Graphs and Trees

CSX2008 Mathematics Foundation for Computer Science

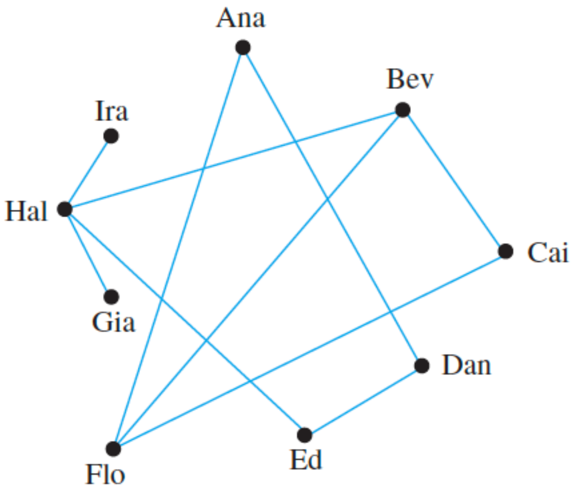
Department of Computer Science
Vincent Mary School of Science and Technology
Assumption University

Session Outline

- Graph
  - Definition of Graphs
  - Basic Properties of Graphs
  - Representation of Graphs
  - Isomorphism
  - Bipartite Graphs
  - Graph Algorithms
- Trees
  - Definition of Trees
  - Representation of Trees
  - Tree Traversal
  - Tree Algorithms

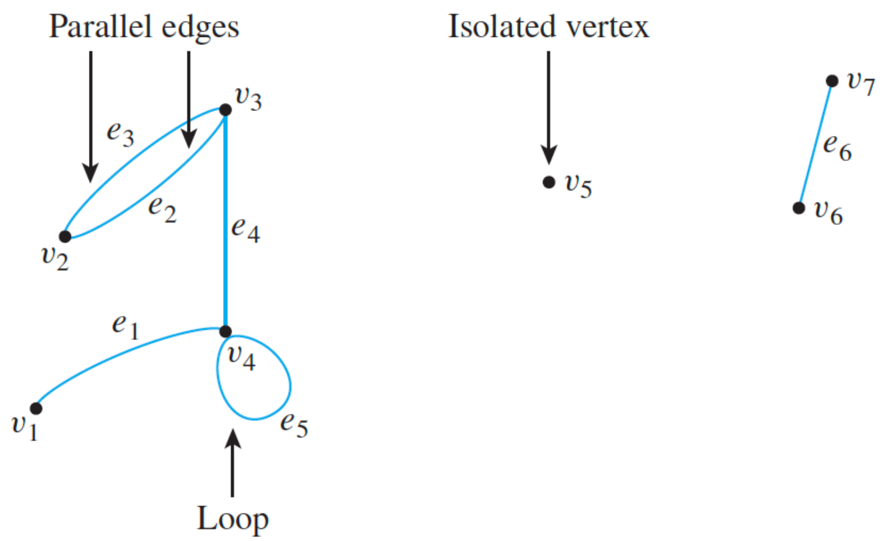
Graph

Name	Past Partners
Ana	Dan, Flo
Bev	Cai, Flo, Hal
Cai	Bev, Flo
Dan	Ana, Ed
Ed	Dan, Hal
Flo	Cai, Bev, Ana
Gia	Hal
Hal	Gia, Ed, Bev, Ira
Ira	Hal

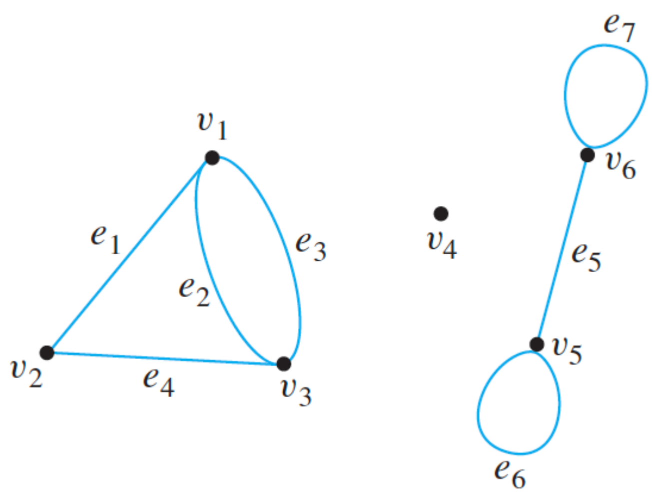


Definition: Graph

- A graph G consists of two finite sets:
  - a nonempty set V(G) of vertices, and
  - a set E(G) of edges, where each edge is associated with a set consisting of either one or two vertices called its endpoints.



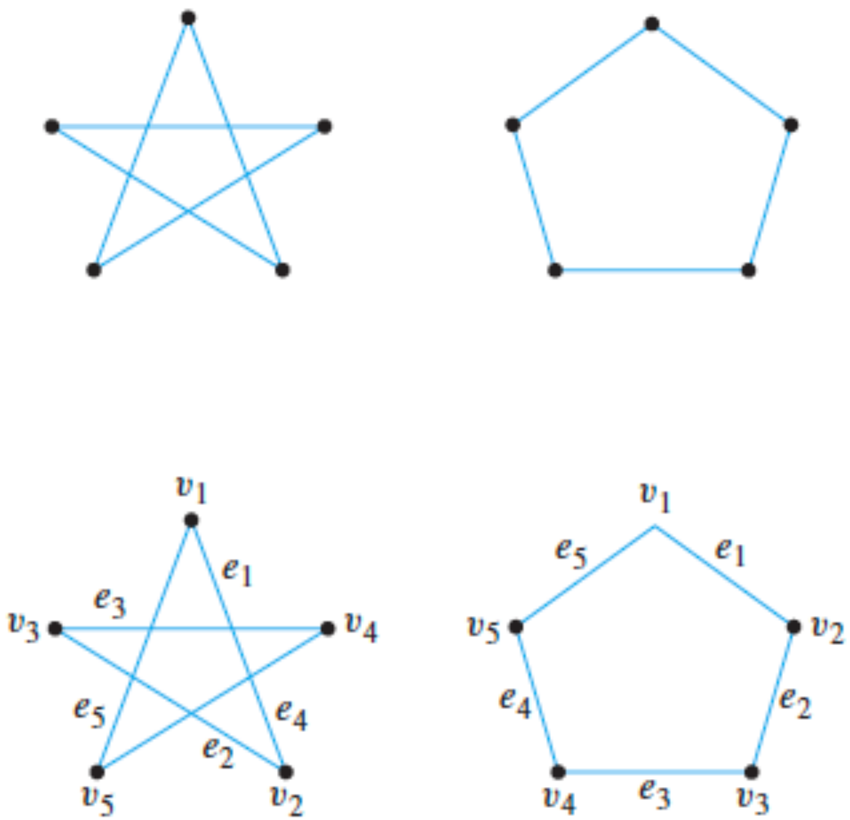
Example



- Write the vertex set and the edge set, and give a table showing the edge-endpoint function.
- Find the following:
  - all edges that are incident on v1,
  - all vertices that are adjacent to v1,
  - all edges that are adjacent to e1,
  - all loops,
  - all parallel edges,
  - all vertices that are adjacent to themselves, and
  - all isolated vertices.

