Binary Lifting

Trees 3

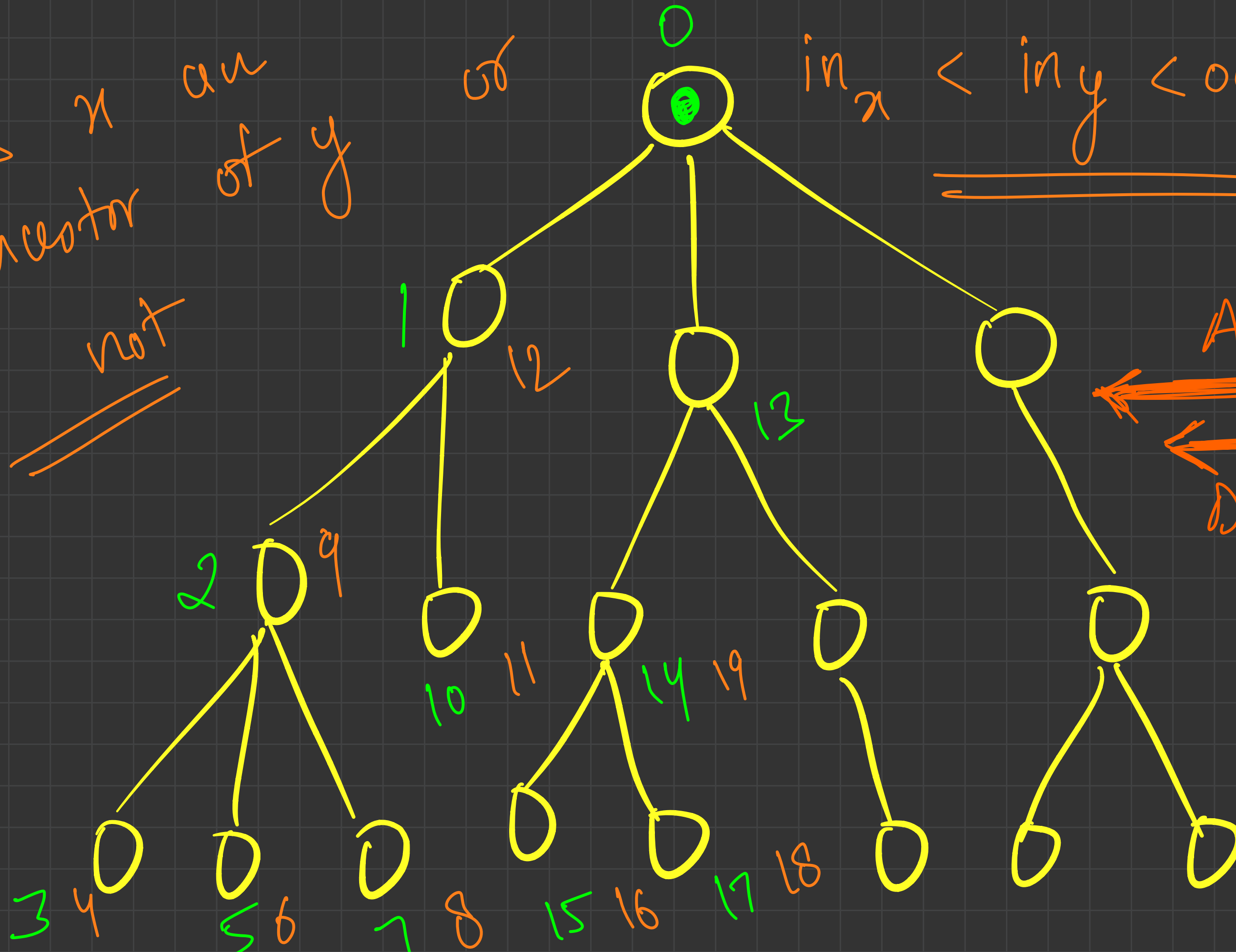
In-Time, Out-Time Continued

- Revision
- Code
- Ancestor-Descendant Relationship

```
bool isAncestor ( int x, int y )    // if x an ancestor of y or not
{
    return inTime[x] < inTime[y] &&
           outTime[y] < outTime[x]
}
```

