Non linear Data structures

$G = (V, E)$ — Defined by 2 sets

Vertices (nodes) — edges (links, Arcs)

V = set of vertices
E = set of edges

A graph is a versatile data structure that is used whenever we want to represent (pair-wise) connections btn objects

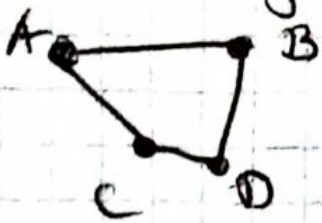
- Network of roads in a city or country
- Computer network
- Social network

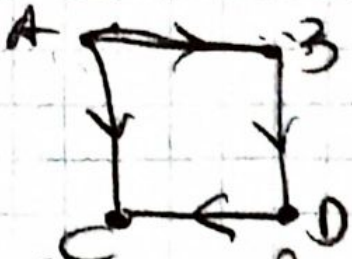
Simple graph: — no self loop
— no parallel edges

Directed graph: — every edge has a direction (specific).

Undirected graph: — every edge is symmetric

path: Sequence of nodes connected by edges

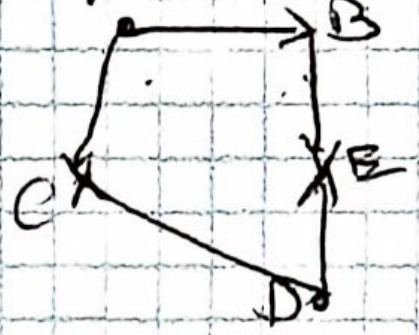




Cycle - closed path

Acyclic graph - graph without a cycle

- Eulerian tour: A path through the graph that uses each edge exactly once.
- Hamiltonian tour: A path through the graph that touches every vertex exactly once.
- Directed graph
  - In degree: the number of incoming edges to a vertex
  - Out degree: number of edges outgoing from a vertex



In deg(A) = 0
Out deg(A) = 2

In deg(B) = 1
Out deg(B) = 1

In deg(C) = 2
Out deg(C) = 0

Out deg(E) = 1
In deg(E) = 2

In deg(D) = 0
Out deg(D) = 2

Undirected graph

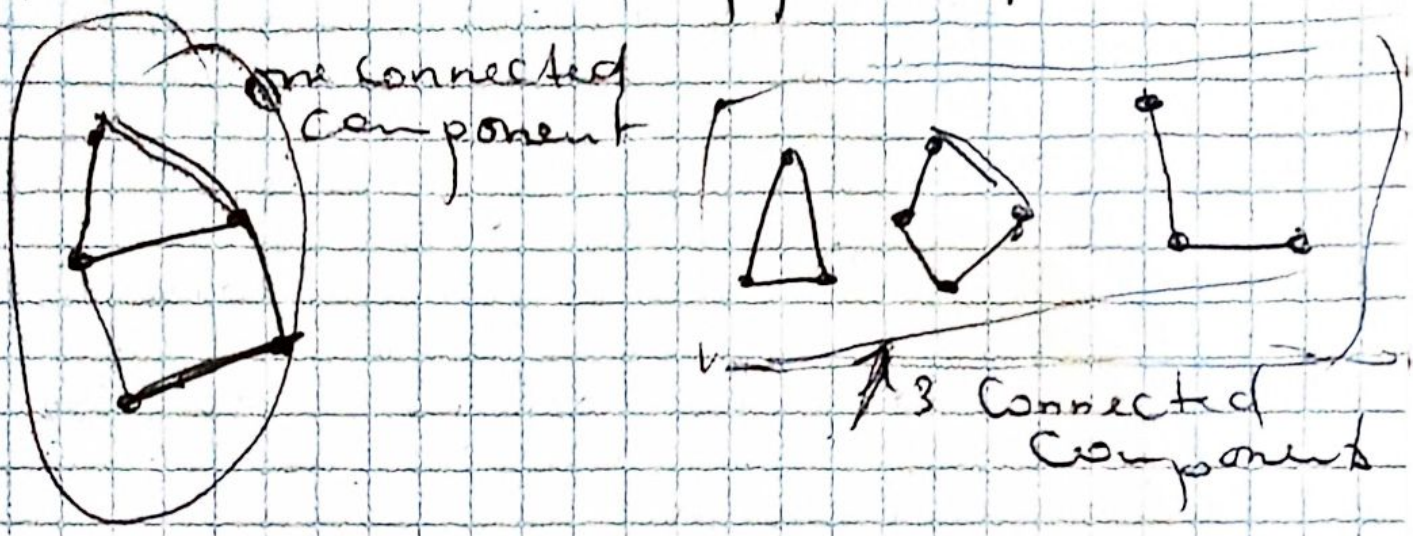
deg(v) = number of edges adjacent to v

$$\sum_{v \in V} \text{degree}(v) = 2|E| \quad \text{(Handshaking lemma)}$$

- Complete Graph is a graph where there is an edge between any pair of vertices. Given a complete Graph with n nodes, how many edges does it have?