Edge	Endpoints
e_1	$\{v_1, v_2\}$
e_2	$\{v_1, v_3\}$
e_3	$\{v_1, v_3\}$
e_4	$\{v_2, v_3\}$
e_5	$\{v_5, v_6\}$
e_6	$\{v_5\}$
e_7	$\{v_6\}$

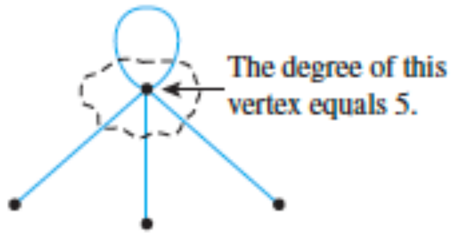
Example



Edge	Endpoints
e_1	$\{v_1, v_2\}$
e_2	$\{v_2, v_3\}$
e_3	$\{v_3, v_4\}$
e_4	$\{v_4, v_5\}$
e_5	$\{v_5, v_1\}$

Degree

- Let G be a graph and v a vertex of G.
  The degree of *v*, denoted deg(*v*), equals the number of edges that are incident on v, with an edge that is a loop counted twice.
- The total degree of *G* is the sum of the degrees of all the vertices of G.

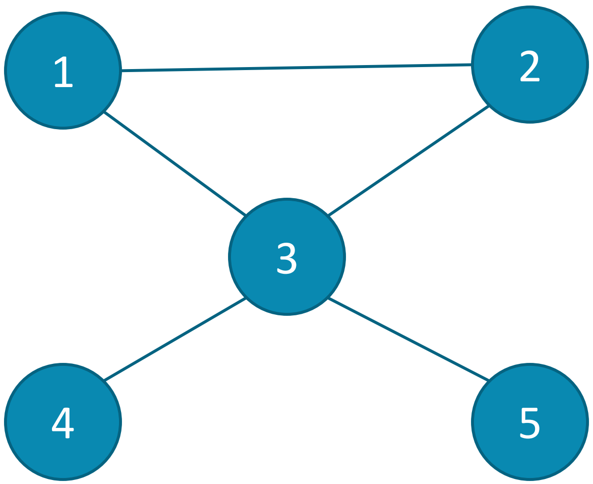


Exercise: Graph

- Consider an undirected graph G = (V, E) where V is the set {1, 2, 3, 4} and E is the set {{1, 2}, {2, 3}, {1, 3}, {2, 4}, {1, 4}}.
  - Draw a graph G.
  - What is the degree of node 1?
  - Indicate a path from node 3 to node 4 on the drawing of G.

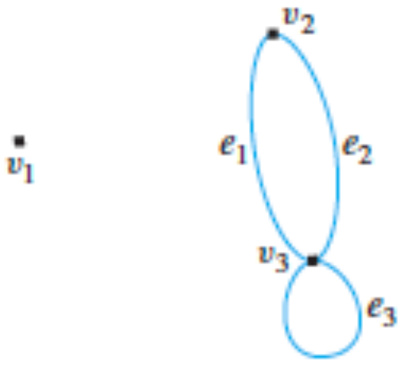
Exercise

- Write a formal description of the following graph.



Exercise: The Concept of Degree

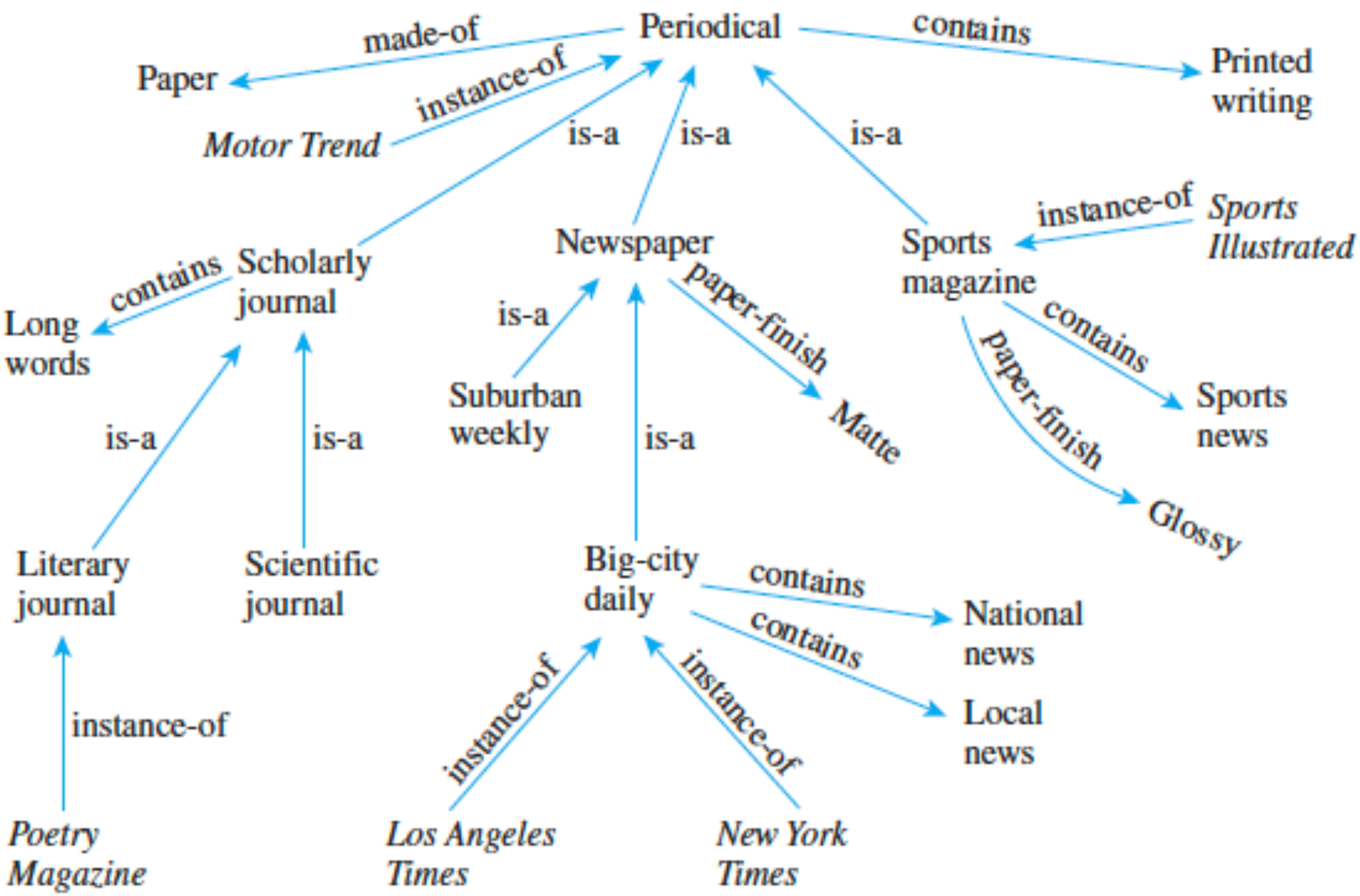
- Find the degree of each vertex of the graph G.
- Find the total degree of G.



Directed Graph

- A directed graph, or digraph, consists of two finite sets:
  - a nonempty set V(G) of vertices and
  - a set D(G) of directed edges, where each is associated with an ordered pair of vertices called its endpoints.
- If edge e is associated with the pair (v, w) of vertices, then e is said to be the (directed) edge from v to w.

Representing Knowledge using Graphs

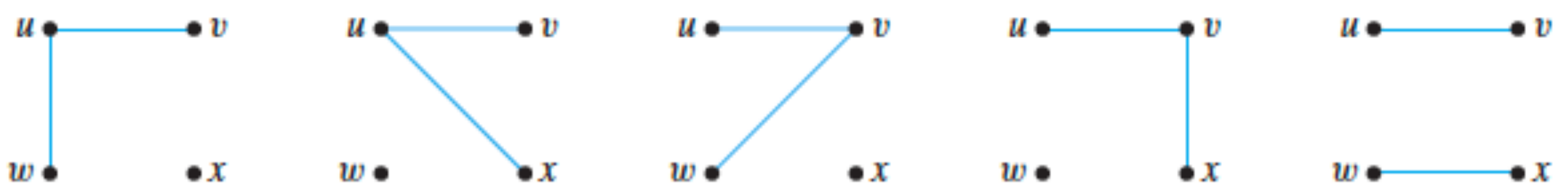


Simple Graph

- A simple graph is a graph that does not have any loops or parallel edges.
- In a simple graph, an edge with endpoints v and w is denoted {v, w}.

