$ then G is spanning sub graph.

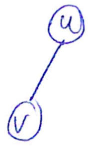
we need not to keep the edges, but it should have vertices.



G



G_1 (SSG)

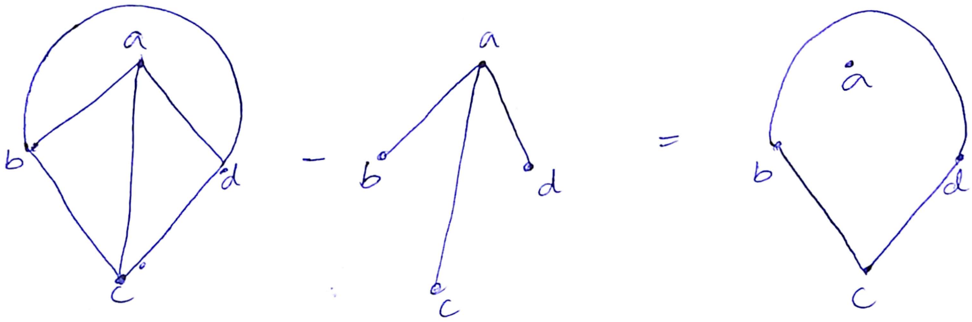


G_2 (not Spanning sub graph)

*** Complement graph:** Let G be a loop free undirected graph on 'n' vertices. The complement of G is ($\bar{G}$) is the subgraph of K_n consists of n vertices in G & all edges that are not in G.

$\bar{G} = (\bar{V}, \bar{E})$; $\bar{V} = V_{K_n}$ & $\bar{E} = E_{K_n} - E$

K_4 =

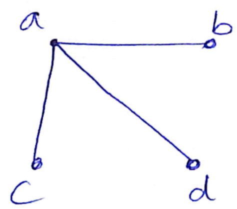


G　　　G_1　　　$\bar{G} - G_1$

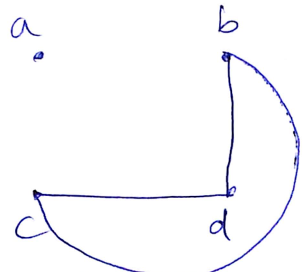
$\bar{G}$ = K_n total edges ${}^{V}C_2$ − E

– $G = K_n$; then $\bar{G}$ is a graph consisting of n vertices & no edges. Such graph is a null graph.

Eg: (1)

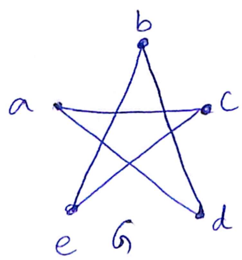


G

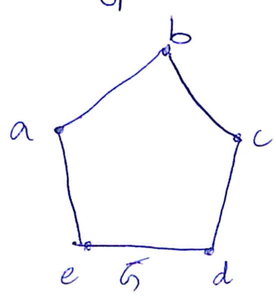


$\bar{G}$

(2)

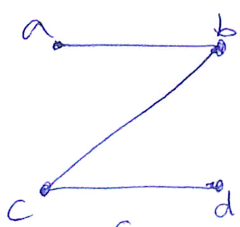


G

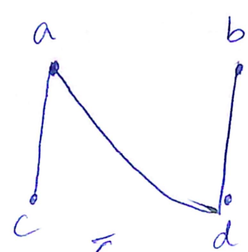


$\bar{G}$

(3)



G



$\bar{G}$

* Graph Isomorphism:

Let $G_1 = (V_1, E_1)$ & $G_2 = (V_2, E_2)$ be two undirected graphs. A function $f: V_1 \to V_2$ is called a graph isomorphism. If

i) f is one-to-one & onto and

ii) for all $a, b \in V_1$, $\{a, b\} \in E_1$ if and only if $\{f(a), f(b)\} \in E_2$.

When such a function exists G_1, G_2 are called isomorphic graphs.

(Or)

$G_1 = (V_1, E_1)$ & $G_2 = (V_2, E_2)$ are isomorphic if:

– There is a one-to one & onto function f from V_1 to V_2 with the property 'a' and 'b' are adjacent in G_1 if and only if $f(a)$ and $f(b)$ are adjacent in G_2, for all a and b in $V1$.

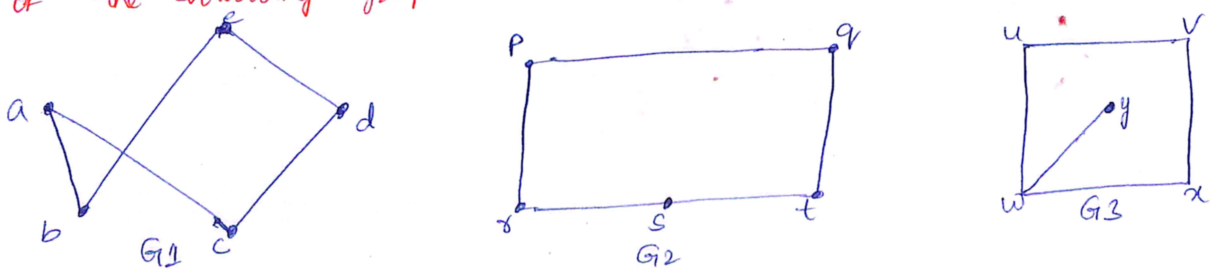
Then function f is called Isomorphism.

- When G_1, G_2 are isomorphic. we can write it as $G_1 \cong G_2$

We can say two graphs are isomorphic if they have

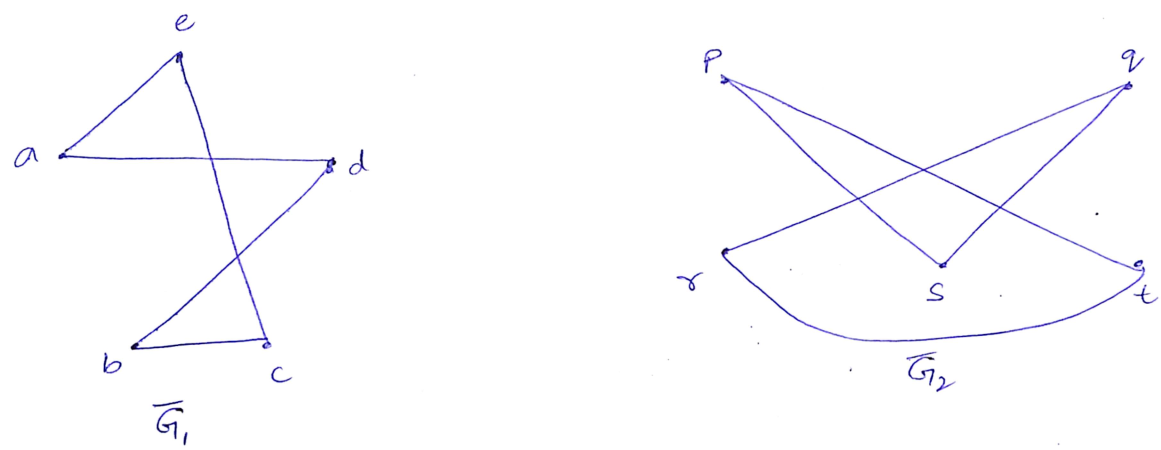
i) Equal no of vertices
ii) equal no of edges
iii) same degree sequence
iv) same no of circuit of particular length.

* Which of the following graphs are isomorphic?



Sol: In G3, w has degree 3 whereas remaining has 2 degree. So, G_3 is not isomorphic to G_1, G_2

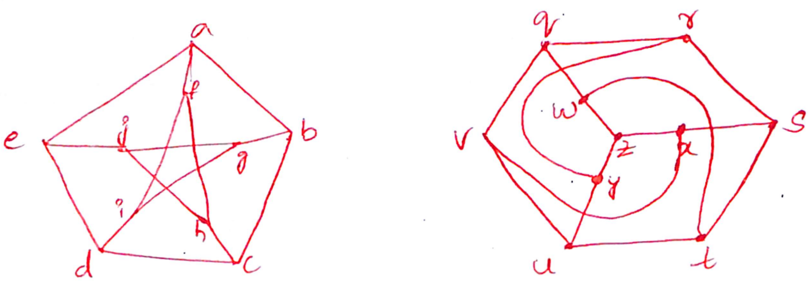
If we will take the complements for G_1, G_2 then,



Here, $\overline{G_1} \cong \overline{G_2}$ so $G_1 \cong G_2$

G_1, G_2 are isomorphic

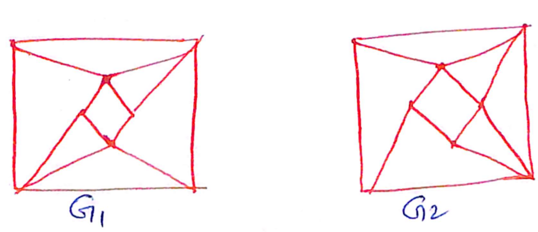
* show that given graphs are isomorphic?



Sol: let us consider a as q, $b \to v$, $e \to r$, $g \to x$, $i \to z$
$c \to u$, $d \to$ ~~x~~ y, $f \to w$, $h \to t$, $j \to s$

Edges are equal & adjacent vertices degree also equal.

