# COL 106 Lecture 10

Topic: Almost-Complete Binary Trees

Attendance Marking Instructions:

~~Reset Schedule, then mark attendance on~~ phone. Mark attendance on phone in your slot.

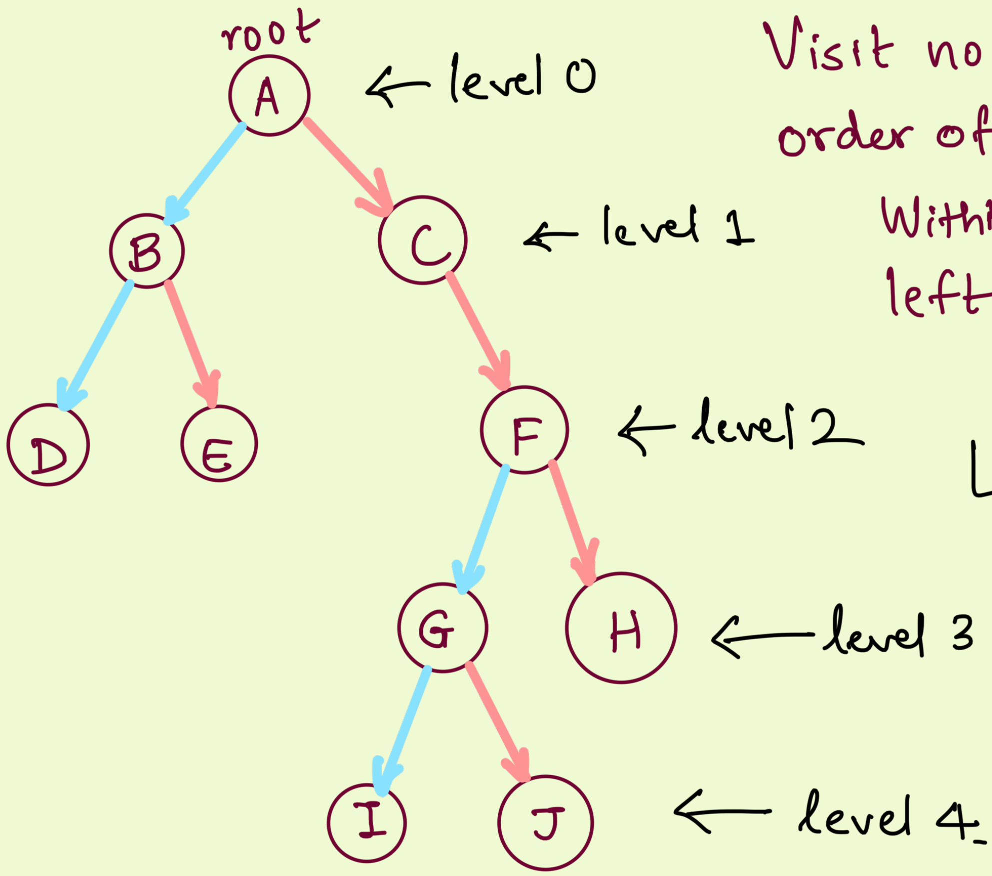
Gentle Reminder:

Assignment 1 due Aug 31 (tomorrow) 23:00.

Recap:

- Priority Queues
- Binary trees
- Parent, left/right child, ancestor, descendant....
- Depth, Levels, Height
- Complete binary trees
- Level - Order Traversal

