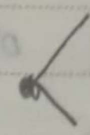
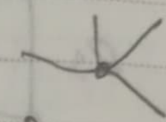
Discrete

Date 28 08 2025

Graph



Degree 2



Degree 4

~~Isometri~~:

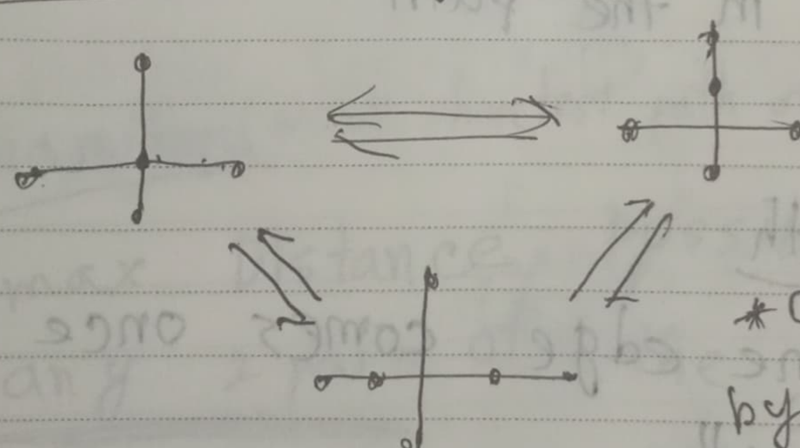
Isomorphic graph:

→ Vertex same

→ Edge "

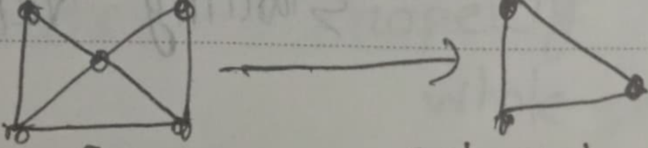
→ Degree "

Homemorphic graph



* originate by deviding the edges by 1 or more vertex from a base graph

subgraph

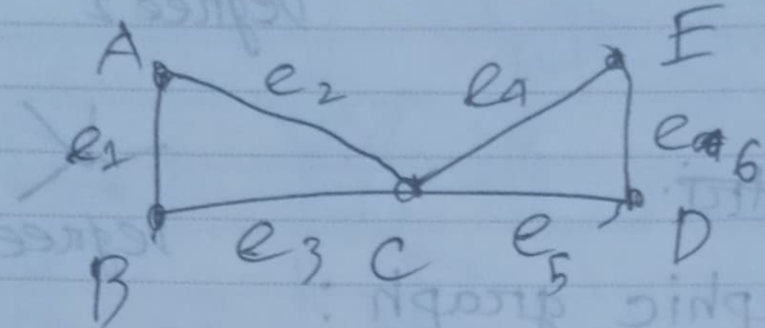


G subgraph of G

SDG CELL www.sid.gov.bd

Path

sequence of vertex, edge ~~from~~ between 2 vertex



A, e_2, C, e_5, D or $A, e_2, C, e_4, E, e_6, D$

simple path

If one vertex comes ~~one time~~ once only in the path

Trail path

If one edge comes once in the path.

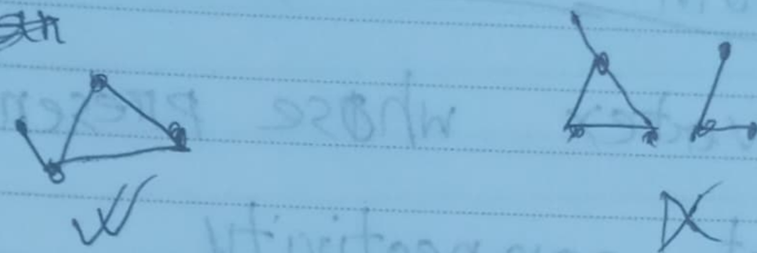
cycle:

can loop backs to same ~~vertex~~ starting vertex.

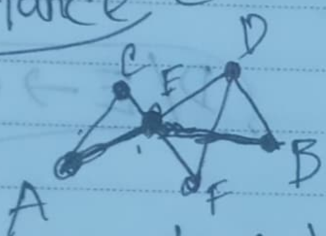
Date

Conectivity

~~Con~~



Distance (shortest path)



shortest path = Distance = 2
(AE, EB)

Diamiter (max shortest path of any pair)

max Distance possible from any 2 pair of the graph.

