Undirected Graphs

CSCI 170 Spring 2021

Sandra Batista

1.1-1.2

Graph Theory Introduction

- **Graph Definitions**
- Paths and Cycles
- Connectivity
- Trees

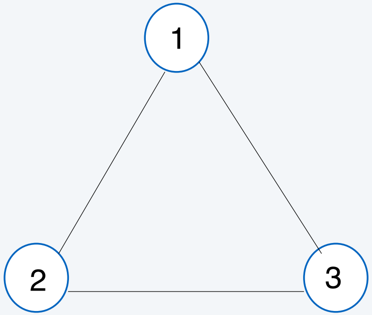
1.1–1.2

What is a graph?

A **graph**, G = (V,E) is a set of **vertices**, V, and a set of **edges**, E.

The set of edges is a subset of the Cartesian product of VxV.

Example:
Graph, G = {{1,2,3}, {(1,2), (2,3), (1,3)}}
The vertices are the set {1,2,3}
The edges are the set {(1,2),(2,3), (1,3)}



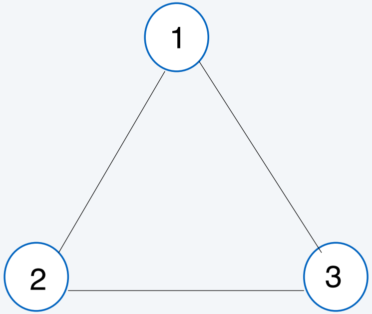
Edges

If an edge is between a vertex and itself, that is a **self-loop**

If we have multiple edges that are the same, that is a **multi-edge**

A graph is **simple** if it has no self-loops or multi-edges.

Example:
Graph, G = {{1,2,3}, {(1,2), (2,3), (1,3)}}
The vertices are the set {1,2,3}
The edges are the set { (1,2),(2,3), (1,3)}



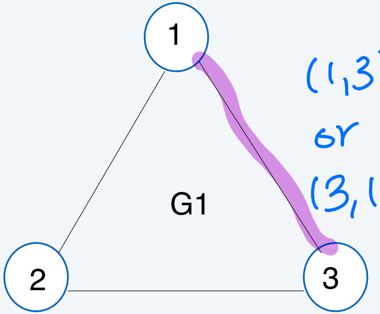
Directed and Undirected Graphs

Edges have **start** and **end**. The edge (u,v) has start u and end v.

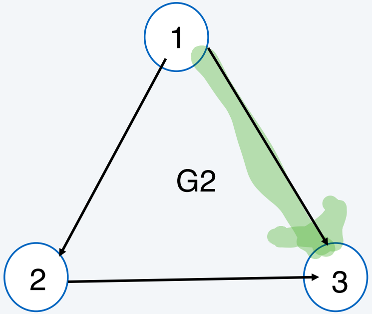
In an **undirected** graph, start and end do not matter. The edge (u,v) is the same as the edge (v,u).

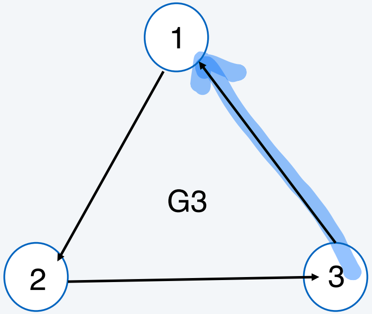
In a **directed** graph, the start (source) and end (sink) do matter. The edge (u,v) is not the same as the edge (v,u) in a directed graph.

Examples:
Undirected: G1 = {{1,2,3}, {(1,2), (2,3), (1,3)}}
Directed: G2 = {{1,2,3}, {(1,2), (2,3), (1,3)}}
Directed: G3 = {{1,2,3}, {(1,2), (2,3), (3,1)}}



(1,3) or (3,1)

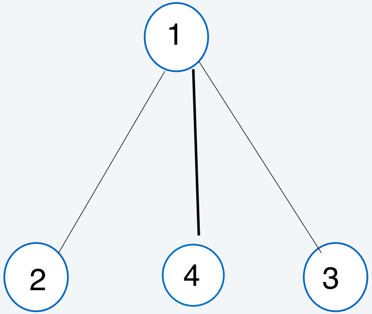




Neighbors

Vertices are called **neighbors** if they share an edge between them. If a graph contains edge, (u,v), vertex u and vertex v are neighbors and are also called **adjacent.**

In an undirected graph the **degree** of a vertex is the number of edges for which it is an endpoint. Such edges touching the vertex or are **incident** to it.