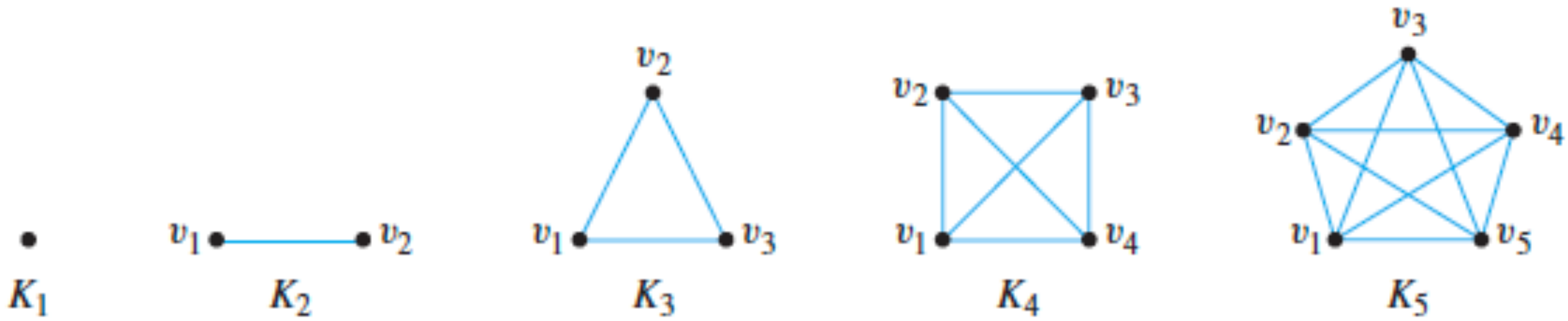
Exercise

- Draw all simple graphs with the four vertices { u, v, w, x } and two edges, one of which is {u, v}



Complete Graph

- Let n be a positive integer.
- A complete graph on n vertices, denoted K_n, is a simple graph with n vertices and exactly one edge connecting each pair of distinct vertices.



(Complete) Bipartite Graph

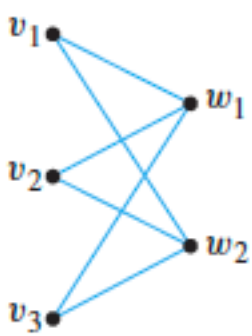
Let *m* and *n* be positive integers.

A complete bipartite graph on (*m*, *n*) vertices, denoted $K_{m,n}$, is a simple graph with distinct vertices $v_1, v_2, \ldots, v_m$ and $w_1, w_2, \ldots, w_n$ that satisfies the following properties:

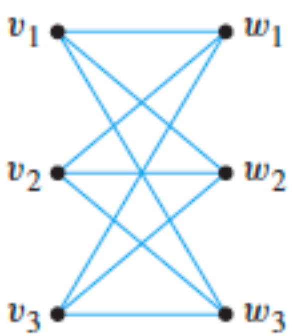
For all i, k = 1, 2, . . . ,m and for all j, l = 1, 2, . . . , n,

1. There is an edge from each vertex v_i to each vertex w_j .
2. There is no edge from any vertex v_i to any other vertex v_k.
3. There is no edge from any vertex w_j to any other vertex w_l.

Example: Complete Bipartite Graph



$K_{3,2}$



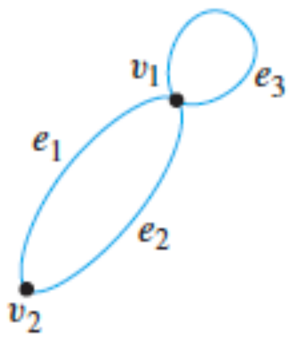
$K_{3,3}$

Subgraph

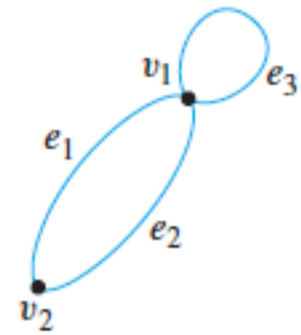
- A graph H is said to be a subgraph of a graph G if, and only if, every vertex in H is also a vertex in G, every edge in H is also an edge in G, and every edge in H has the same endpoints as it has in G.

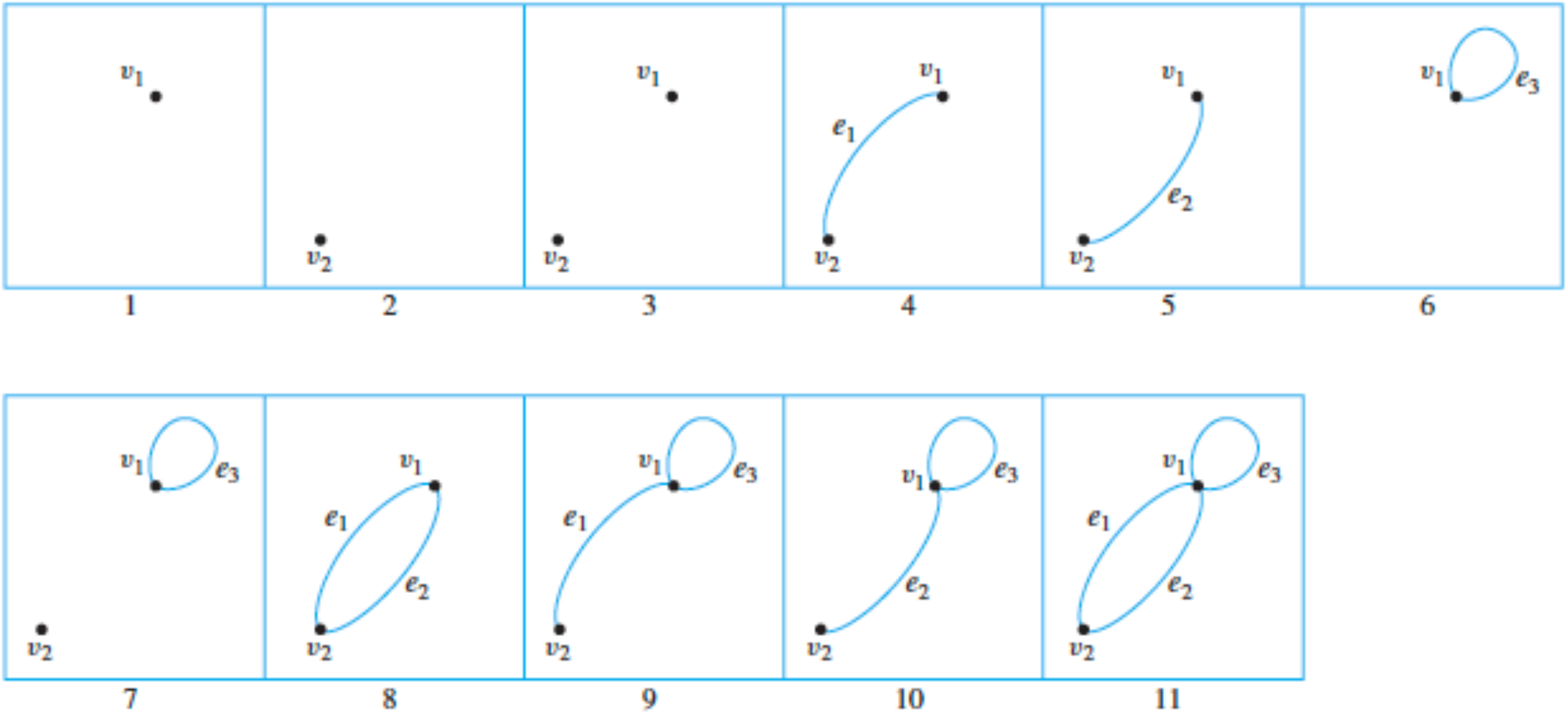
Exercise: Subgraph

- List all subgraphs of the graph *G* with vertex set $\{v_1, v_2\}$ and edge set $\{e_1, e_2, e_3\}$, where the endpoints of e_1 are v_1 and v_2, the endpoints of e_2 are v_1 and v_2, and e_3 is a loop at v_1.
- What does the graph *G* look like?



