Graphs

Terminology and Representations

A many to one relation between entities can be represented by a Graph. A Graph G is made up of two sets V and E.

$G = \{V, E\}$ $\quad V \Rightarrow$ finite non empty set of vertices.

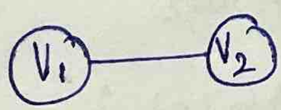
$E \Rightarrow$ Set of pairs of vertices.

These pairs are edges

E can be an empty set also. E can be empty but not V.

Undirected Graph:

The pair of vertices represented in the edge is undirected.

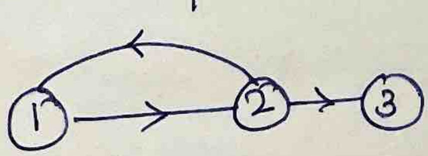


$(V_1, V_2) = (V_2, V_1)$.

Direction is represented by arrows and no arrow is present.

Directed Graph

Each edge is represented by a directed pair $\langle V_1, V_2 \rangle$. $V_1 \Rightarrow$ tail, $V_2 =$ head.

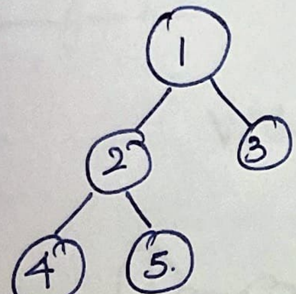


$V(G) = \{1, 2, 3\}$

$E(G) = \{ (1,2), \langle 2,1 \rangle, \langle 2,3 \rangle \}$

Edges of a Graph in a duected tree are drawn with an arrow from tail to head.

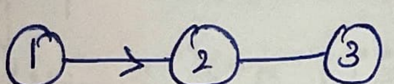
Tree is a Subset of a Graph.



$V = \{1, 2, 3, 4, 5\}$

$E = \{(1,2), (1,3), (2,4), (2,5)\}$

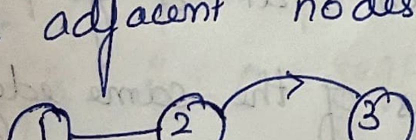
Mixed Graphs Not all edges in a Graph will be directed or undirected. Some of them can be directed and some of them are undirected. This is Mixed Graph.



$G = \{V, E\}$

$V = \{1, 2, 3\}$ $E = \{(1,2), (2,3)\}$

Adjacent Nodes : Two nodes connected by an edge are adjacent nodes



$(1,2)$ & $(2,3)$ are adjacent.

Isolated Node A graph node which is not adjacent to any other node is called isolated node.

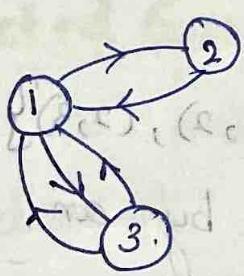
NULL GRAPH A graph that does not have edges.

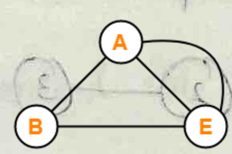
V is non-empty. $E = \{ \}$

LOOP is an edge that connects the vertex to itself.

Direction is immaterial.

PARALLEL EDGES In some Graphs, there may be more than one edge connecting two nodes in same manner / direction. They are parallel edges.





Example of Multi Graph

MULTIGRAPH when the restriction that "a graph may not have multiple occurrences of the same edge" is removed. The resulting Graph is a multigraph.

MAXIMUM # OF EDGES IN UNDIRECTED GRAPH WITH NO RINGS

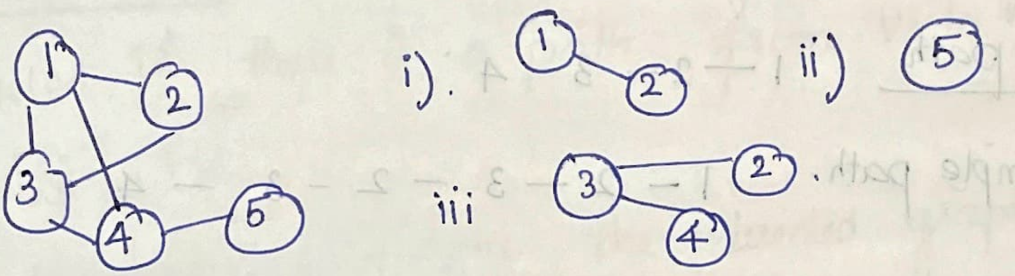
$n(n-1)/2$ where $n \Rightarrow$ vertices.

An 'n' vertex Graph with exactly $n(n-1)/2$ edges is called complete undirected graph.

In a complete directed graph there will be $n(n-1)$ edges.