* show that the following graphs are not isomorphic?



Sol: In G_1, It has pair vertices of degree 4 which are not adjacent
G_2, " " " " are adjacent

So, Not isomorphic

* Isomorphic graphs:

Two graphs G_1 and G_2 are said to be isomorphic $G_1 \equiv G_2$ if

i) $|V(G_1)| = |V(G_2)|$

ii) $|E(G_1)| = |E(G_2)|$

iii) Degree sequences of G_1 and G_2 are same.

(iv) If the vertices $\{v_1, v_2 \ldots v_k\}$ forms a cycle of length k in G_1, then the vertices $\{f(v_1), f(v_2), \ldots f(v_k)\}$ should form a cycle of length k in G_2.

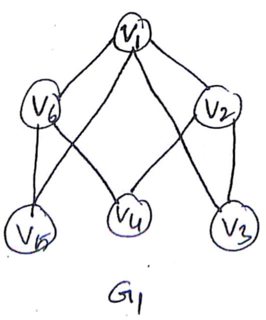
(v) Edge connectivity is retained. {∵ vertices & edges should be mapped}

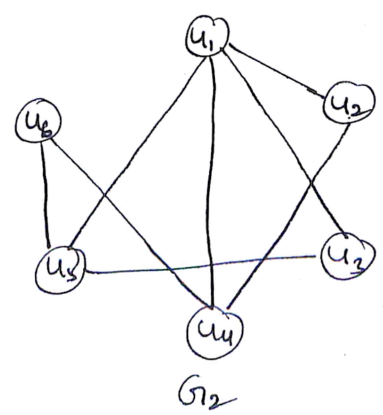
(vi) $G_1 \equiv G_2$ iff $\overline{G_1} \equiv \overline{G_2}$ where G_1 and G_2 are simple graph (∵ contains no self loops & no parallel edges).

(vii) $G_1 \equiv G_2$ if the adjacency matrices of G_1 and G_2 are same.

(viii) $G_1 \equiv G_2$ iff the corresponding subgraphs of G_1 and G_2 are isomorphic (∵ delete some vertex in G_1 and also image vertex in G_2).

* show that the following graphs are isomorphic.





Sol: i) No of vertices in $G_1 = 6$, $G_2 = 6$

ii) No of Edges in $G_1 = 8$, $G_2 = 8$

iii) Degree sequence in $G_1 = \{2, 2, 2, 3, 3, 4\}$

$G_2 = \{2, 2, 2, 3, 3, 4\}$

(iv) $V_1 = u_1$

$V_2 = u_4$

$V_3 = u_2$

$V_4 = u_6$

$V_5 = u_5$

$V_6 = u_3$

Now, we should map edges.

$\{V_1, V_2\} = \{u_1, u_4\}$ $\quad$ $\{V_1, V_5\} = \{u_1, u_3\}$ $\quad$ $\{V_1, V_6\} = \{u_4, u_5\}$

$\{V_1, V_3\} = \{u_1, u_2\}$ $\quad$ $\{V_4, V_2\} = \{u_6, u_4\}$ $\quad$ $\{V_4, V_6\} = \{u_6, u_5\}$

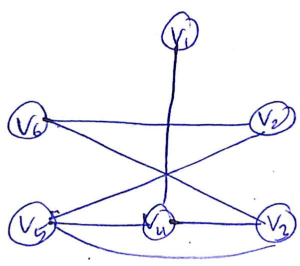
$\{V_5, V_6\} = \{u_5, u_3\}$ $\quad$ $\{V_2, V_3\} = \{u_4, u_2\}$

(v). No. of cycles in G_1 is 3 {one cycle with 4 vertices & 2 cycles with 3 vertices}

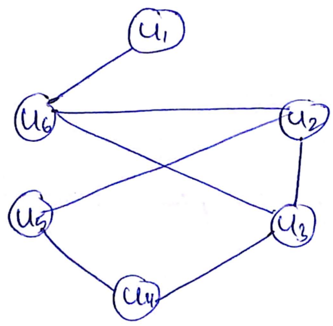
No. of cycles in G_2 is 2 {with 3 vertices}

So smallest cycle length is 3 in both G_1, G_2

(vi) Complement of G_1, G_2 are,



$\overline{G_1}$

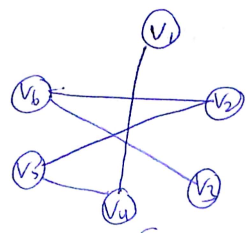


$\overline{G_2}$

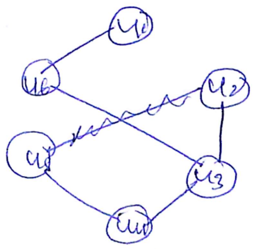
$\overline{G_1}$, $\overline{G_2}$ are simple graphs

(viii) sub graph same.

remove $\{V_3, V_5\}, \{V_3, V_4\}$



G_1



G_2

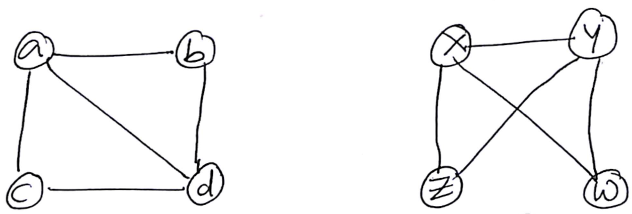
(vii) adjacency matrix

	V_1	V_2	V_3	V_4	V_5
V_1	-	1	1	∞	1
V_2	1	-	1	1	∞
V_3	1	1	-	∞	∞
V_4	∞	1	∞	-	∞
V_5	1	∞	∞	∞	-

for G_2 also same adjacency matrix

So, G_1, G_2 are isomorphic graphs

* Show that following graphs are isomorphic graphs. (6)



G_1 G_2

Sol: (i) No. of vertices in $G_1 = 4$, $G_2 = 4$

(ii) No. of Edges in $G_1 = 5$, $G_2 = 5$

(iii) Degree sequence in $G_1 = 2, 2, 3, 3$

$G_2 = 2, 2, 3, 3$

(iv) we should map the vertices in G_1 with G_2

$f(a) = x$ { ∵ both have degree 3 & connected to same edges }

$f(b) = z$ "

$f(c) = w$ "

$f(d) = y$ "

(v) Edges mapping

$ac = xw$ $bd = zy$

$ad = xy$ $dc = yw$

$ab = xz$

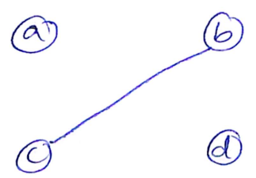
(vi) No. of cycles in G_1 is 2, with 3 vertices.

G_2 is 2

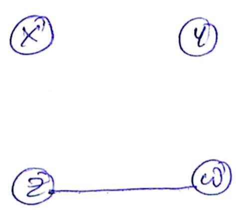
(vii) If we will do complement then, G_1, G_2 should be simple graph

That means, it will not have any self loops or parallel edges

$\overline{G_1}$ =



$\overline{G_2}$ =



(viii) G_1, G_2 ajacency matrices should be same.

G_1

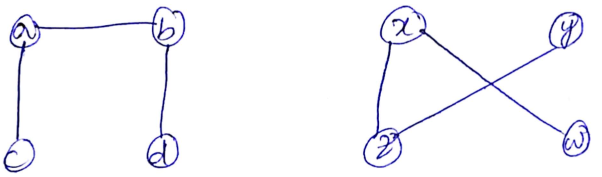
	a	b	c	d
a	–	1	1	1
b	1	–	∞	1
c	1	∞	–	1
d	1	1	1	–

G_2

	x	y	z	w
x	–	1	1	1
y	1	–	1	1
z	1	1	–	∞
w	1	1	∞	–

(ix) Sub graphs also should be isomorphic.

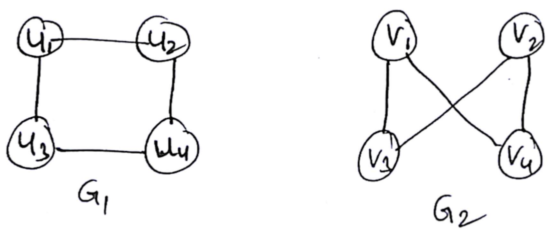
we should remove some edge in G_1 & same image in G_2



G_1

So, the given graphs are isomorphic graphs.

* Show that following graphs are isomorphic (or) not ?



Sol: Given graphs are isomorphic.

