INT102
Algorithmic Foundations And Problem Solving
Graph Theory

Dr Yushi Li
Department of Intelligent Science

西交利物浦大学
Xi'an Jiaotong-Liverpool University

Learning outcomes

- ✓ Able to tell what is an undirected graph **and what is a directed graph**
- ✓ Know how to represent an undirected graph using matrix and list
- ✓ Understand what Euler path / circuit and able to determine whether such path / circuit exists in an undirected graph
- ✓ Able to apply BFS and DFS to traverse a graph
- ➢ Know how to represent a graph using matrix and list
- ➢ Able to tell what a tree is

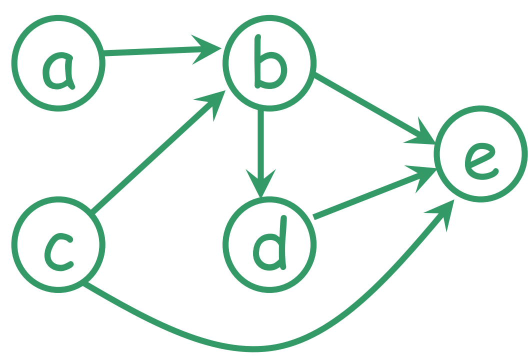
Directed graph …

Directed graph

Given a directed graph G, a vertex **a** is said to be **connected to** a vertex **b** if there is a path from **a** to **b**.

E.g., G represents the routes provided by a certain airline. That means, a vertex represents a city and an edge represents a flight from a city to another city. Then we may ask question like: Can we fly from one city to another?

Reminder: A directed graph G=(V,E) consists of a set of vertices V and a set of edges E. Each edge is an ordered pair of vertices.



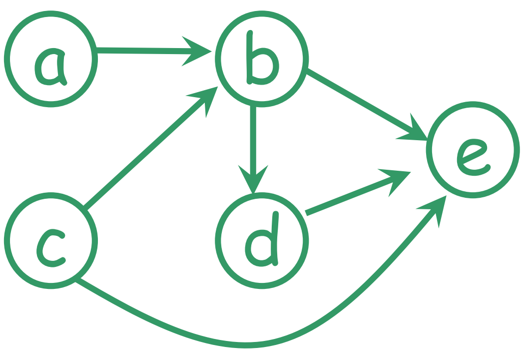
E = { (a,b), (b,d), (b,e), (c,b), (c,e), (d,e) }

N.B. (a,b) is in E, but (b,a) is NOT

In/Out degree (in directed graphs)

The **in-degree** of a vertex *v* is the number of edges *leading to* the vertex *v*.

The **out-degree** of a vertex *v* is the number of edges *leading away* from the vertex *v*.



	in-deg(v)	out-deg(v)	
a	0	1	
b	2	2	
c	0	2	
d	1	1	
e	3	0	
sum:	6	6	Always equal?

Representation (of directed graphs)

Similar to undirected graph, a directed graph can be represented by **adjacency matrix**, **adjacency list**, **incidence matrix** or **incidence list**.

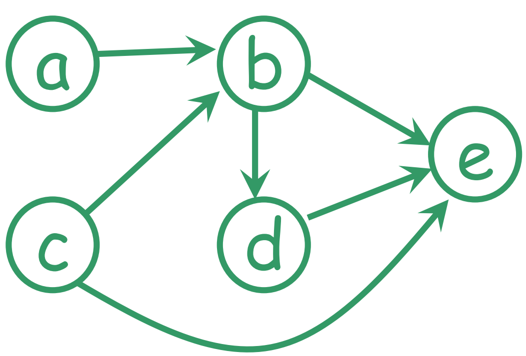
Adjacency matrix / list

Adjacency matrix M for a **directed** graph with n vertices:

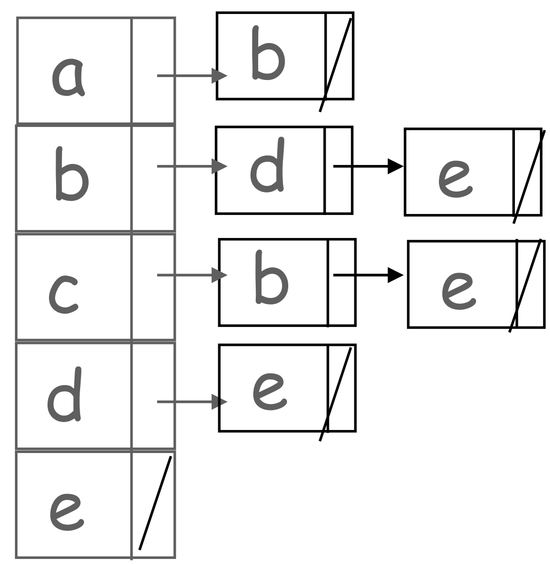
- M is an nxn matrix
- M(i, j) = 1 if (i,j) is an edge

Adjacency list:

- each vertex ***u*** has a list of vertices pointed to by an edge leading away from ***u***