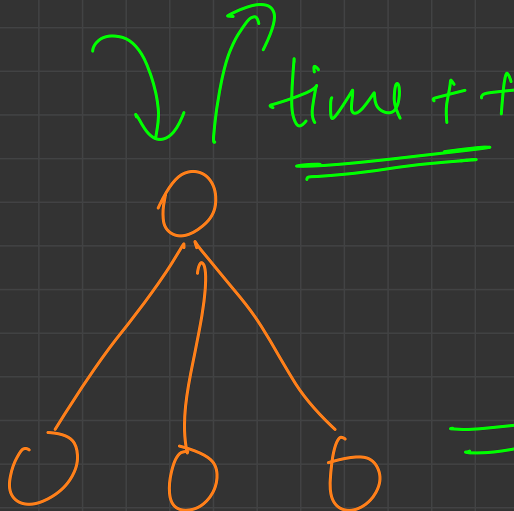
is x an ancestor of y or not

$in_x < in_y < out_y < out_x$



Ancestor

Descendant




```
vector <int> inTime(n);
vector <int> outTime(n);
int globalTime = 0;
void dfs(curr, edges, parent)
{
    inTime[curr] = globalTime++;
    for (neighbour : edges[curr])
        if (neighbour != parent)
            dfs(neighbour, edges,
                curr)
} outTime[curr] = globalTime++
```

```
dfs(root, edges, -1)
```