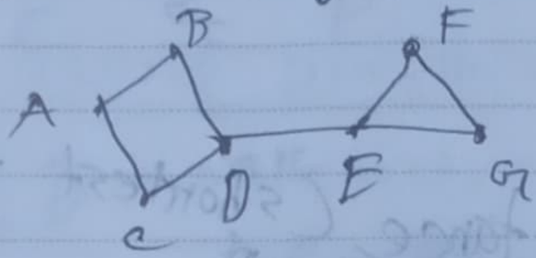
Distance $\longrightarrow$ Between 2 vertex

Diameter $\longrightarrow$ Property of the whole graph.

~~Grease~~

Cutpoint (A vertex)

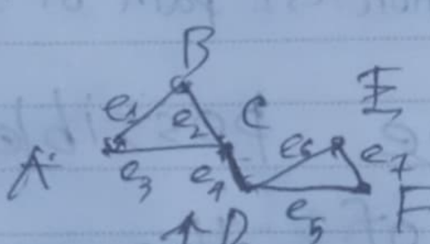
A vertex whose presense affect connectivity.



D/E → Cut point

Grease : (An edge)

cutpoint equivivalant ~~of~~ for edge

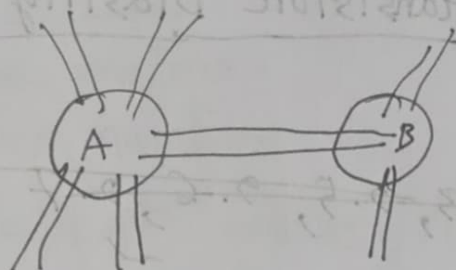


e_4 grease

04/09/25

বৃহস্পতি

Discrete



* can you traverse each bridge ~~atleast~~ only once..?

→ Ans: NO

* only possible if:
  - → All vertex need to have even degree.
  - or,
  - → Only 2 odd degree vertex

Eulerian graph:

→ A graph where it is possible to ~~only once~~ traversing all ~~vert~~ edge but only once each ~~and reaching starting point~~.

Hamilto graph:

→ Same as Eulerian graph ~~by~~ but traverse vertex.

→ only if:
if $n \geq 3$ and $n \leq$ degree(vertex) for each vertex

circuit

→ Can come back to same node after traversing

Path:

→ Cannot come back to same node but fullfill Eulerian/Hamiton condition.

weighted graph

→ has cost for each vertex

complete graph

→ All node connected with all node

Regular ghaph

→ when all vertex has same degree = k the it called ~~k~~ regu graph.
denoted by (k- regular)

* All complete graph ≡ regular graph
" regular " != complete "

7/9/25
রবিবার

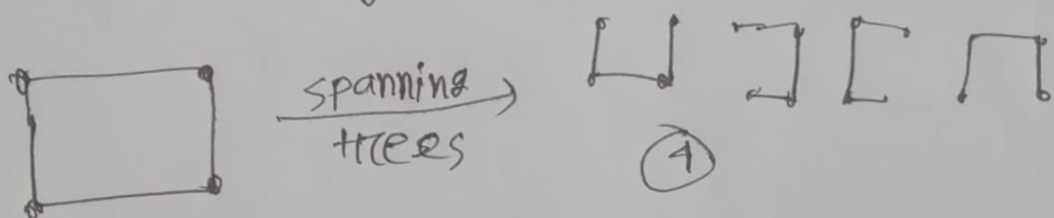
Discrete

Tree

All Tree are Graph
Not ,, Graph ,, Tree

~~ei~~ Graph → Has cycle
Tree → No cycle
→ ~~No cyel~~ Break any cycle

The number of Tree possible by ~~changi~~ removing ~~any one~~ edges of graph is called spaning tree



Spanning tree algorithms (Minimum spannig tree)

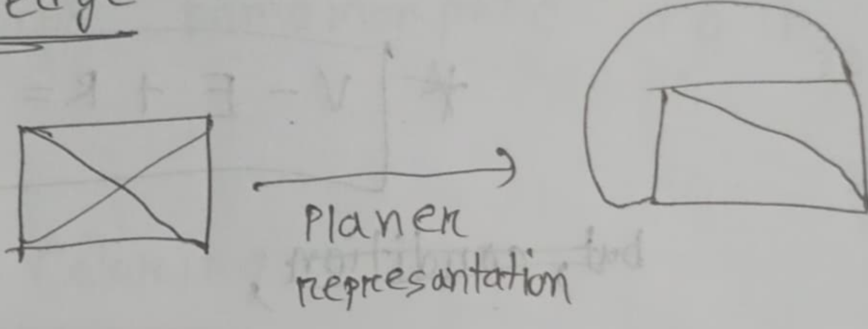
→ Pri~~em~~m's algorithm → Find minimum spanning tree from weighted graph, ~~chec~~ check each removable edge.