$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

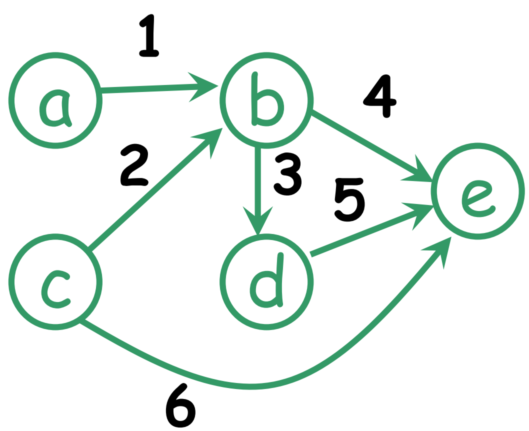


Incidence matrix / list

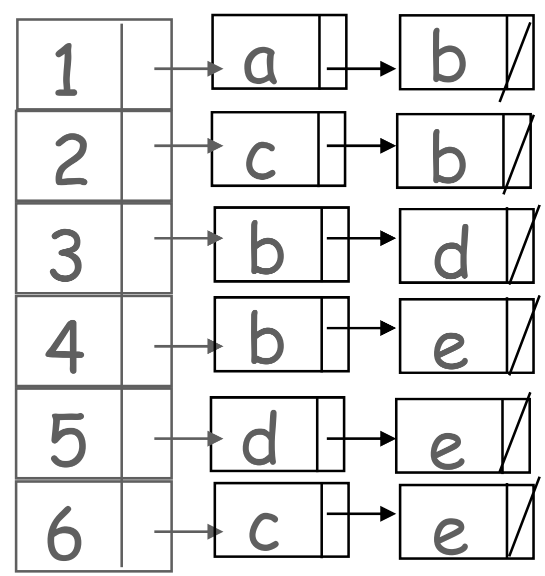
Incidence matrix M for a **directed** graph with n vertices and m edges is an mxn matrix

- M(i, j) = 1 if edge i is leading away from vertex j
- M(i, j)= -1 if edge i is leading to vertex j

Incidence list: each edge has a list of two vertices (leading away is 1st and leading to is 2nd)

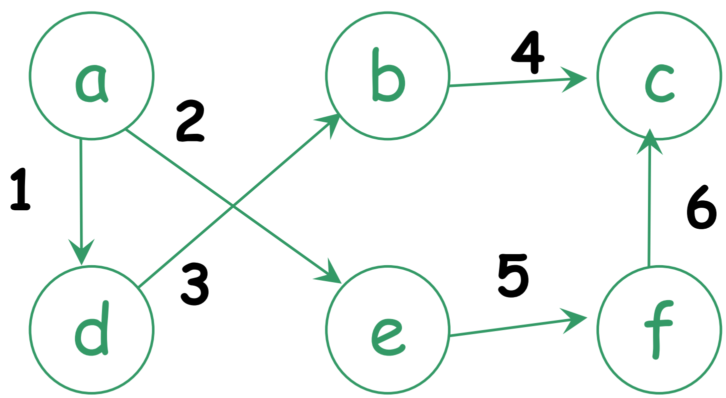


$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 & -1 \end{pmatrix} \end{array}$$



Exercise

Give the adjacency matrix and incidence matrix of the following graph



labels of edge are edge number

Learning outcomes

- ✓ Able to tell what is an undirected graph and what is a directed graph
  - ✓ Know how to represent a graph using matrix and list
- ✓ Understand what Euler path / circuit and able to determine whether such path / circuit exists in an undirected graph
- ✓ Able to apply BFS and DFS to traverse a graph
- ➢ **Able to tell what a tree is**

Tree …

Trees

An undirected graph $G=(V,E)$ is a tree if G is connected and acyclic (i.e., contains no cycles)

Other equivalent statements:

1. There is exactly one path between any two vertices in G
2. G is connected and removal of one edge disconnects G
3. G is acyclic and adding one edge creates a cycle
4. G is connected and $m=n-1$ (where $|V|=n$, $|E|=m$)

Trees

An undirected graph $G=(V,E)$ is a tree if G is connected and acyclic (i.e., contains no cycles)

Other equivalent statements:

1. There is exactly one path between any two vertices in G
   (coz G is connected and acyclic)
2. G is connected and removal of one edge disconnects G
   (removal of an edge {u,v} disconnects at least u and v because of [1])
3. G is acyclic and adding one edge creates a cycle
   (adding an edge {u,v} creates one more path between u and v, a cycle is formed)
4. G is connected and $m=n-1$ (where $|V|=n$, $|E|=m$)

Lemma: P(n): If a tree T has n vertices and m edges, then $m=n-1$.

optional, self-study

Proof: By induction on the number of vertices.

Basic step: A tree with single vertex does not have an edge.

Induction step: $P(n-1) \Rightarrow P(n)$ for $n > 1$?

Remove an edge from the tree T. By [2], T becomes disconnected. Two connected components T_1 and T_2 are obtained, neither contains a cycle (the cycle is also present in T otherwise).

