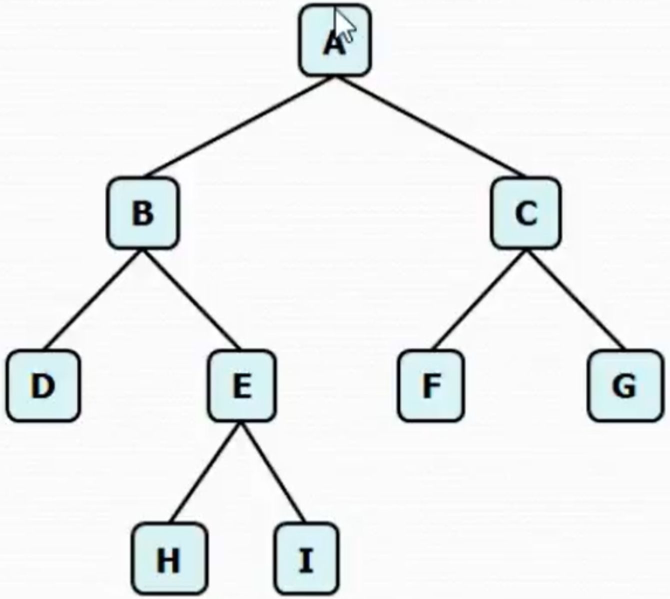
Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

InOrder Traversal: Example

Output: D B H E I A F C G



Alka Jindal

PostOrder Traversal

- Visit the left sub-tree in PostOrder
- Visit right sub-tree in PostOrder
- Visit root node

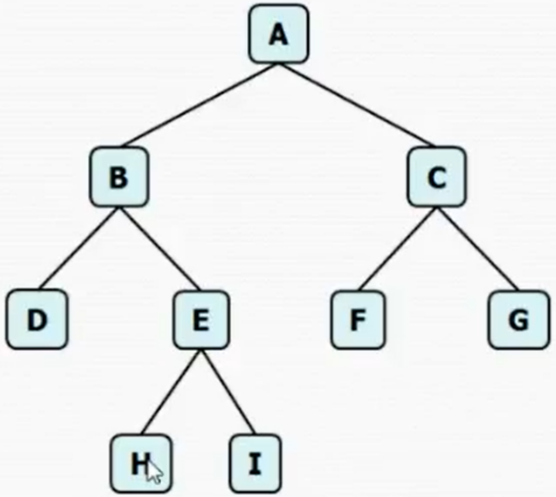
```
PostOrderTraversal(Tree) {
        if (isEmpty(Tree)) return;
        else {
                PostOrderTraversal(tree->left);
                PostOrderTraversal(tree->right);
                print (tree->data);
        }
}
```

A
Alka Jindal

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

PostOrder Traversal: Example

Output: D H I E B F G C A

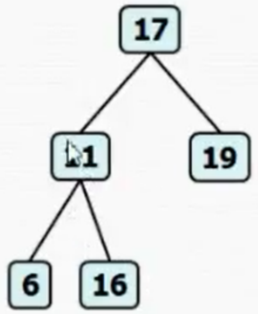


A
Alka Jindal

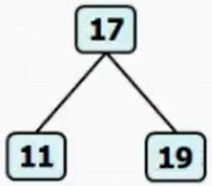
Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Full Binary Tree

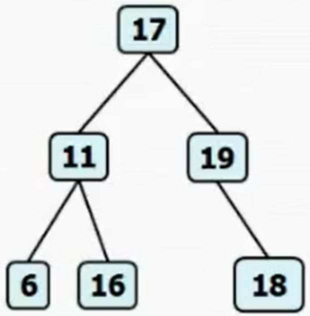
- Either a node has 2 children or no child in a binary tree



Full Binary Tree



Full Binary Tree



Not Full Binary Tree

Alka Jindal

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Perfect Binary Tree

- All internal nodes has two children and all leaves are at same level/depth

full; not perfect

Full; perfect

Not full; not perfect

A
Alka Jindal

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Perfect Binary Tree

- Level i has 2^{i-1} nodes
- If leaves are level at h
  - Number of leaves is 2^{h-1}
  - Number of internal node = $1+2+2^2+2^3+\ldots+2^{h-2} = 2^{h-1} - 1$

    = number of leaves - 1
  - Total number of nodes = $2^{h-1} + 2^{h-1} - 1 = 2^h - 1$

Alka Jindal

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Perfect Binary Tree

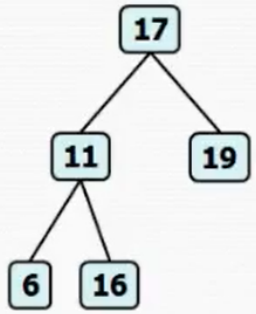
- Level i has 2^{i-1} nodes
- If leaves are level at h
  - Number of leaves is 2^{h-1}
  - Number of internal node = $1+2+2^2+2^3+\ldots+2^{h-2} = 2^{h-1} - 1$
    = number of leaves - 1
  - Total number of nodes = $2^{h-1} + 2^{h-1} - 1 = 2^h - 1$
- If total number of nodes is n
  - Number of leaves = $(n+1)/2$
  - Height of tree = $\log_2$ (number of leaves) = $\log_2((n+1)/2)$