Degree(1) = 3
Incident edges are (1,2), (1,4), and (1,3)

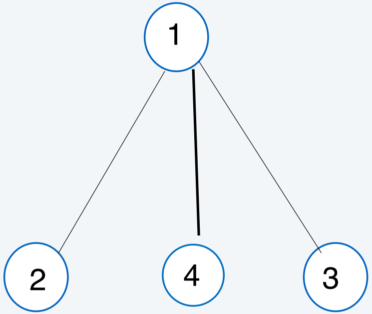
The neighbors of 1, N(1) = {2,3,4}

Handshaking Lemma for Undirected Graphs

The sum of the degrees of all the vertices of the graph equals twice the number of edges:

$$\sum_{v \in V} d(v) = 2|E|$$

Exercise:
1)Verify this on the following graph.
2) How to prove this?



deg(1) = 3
deg(2) = 1
deg(4) = 1
deg(3) = 1

6 = 2|E| = 2*3 ✓

Proof: Each edge has 2 endpts (and endpts contribute to degree of the vertex)

Graph Theory Introduction

- Graph Definitions
- **Paths and Cycles**
- Connectivity
- Trees

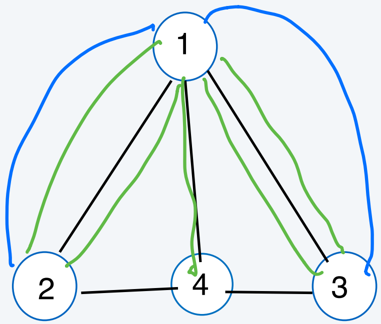
1.1–1.2

Walks and Paths

A **walk** is a sequence of vertices in the graph that traverse edges in the graph.

A **path** is a walk that does not repeat any edges

The **length** of a walk is the number of edges it traverses.



Example walk: 1,3,1,2,1,4
Its length is 5.

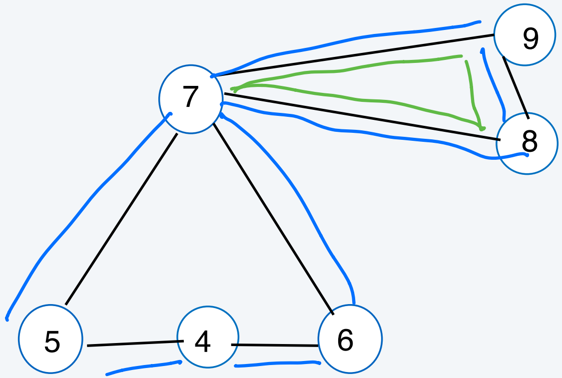
Example path: 2, 1, 3
Its length is 2.

A **circuit** is a path that ends where it begins.

A **cycle** is a circuit that only repeats the first and last vertices.

Example circuit: 7,5,4,6,7,8,9,7
Length: 7

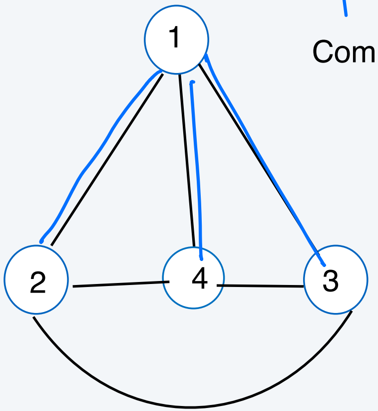
Example cycle: 7,8,9,7
Length: 3



Cycle Graph and Complete Graphs

If G is a graph with n vertices, and the entire graph is a simple cycle on the n vertices, G is called a **cycle graph.**

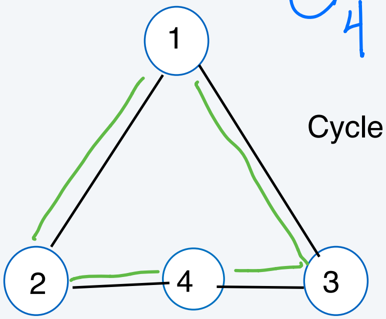
If G is a graph with n vertices and every possible edge exists between each pair of vertices, G is called a **complete graph.**