Exercise: Subgraph





Walk, Trail, Path and Circuit

	Repeated Edge?	Repeated Vertex?	Starts and Ends at Same Point?	Must Contain at Least One Edge?
Walk	allowed	allowed	allowed	no
Trail	no	allowed	allowed	no
Path	no	no	no	no
Closed walk	allowed	allowed	yes	no
Circuit	no	allowed	yes	yes
Simple circuit	no	first and last only	yes	yes

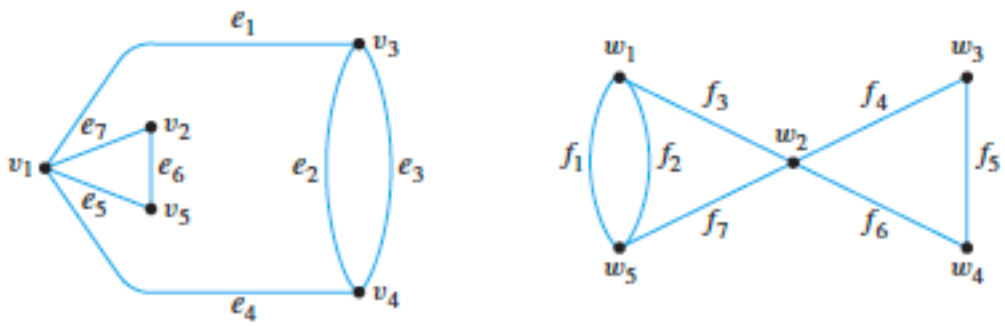
Isomorphism

- Let G and G$ be graphs with vertex sets V(G) and V(G’) and edge sets E(G) and E(G’), respectively.

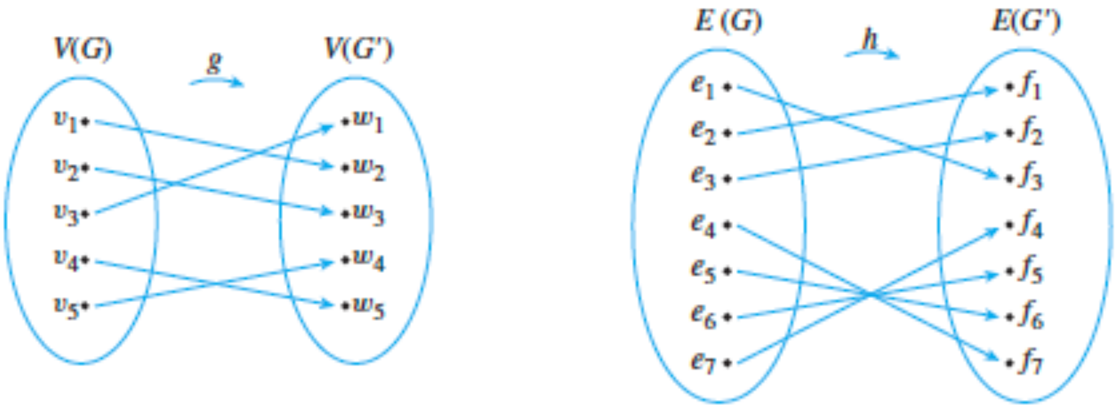
- G is isomorphic to G if, and only if, there exist one-to-one correspondences $g: V(G) \to V(G')$ and $h: E(G) \to E(G')$ that preserve the edge endpoint functions of G and G’ in the sense that for all $v \in V(G)$ and $e \in E(G)$,

$$v \text{ is an endpoint of } e \Leftrightarrow g(v) \text{ is an endpoint of } h(e).$$

Example: Isomorphic Graphs



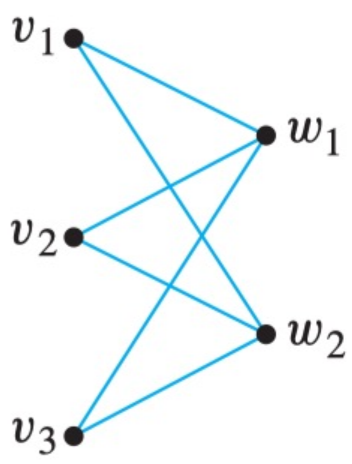
- Show that these two graphs are isomorphic.

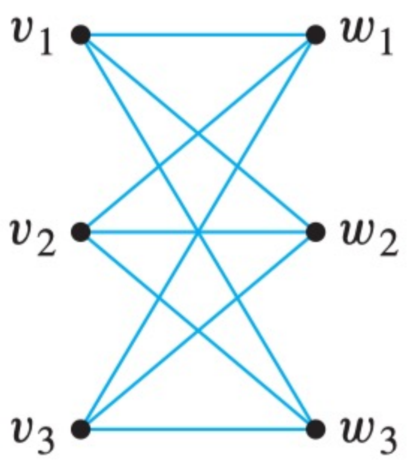


Bipartite Graph

- A graph is bipartite

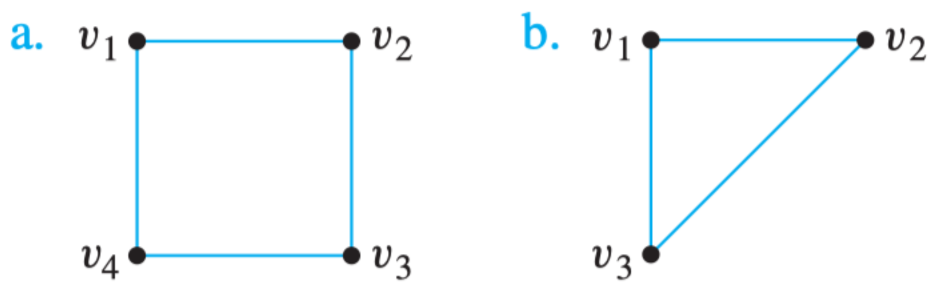
Example: Bipartite Graphs





Exercise 1

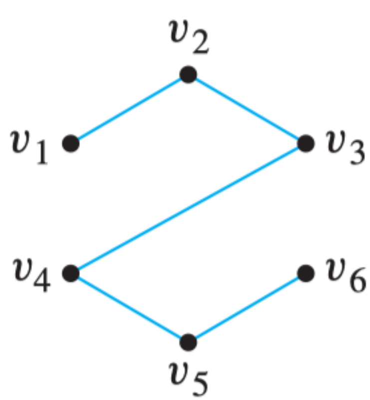
- Which of the following graphs is bipartite? Redraw the bipartite graphs so that their bipartite nature is evident



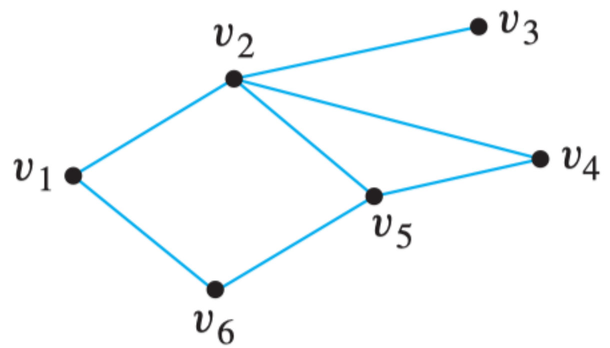
Exercise 2

- Which of the following graphs is bipartite? Redraw the bipartite graphs so that their bipartite nature is evident

c.