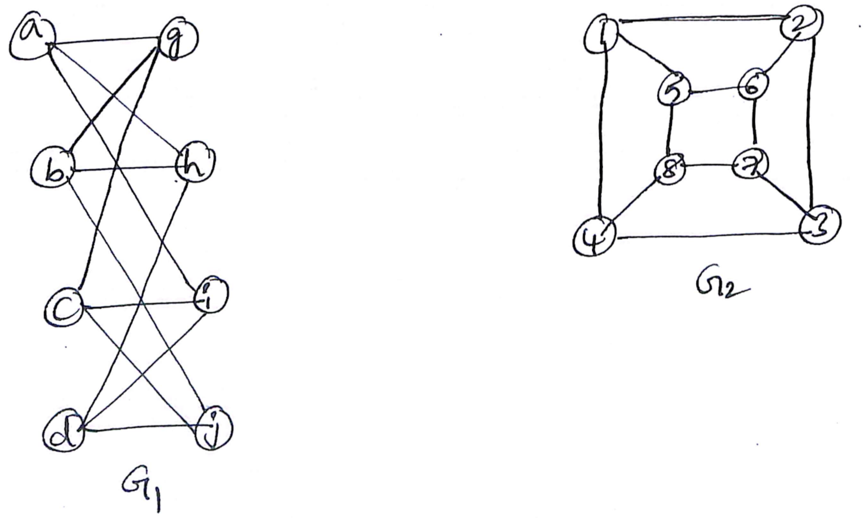
$f(u_1) = v_1$

$f(u_2) = v_4$

$f(u_3) = v_3$

$f(u_4) = v_2$

* Given graphs are isomorphic or not ?



Sol: G_1, G_2 are isomorphic

$f(a)=1$	$f(c)=8$	$f(g)=5$	$f(i)=4$
$f(b)=6$	$f(d)=3$	$f(h)=2$	$f(j)=7$

* Walk:

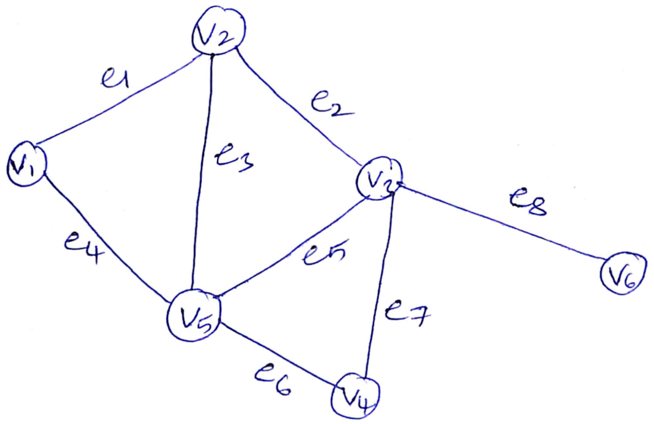
Consider a graph G having atleast one edge in G. Alternate sequence of vertices and edges of the form,

$$v_i\, e_j\, v_{i+1}\, e_{j+1}\, v_{i+2}\, e_{j+2} \cdots v_k\, e_m\, v_m$$

which begins & ends with the vertices

- Open walk is a walk in which starting & ending vertices should not be same. It should be different.
- Closed walk is a walk in which starting & ending vertices should be same.
- Length of the walk is the no. of edges present in that walk.

Eg:



Walk: $V_1 e_1 V_2 e_2 V_3 e_8 V_6$

Open walk: $V_1 e_1 V_2 e_3 V_5 e_6 V_4$

Closed walk: $V_1 e_4 V_5 e_3 V_2 e_1 V_1$

length of closed walk is 3.

* trail & circuit:

- If, in open walk, no edge appears more than once is called trail.
- If, in closed walk, no edge appears more than once is called circuit.

Eg: $V_1 e_1 V_2 e_2 V_3 e_8 V_6$ is a trail

$V_1 e_1 V_2 e_3 V_5 e_4 V_1 e_1 V_2$ is not a trail.

$v_1 e_1 v_2 e_3 v_5 e_4 v_1$ is a circuit

$v_2 e_2 v_3 e_7 v_4 e_6 v_5 e_3 v_2 e_2 v_3 e_5 v_5 e_3 v_2$ is not a circuit as e_3 repeating twice

* path & cycle:

- path is an open walk in which neither vertex nor edge appears more than once.

- cycle is closed walk in which neither vertex nor edge appears more than once except starting & ending vertices.

- Every path is a trail, but trail is not a path always.

- Every cycle is a circuit, but circuit need not be a cycle.

* A circuit is a path begins & ends at the same vertex & no edge appears more than once.

* cycle is a circuit that doesn't repeat vertices except first & last vertices.

Eg: $v_1 e_1 v_2 e_3 v_5 e_5 v_3 e_7 v_4$ is a path

$v_1 e_4 v_5 e_3 v_2 e_2 v_3 e_5 v_5 e_6 v_4$ is not a path {$\because$ v_5 repeated}

$v_2 e_2 v_3 e_5 v_5 e_3 v_2$ is a cycle

$v_2 e_1 v_1 e_4 v_5 e_5 v_3 e_4 v_4 e_6 v_5 e_3 v_2$ is not a cycle {$\because$ e_4, v_5 both repeated}

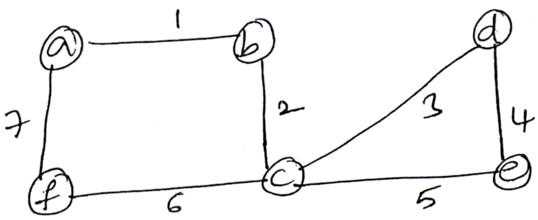
* Euler graph:

A graph G said to be Euler graph, if G contains Euler circuit.

- A graph G said to be semi euler graph, if G contains Euler trail.

- Euler graph contains Euler trail which includes every edge.
- Euler path uses every edge in the graph exactly once & starting, ending vertices are different.
- Euler circuit uses every edge exactly once & starting, ending vertices should be same.
- A graph G is Euler graph when all vertices are of even degree.
- A graph G is Eulerian. If its edge set can be decomposed into cycles

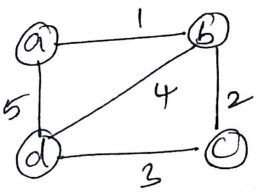
* Eg:



a1b2c3d4e5c6f7a

It is Euler circuit so, Graph is Euler

*

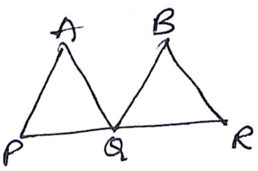


deg(a) = 2
d(b) = 3
d(c) = 2
d(d) = 3

so, degree of b, d = 3 (odd).

so, graph is non Euler.

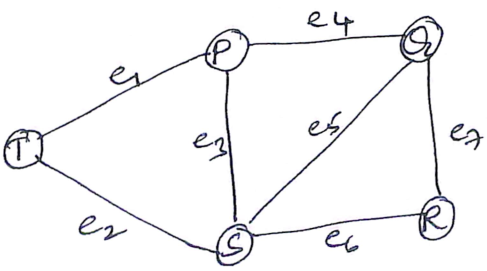
*



PAQBRQP

It is Euler circuit.

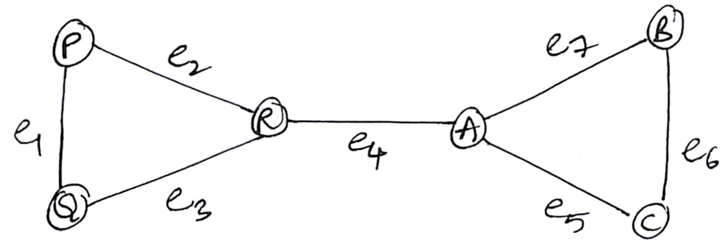
*



$Pe_1Te_2Se_3Pe_4Qe_5Se_6Re_7Q$

It is Euler trail.

*

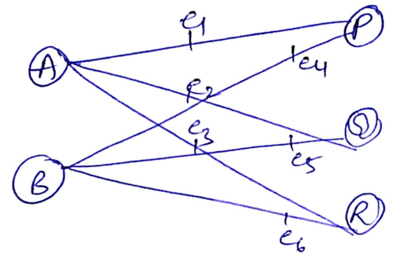


$P e_1 Q e_3 R e_4 A e_5 C e_6 B e_7 A$

Not Euler circuit

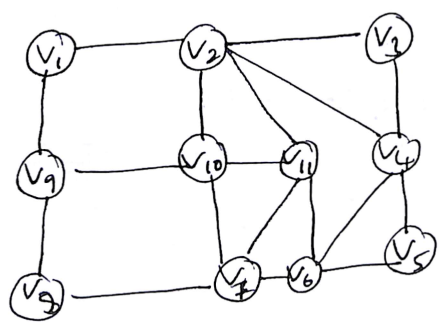
*. Show that complete Bipartite graph $K_{2,3}$ contains Euler trail.

Sol:



$A e_1 P e_4 B e_6 R e_3 A e_2 Q e_5 B$.

* Find Euler circuit ~~for~~ the given graph.



$V_1 V_2 V_{10} V_{11} V_7 V_{10} V_{11} V_6 V_4 V_2 V_3 V_4 V_5 V_6 V_7 V_8 V_9 V_1$

* Euler ~~f~~ormula:

$r = e - v + 2$

~~If~~ In a graph G, ~~If~~ we have v vertices, e edges, r regions.

* In a graph G, ~~If~~ we have 20 vertices of each degree 3, then find the number of regions?