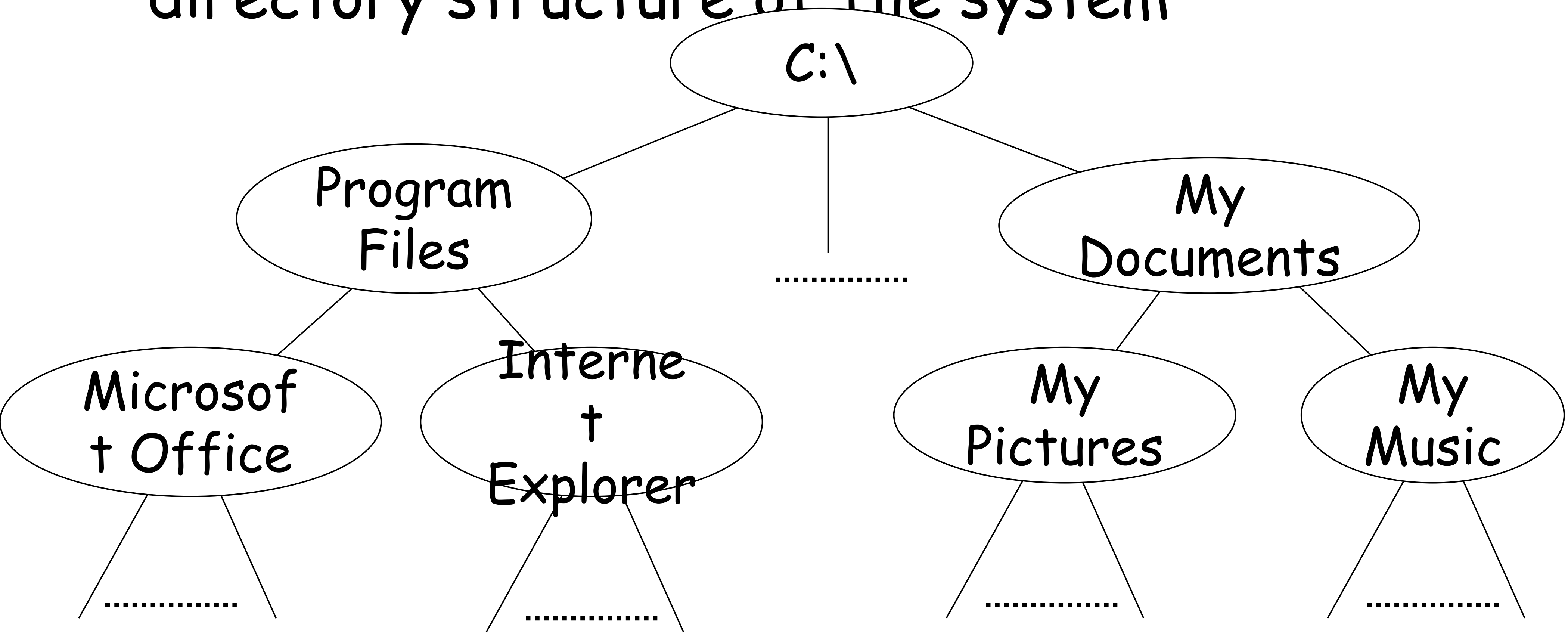
Therefore, both T_1 and T_2 are trees. Let n_1 and n_2 be the number of vertices in T_1 and T_2. $[n_1+n_2 = n]$

By the induction hypothesis, T_1 and T_2 contains n_1-1 and n_2-1 edges.

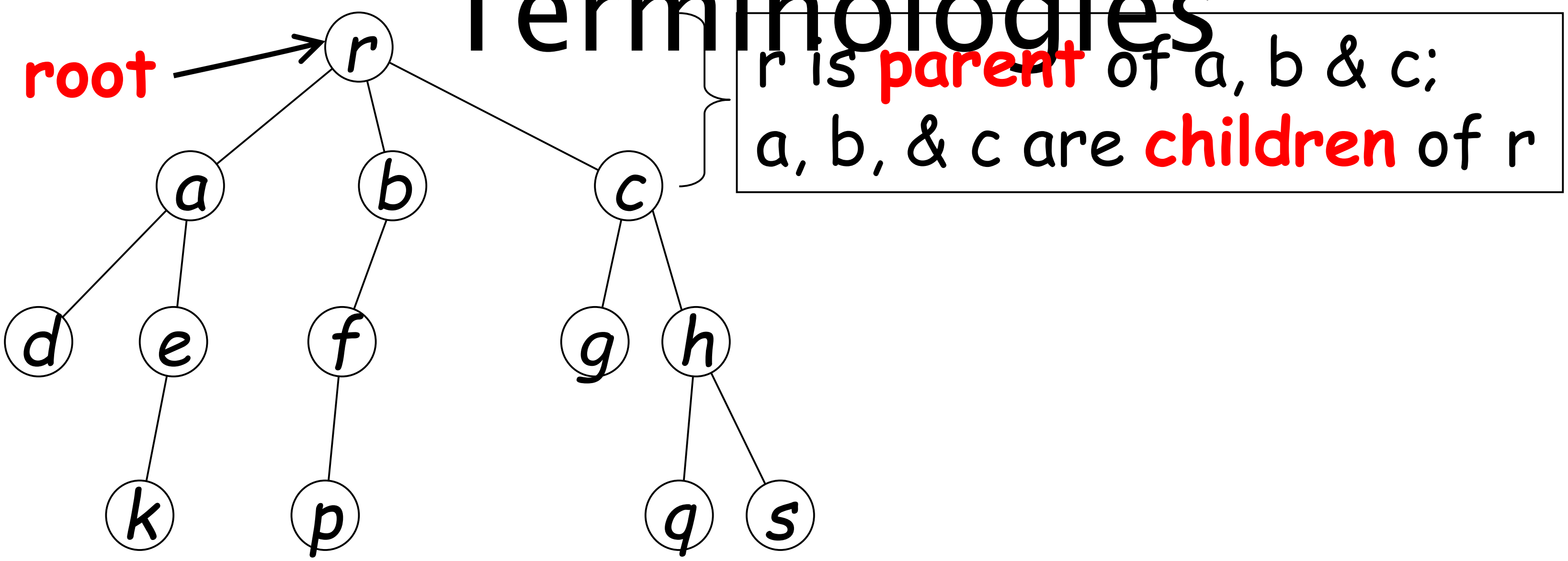
Hence, T contains $(n_1-1) + (n_2-1) + 1 = n-1$ edges.