K_4

Complete Graph on 4 vertices:

K_n



C_4

Cycle graph on 4 vertices:

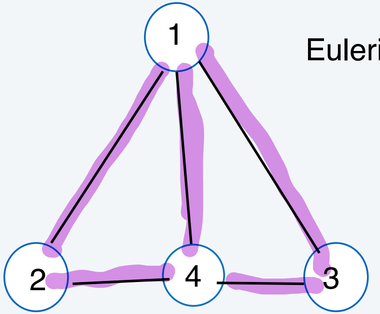
C_n

Eulerian walks and circuits

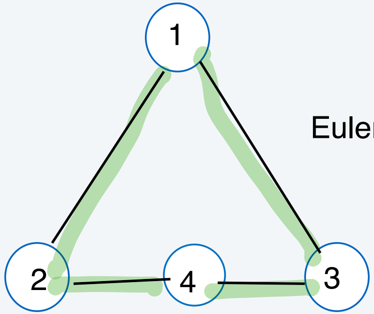
A **Eulerian walk** is a walk that traverses every edge of the graph exactly once.

A **Eulerian circuit** is a Eulerian walk that starts and ends at the same vertex.

Named in honor of Euler after he pondered if all 7 bridges of Könisberg could traversed exactly once.



Eulerian walk: 1, 2,4,1,3,4



circuit

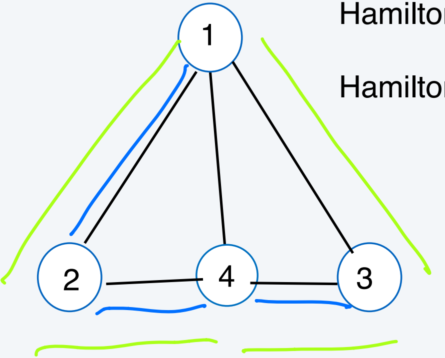
Eulerian ~~cycle~~: 1,2,4,3,1

Hamiltonian paths and cycles

A **Hamiltonian path** is a path that visits every vertex exactly once.

A **Hamiltonian cycle** is a Hamiltonian path that ends where it begins.

Finding a Hamiltonian path/cycle is an NP-Complete problem.



Hamiltonian path: 1, 2,4,3

Hamiltonian cycle: 1,2,4,3, 1

Algorithm to Find Eulerian walk

Given graph G:

1. Check that there are at most 2 vertices of odd degree. If not, no Eulerian walk.

2. Start with vertex of odd degree, v*. Otherwise v* can be any vertex in G.

3. Let G'= G.

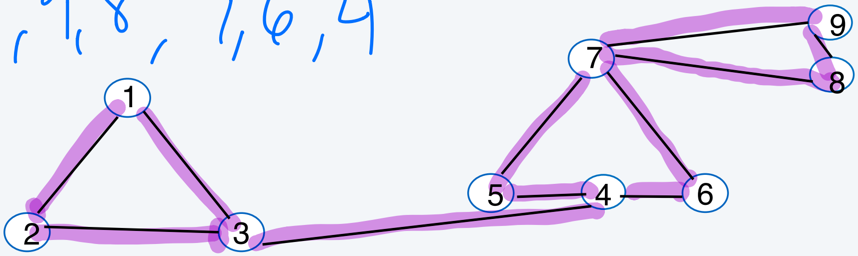
4. While there exists edges in G',

    Let edge (u,v*) be an edge incident to v* that is not a bridge or only edge from v*.

    Traverse (u,v*) in walk by removing from G';

    Let v* = u;

3, 2, 1, 3, 4, 5, 7, 9, 8, 7, 6, 4



Graph Theory Introduction

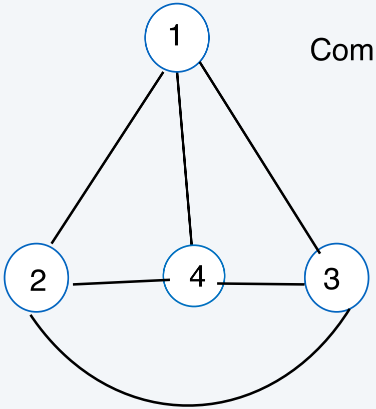
- Graph Definitions
- Paths and Cycles
- Graph Coloring
- Trees
- **Connectivity**

1.1–1.2

Connectivity in Undirected Graphs

Two vertices are **connected** if there exists a path between them.