Sol: By hand shaking theorem, $\sum \deg(v) = 2e$

$20(3) = 2e \Rightarrow e = 30$

$r = 2 - v + e$

$r = 2 - 20 + 30$

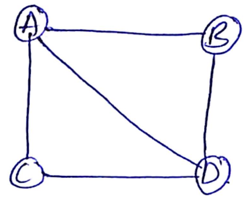
$r = 12$

* Hamiltonian cycles and paths:

Let G be a connected graph if there is a cycle in G that contains all vertices of G then that cycle is called Hamiltonian cycle.

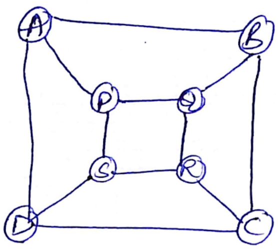
- A graph G contains hamiltonian cycle is called hamiltonian graph.

Eg:



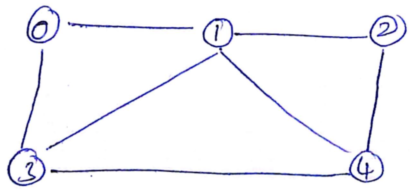
Here ABDCA is forming a cycle which included all vertices. So, it is hamiltonian cycle. the graph is hamiltonian graph.

Eg:



Here, ABCDSRQPA is a hamiltonian cycle. Because, it included all the vertices. It's not required to have all the edges.

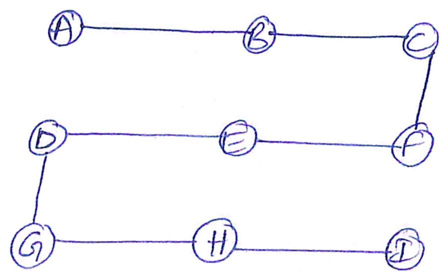
Eg:



Here, {0,1,2,4,3,0} is forming hamiltonian cycle.

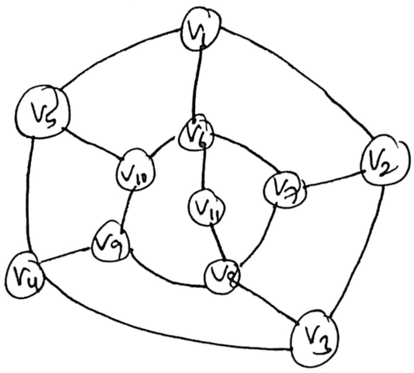
Eg: If G is a graph with the edges {A,B}, {B,C}, {C,F}, {F,E}, {E,D}, {D,G}, {G,H}, {H,I} yields a hamiltonian path. But, G have hamiltonian cycle ?

Sol:



It doesn't have hamiltonian cycles

* show that the following graph is hamiltonian.



Sol: $V_1 V_6 V_7 V_2 V_3 V_4 V_5 V_{10} V_9 V_8 V_{11}$ → hamiltonian path

$V_1 V_2 V_7 V_6 V_{11} V_8 V_3 V_4 V_9 V_{10} V_5 V_1$ → hamiltonian cycle.

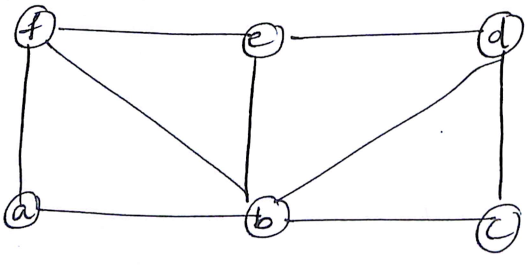
- As, It is having hamiltonian cycle, the given graph is hamiltonian.

* Hamiltonian path is an ~~undirected~~ path in undirected graph that visits each vertex exactly once. Not every edge.

* Hamiltonian cycle is a hamiltonian path such that there is an edge from last vertex to the first vertex of the hamiltonian path.

* Euler's circuit contains each edge of the graph exactly once.
* In hamiltonian cycle, some edges of the graph can be skipped.

Eg:



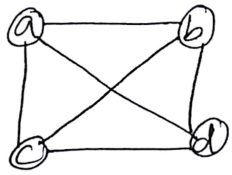
In this graph, hamiltonian cycle & path exists

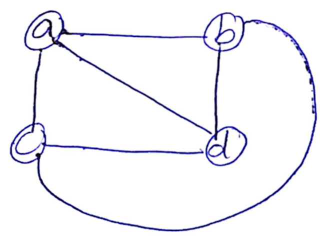
- But, Euler path & circuit doesn't exist. Because Euler path & circuit both uses every edge exactly once.

* planar graphs:

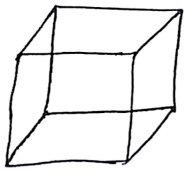
A planar graph is a graph which can be drawn on the plane without having any crossed edges.
Otherwise, It is non planar graph.

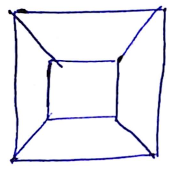
Eg:



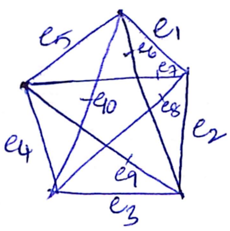


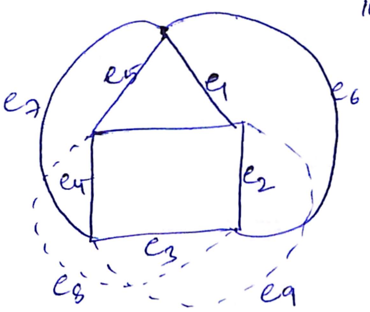
Eg:





Theorem: A complete graph of 5 vertices is non planar.
10 edges





$3v - e \geq 6$

$3(5) - 10 = 15 - 10 = 5 \geq 6$

false

so non planar

proof: As it is having crossing edges, it is non planar.

* properties of planar graphs:

(i) If connected planar graph G has e edges and r regions, then $r \leq \frac{2}{3}e$.

(ii) " " " has v vertices, then $v - e + r = 2$.

(iii) $3v - e \geq 6$

(iv). A complete graph K_n is a planar if and only if $n < 5$.

(v). A complete bipartite graph K_{mn} is planar if and only if $m < 3$ or $n > 3$.

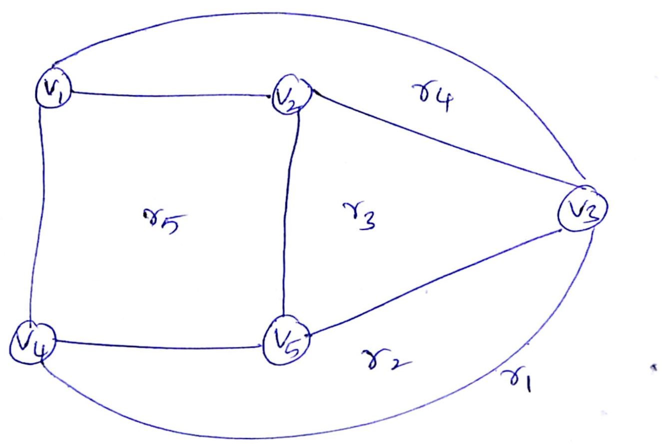
* Region of a graph:

A region is defined to be an area of the plane that is bounded by edges and can't be further subdivided. A planar graph divides the plane into one or more regions.

Finite region:

- If the area of the region is finite.
- If the area of the region is infinite, then infinite region.

Eg:



total regions r_1, r_2, r_3, r_4, r_5

finite regions r_2, r_3, r_4, r_5

infinite regions : r_1

* prove that Complete graph K_4 is planar.

Sol: K_4 contains 4 vertices & 6 edges

$3v - e \geq 6 \Rightarrow 3(4) - 6 = 12 - 6 = 6 \geq 6$ ✓

So, K_4 is planar graph.

* Graph Coloring:

For a given graph G, If we assign colors to its vertices in such a way that no two adjacent vertices have the same color.

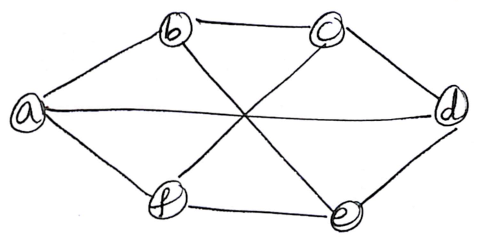
Chromatic number:

The min no. of colors required for coloring of the graph.

It is denoted by $\chi(G)$.

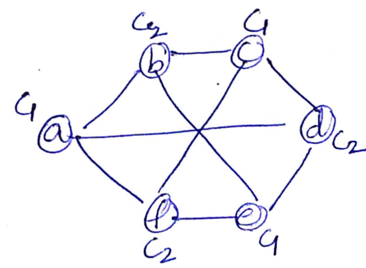
- A graph G is said to be K-colorable, if we can properly color it with K no. of colors.
- 'K' chromatic graph is a graph that can be properly colored with K-colors but not less than K-colors.

Eg: Find the chromatic no. for the following graph?