Rooted trees

Tree with hierarchical structure, e.g., directory structure of file system

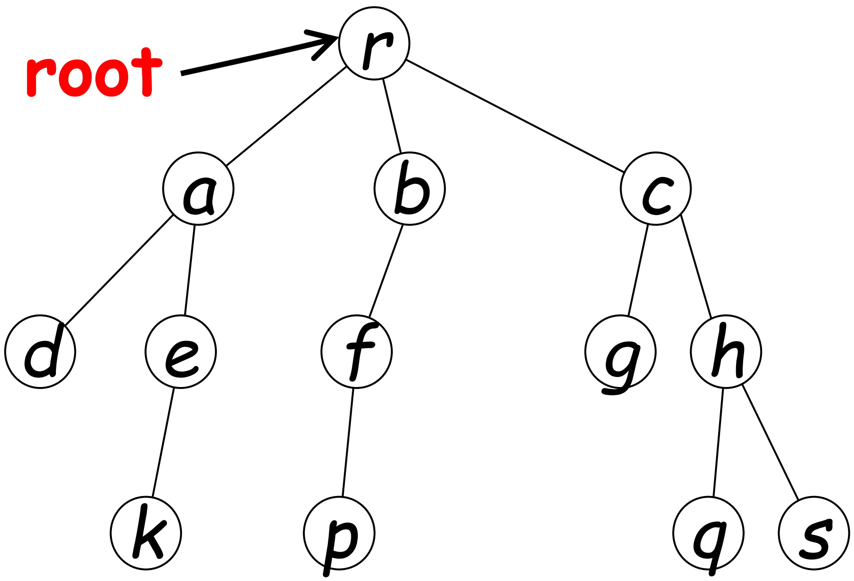


Terminologies



- Topmost vertex is called the **root**.
- A vertex ***u*** may have some **children** directly below it, ***u*** is called the **parent** of its children.
- **Degree** of a ***vertex*** is the no. of children it has. (N.B. it is different from the degree in an unrooted tree.)
- Degree of a ***tree*** is the max. degree of all vertices.
- A vertex with no child (degree-0) is called a **leaf**. All others are called **internal vertices**.