Alka Jindal

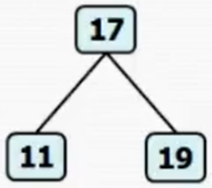
Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Complete Binary Tree

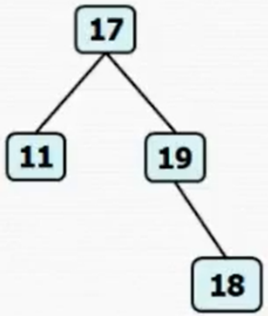
- All levels are completely filled except last. Also, all nodes in last level are as far left as possible



Full;
Not perfect;
Complete



Full;
Perfect;
Complete



Not full;
Not perfect;
Not Complete

A
Alka Jindal

emu-juny-upp (2021-10-29 at 01:49 GMT-7)

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Complete Binary Tree

- Perfect tree is a special case of complete tree with last level completely filled.
- In some literature, Perfect binary tree is referred as Complete binary tree. In that case, Complete binary tree is referred Almost Complete binary tree.

A

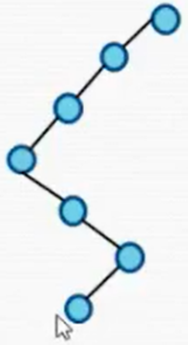
Alka Jindal

22:27 / 45:37

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Height of a Binary Tree

- If a binary tree has n nodes and height h, then
  - Level i has at most 2^{i-1} nodes
  - $n <= 2^{h+1} - 1$
  - Hence, $h >= \log_2(n+1)/2$ i.e. minimum height of a tree with n nodes is $O(\log_2 n)$
  - Maximum height of a tree with n nodes is n-1 which is obtained when every non-leaf node has exact one child



A

Alka Jindal

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]
emu-juny-upp (2021-10-29 at 01:49 GMT-7)

Linear Representation of Binary Tree

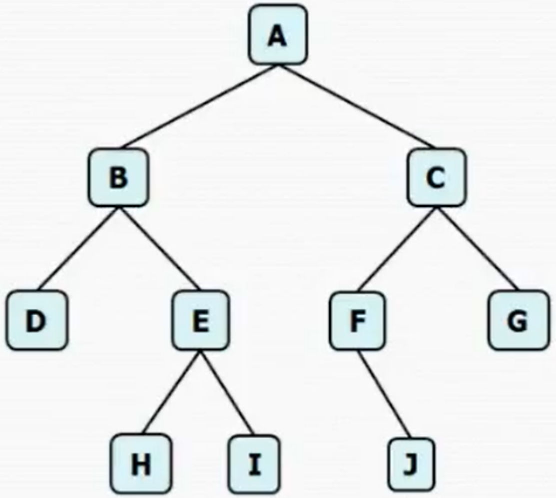
- Binary tree can also be represented using arrays
- Store root node at index 0
- Store left child of a parent node is at $2*i+1$ where i is the index of parent node in array
- Store right child of a parent node is at $2*i+2$ where i is the index of parent node in array
- Parent of a node at index i can be found at $(i-1)/2$ except for root node

Alka Jindal
A
27:32 / 45:37

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Example

A	**B**	**C**	**D**	**E**	**F**	**G**			**H**	**I**		**J**
0	1	2	3	4	5	6	7	8	9	10	11	12



Alka Jindal

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]
emu-juny-upp (2021-10-29 at 01:49 GMT-7)

Linear Representation of Binary Tree

- If a node doesn't have a left or/and right child, indices for left or/and right child are empty
- If index of a child is greater than size of array, child of that node does not exist

Alka Jindal
30:54 / 45:37

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]
emu-juny-upp (2021-10-29 at 01:49 GMT-7)

Searching a Binary Tree

- BFS or DFS
- Worst case time complexity = O(n) where n is the number of nodes in binary tree
- Can we improve searching?

Alka Jindal
39:16 / 45:37

21221_CS1 | emu-ju | wab-jvws-t | mas-gcqw- | rja-wdxe-m | uwv-twpt-j | fre-ibqp-yd | rwf-fmuu-y

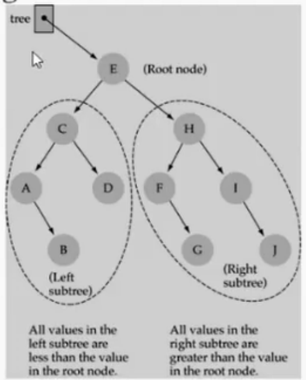
https://drive.google.com/file/d/1iqYnYA-pO79jzibWynO0Zr_sy1hJrgNO/view

emu-juny-upp (2021-10-29 at 01:49 GMT-7)

Microsoft PowerPoint - [PowerPoint Slide Show - [7_TreeCLASS]]

Binary Search Tree

- Root is greater than left child and less than right child
- Equal keys can be kept in left or right sub-tree
- Same strategy should be followed for whole tree for repeated keys



A

Alka Jindal

Links Mars4_5 - Shortcut 99+ 27°C ENG 16:55 21-11-2021