# Level-Order Traversal




Visit nodes in increasing order of levels

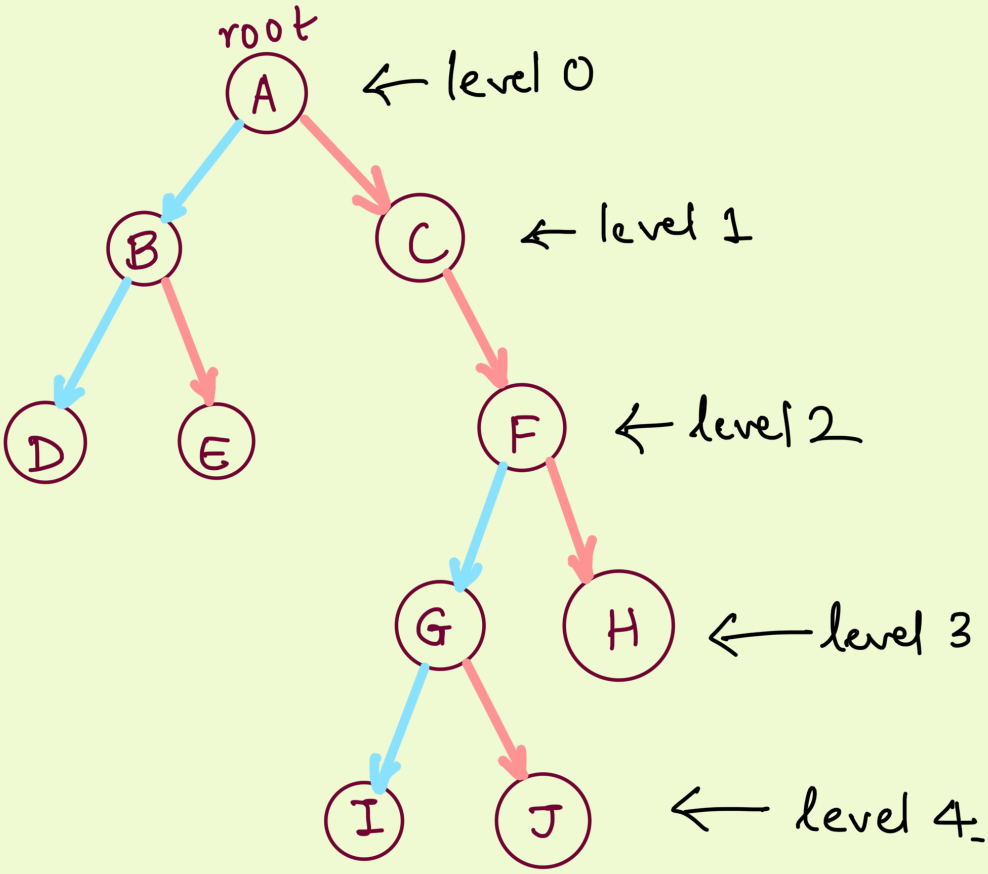
Within each level from left to right.