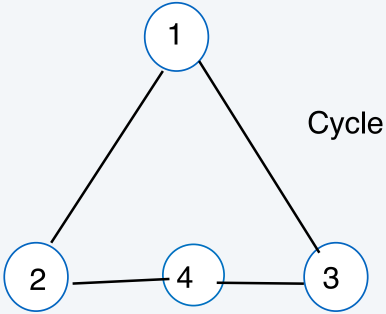
An undirected graph is **connected** if there exists a path between every pair of vertices.



K_4

Complete Graph on 4 vertices:

K_n



C_4

Cycle graph on 4 vertices:

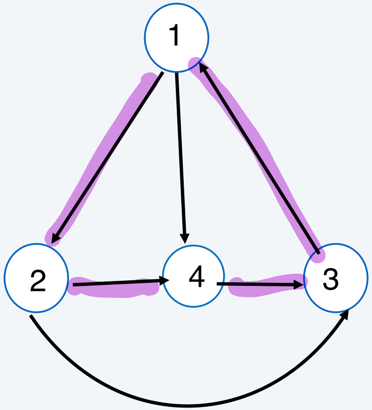
C_n

Connectivity in Directed Graphs

A directed graph is **weakly connected** if it is connected if the direction of edges are ignored.

A directed graph is **connected** if for every pair of vertices u and v, there exists a path u to v or a path v to u in the graph.

A directed graph is **strongly connected** if for every pair of vertices u and v there exists a path from u to v and v to u.



Example:

This graph is weakly connected. Why? because underlying undirected graph is K_4

This graph is connected. Why? (highlighted directed cycle)

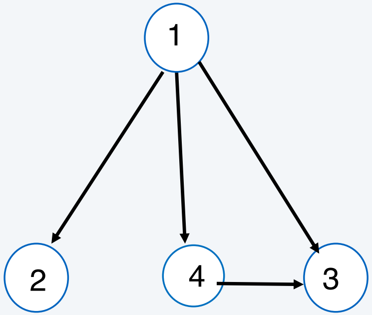
This graph is strongly connected. Why?

Connectivity in Directed Graphs

A directed graph is **weakly connected** if it is connected if the direction of edges are ignored.

A directed graph is **connected** if for every pair of vertices u and v, there exists a path u to v or a path v to u in the graph.

A directed graph is **strongly connected** if for every pair of vertices u and v there exists a path from u to v and v to u.



Example:

This graph is weakly connected. Why? undirected underlying graph is connected

This graph is not connected. Why? no, path 2 to 4 or 4 to 2

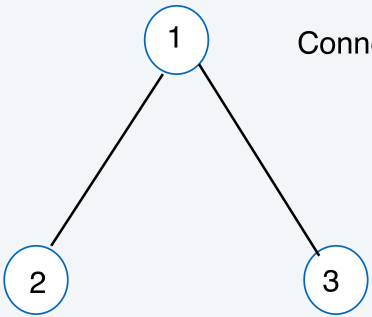
This graph is not strongly connected. Why?

since not connected

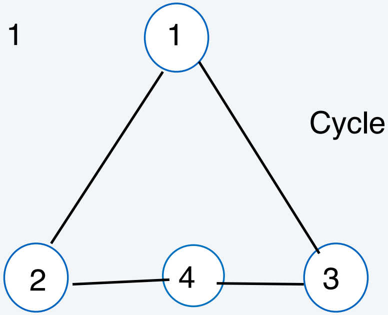
Connected Components

A **connected component** is a subgraph consisting of a vertex, v*, and all vertices and edges connected to v*.

A **connected graph** is a single connected component.



Connected component of vertex 1



C_4

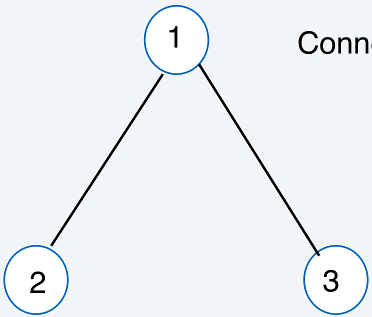
Cycle graph on 4 vertices:

$$C_n$$

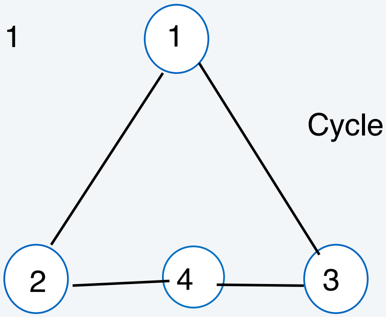
Connected Components

A **connected component** is a subgraph consisting of a vertex, v*, and all vertices and edges connected to v*.

