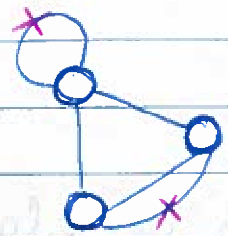
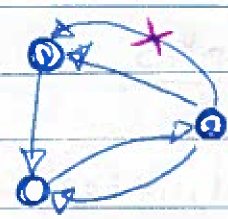
(11 - 12.30am)

EE360C ALGORITHMS - FALL 2018 - Sep 4 2018

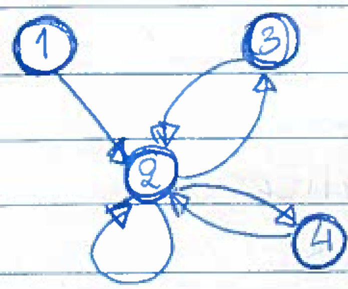
A path from vertex u to v is a sequence of vertices $(v_0, v_1, \ldots, v_k)$ s.t. $v_0 = u$, $v_k = v$ and $\forall i = 1, \ldots, k$ $(v_{i-1}, v_i) \in E$, if graph is directed
$\{v_{i-1}, v_i\} \in E$, if undirected





$E = \{\ldots, (2,1), \cancel{(2,1)}, \ldots\}$

~~multigraphs~~



There is a path from 1 to 3, but no path from 3 to 1.

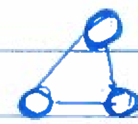
v is reachable from u if $\exists$ path from u to v.
Note: v is always reachable from itself.

A path is simple if all its vertices are distinct.

e.g. In the graph above: path (1,2,4) simple
path (1,2,3,2,4) not simple

In digraph: path forms a cycle if $v_0 = v_k$, 
and $k \geq 1$ at least one edge

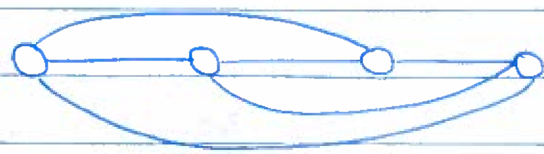
Simple cycle: All vertices other than v_0 and v_k
are distinct.

n undirected graph: path forms a cycle if $v_0 = v_k$ 
and $k \geq 3$ at least 3 edges.

Special Types of Graphs

- COMPLETE GRAPH: Undirected graph with an edge between all pairs of nodes

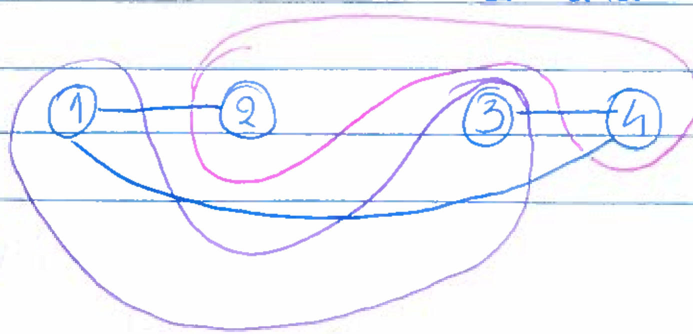
e.g.



NOTE: If a complete graph has n vertices, it will have $\frac{n(n-1)}{2} = \binom{n}{2}$ edges.

- BIPARTITE GRAPH: Undirected graph in which V can be partitioned into V_1 and V_2 s.t. every edge $\{u, v\}$ has $u \in V_1$ and $v \in V_2$.

e.g.



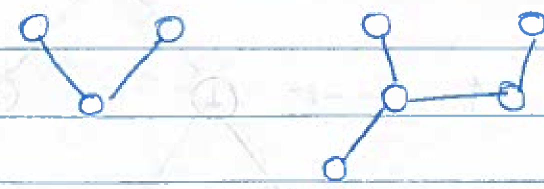
V_1

V_2

- FOREST: Undirected acyclic graph
- TREE: Undirected acyclic connected graph
  ↓
  there is a path between any 2 vertices

NOTE: A graph is acyclic if $\nexists$ path that's a cycle.

e.g. this is a forest:



TREE THEOREM

Let $G = (V, E)$ be an undirected graph.

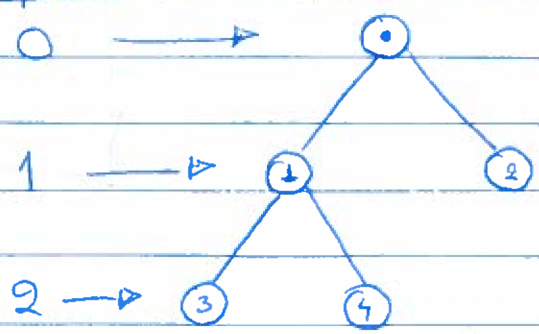
Then the following are equivalent:

1. G is a tree
2. Any two vertices are connected by a unique simple path.
3. G is connected, but if any edge is removed, it's not. "minimally connected"
4. G is acyclic, but if any edge is added, it becomes cyclic. "maximally acyclic"
5. G is connected and $|E| = |V| - 1$
6. G is acyclic and $|E| = |V| - 1$

TERMS FOR ROOTED TREES

1. parent / child
2. leaf : vertice without a child (□)
3. degree of vertex: # of children it has
4. depth of vertex u: length of the simple path from root to u
5. height of tree: largest depth

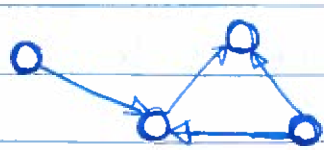
e.g. depth



- The root is the parent of nodes 1 and 2
- Node 3 is a child of 1
- The root's degree is 2.
- The depth of node 4 is 2
- The height of the tree is 2.

DAG : Directed acyclic graph

e.g.



} this is a DAG.

A graph of that type can be used for showing prerequisites of courses:

e.g. The nodes are courses, and there exists an edge Ⓐ→Ⓑ if A is a prerequisite for B.