$$\frac{a_2(n-1)}{2}$$

A Graph is connected if there is a path between any pair of vertices



- Tree: A cyclic Connected Graph
- Spanning tree of graph G:
  Tree with all the vertices in G and some of the edges in G.
- Rooted tree: Tree in which a vertex is specifically designated as the root of the tree (defines a parent-child relationship)

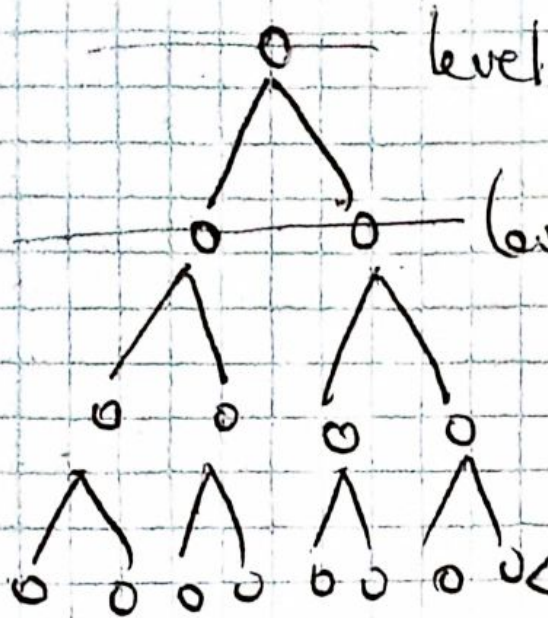
Between vertices: every node, except the root has exactly one parent

Binary tree: Rooted tree in which every node has at most 2 children

- left child
- Right child

(Leaf node has no child
Internode: not a leaf)

Full binary tree: Binary tree in which all the leaves are at the same level and all internal nodes have 2 children.



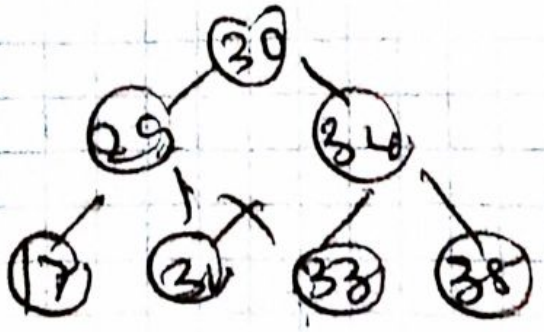
Height of a tree = longest path from root to leaf

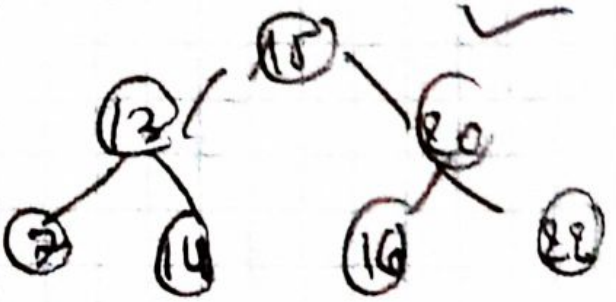
$$\sum_{l=0}^{h} 2^l = 2^{h+1}$$

full binary tree of height h, how many nodes does it have?

Binary Search tree

Binary tree such that the key in every node is ≥ the key in its left child and ≤ the key in its right child.





BST (ADT)

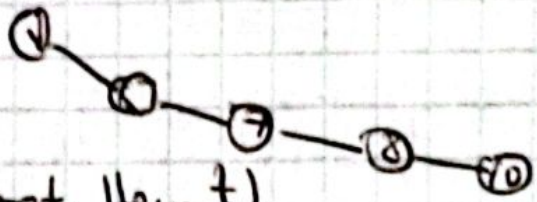
- Values stored are of any type
- Find item (t)
- Insert item (t)

• Delete (item t)

Search (Node root, Item t)

```
if root.key == t
    return found
else if (root.key < t)
    return search (root.left, t)
else return search (root.Right, t)
```

(O(h))



Insert (Node root, Item t)

```
if (root == nullptr)
  root.key = t
else if (root.key ≤ t)
    Insert (root.left, t)
  else insert (root.right, t)
```

O(h)

Delete (Item t)

(O(h))

If Item t is in leaf node just detach the node from the tree

If item t is not in a leaf node (then, predecessor(t)) delete predecessor(t).