A **connected graph** is a single connected component.



Connected component of vertex 1



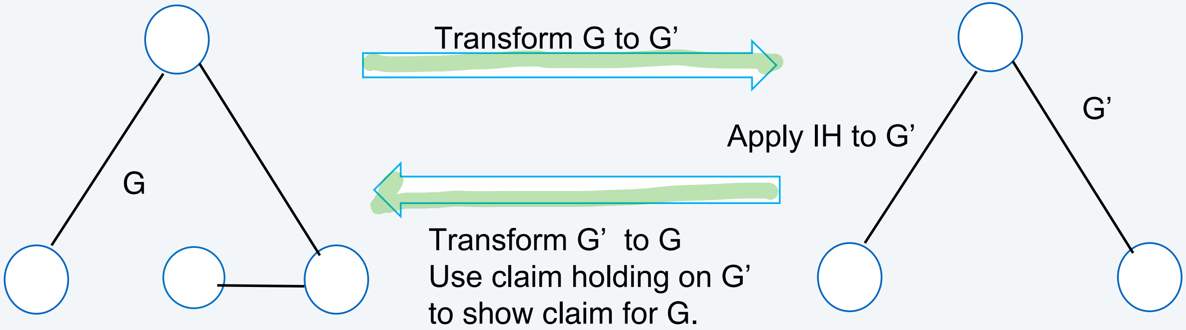
Cycle graph on 4 vertices:

$$C_n$$

Graph Induction Template

Shrink Down, Build Up Approach

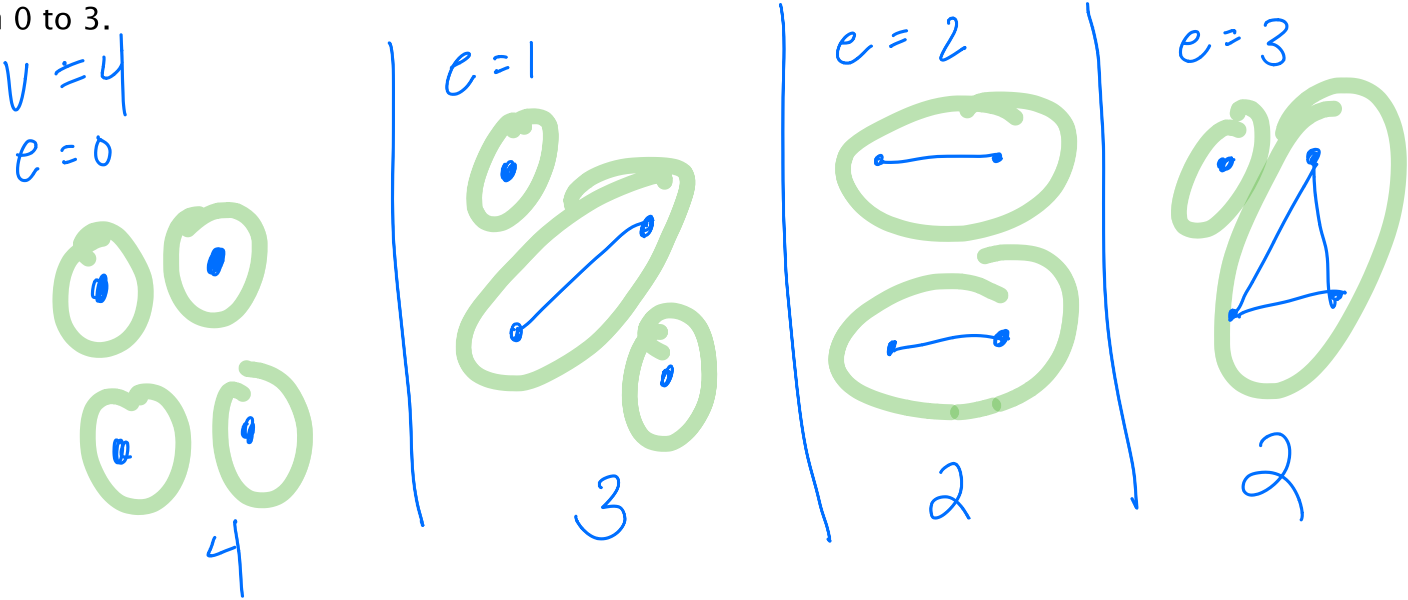
1. Start from G a graph such that the premise for the inductive step holds
2. Perform graph operations on G to construct G' a smaller graph such that the premise of an inductive hypothesis holds
3. Use the Inductive Hypothesis to assert that the claim holds for G'
4. Construction G from G'. Use the claim holding from the inductive hypothesis on G' to show that the claim holds for G in the inductive step.