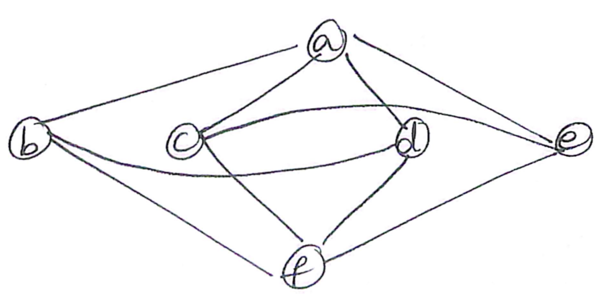
Sol:

Vertices	a	b	c	d	e	f
colors	C_1	C_2	C_1	C_2	C_1	C_2



So, the min no. of colors are '2'. $\therefore \chi(G) = 2$

Eg: Find the chromatic no. for the following graph?

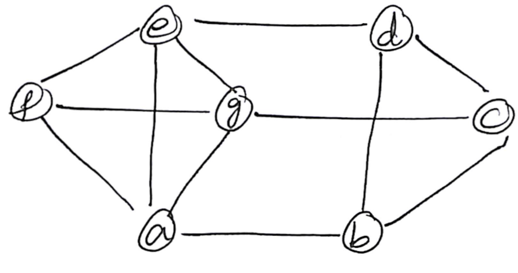


Sol:

vertices	a	b	c	d	e	f
colors	C_1	C_2	C_2	C_3	C_3	C_1

$\therefore \chi(G) = 3$

* Find the chromatic no. of the following graph?

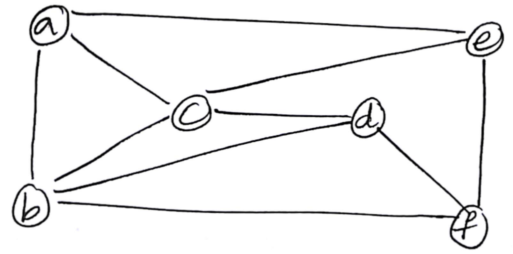


Sol:

vertices	a	b	c	d	e	f	g
colors	c_1	c_2	c_3	c_4	c_3	c_2	c_4

$\chi(G) = 4$

* $\chi(G) = ?$



Sol:

a	b	c	d	e	f
c_3	c_1	c_2	c_3	c_4	c_2

$\chi(G) = 3$

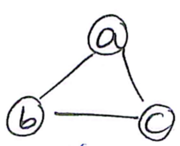
* Graph coloring applications:

- map coloring
- scheduling the tasks
- preparing time table
- Assignment
- Conflict resolution
- sudoku.

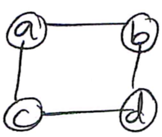
*. Chromatic number for cycle graph:

- If no. of vertices in cycle graph is even, then chromatic no = 2.
- If odd = 3.

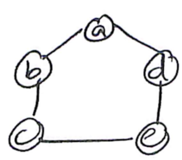
Eg:



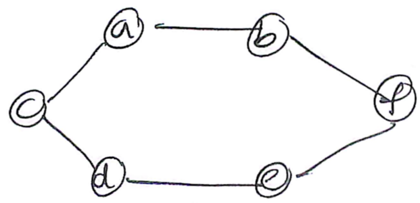
$\chi(G) = 3$



2



3



2

* chromatic number for planar graphs:

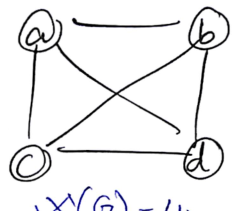
" " " " is less than or equal to 4.

Eg: the above cycle graphs are planar, which are having chromatic no ≤ 4.

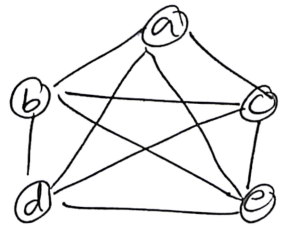
* chromatic no. for complete graphs:

" " " " is no. of vertices in that complete graph.

Eg:



$\chi(G) = 4$

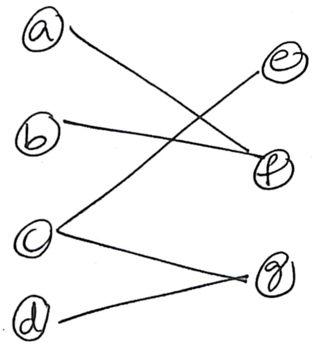


$\chi(G) = 5$

* chromatic no. for bipartite graphs:

" " " " = 2.

Eg:

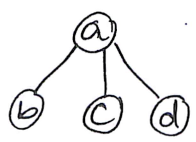


$\chi(G) = 2$.

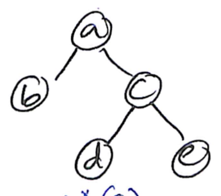
* chromatic no. for trees:

" " " " = 2.

Eg:



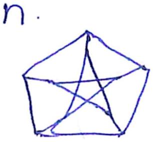
$\chi(G) = 2$



$\chi(G) = 2$

* For complete graph (K_n), $\chi(G) = n$.

* For peterson graph $\chi(G) = \cancel{4}\ 3$



* For wheel graph, ~~If~~ n is even $\chi(W_n) = 3$

" " odd $\chi(W_n) = 4$

n = vertices of wheel graph.

* Graph matching:

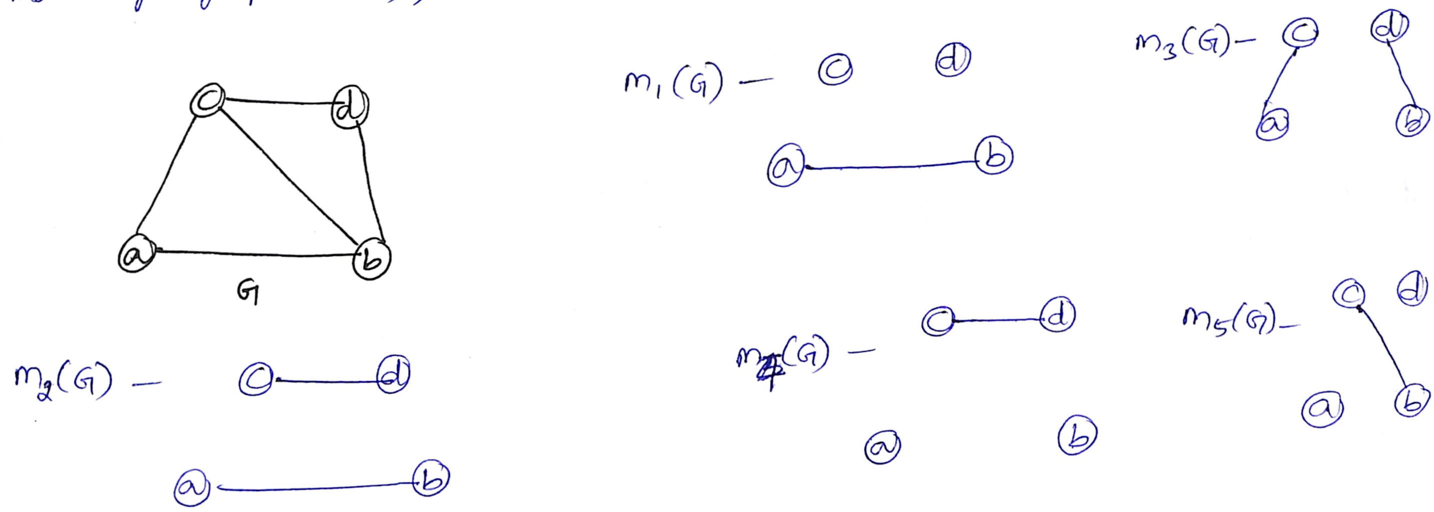
A matching graph is a subgraph of a graph where there are no edges adjacent to each other. i.e., there should not be any common vertex b/w 2 edges.

- Let $G(V,E)$ is a graph. A subgraph is called matching $M(G)$, if each vertex of G is incident with atmost one ~~degree~~ edge in M.

i.e, $deg(v) \leq 1 \ \forall \ v \in G$.

In matching graph $M(G)$, the vertices should have a degree of 1 or 0,

Eg:



In matching, no 2 edges are adjacent. Because the degree will be 2. It voilates matching rule.

If $deg(v)=1$, then matched

$deg(v)=0 \Rightarrow$ not matched.

* maximal matching:

" " means, if no other edges of G can be added to M.

In above example, m_2, m_3, m_5 are maximal matching of G.

Because, If we add any edge, degree will be more than 1.

* Maximum matching:

It is also called as largest maximal matching. i.e., Maximal matching with maximum no. of edges.

- In the above example m_2, m_3 are maximum matching because the no of edges are 2 in both matchings.

* perfect matching:

A matching M of graph G is said to be perfect match, if every vertex of graph G is incident to exactly one edge of the matching.

$\deg(v) = 1 \ \forall v.$

- In above example, m_2, m_3 are perfect matching. Because degrees of all vertices is exactly one.
- If G has perfect match, the no. of vertices is even.
- If no. of vertices are odd, then some vertex has pair up with other vertex. It clearly violates the rule. As it forms degree more than 1.

* Trees:

A tree is an undirected, connected & acyclic graph. i.e., A connected graph that doesn't contain any cycle.

- The elements of tree are called nodes & the edges of the tree are called as branches.
- A tree with n vertices has (n-1) edges.
- A leaf in a tree is a vertex of degree 1 or any vertex having no children

Properties:

(i) Every tree which has atleast 2 vertices of degree two.