→ Kruskal's ,, → Same as prim but from reverse, returns only the nonremovabe edges.

removable edge → does not disconect graph and

Planer graph:

→ Can be drawn/redraw without crossing edge.

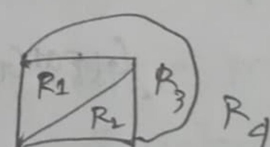


*map:

* A planer Graph/representation is called map

*region:

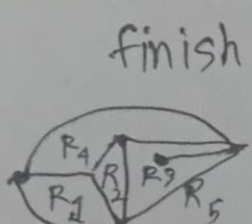
∵ Bounded cycles in map are called region



* Degree of map region:

~~→ number of edges that form the boundary of a region~~

→ number of edge travaral needed to finish a cycle in a bounded region



deg($R_1 = 3$ $R_4 = 4$

$R_2 = 3$ $R_5 = 3$

$R_3 = \underline{5}$

* Number of region in a map/Planer graph,

Eular's Theorem,

* $\boxed{V - E + R = 2}$

V → vertex
E → Edge
R → Region

~~but, condition~~,

* possible Condition of planer graph

if, $p >= 3$

if, $q >= 3P - 6$ → Not 100% accurate, only for simple cases.

~~Grap~~ Graph is planer

$\boxed{\begin{array}{l} p \rightarrow \text{vertex} \\ q \rightarrow \text{edge} \end{array}}$

$P = 6 \geq 3$

$q = 12 \geq 3P - 6$

So planer represantation possible

* condition of non-planer graph

→ If and only if the graph contains subgraph homomorphic to $K_{3,3}$ and K_5

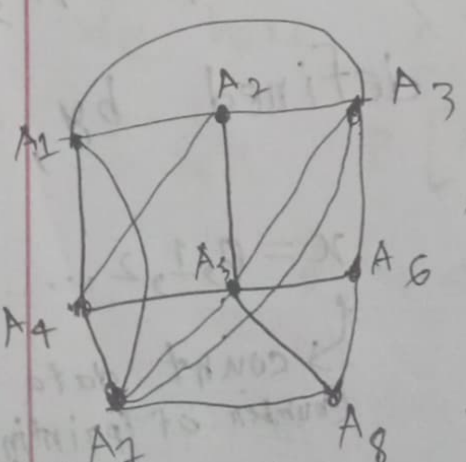
Graph Coloring

* Adjacent vertex in graph cannot have same color.

*** Algorithm:**

→ Well's Panel's algorithm

↳ Determine minimum number of ~~col~~ color needed to do graph coloring.



step-1:

* sort vertex in descending order of degree

→ $A_5, A_3, A_7, A_1, A_2, A_4, A_6, A_8$

step-2: start coloring for start and also give same color to non conneted graph.

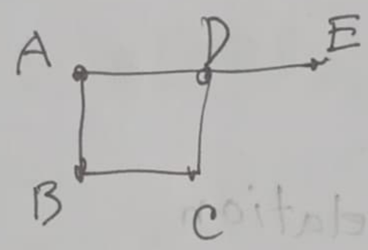
A_5	A_3	A_7	A_1	A_2	A_4	A_6	A_8
c_1	c_2	c_3	c_1	c_3	c_2	c_3	c_2

$\boxed{\text{3 colors}}$

11/09/2025

Discrete

Graph representation in Data Structure



$$a_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

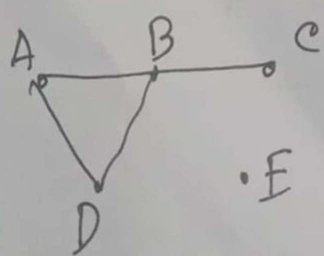
	A	B	C	D	E
A	0	1	1	0	0
B	1	0	1	0	0
C	0	1	0	1	0
D	1	0	1	0	1
E	0	0	0	1	0

~~$V_i \to$ number of edge~~

$V_x \to$ ~~xth~~ x^{th} vertex

① * ~~Ma~~ Array representation

② * Linked List representation



vertex	adjacant list
A	B, D
B	A, C, D
C	B
D	A, B
E	φ