Theorem: Connected Components

Theorem: Every graph with v vertices and e edges has at least v-e connected components.

First, let's see what this theorem means when there are 4 vertices and the number of edges increases from 0 to 3.



Theorem: Connected Components

Theorem: Every graph with v vertices and e edges has at least v-e connected components.

Let's prove by induction over the number of edges.

P(e): Every graph with v vertices and e edges has at least v-e connected components.

Base case: P(0): Every graph with v vertices and no edges has at least v connected components. Every vertex is a connected component, so the claim holds.

Assume the claim holds for some fixed arbitrary number of edges k (>= 0), P(k):

A graph with v vertices and k edges has at least v-k connected components.

Show the claim holds for a graph with v vertices and k+1 edges.

Theorem: Every graph with v vertices and e edges has at least v-e connected components.

Proof (continued): Inductive step

Let G be a graph with v vertices and k+1 edges.

(Transform G to G')

Remove any edge e^* from G to construct G'.

G' has v vertices and k edges.

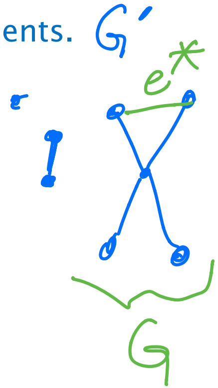
$\Rightarrow$ by IH G' has v - k connected components

Theorem: Connected Components

Theorem: Every graph with v vertices and e edges has at least v-e connected components.

Proof (continued): Inductive step

(Transform G' back to G and show claim holds)

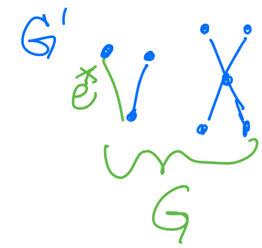


Add e^* to G' to form G again.

Case 1: e^* is added within a connected component of G'

Since G' has $\geq v-k$ connected components and $v-k \geq v-(k+1)$

$\Rightarrow$ G has $v-k$ connected components & claim holds case 2: e^* connects 2 connected components of G' in recreating G



Since G' has $\geq V-K$ connected components

$\Rightarrow$ G has $V-K-1 \geq V-(K+1)$ connected components so claim hold

Graph Theory Introduction

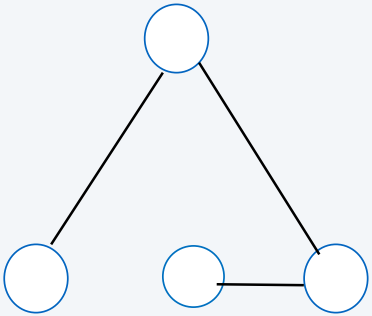
- Graph Definitions
- Paths and Cycles
- Graph Coloring
- **Trees**
- Connectivity

1.1–1.2

Trees

An undirected graph is a **tree** if it is connected and has no cycles.

Trees are bipartite and two-colorable.



Rooted M-ary Trees

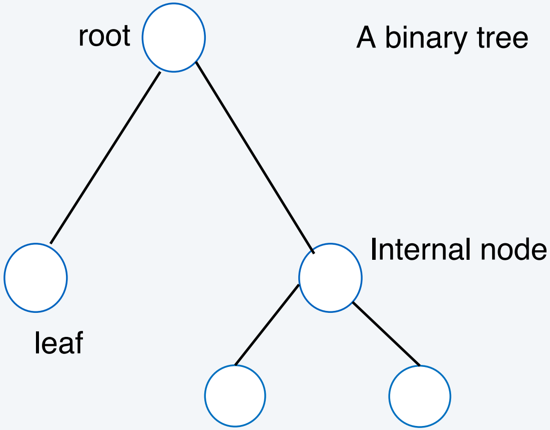
A **root node**: Node designated as start of the tree