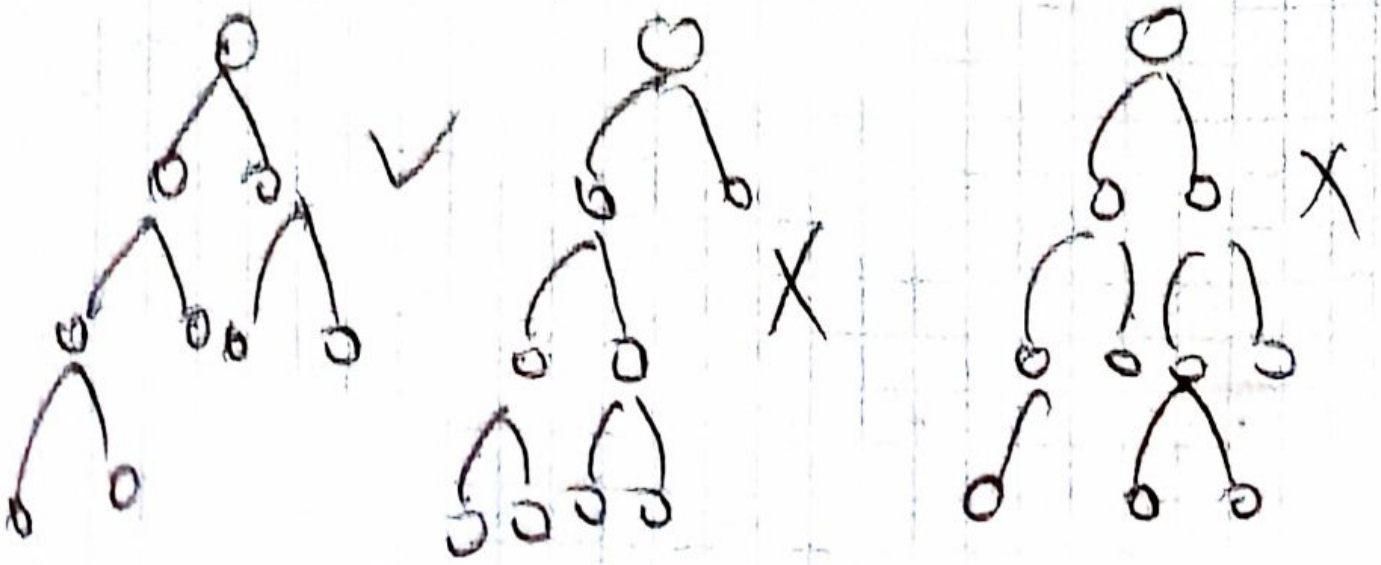
NB :
- Predecessor(t) = Right most node in the left subtree of t
- Successor(t) = left most node in the right subtree of t

AVL
RB
B

Complete BT

- Binary tree with all levels full except possibly the last level
- In the last level all the nodes are backed to the left without a gap

Heap (Max or Min)



Heap (Max or Min)

Complete Binary tree

- Min heap ⇒ key in a node is < keys in its children
- Max heap ⇒ key in a node is > keys in its children

⑥ Given a heap of height h, what is the maximum number of nodes it can have? ix the minimum number of nodes a heap of height h can have?

Maximum (full complete binary tree)

$\Rightarrow 2^{h+1} - 1$

Minimum 2^h

The height of a heap is always $O(\lg n)$

Tree traversal

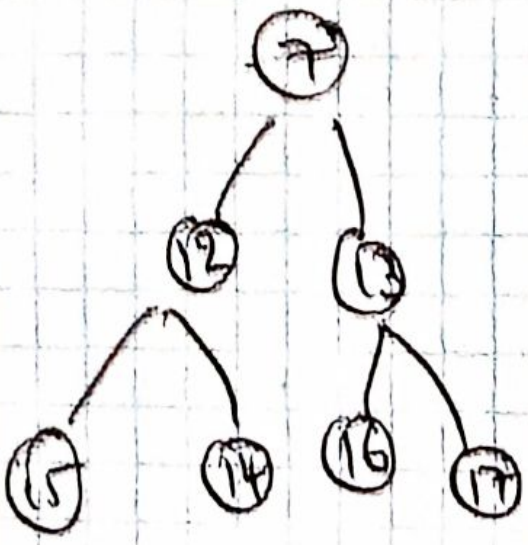
In-order

15 12 14 7 16 13 17

Pre order

7 12 15 14 13 16 17

root - left - right



Post order

left - right - root

15 14 12 16 17 13 7