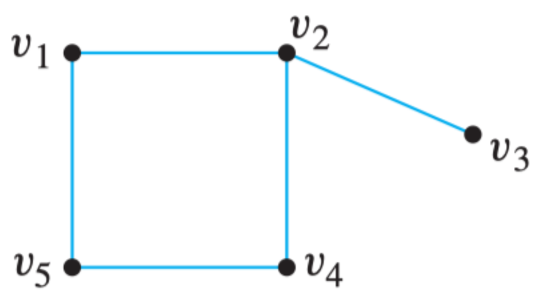
d.



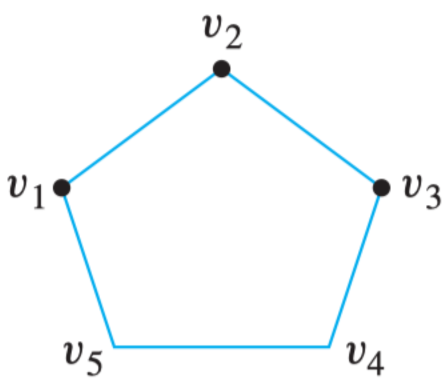
Exercise 3

- Which of the following graphs is bipartite? Redraw the bipartite graphs so that their bipartite nature is evident.

e.



f.



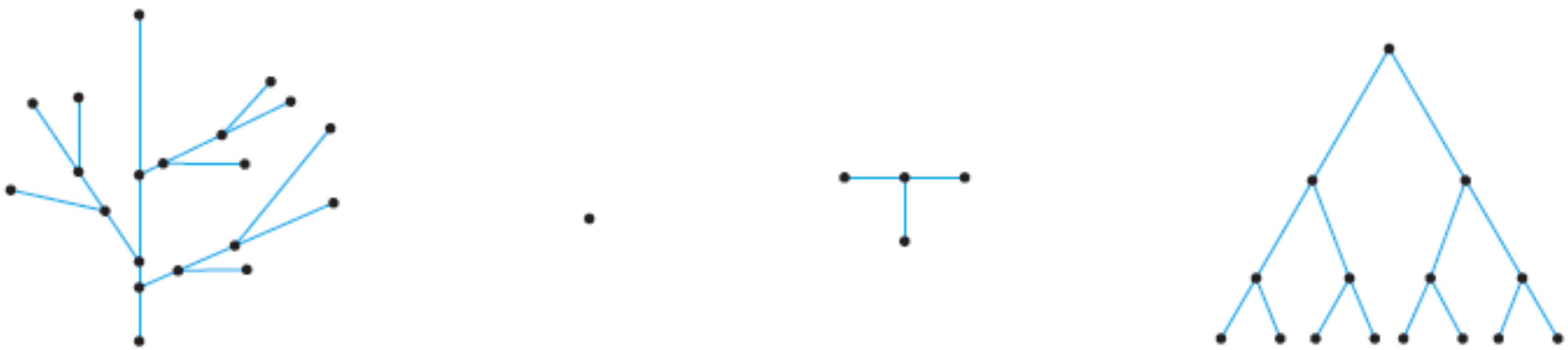
Tree Definition

- Trees are a particular type of graph.
- A tree is a directed graph that
  - has no cycles,
  - has a distinct vertex with no incoming edges called the root.
- Leaf
  - A vertex with no outgoing edges
- Level
  - The number of edges in the path from the root to that vertex
- Height
  - The largest level number of any vertex

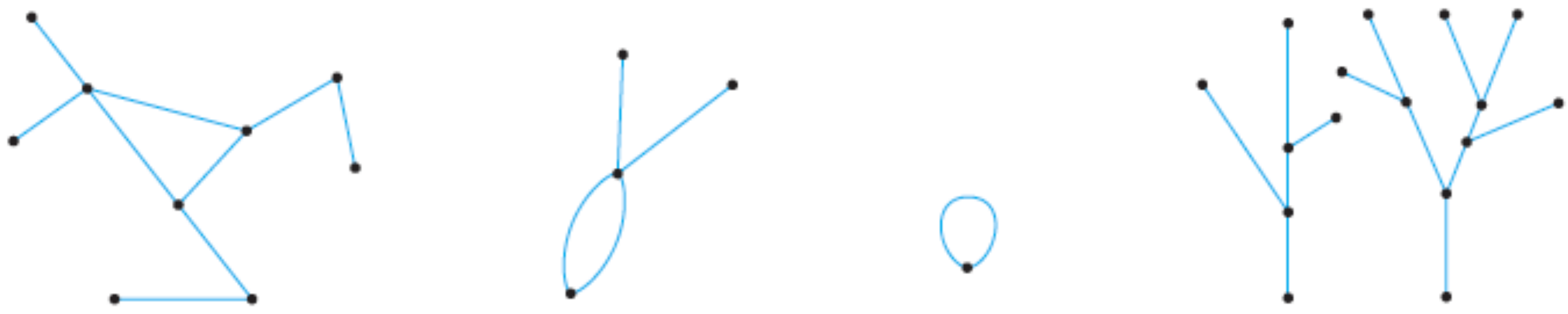
Tree

- A graph is said to be circuit-free if, and only if, it has no circuits (cycles).
- A graph is called a tree if, and only if, it is circuit-free and connected.
- A trivial tree is a graph that consists of a single vertex.
- A graph is called a forest if, and only if, it is circuit-free and not connected.