(ii). Trees has following characterizations:

If T is a tree

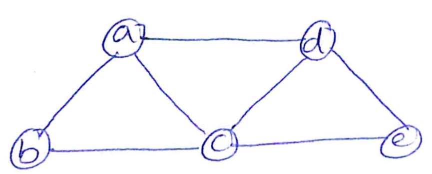
- T contains no cycles & (n-1) edges
- T is connected
- In T, every edge is a cut edge.
- Any two vertices are connected by exactly one path.
- If we add any new edge e, then T+e has exactly one cycle.

(iii). Every connected graph contains a spanning tree.

* Spanning tree:

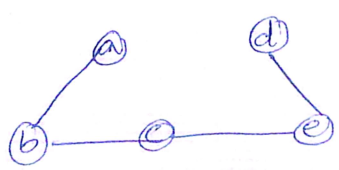
A spanning tree in a connected graph G is a sub graph H of G that includes all the vertices of G & is also tree.

Eg:

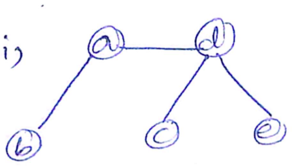


From this graph we can implement 3 spanning trees

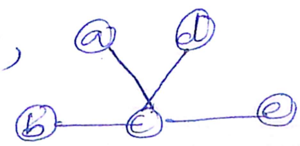
(i)



(ii)

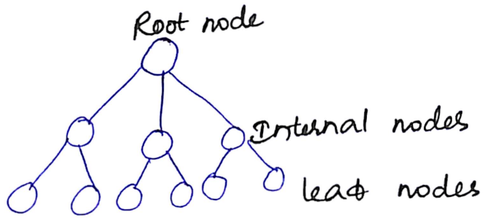


(iii)



*Rooted tree:

-A rooted tree G is connected acyclic graph with a special node is called the root of the tree & every edge is directly or indirectly originates from the root.



--An ordered rooted tree is a rooted tree where the children of each internal vertex are ordered.

- If every internal vertex of a rooted tree has not more than 'm' children, it is called an m-ary tree.

- If every internal vertex of a rooted tree has exactly m children, It is called full m-ary tree

- If m=2, the rooted tree is called a binary tree.

We have 2 types of spanning trees. i) minimum spanning tree
ii) ~~Mini~~ maximum spanning tree

If we will construct the spanning tree by taking all least weight edges then it is minimum spanning tree. If it is by taking max weight, that is maximum spanning tree.

We have different algorithms to find minimum spanning tree.

i) Kruskal's algorithm

ii) prim's algorithm.

* Kruskal's algorithm:

i) Sort all edges in non decreasing order of their weight.

ii) pick smallest edge

Check if this edge is forming any cycle or not.

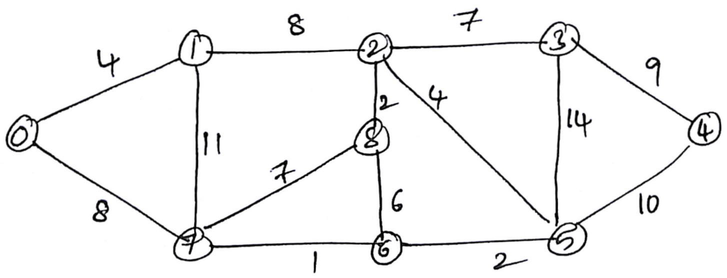
If it is forming cycle then discard that edge

If not, include that edge into spanning tree.

(iii) Repeat step 2, until there are (V-1) edges in spanning tree.

Example:

Find MST for following.



Sol: we have 9 vertices, 14 edges. So, In MST V-1 = 8 edges should be there.

{7,6} = 1

{2,8} = {6,5} = 2

{0,1} = {2,5} = 4

{8,6} = 6

{2,3} = {7,8} = 7

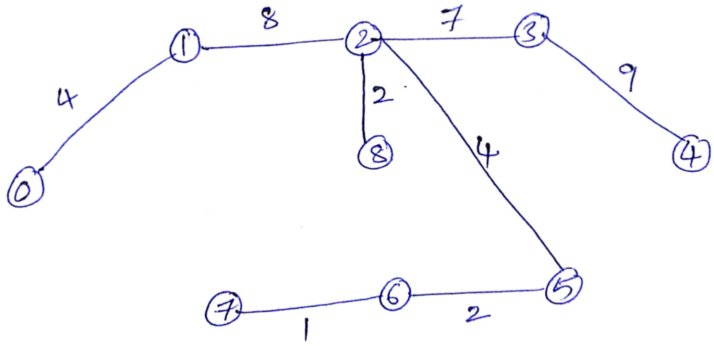
{1,2} = {0,7} = 8

{3,4} = 9

{5,4} = 10

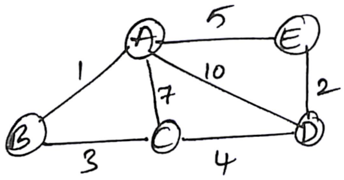
{1,7} = 11

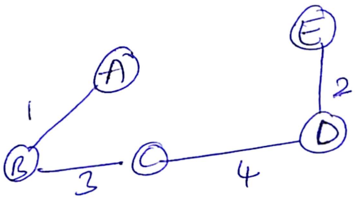
{3,5} = 14



Cost (MST) = 37.

* Find MST





Sol:

$\{A,B\} = 1$

$\{D,E\} = 2$

$\{B,C\} = 3$

$\{C,D\} = 4$

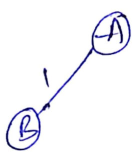
$\{A,E\} = 5$

$\{A,C\} = 7$

$\{A,D\} = 10$

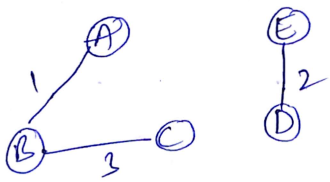
Cost (MST) = 10.

Step 1:

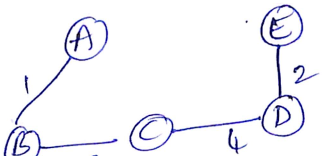


Step 2:

Step 3:

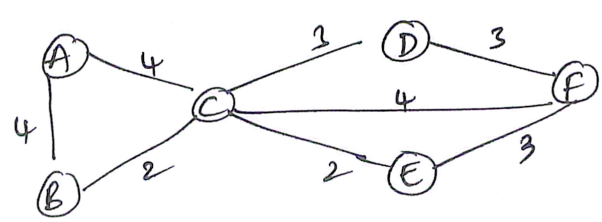


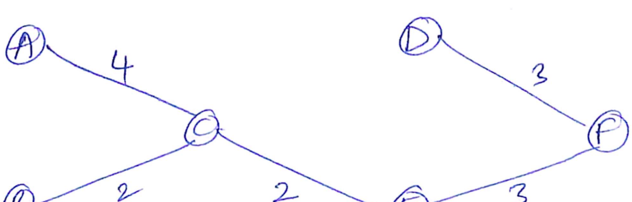
Step 4:



If we add $\{A,E\}$ or $\{A,C\}$, (or) $\{A,D\}$ It will form a cycle. So we can neglect.

* Find MST.





Sol:

$\{B,C\} = \{C,E\} = 2$

$\{D,F\} = \{E,F\} = \{C,D\} = 3$

$\{A,B\} = \{A,C\} = \{C,F\} = 4$

Cost (MST) = 14

* Prim's Algorithm:

It is used to find the min spanning tree from a graph. Prim's algorithm finds the subset of edges that includes every vertex of the graph such that the sum of the weights of the edges can be minimized.

- Prim's algorithm starts with a single node and explore all the adjacent nodes with all the connecting edges at every step & It should not have cycles

Algorithm:

Step 1: Select a starting vertex.

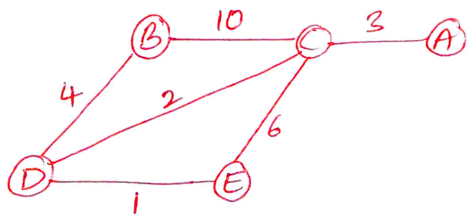
Step 2: Repeat steps 3 and 4 until there are fringe vertices

Step 3: Select an edge e connecting the tree vertex & fringe vertex that has min weight.

Step 4: Add the selected edge & the vertex to the minimum spanning tree T.

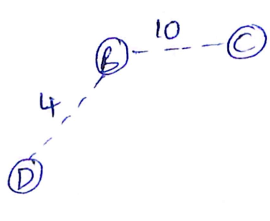
Step 5: Exit.

