Graph

A graph is a data stoucture made up of set of nodes (also called vertices) and edges (connection between pair of nodes).

Real life examples

→ Social Network in

→ City map and roads

→ Flight route and airports

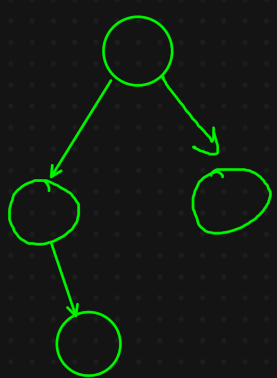
Industry use cases

→ Google map

→ Search Engines

→ Social Network websites

A tree is a special kind of graph.



1) Every node is connected To each other in someway. (No isolated node possible)

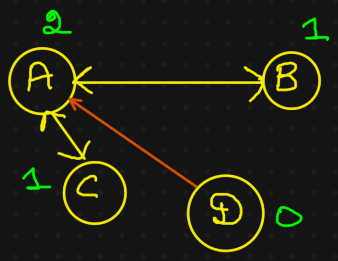
2) Trees don't have a cycle.

Terminologies in Graph

1. Node (vertex) → individual element in a graph

2. Edge → Connection between 2 vertices.
   Unidirection or bidirectional

3. Adjacent vertex → Two vertices are said to be adjacent if connected by an edge.



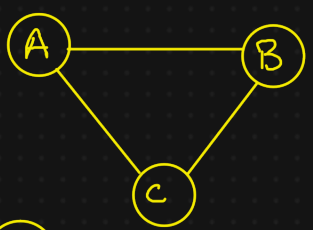
4. Degree → no. of edges connected to a node/vertex
   in degree vs out degree
   for A 3 2

5. Path → a path of length n from a node u_1 to a node u_n is defined as sequence of (n+1) nodes
   $P = \{u_0, u_1, \ldots, u_n\}$

6. Cycle
   ↓
   If a path start and end at the same vertex.
   $u_0 = u_n$

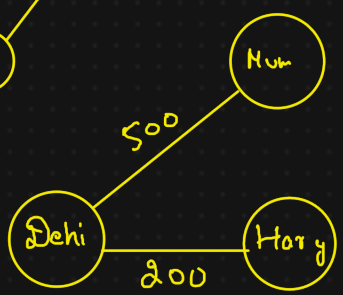
Different Kind of Graphs

1. Undirected Graph (friends on fb)

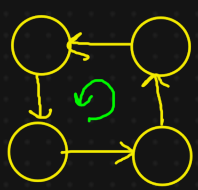


2. Directed Graph (follow on Instagram)

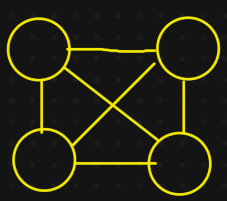
3. Weighted Graph / Unweighted



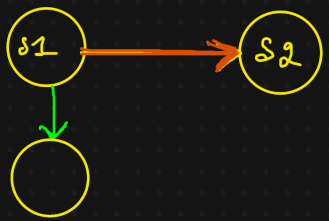
4. Cyclic / Acyclic Cyclic ⇒ contains one cycle atleast



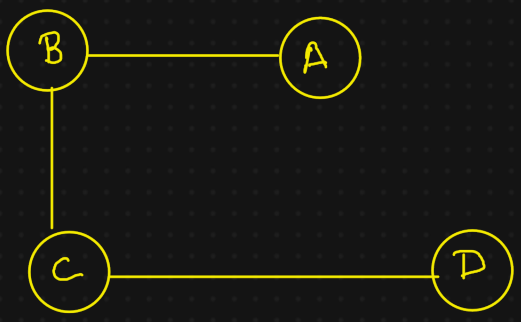
5. Completed Graph
   (Dense)