Example: Trees



Example: Non-Trees



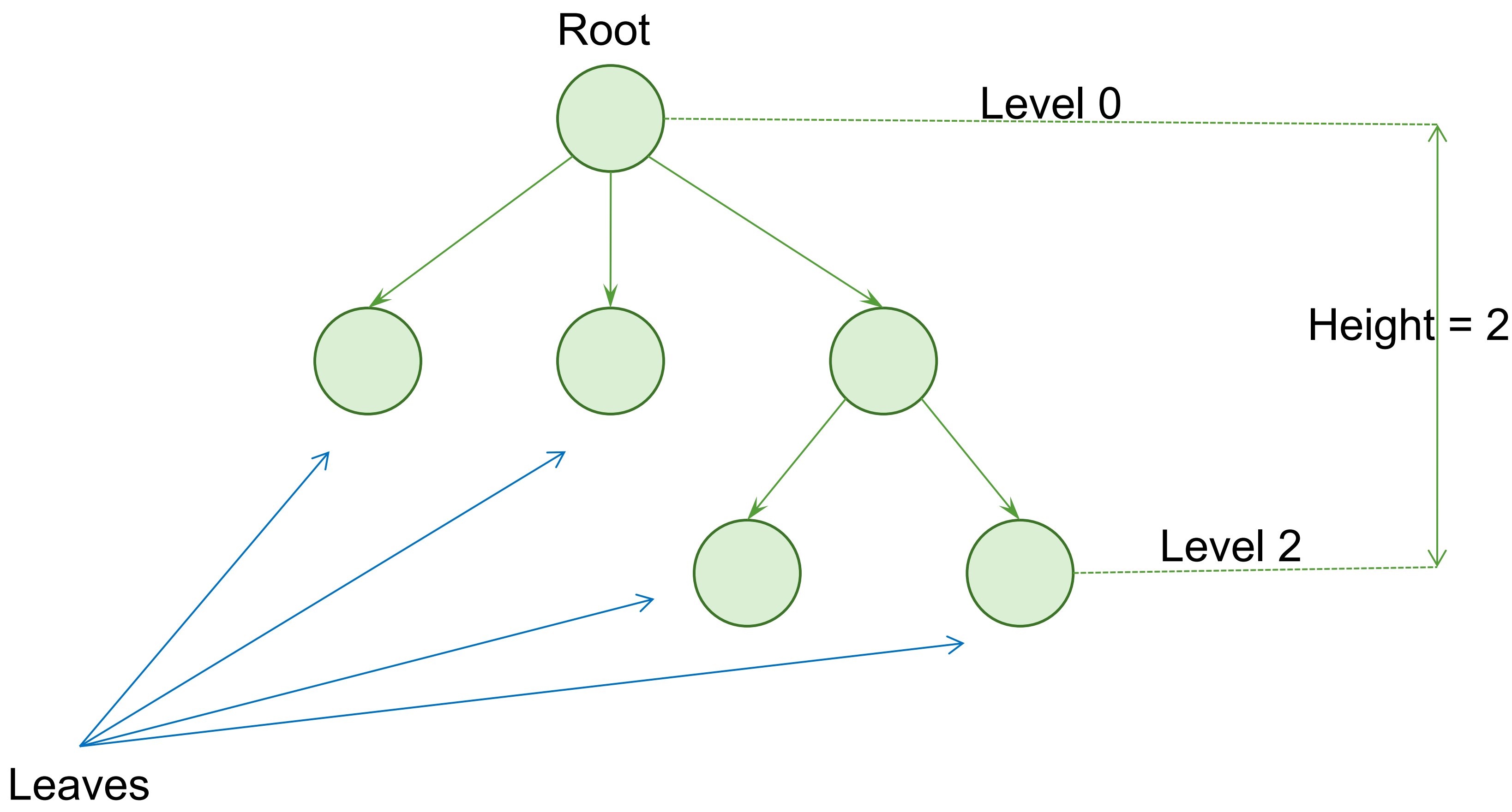
Leaf and Branch Vertex

- Let T be a tree.
- If T has only one or two vertices, then each is called a terminal vertex.
- If T has at least three vertices, then a vertex of degree 1 in T is called a terminal vertex (or a leaf), and a vertex of degree greater than 1 in T is called an internal vertex (or a branch vertex).

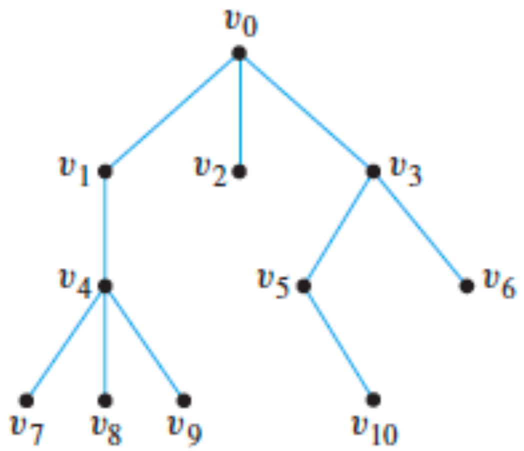
Rooted Tree

- A rooted tree is a tree in which there is one vertex that is distinguished from the others and is called the root.
- The level of a vertex is the number of edges along the unique path between it and the root.
- The height of a rooted tree is the maximum level of any vertex of the tree.
- Given the root or any internal vertex v of a rooted tree, the children of v are all those vertices that are adjacent to v and are one level farther away from the root than v.
- If w is a child of v, then v is called the parent of w, and two distinct vertices that are both children of the same parent are called siblings.
- Given two distinct vertices v and w, if v lies on the unique path between w and the root, then v is an ancestor of w and w is a descendant of v.

Example: Rooted Tree

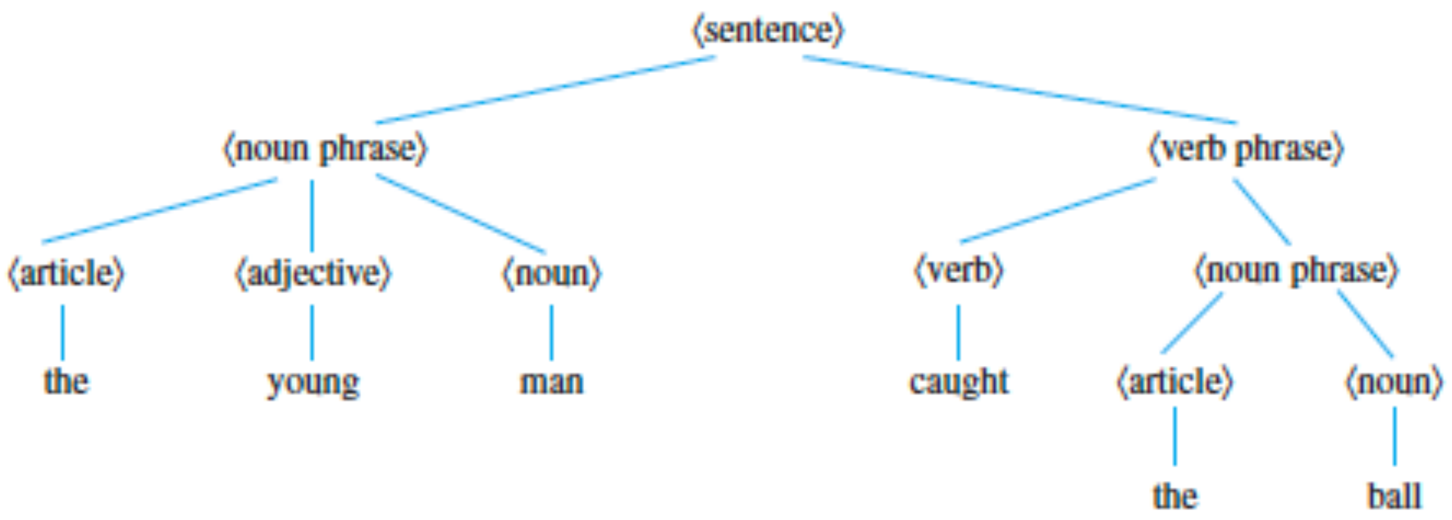


Exercise: Rooted Trees



a. What is the level of v_5?
b. What is the level of v_0?
c. What is the height of this rooted tree?
d. What are the children of v_3?
e. What is the parent of v_2?
f. What are the siblings of v_8?
g. What are the descendants of v_3?