<u>Subgraph</u> A Subgraph of G_1 (G') is a graph such that

$$V(G') \subseteq V(G) \ \& \ E(G') \subseteq E(G)$$



<u>PATH</u> A path from vertex V_p to V_q in a graph is a sequence of ~~vertices~~ edges. $(V_p, V_{i1})(V_{i2}, V_{i2}) \ldots (V_{in}, V_q) \subseteq E(G)$

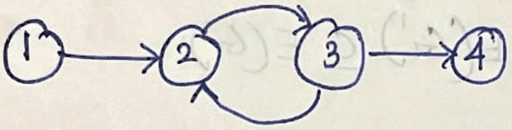
<u>PATH</u> A path from vertex V_p to Vertex V_q in a graph is a sequence of vertices $V_p, V_{i1}, V_{i2} \ldots, V_{iq}$ Such that $(V_p, V_{i1}), (V_{i1}, V_{i2}), \ldots (V_{in}, V_q) \subseteq E(G)$. are edges in G.

if G is a directed then

$\langle V_p, V_{i1} \rangle, \langle V_{i1}, V_{i2} \rangle \ldots \langle V_{in}, V_q) \subseteq E(G)$

are edges in G.

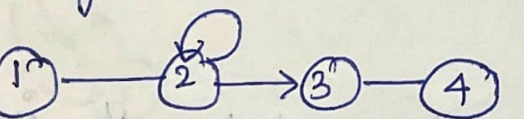
Length of the Path is the number of edges on the path. The path in which the edges are distinct is called a Simple path.



Simple path 1 — 2 — 3 — 4

Not Simple path. 1 — 2 — 3 — 2 — 3 — 4.

Elementary path path in which all nodes through which it traverses are distinct is called an elementary path.



elementary 1 — 2 → 3 — 4

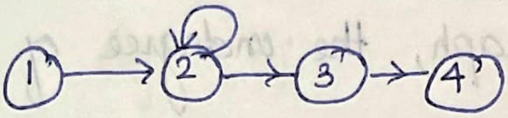
non-elementary 1 – 2 – 2 – 3 – 4.

CYCLE

In a path first and last node could be the same.

A path which originates and terminates on same node is called cycle.

A cycle is called elementary if it does not traverse through any node more than once.



1 -> 2 -> 3 - 4 (elementary cyle).

Acyclic Graph

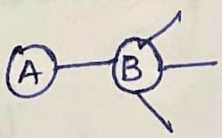
A graph that does not have a cycle is called acyclic graph.

Connected Graphs An undirected graph is said to be connected if there is a path from V_i to V_j for all i, j, $j \neq i$

Strongly connected Graphs The directed graph is said to be strongly connected if for every pair (V_i to V_j) there is a directed path from V_i to V_j $i \neq j$. It is said to be weakly connected if there is a path from V_i to V_j and not from V_j to V_i the graph is a weakly connected Graph.

WEAK LINK Refers to an edge whose removal would significantly impact the connectivity of the graph.

DEGREE OF VERTEX # of edges incident on it.



Degree of A => 1

" " B => 3.

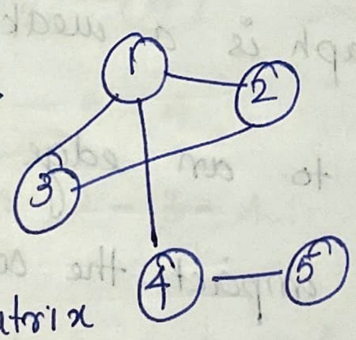
In case of a directed graph, the indegree of the vertex is the # of edges for which it is the head.

Outdedgree => # of edges for which it is a tail.

<u>Tree</u> A tree is a directed graph with no loops + circuits with a indegree of 1 & out degree varying for all nodes except for one node with an indegree 0 called root. A node with 0 out degree is called sink node. A node with 0 indegree is called source node.

Data structures for Graphs

<u>Adjacency matrix</u>



It is a Boolean matrix

	1	2	3	4	5
1	0	1	1	1	0
2	1	0	1	0	0
3	1	1	0	0	0
4	1	0	0	0	1
5	0	0	0	1	0

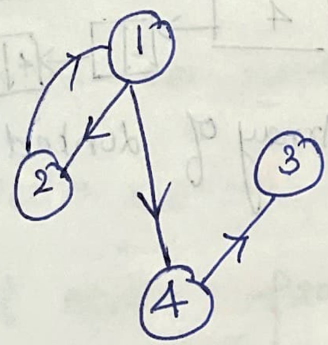
For an undirected Graph Adjacency matrix is a symmetric matrix.

$$A_{ij} = A_{ji}$$

Degree - number of 1s in any row/column.

Degree of a node $= \sum_{j=1}^{n} a_{ij}$ $n \Rightarrow$ # of nodes

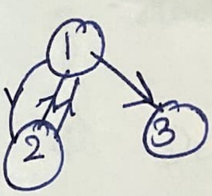
Directed Graphs



$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 1 & 0 & 1 \\ 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 1 & 0 \end{array}$$

In Degree

Out Degree



$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 1 & 1 \\ 2 & 2 & 0 & 0 \\ 3 & 0 & 0 & 0 \end{array}$$

# of times

$$\sum_{j=1}^{n} a_{ij} = \text{out degree}$$

$$\sum_{i=1}^{n} a_{ij} = \text{in degree}.$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} = \text{number of edges}$$

for undirected graph: number of edges.