Example:

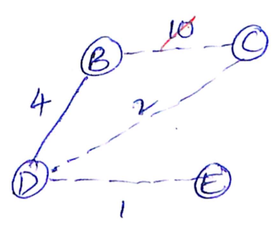


Sol: Step 1: Choose vertex 'B'

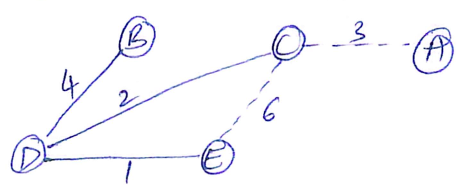
Step 2:



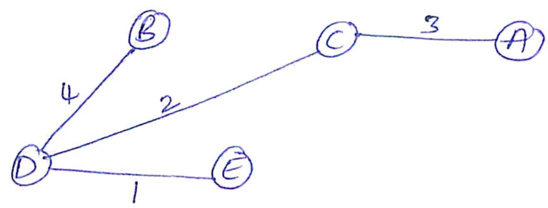
Step 3:



Step 4:

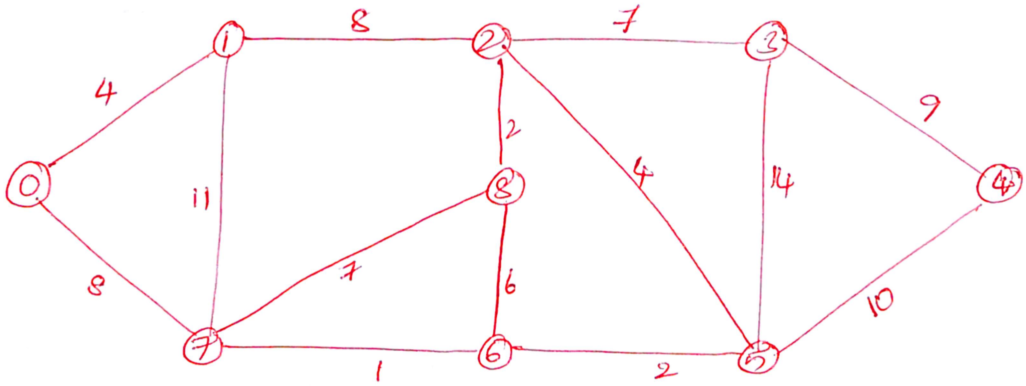


Step 5:

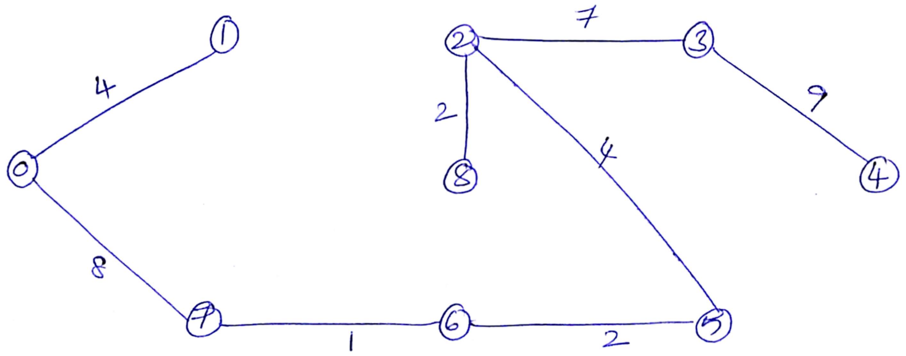


Cost (MST) = 4+2+1+3 = 10

Eg.2:

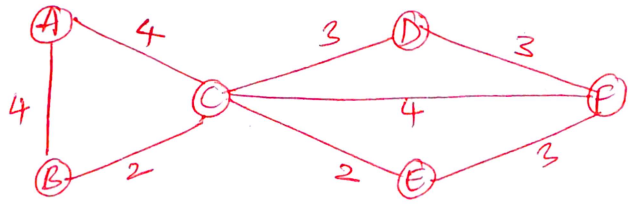


Sol:



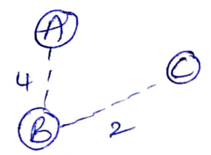
Cost(MST) = 4 + 8 + 1 + 2 + 2 + 4 + 7 + 9 = 37.

Eg.3:

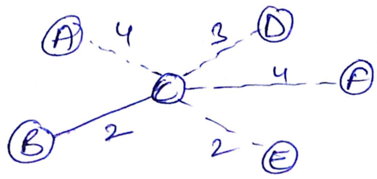


Sol: step 1:

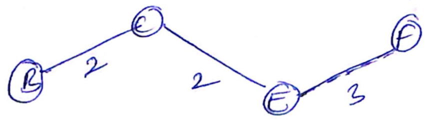
step 2:



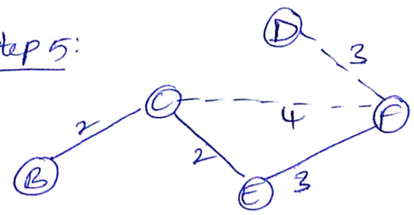
step 3:



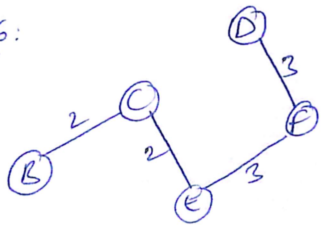
step 4:



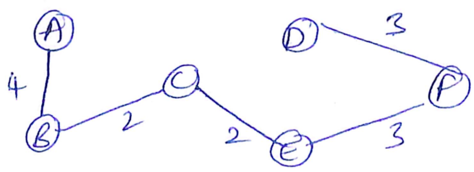
step 5:



step 6:



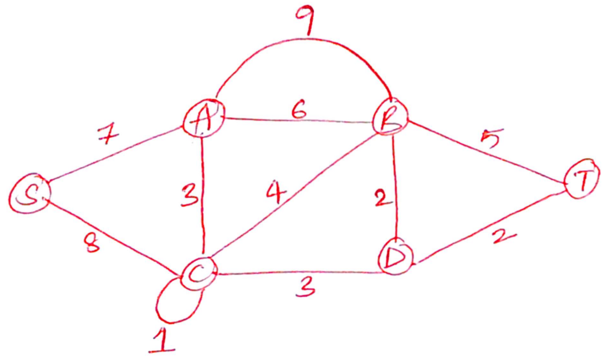
step 7:



Cost (MST) = 4 + 2 + 2 + 3 + 3

= 14

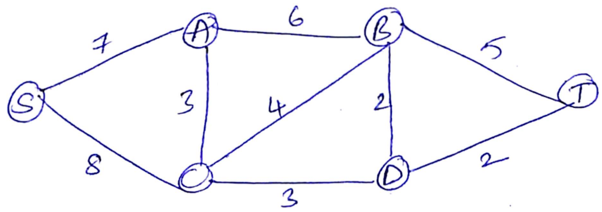
Eg 4:



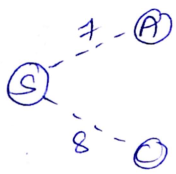
Sol:

Step 1: remove all self loops & parallel edges

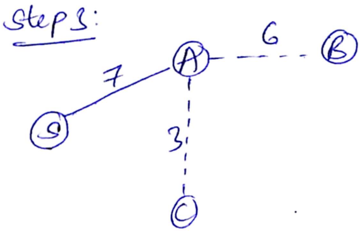
In parallel edge, we should consider least cost edge.



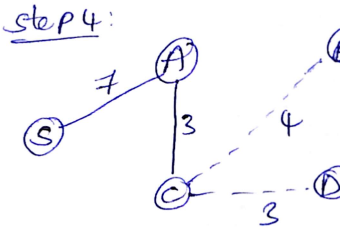
Step 2:



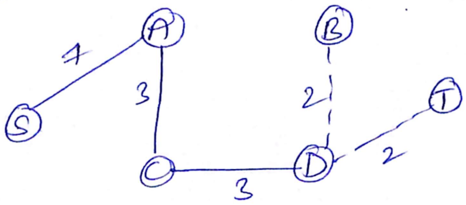
Step 3:



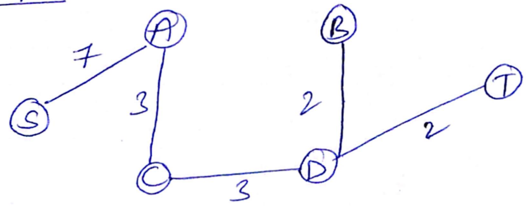
Step 4:



Step 5:

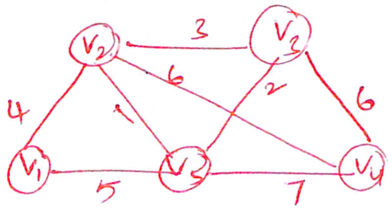


Step 6:



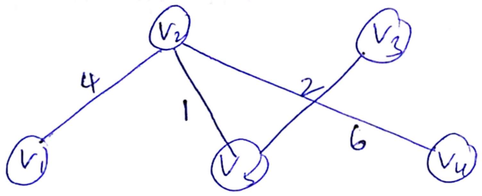
Cost(MST) = 7+3+3+2+2 = 17.