Level-order traversal:
A, B, C, D, E, F, G, H, I, J

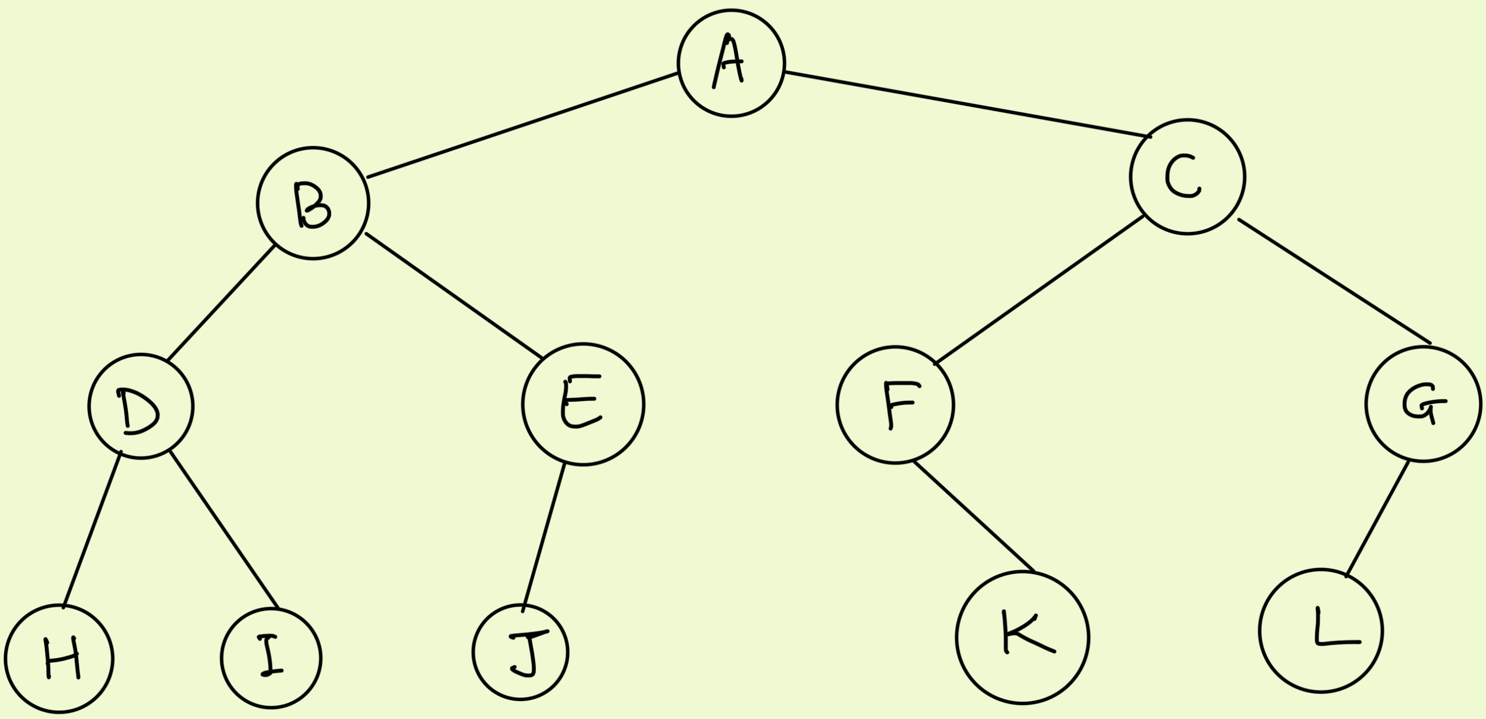
# Level-Order Traversal Algorithm

If root is not None:
    Enqueue root into an empty queue.
    While queue not empty:
        Dequeue a node, say $x$.
        Visit $x$.
        If $x$.left is not None, enqueue $x$.left.
        If $x$.right is not None, enqueue $x$.right.

Exercise: Prove that the above algorithm performs level-order traversal.