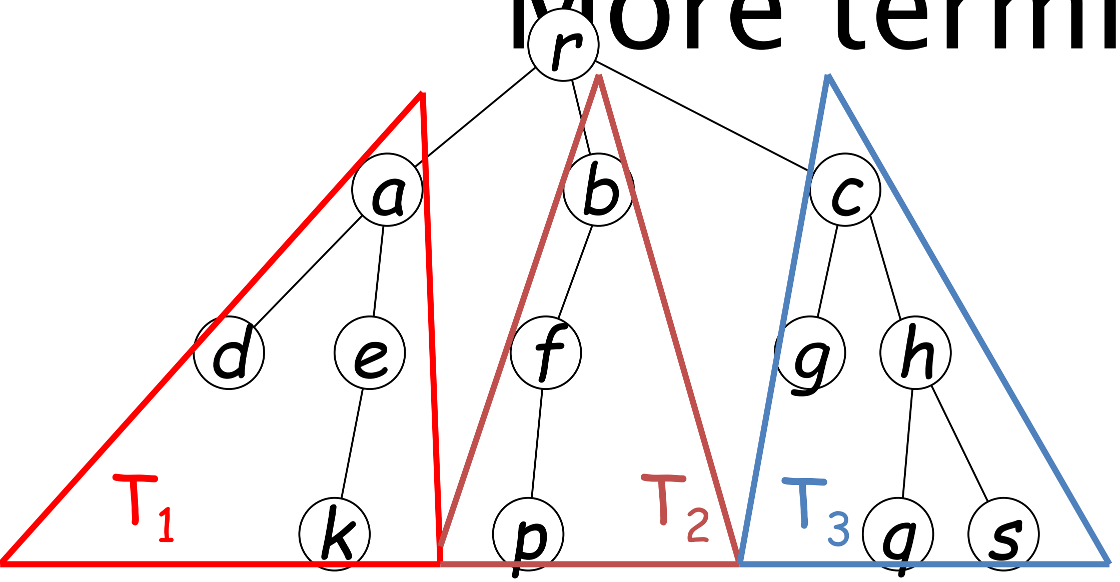
Terminologies

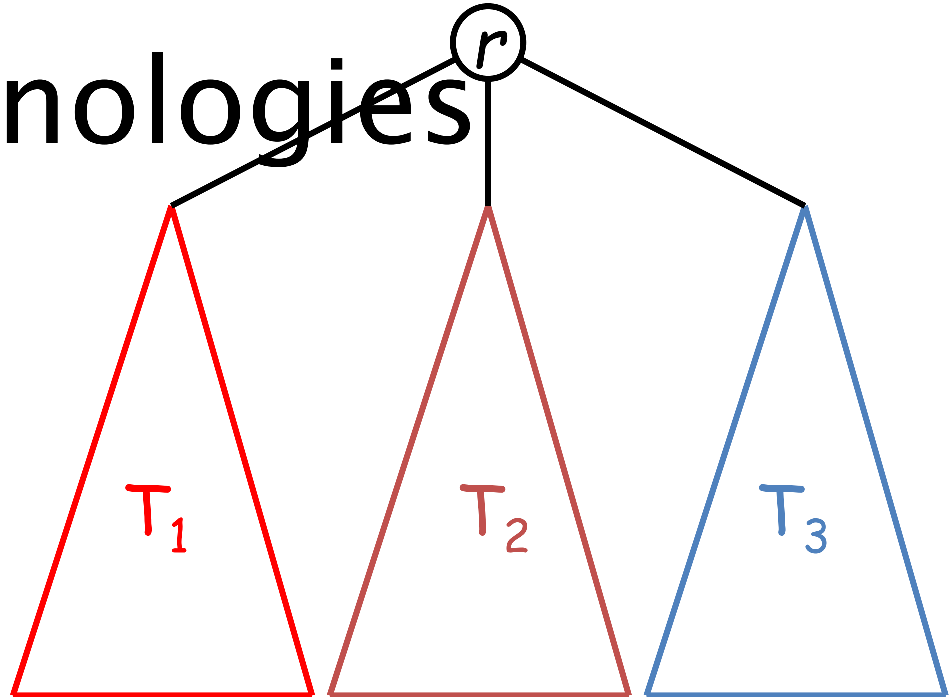


deg-0:
deg-1:
deg-2:
deg-3:

What is the degree of this tree?

More terminologies





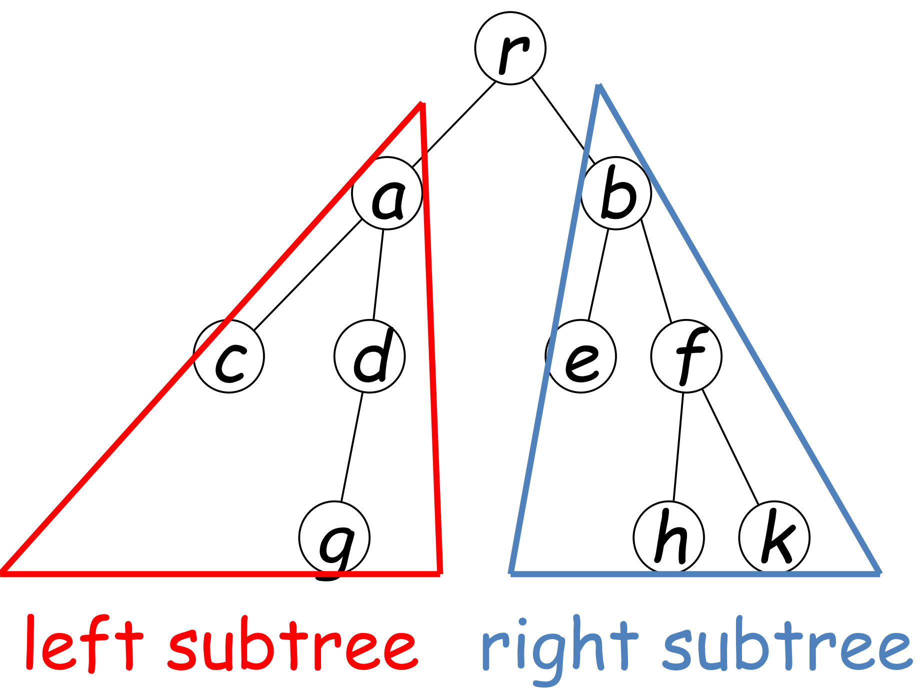
three subtrees

- We can define a tree **recursively**
  - A single vertex is a tree.
  - If $T_1, T_2, \ldots, T_k$ are **disjoint** trees with roots $r_1, r_2, \ldots, r_k$, the graph obtained by attaching a *new vertex r* to each of $r_1, r_2, \ldots, r_k$ with a single edge forms a tree T with root r.
  - $T_1, T_2, \ldots, T_k$ are called **subtrees** of T.

which are the roots of the subtrees?