Binary Lifting

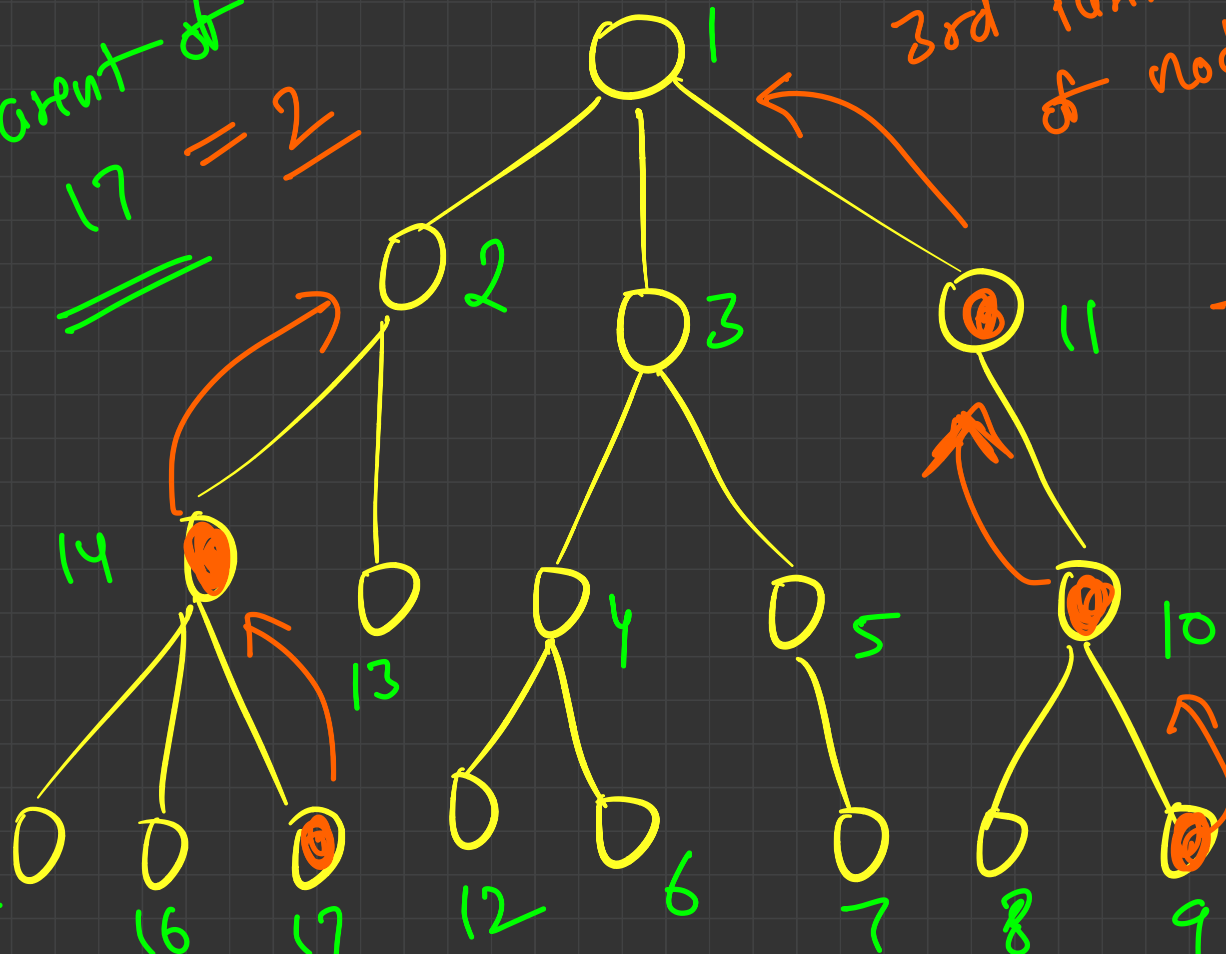
Most Important

- Find kth Parent of any node in a Tree Problem Link
- Find LCA of 2 nodes Problem Link
  - O(logn * logn)
  - Precomputation of logs -> O(logn)
  - Using in-time out-time trick -> O(logn)