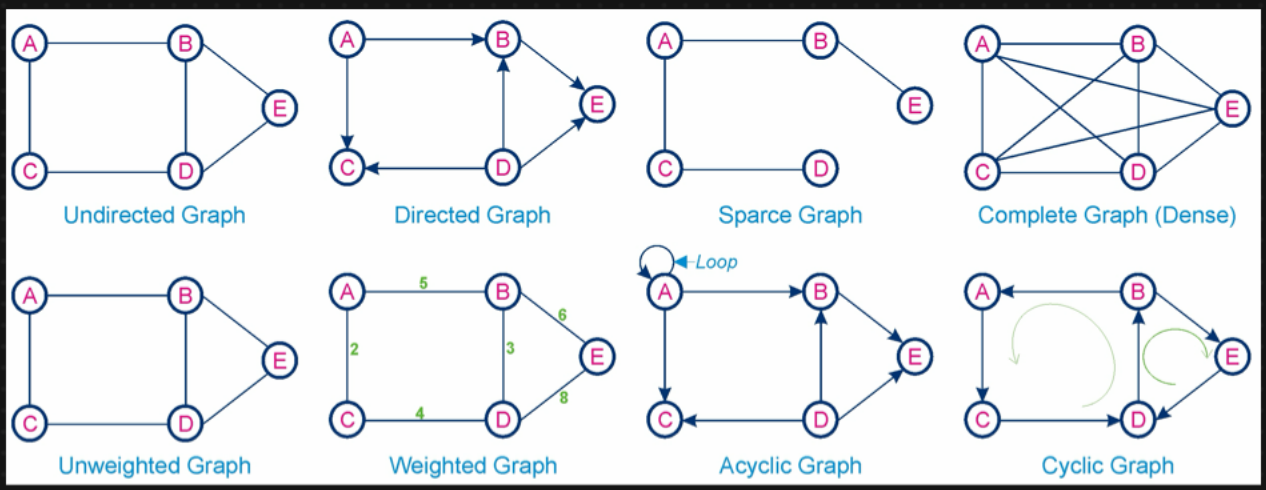


6. Labelled Graph



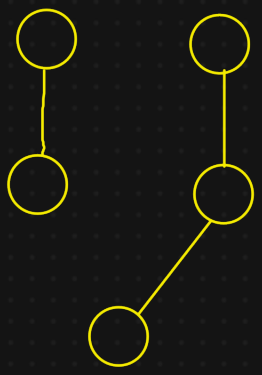
7. Connected Graph

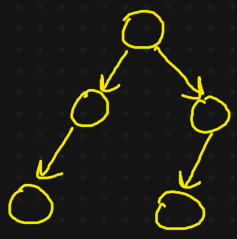

B
A
C
D


A
B
C
D
E
Undirected Graph
Directed Graph
Sparce Graph
Complete Graph (Dense)
5
6
2
3
4
8
Loop
Unweighted Graph
Weighted Graph
Acyclic Graph
Cyclic Graph

Graph Implementation

1. Nodes
2. edges





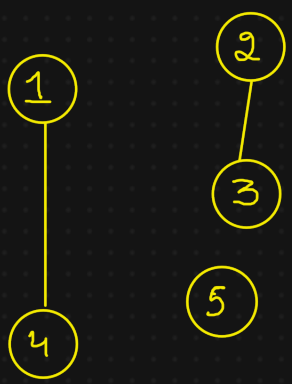
Class Graph:

vertices

edges

1. store the vertices and store all the edges

(1,4) , (2,3) Edge list



how many edges in a n node complete graph ⟶ $^nC_2 = \frac{n(n-1)}{2}$

~ n^2 complexity

2. Adjacency list

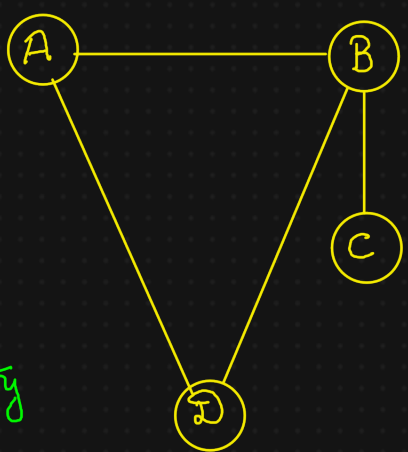
{C, D}

A

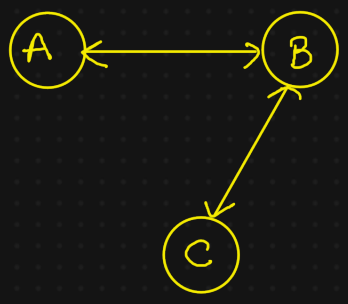
O(n)

~O(n)