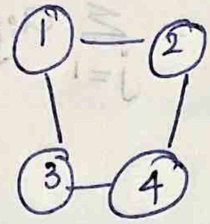
$$\frac{\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}}{2}$$

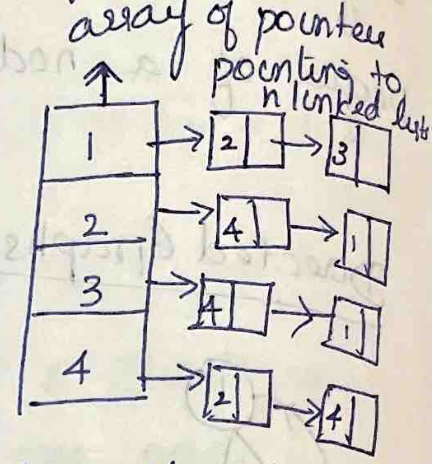
Adjacency list

Length of linked list = number of adjacent nodes.

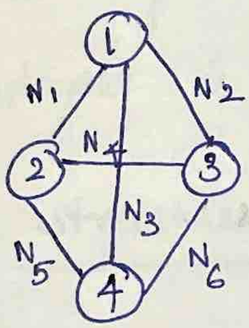
= degree (undirected)

= outdegree (directed)





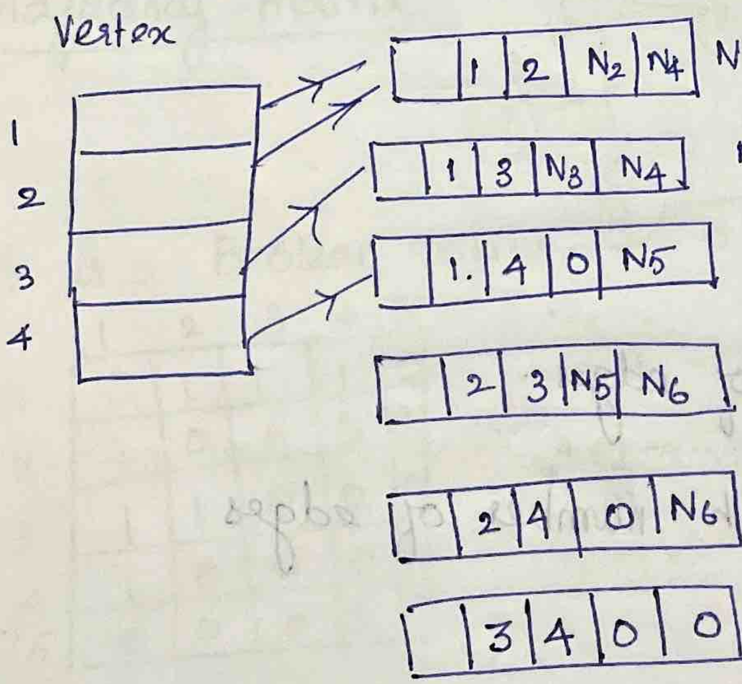
Adjacency Multilists



M	1	2	N_2	N_4

M	V_1	V_2	link1 for V_1	link2 for V_2

One node for every edge



N_1 edge (1,2)

N_2 edge (1,3)

N_3 edge (1,4)

N_4 edge (2,3)

N_5 edge (2,4)

N_6 edge (3,4)

Lists are

Vertex 1 : $N_1 \rightarrow N_2 \rightarrow N_3$

" 2 : $N_2 \rightarrow N_4 \rightarrow N_6$

" 3 : $N_2 \rightarrow N_4 \rightarrow N_6$

" 4 : $N_3 \rightarrow N_5 \rightarrow N_6$

Graph Traversals

Given an undirected Graph $G = (V, E)$ and a vertex v, Graph traversal is a way of reaching all nodes from vertex v. There are two ways of doing this.

1. Depth First search.
2. Breadth First search.

Depth First Search

function DFS(V)

Visited (V) ← 1;

for each vertex w adjacent to V do

if visited(w) = 0 then DFS(w)

end end

Breadth First Search

Function BFS(V)

Visited (V) ← 1

loop

for all vertices w adjacent to V do

if visited(w) = 0 then

[visited(w) ← 1 ; Add(Q, w);] //Adding w to queue Q//

if Q is empty return ;

V ← Delete(Q); // Queue Deletion//

forever.

WARSHALL'S ALGORITHM

function warshall (A, P, n) — A: given matrix, P: Path matrix, n: # of nodes.

P ← A

for k = 1 to n do

for i = 1 to n do

for j = 1 to n do

$A^k[i,j] = \min[A^{k-1}[i,j], (A^{k-1}[i,k] + A^{k-1}[k,j])]$

~~Pij ← Pij~~

TO DETERMINE ALL CONNECTED COMPONENTS OF A GRAP

procedure COMP (G, n) // determine the connected components of G. G has $n \geq 1$ vertices. VISITED is now a local array //

for i ← 1 to n do
    VISITED (i) ← 0 // initialize all vertices as unvisited //
end
for i ← 1 to n do
    if VISITED(i) = 0 then [call DFS(i); // find a component // output all newly visited vertices together with all edges incident to them]
end
end COMP