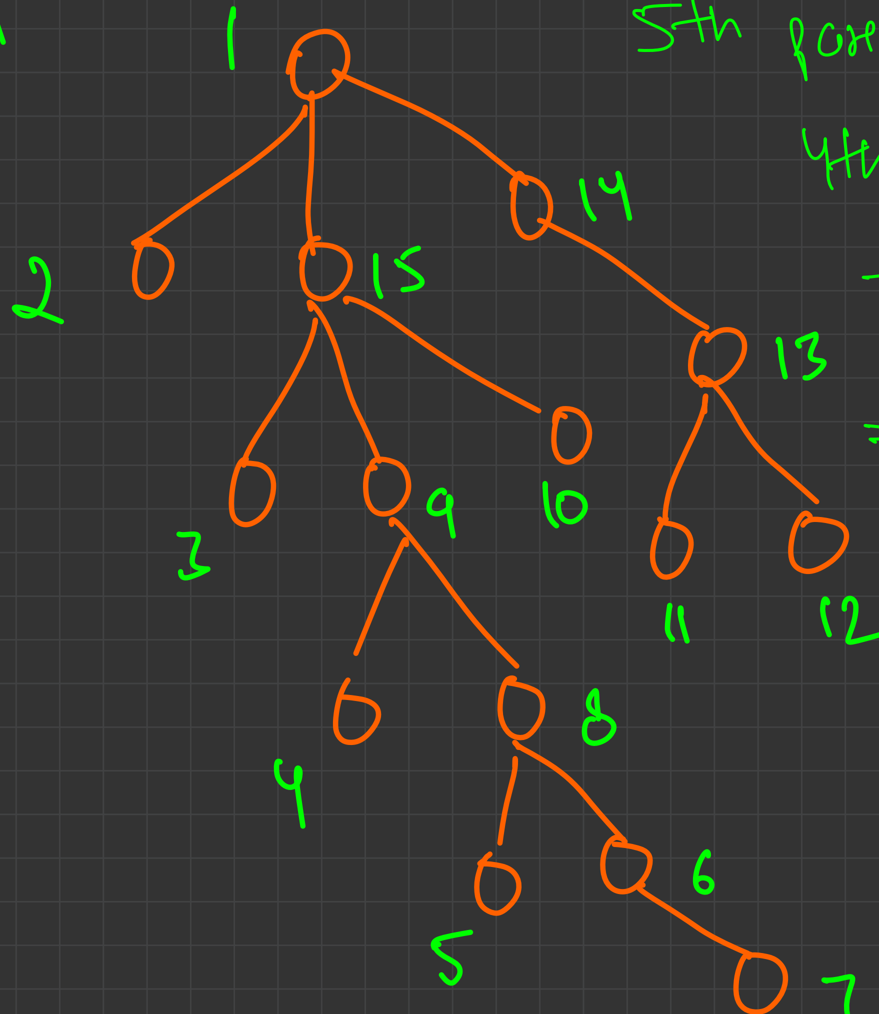
efficiently

2nd Parent of node 17 = 2

3rd Parent of node 9 = 1



5th parent of 7



5th parent of 7 =

4th parent of 6 =

3rd parent of 8

= 2nd parent of 9

= 1st parent of 15

= 1

```
int kthParent( int x, int k)
{
    while( k > 0){
        x = parent[x];
        k--;
    }
    return x;
}
```

$O(k)$

$= O(n)$

$k = n-1$

find kth parent → $O(k)$

find out 15th parent of a node

$O(\log k)$

15

15 = 1111 ✓ 8th parent of

X = Y

15th parent of X ??

↓

4th parent of

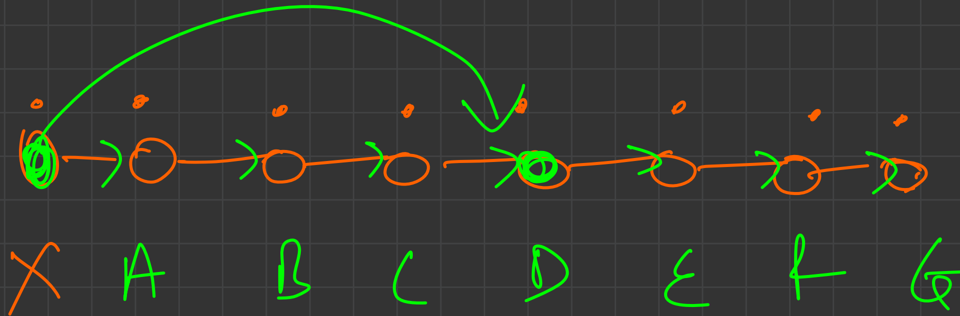
Y = Z

7th parent of Y

↓

3rd parent of Z

X ⟶ Y ⟶ Z ⟶ G



7th parent of X = G

4th parent of X = D

7th parent of X = 3rd parent of D

for every node you know the following :

1st parent, 2nd parent, 4th parent, 8th parent

— — — — — — 2^kth parent

5 steps above ⟶ 4 steps above + 1 step above

15 steps after $\longrightarrow$ 1111

$$= \underbrace{8 \text{ steps}}_{} \overset{O(1)}{} \text{ after} + \underbrace{4 \text{ steps after}}_{} \overset{O(1)}{}$$