time complexity



3. Adjacency Matrix



→ easy to implement

→ Quick

→ space heavy
$O(n^2)$

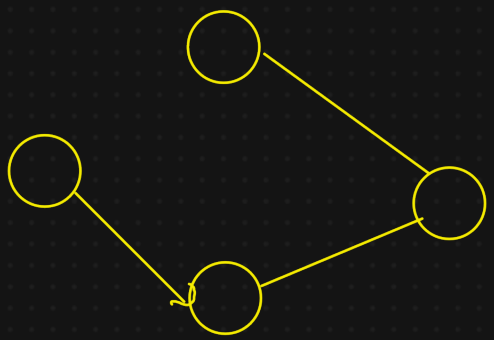
→ not recommended for sparse graph

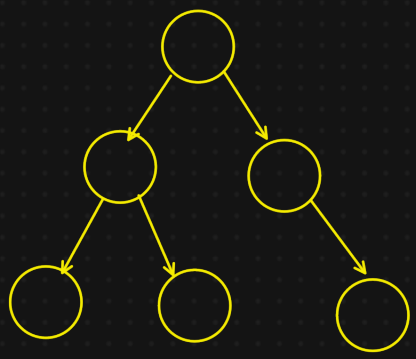
	A	B	C
A	0	1	0
B	1	0	1
C	0	1	0

n^2 matrix

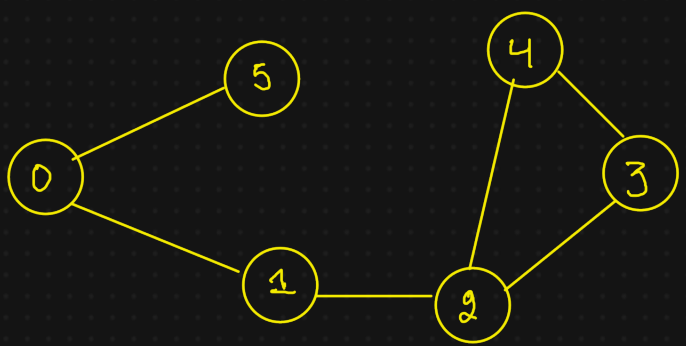
Traversals in a Graph

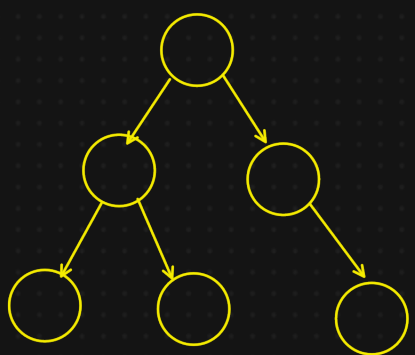
A:





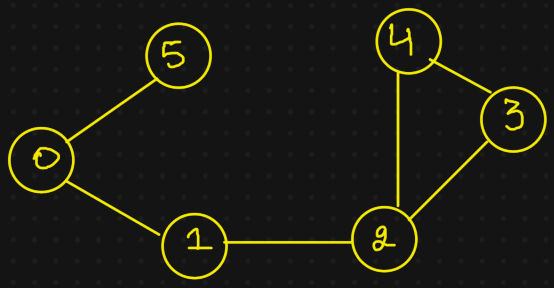
Depth First Search (DFS)

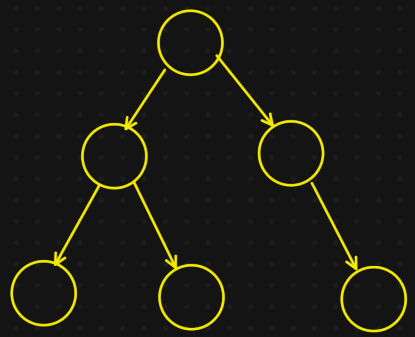




→ 0, 1, 2, 3, 4, 5

Breadth First Search (BFS)



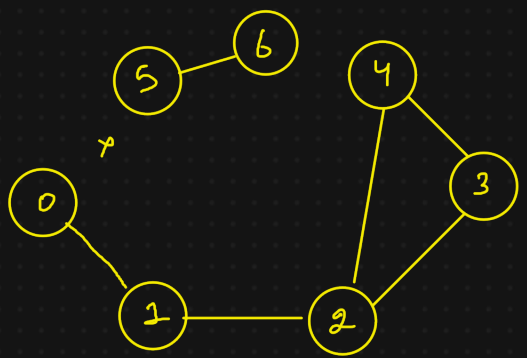


Queue

0, 5, 1

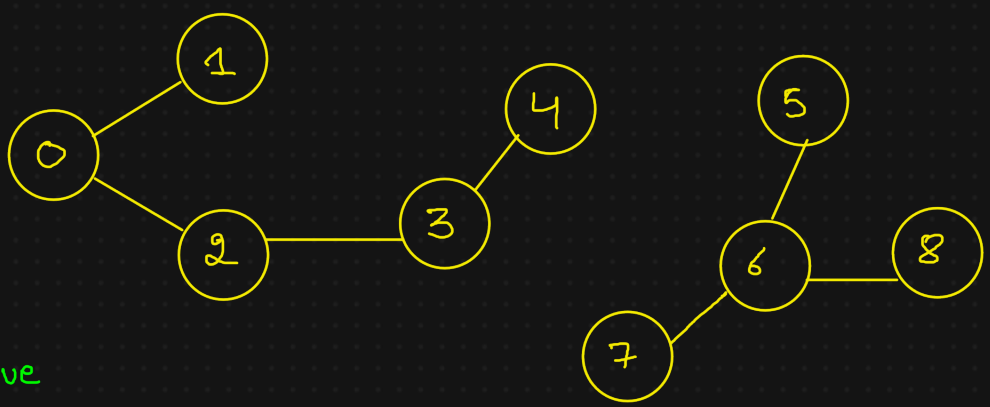
→ 0, 1, 5, 2, 3, 4

Traversal on a Disconnected Graph



Number of Components

Has path



0 , 4 True

7 , 1 False

no need for complete traversal after finding the node.
end