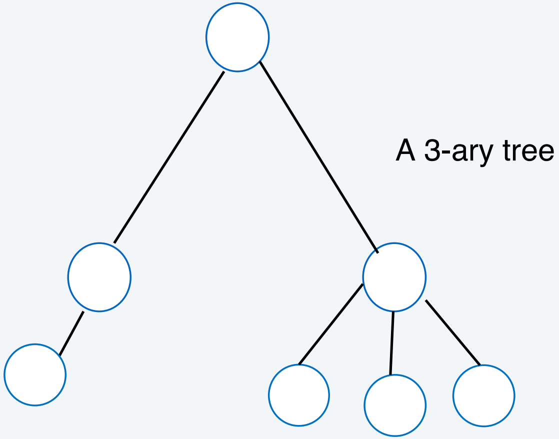
Leaf node: A node with no children (More generally a leaf node in a tree has degree 1.)

Internal nodes: Nodes with at least one child node

Internal nodes have at most m children

If m=2, **binary** tree





Binary Search Trees

A **binary search tree (BST)** is a binary tree with the **binary search tree (BST) property**.

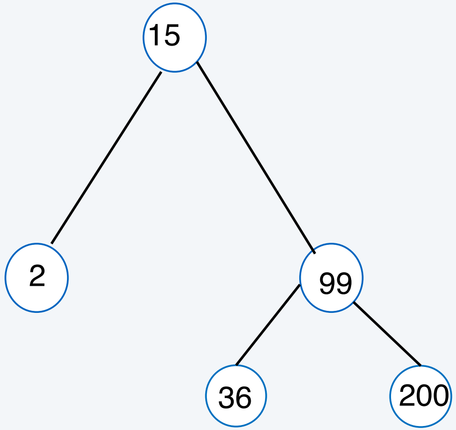
A BST is empty or two disjoint BSTs a left and right.

The **BST property** is that every node has a key value, x, such that

i) Every value of nodes in its left subtree are less than x

ii) Every value of nodes in its right subtree are greater than x

.



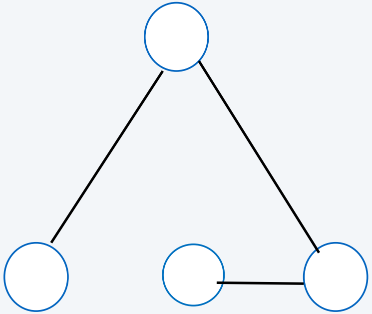
Theorems on Trees

Theorem: If a tree has at least two vertices, then it has at least two leaf nodes.

Theorem: A tree on n vertices has exactly n-1 edges for all n greater than or equal to 1.

Theorem: We can show that any two of the following properties imply the third:

- G is connected
- G has no cycles
- G has n-1 edges



Theorem: The number of edges of a tree

Theorem: A tree on n vertices has exactly n-1 edges for all n greater than or equal to 1.

Let's prove by induction over the number of vertices

P(n): A tree on n vertices has exactly n-1 edges.

Base case: P(1): A tree with 1 vertex, has no edges so claim holds.

Assume the claim holds for some fixed arbitrary number n (>=1), i.e. P(n) holds for some fixed n >=1.

Show the claim holds for n+1, i.e. a tree on n+1 vertices has n edges.

Theorem: The number of edges of a tree (cont.)

Show the claim holds for n+1, i.e. a tree on n+1 vertices has n edges

Let G be graph w/ n+1 vertices. G tree - acyclic & connected, G has at least 2 vertices ⇒ G has a leaf node, v^*.

- Remove v^* and edge incident to it e^* to construct G'
- Consider G' ⇒ G' has n vertices

  ⇒ G' is acyclic (since G is and removing edge & vertex cannot create cycle)

$\Rightarrow G'$ is connected (because removed leaf node & unique edge connected to it)

$\Rightarrow G'$ is a tree on n vertices

$\Rightarrow$ by IH G' has $n-1$ edges

Add v^* and e^* back to G' to reconstruct G $\Rightarrow$ (since v^* is leaf and e^* its unique incident edge G still tree) $\Rightarrow G$ has n edges