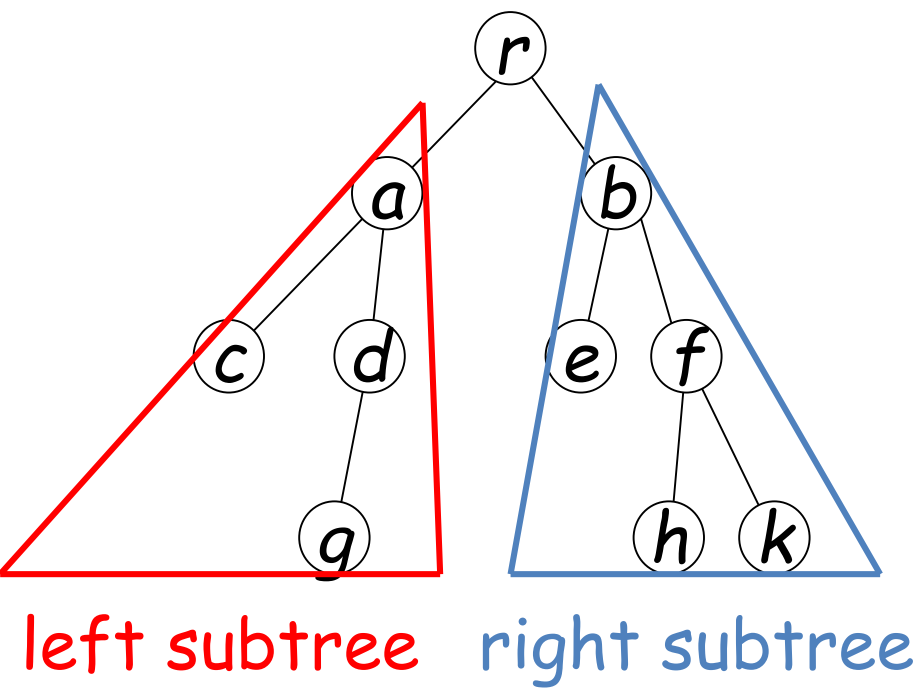
Binary tree

- a tree of degree at most TWO
- the two subtrees are called left subtree and right subtree (may be empty)



Binary tree

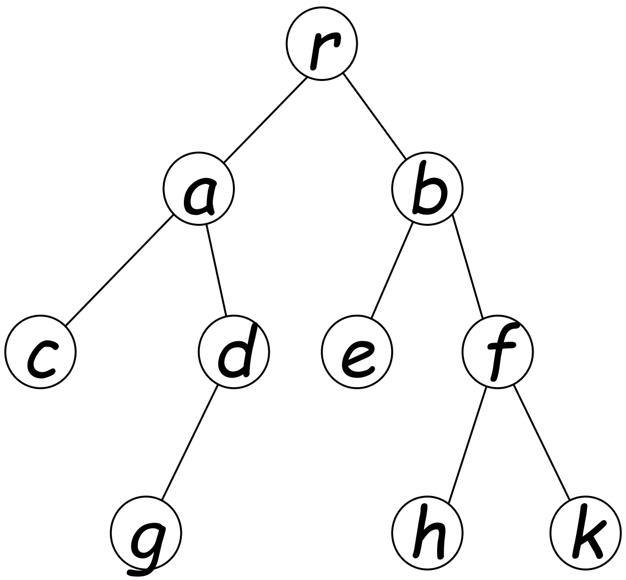
- a tree of degree at most TWO
- the two subtrees are called left subtree and right subtree (may be empty)



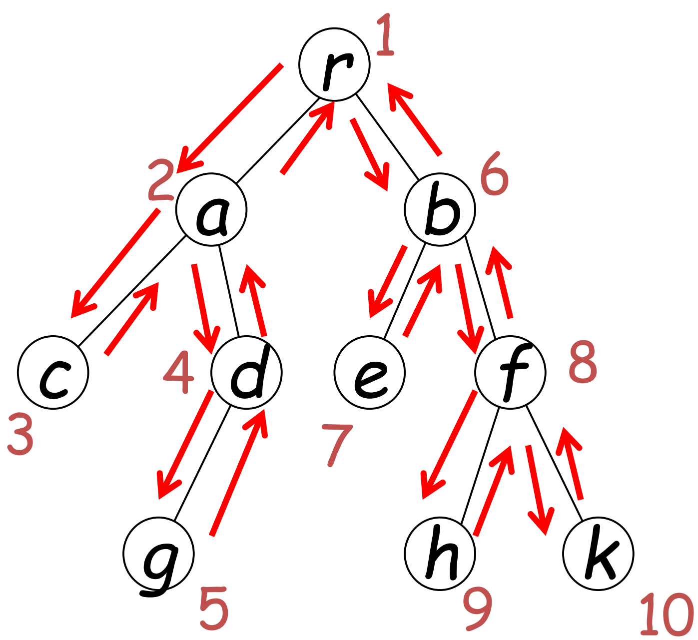
There are *three* common ways to traverse a binary tree:

- **preorder** traversal - vertex, left subtree, right subtree
- **inorder** traversal - left subtree, vertex, right subtree
- **postorder** traversal - left subtree, right subtree, vertex

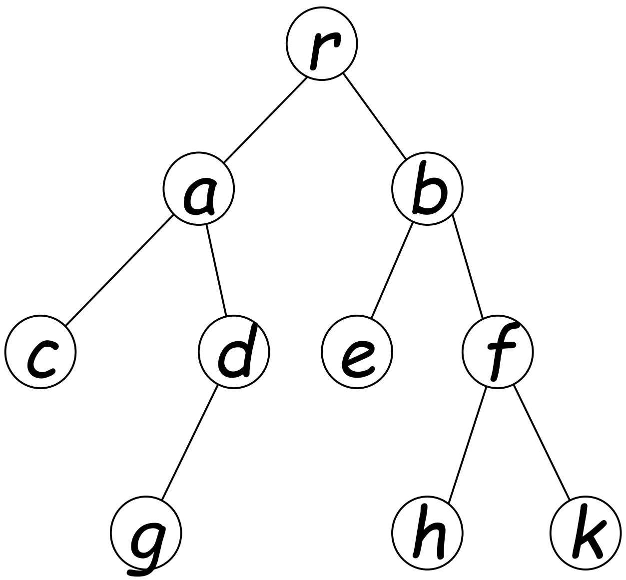
Traversing a binary tree



preorder traversal
- vertex, left subtree, right subtree

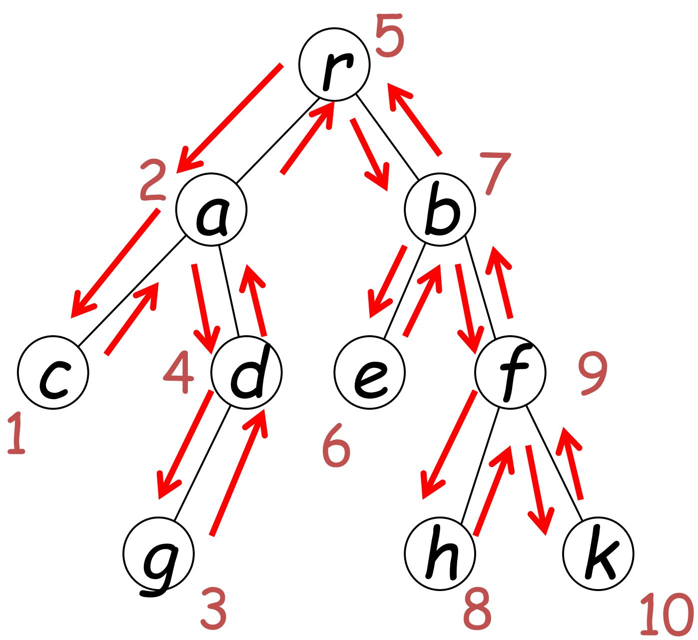


Traversing a binary tree

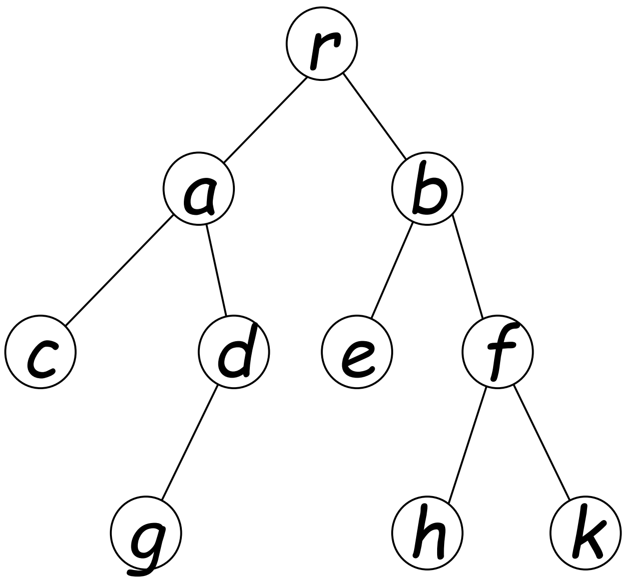


inorder traversal

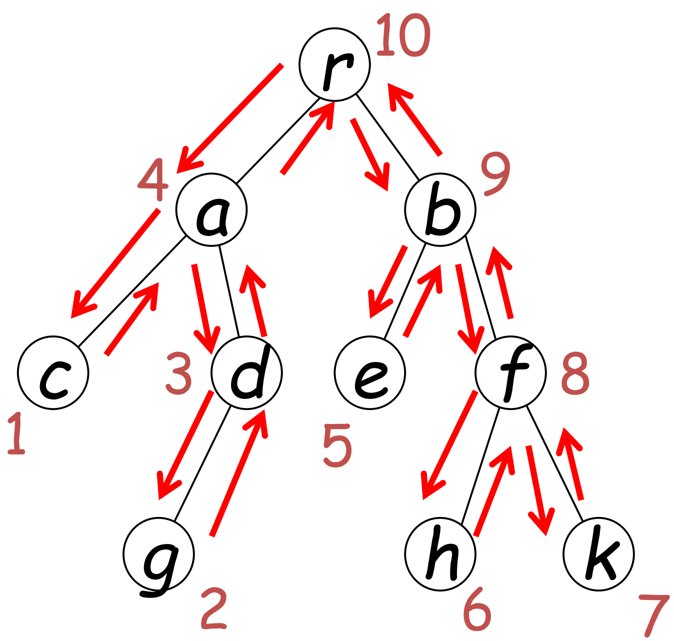
- left subtree, vertex, right subtree



Traversing a binary tree