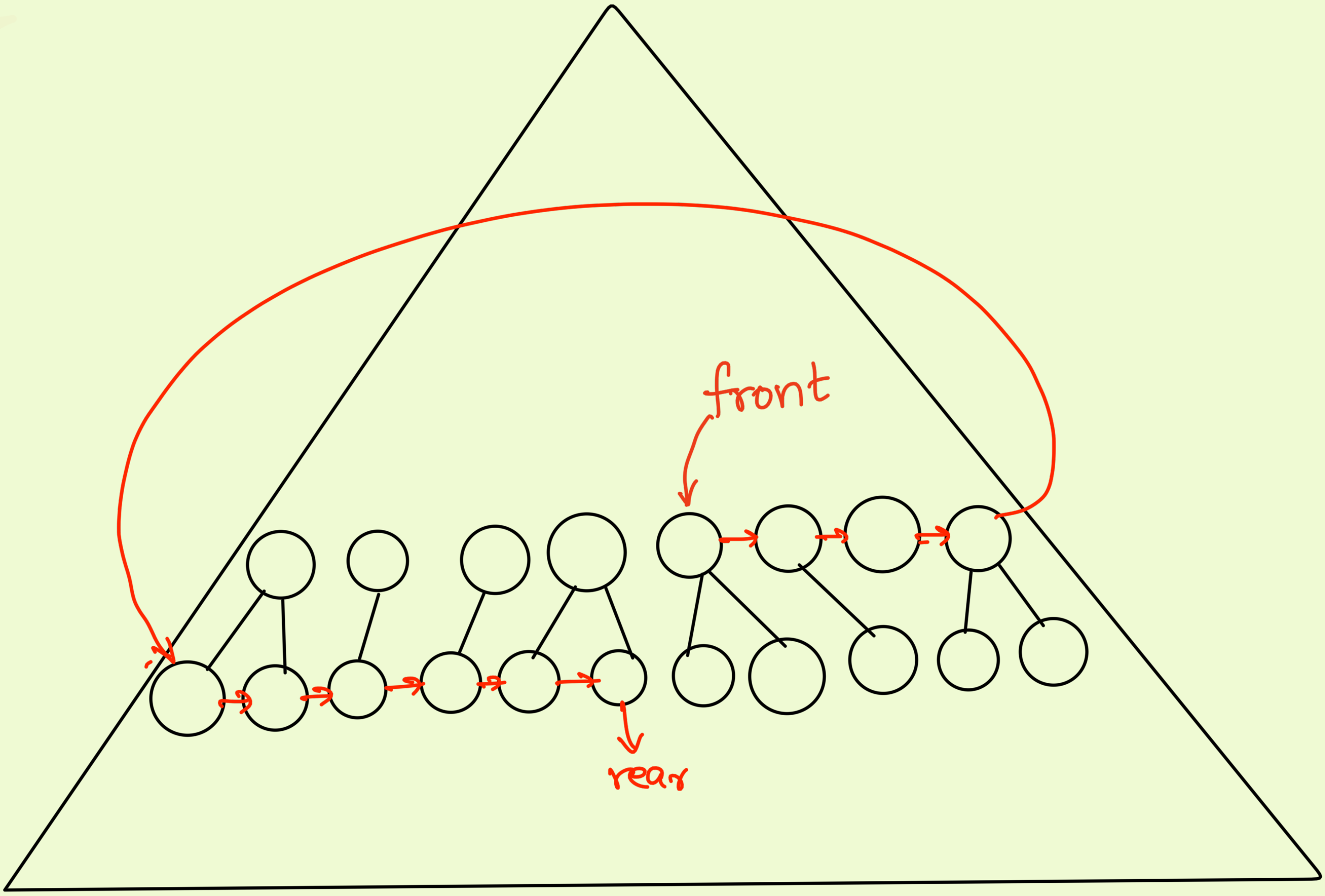



| Visited node | Queue |
|---|---|
| | A |
| A | B C |
| B | C D E |
| C | D E F |
| D | E F |
| E | F |
| F | G H |
| G | H I J |
| H | I J |
| I | J |
| J | — |




Question: When F is in the front of the queue, what does the queue contain?

Answer: F G H I J




Claim: Suppose node $x$ is in the front of the queue. Let $d$ be the depth of $x$. Then the queue contains:

- $x$, followed by
- Nodes in level $d$ to the right of $x$, in a left-to-right order,

followed by.

- Children of nodes in level $d$ to the left of $x$, in a left-to-right order.

Proof: By induction on # iterations of the while loop.

## Challenge

Given a sequence $k_0, k_1, \ldots, k_{n-1}$ of $n$ keys, find the number of binary trees with keys $k_0, \ldots, k_{n-1}$ that have this sequence as their level order traversal.

Let $C_n$ denote this number. Try to express $C_n$ in terms of $C_{n-1}, C_{n-2}, \ldots, C_0$.

