🌳Teaching Notes: Trees (Unit 5, Part 1)

---

1. Introduction to Trees

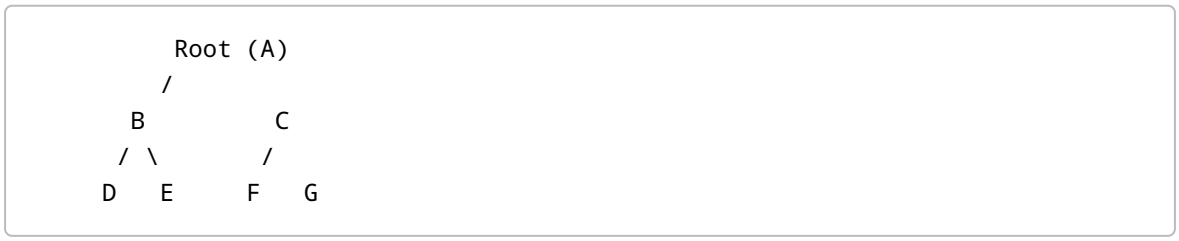
Definition of Tree

- A **Tree** is a **non-linear hierarchical data structure** made up of **nodes** connected by **edges**.
- Unlike arrays and linked lists (linear), trees represent data in a **hierarchical form**.
- **Tree is a special type of graph** with no cycles and exactly one path between any two nodes.

Hierarchical Structure and Properties

- Starts with a **root node**.
- Each node can have **zero or more children**.
- Node with no children is called a **leaf**.

Example Diagram:



Real-life Examples

1. Family Tree
2. File System
3. Organization Hierarchy
4. Decision Trees in AI

Applications of Trees

- Database Indexing (B-trees)
- File Systems
- Network Routing
- Compilers (Parse Trees)
- AI/ML (Decision Trees)

---

2. Tree Terminologies

1. **Root** → Topmost node (no parent)
2. **Parent** → Node with child(ren)
3. **Child** → Node descending from parent
4. **Leaf Node** → Node with no children
5. **Sibling** → Nodes sharing the same parent

6. **Ancestor** → Node above in path from root
7. **Descendant** → Node below in path
8. **Degree of Node** → Number of children
9. **Degree of Tree** → Maximum degree among nodes
10. **Depth** → Distance from root
11. **Height of Node** → Longest path to leaf
12. **Height of Tree** → Height of root node
13. **Level** → Root at level 0 (or 1)
14. **Path** → Sequence of nodes from parent to child
15. **Subtree** → Node with all descendants

Example:

```
Subtree rooted at B:
   B
  /
 D   E
```

---

3. Binary Tree

Definition

- A **Binary Tree** is a tree where each node has **at most two children**: Left and Right.

Example:

```
        10
       /
     5     15
    / \
   2   7     20
```

Properties

1. Maximum nodes at level `i` = `2^i`
2. Maximum nodes with height `h` = `2^(h+1) - 1`
3. Minimum height with `n` nodes = `⌈log2(n+1)⌉ - 1`
4. Worst case height (skewed tree) = `n - 1`

Representations

(a) Linked Representation

- Node structure:

```c
struct Node {
    int data;
```

```
    struct Node *left, *right;
};
```

(b) Array Representation

- Stored in **level order**.
- Node at index `i`: Left Child = `2*i + 1`, Right Child = `2*i + 2`, Parent = `(i-1)/2`

Example:

```
      10
     /
   20    30
  /
40   50
```

Array: [10, 20, 30, 40, 50]

(End of Part 1: Introduction, Terminologies, Binary Tree)