$$+ \; 2\text{st} \; + \; 1 \text{ step}$$

10 $\longrightarrow$ 1010 $\longrightarrow$ 8 steps after +

2 steps after

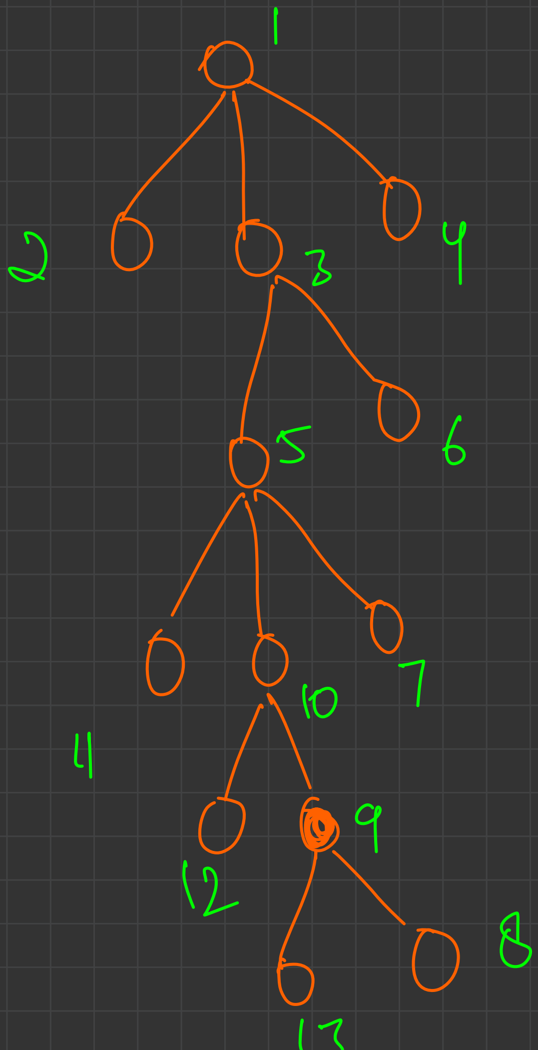
Every k has a unique binary representation

$$\underline{\underline{k}} \rightarrow \left(\underline{\underline{\overset{64\ \ 16\ \ 4}{0010101100}}}\right)$$

$X \rightarrow$ $\underline{\underline{84}}$

$O(\text{no of set bits in } \underline{\underline{k}})$ $\rightarrow$ $\underline{\underline{O(\log k)}}$

$\underline{\underline{64}} \rightarrow \underline{\underline{16}} \rightarrow$ $\underline{\underline{4 \text{ steps}}}$



1st Parent

O(n)

1	2	3	4	5	6	7	8	9	10	11	12	13
-1	1	1	1	3	3	5	9	10	5	5	10	9

2nd Parent

9 → 10 → 5

7 → 5 → 3

1	2	3	4	5	6	7	8	9	10	11	12	13
-1	-1	-1	-1	1	1	3	10	5				

2nd Parent of 9 = 5

1st parent

2nd parent

2nd parent of x = 1st parent of [1st parent of x]

4th parent

4th parent of x = 2nd parent of [2nd parent of x]