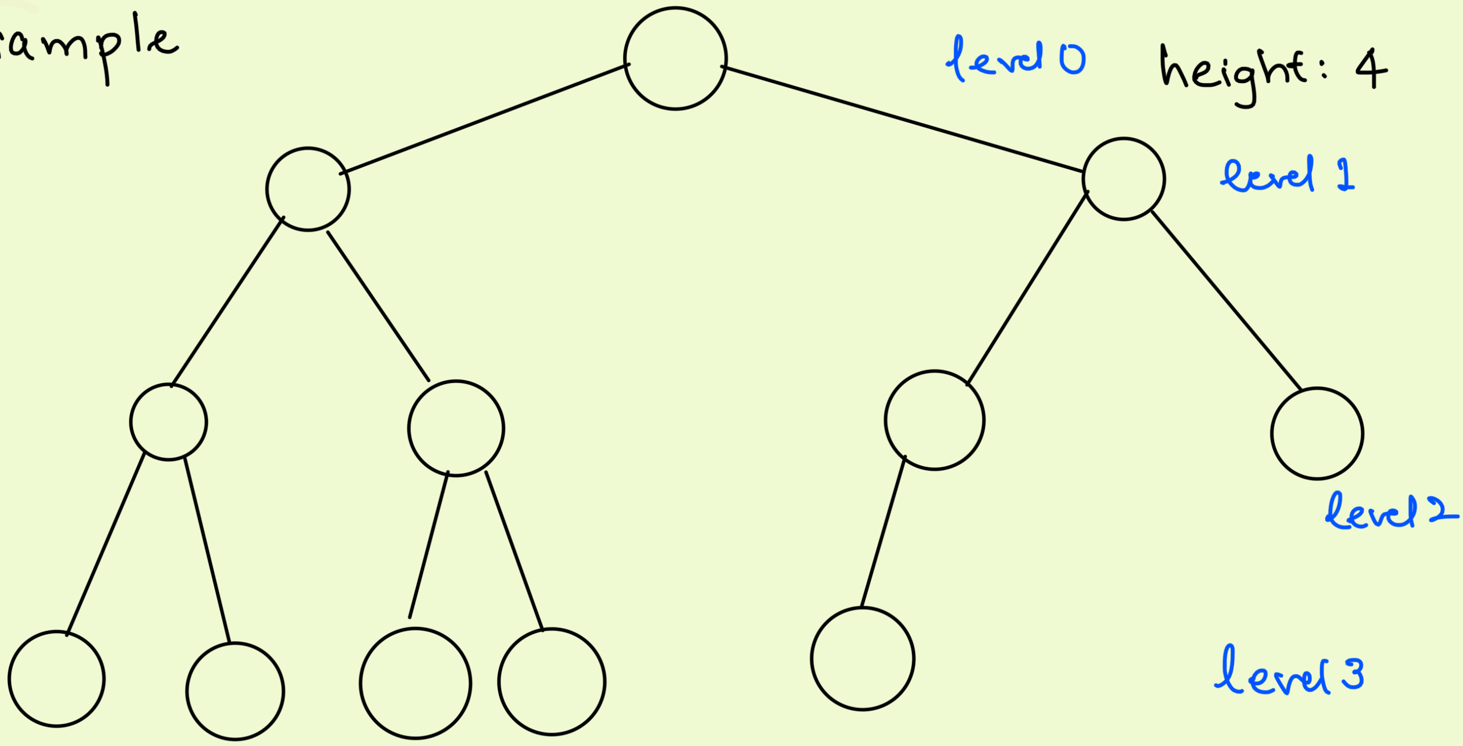
## Almost-Complete Binary Trees

Definition :

An almost-complete binary tree of height $h$ is one in which

- Each level $\ell \in \{0, \ldots, h-2\}$ has $2^\ell$ nodes (ie. it is full).
- Nodes in level $h-1$ (all leaves) are as much to the left as possible.