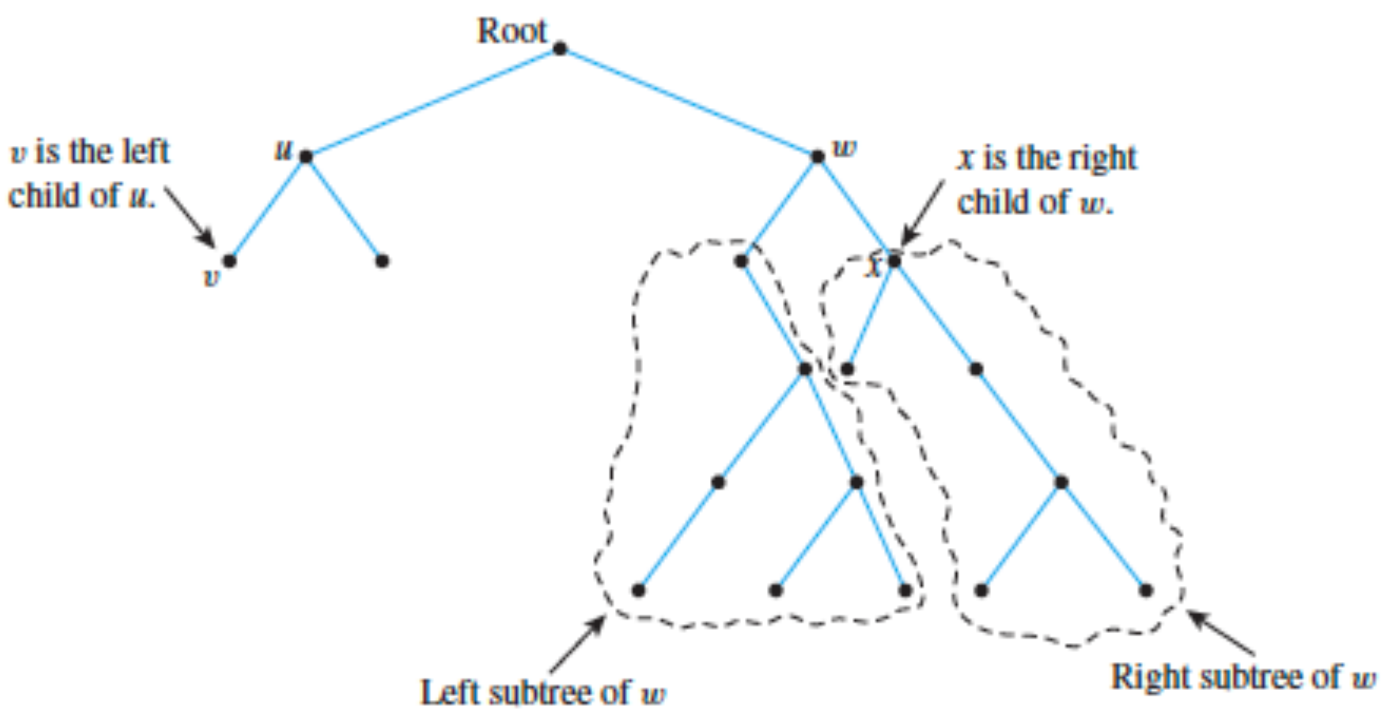
Example: Parse Tree



Binary Tree

- A binary tree is a rooted tree in which every parent has at most two children.
- Each child in a binary tree is designated either a left child or a right child (but not both), and every parent has at most one left child and one right child.
- A full binary tree is a binary tree in which each parent has exactly two children.

Example: Binary Tree

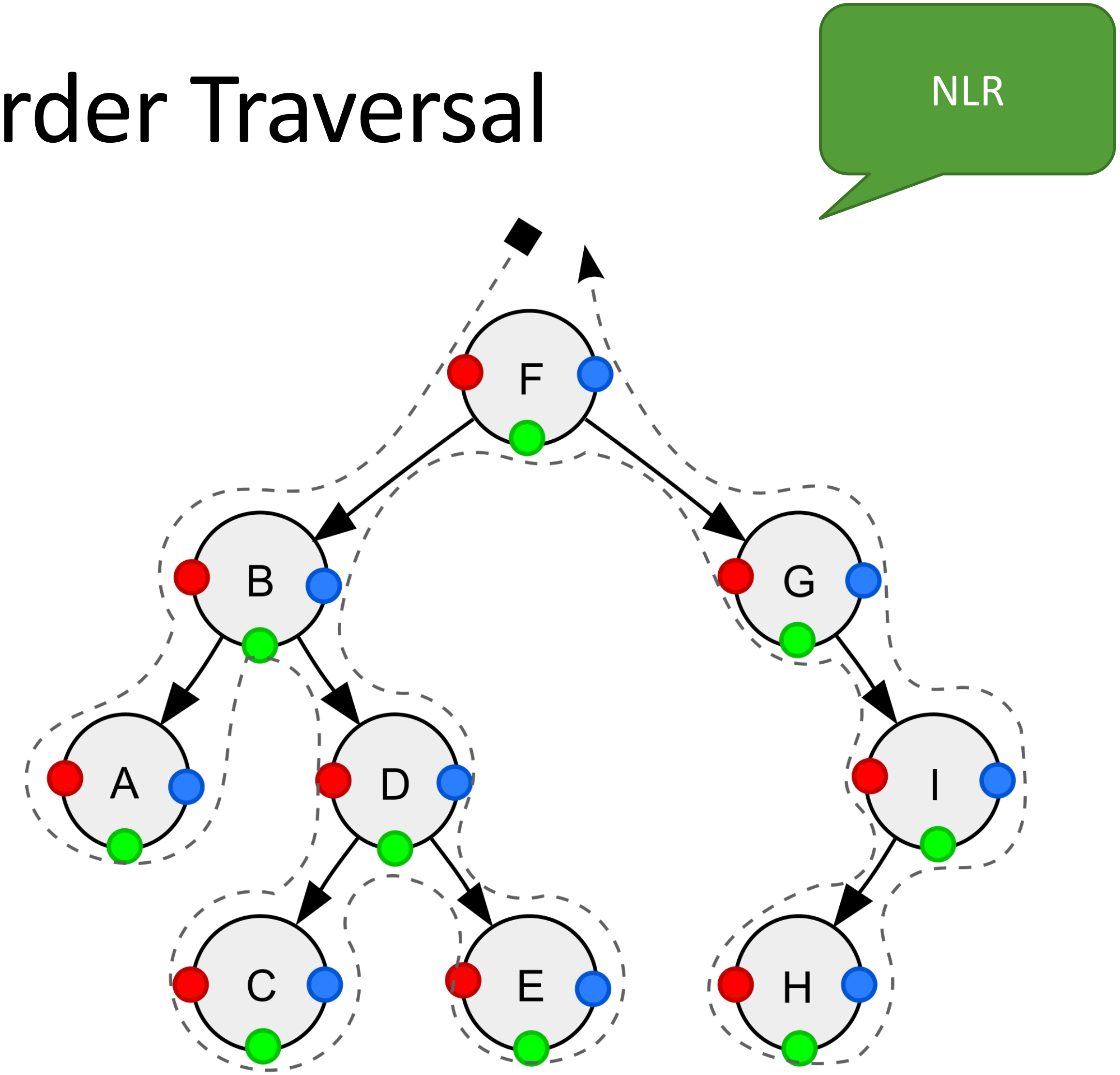


Tree Traversal

- Preorder Traversal
- Inorder Traversal
- Postorder Traversal

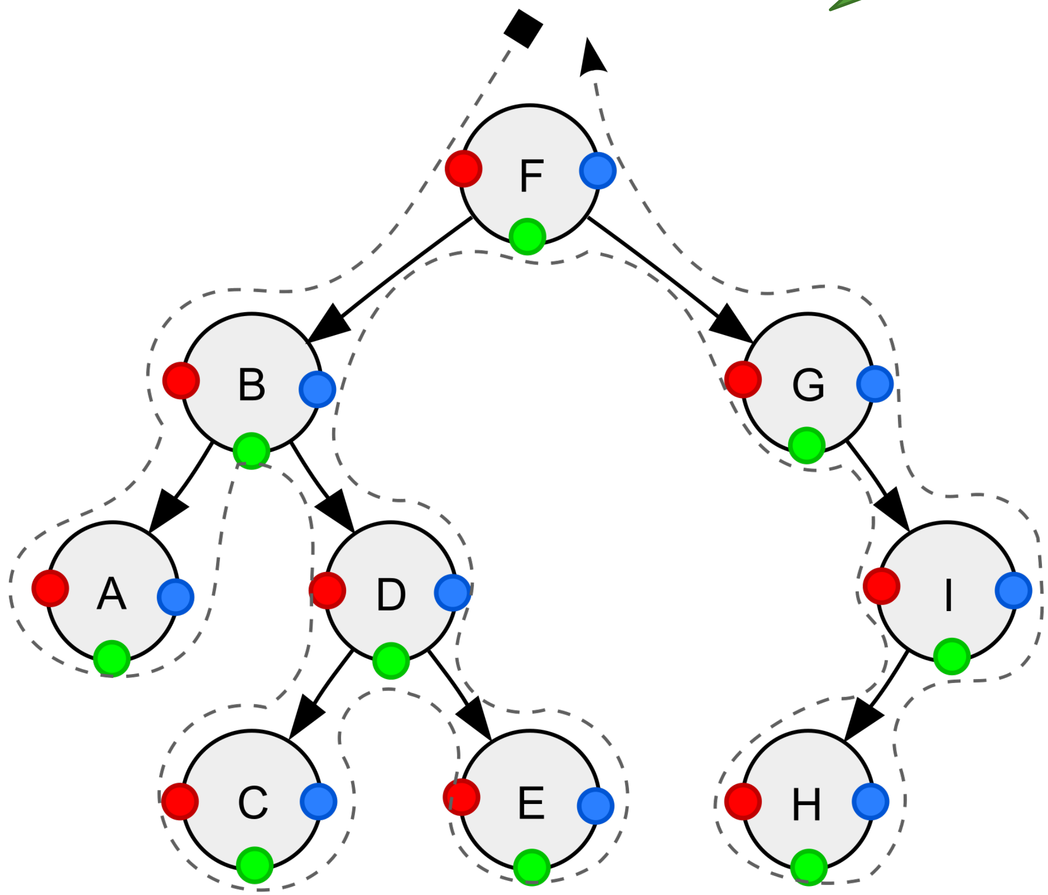
- Traversal involves visiting nodes in specific order:
  - Visit the current node.
  - Recursively traverse the current node's left subtree.
  - Recursively traverse the current node's right subtree.