1st parent

$O(n)$

2nd parent

$O(n)$

4th parent

$O(n)$

$1, 2, 4, 8 \text{ ---- } 2^{\log n}$

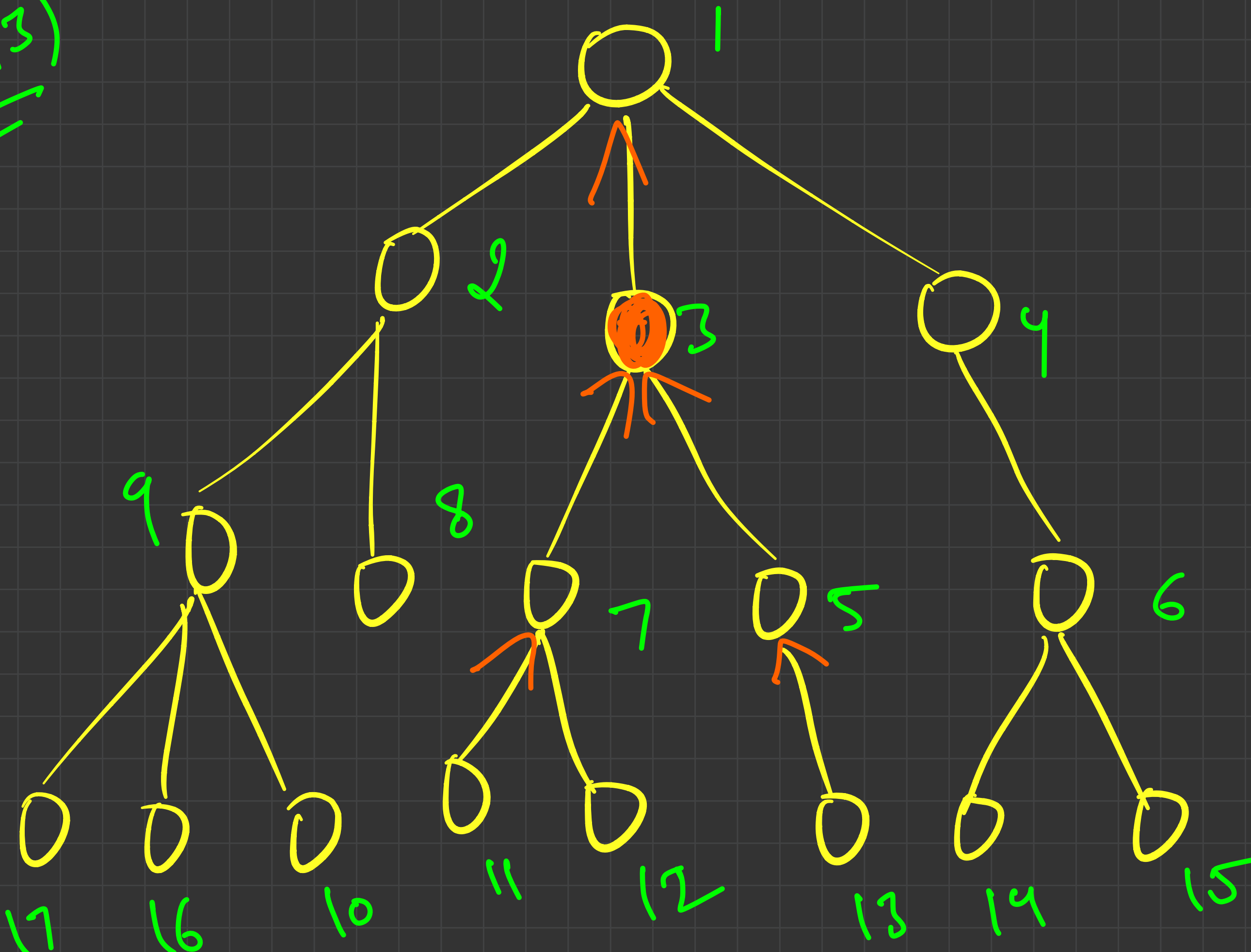
kth point of mid

1st 2nd 4th 8th ---- 2^{m}th

$m \le \log n$

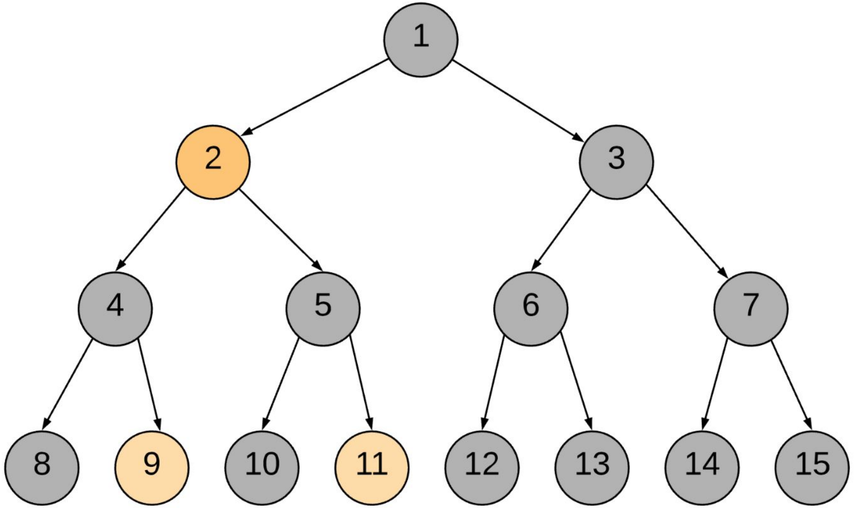
$2^m \le n$

2nd parent of 9 = 1st parent of the 1st parent of 9 lca(11, 13)



Distance between any 2 nodes Link

$$dist(A, B) = level_A + Level_B - 2 * Level_{LCA}$$



Lowest Common Ancestor for **Node 9** and **Node 11** is **Node 2**