Example



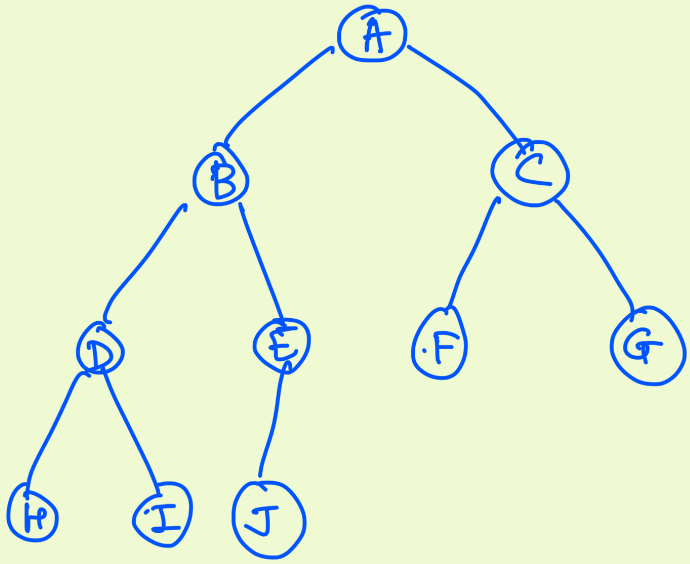

All complete binary trees are almost-complete binary trees.

Claim: Let $n$ be the # of nodes in an almost complete binary tree and $h$ be its height.
Then $2^{h-1} \le n \le 2^h - 1$.
i.e. $h = \lceil \log_2 (n+1) \rceil$.

Proof: For $i \in \{0, \dots h-2\}$ : # nodes in level $i = 2^i$
# nodes in level $h-1$ is between $1$ and $2^{h-1}$.

Construct all possible almost complete binary trees having level-order traversal

A B C D E F G H I J.




Claim: There exists a unique almost complete binary tree having a given sequence as its level order traversal.

Proof: Let $k_0\ k_1, \ldots, k_{n-1}$ be the level order traversal.

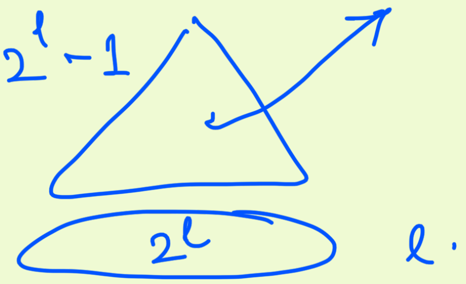
Then height $h = \lceil \log_2(n+1) \rceil$. We are forced to put

$k_0$ in layer 0 $\qquad 1 + 2 + \cdots + 2^{l-1} \rightarrow$ upper layers

$k_1, k_2$ in layer 1 $\qquad = 2^l - 1$

$\vdots$

$k_{2^l-1}, \ldots, k_{2^{l+1}-2}$ in layer $l$




$\vdots$

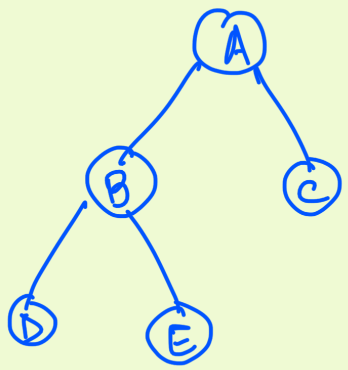
$k_{2^{h-1}-1}, \ldots, k_{n-1}$ in layer $h-1$, where $h = \lceil \log_2(n+1) \rceil$.

Claim: Let $k_0, k_1, \ldots, k_{n-1}$ be the level-order traversal of an almost-complete binary tree. Then the parent of $k_m$ is $k_{\lfloor \frac{m-1}{2} \rfloor}$ (?), its left child is $k_{2m+1}$ (?), and its right child is $k_{2m+2}$ (?).

Exercise: Figure out the correct claim and prove it before the next class.

## Array / List representation of Almost-Complete Binary Trees

Idea: Represent an almost-complete binary tree by its level-order traversal




A B C D E