Eg 5:

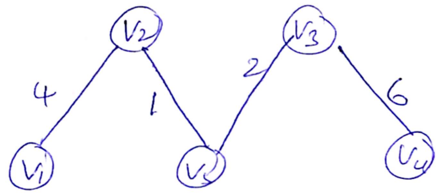


Sol:

	V_1	V_2	V_3	V_4	V_5
V_1	—	4	∞	∞	5
V_2	4	—	3	6	1
V_3	∞	3	—	6	2
V_4	∞	6	6	—	7
V_5	5	1	~~4~~2	7	—

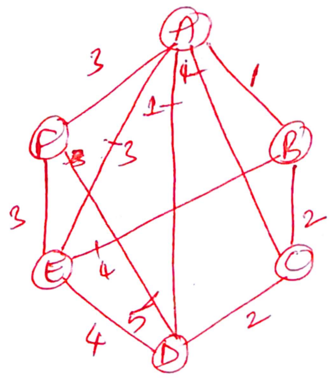


(or)



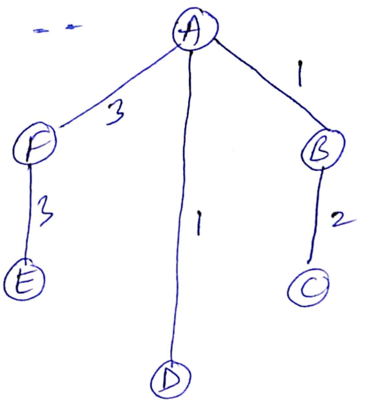
Cost (MST) = 4 + 1 + 2 + 6 = 13

Eg: 6



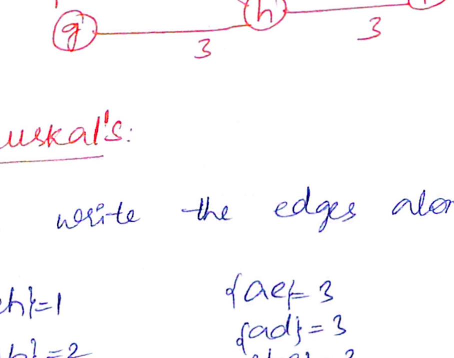
Sol:

	A	B	C	D	E	F
A	—	1	4	1	3	3
B	1	—	2	∞	4	∞
C	4	2	—	2	∞	∞
D	1	∞	2	—	4	5
E	3	4	∞	4	—	3
F	3	∞	∞	5	3	—



cost(MST) = 3 + 3 + 1 + 1 + 2 = 10

Eg 7:



Find MST using Kruskal's & Prim's!

Sol: Kruskal's:

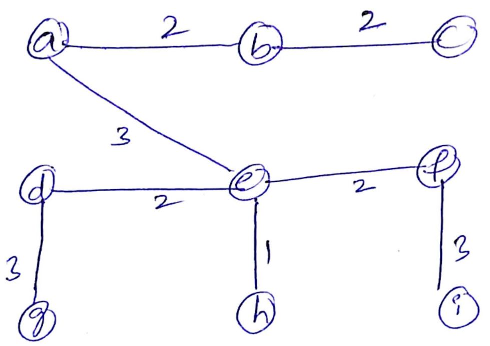
step 1: write the edges along with weights in increasing order.

{eh} = 1
{ab} = 2
{bc} = 2
{de} = 2
{ef} = 2

{ae} = 3
{ad} = 3
{be} = 3
{bf} = 3
{cf} = 3

{dg} = 3
{fi} = 3
{gh} = 3
{hi} = 3

{ei} = 4
{dh} = 5

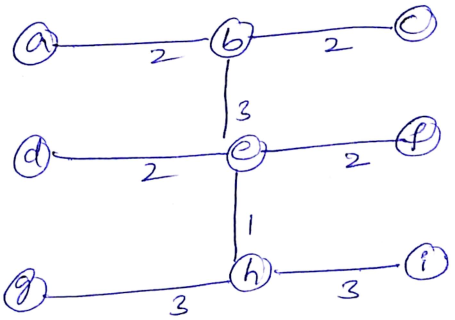


Cost(MST) = 4+3+4+7 = 18

Prim's:

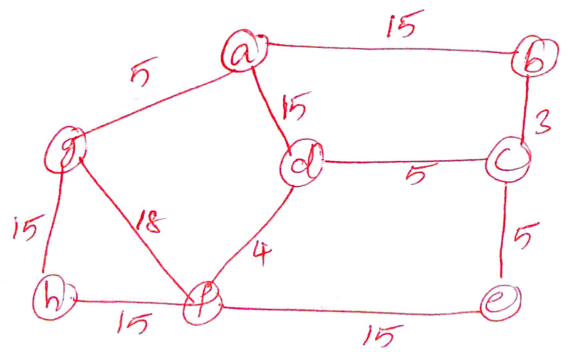
Draw the adjacency matrix & then find the min weight for each vertex

a
{a b} = 2
{b c} = 2
{c b} = 2
{d e} = 2
{e h} = 1
{f e} = 2
{g d} = {g h} = 3
{h e} = 1
{i h} = {i f} = 3



Cost(MST) = 4+3+4+1+6 = 18

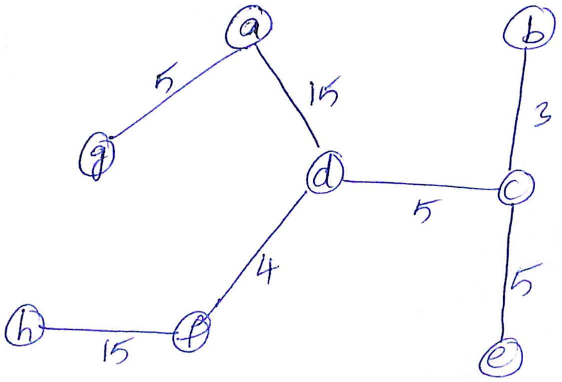
Eg (8): Find MST using Kruskal's & Prim's for the following.



Sol:

Kruskal's:

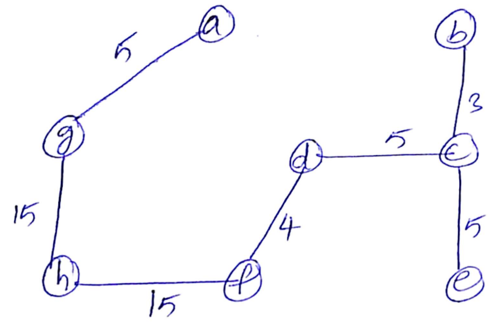
{b c} = 3
{d f} = 4
{a g} = 5
{c d} = {c e} = 5
{a d} = 15
{a b} = 15
{e f} = 15
{f h} = 15
{g h} = 15
{f g} = 18



Cost (MST) = 23+5+24 = 47+5
= 52

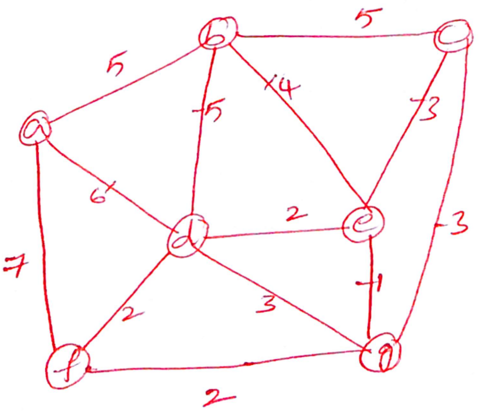
prim's:

ag = 5 , bc = 3, cd = ce = 5, df = 4 , ec = 5, fd = 4, ga = 5, hg = hf = 15



Cost(MST) = 30 + 15 + 7 = 45 + 7 = 52

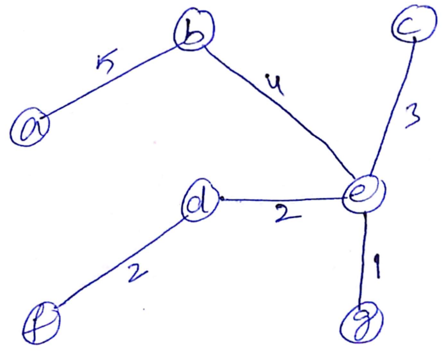
*** Find MST for the following: using kruskal's & prim's.**



Sol:

kruskal's:

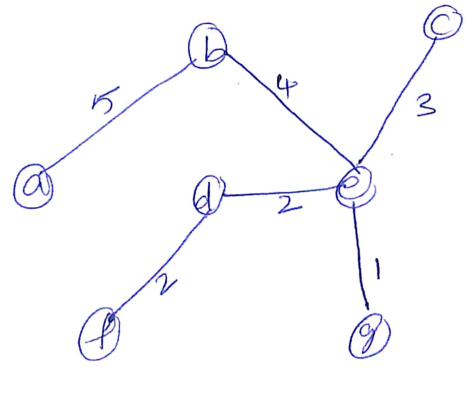
{eg} = 1
{de} = 2
{df} = 2
{fg} = 2
{ce} = 3
{cg} = 3
{dg} = 3
{be} = 4
{a,b} = 5
{b,d} = 5
{b,c} = 5
{a,d} = 6
{a,f} = 7



Cost(MST) = 12 + 5 = 17

prim's:

	a	b	c	d	e	f	g
a	—	5	∞	6	∞	7	∞
b	5	—	5	~~5∞~~	4	∞	∞
c	∞	5	—	∞	3	∞	3
d	6	5	∞	—	2	2	3
e	∞	4	3	2	—	∞	1
f	7	∞	∞	2	∞	—	2
g	∞	∞	3	3	1	2	—



Cost(MST) = 17