Preorder Traversal

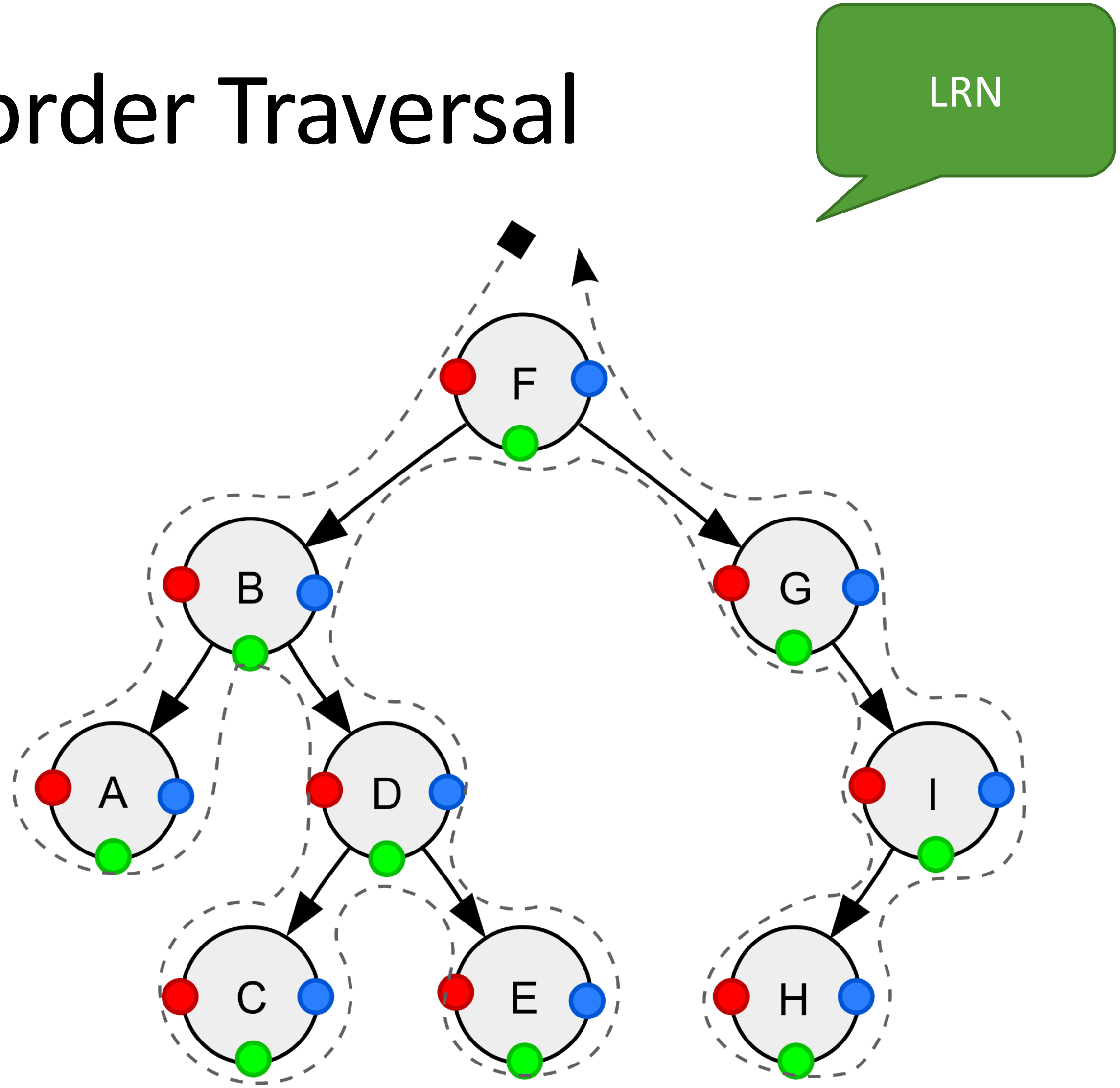


Inorder Traversal

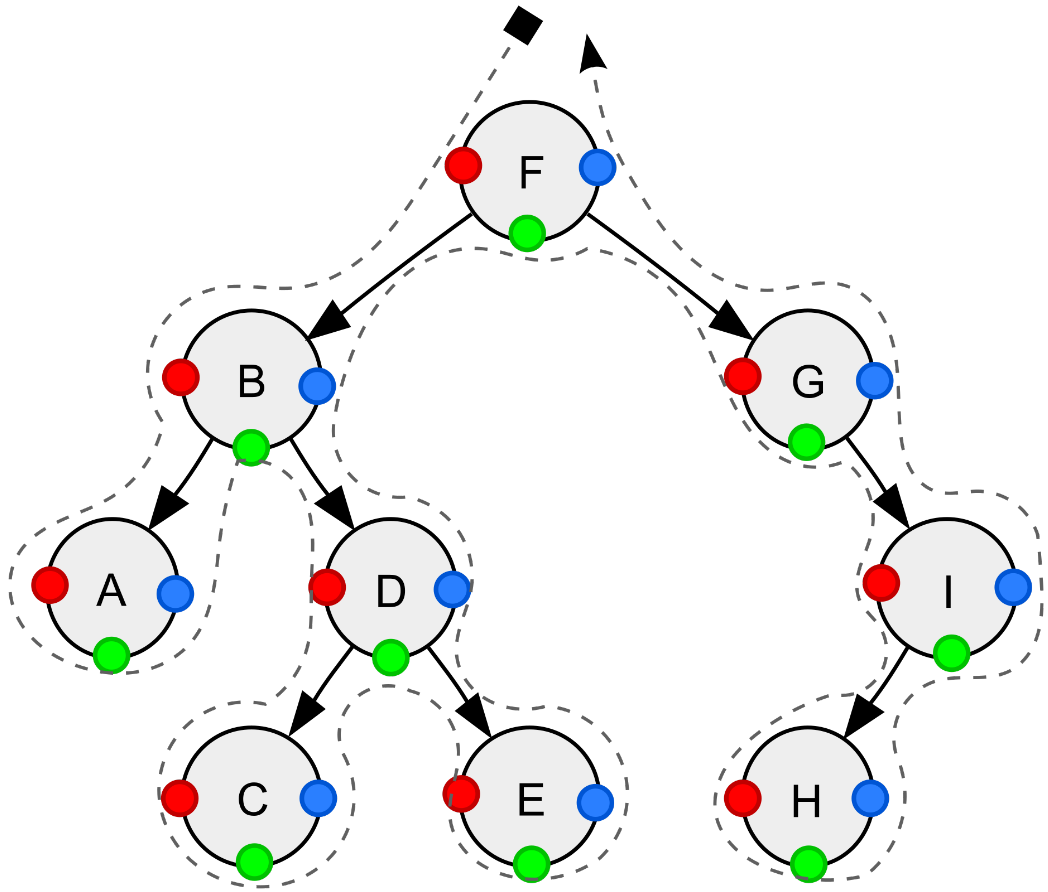
LNR



Postorder Traversal



Depth-First Search



Breadth-First Search