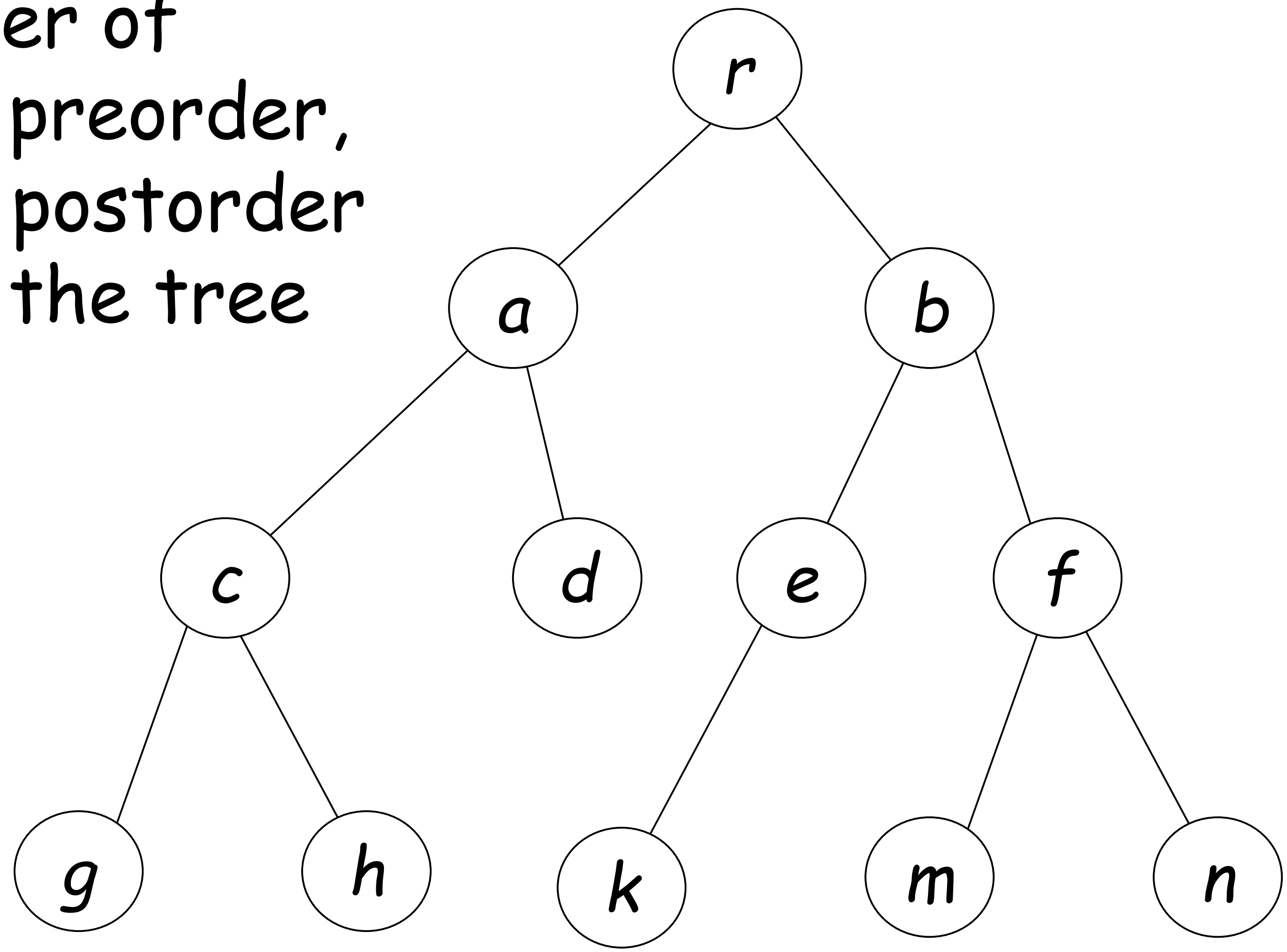


postorder traversal
- left subtree, right subtree, vertex



Example

Give the order of traversal of preorder, inorder, and postorder traversal of the tree



preorder:
inorder:
postorder:

Learning outcomes

- ✓ Able to tell what is an undirected graph and what is a directed graph
  - ✓ Know how to represent a graph using matrix and list
- ✓ Understand what Euler path / circuit and able to determine whether such path / circuit exists in an undirected graph
- ✓ Able to apply BFS and DFS to traverse a graph
- ✓ Able to tell what a tree is