Week 6: Wednesday

EECS 281

MAXIM ALEKSA

maximal.io

Split string

Interview Question

Given an input string and a dictionary of words, split the input string into a space-separated sequence of dictionary words if possible.

For example, if the input string is `"applepie"` and dictionary contains a standard set of English words, then we would return the string `"apple pie"` as output.

http://thenoisychannel.com/2011/08/08/retiring-a-great-interview-problem

Split string

Interview Question

Agenda

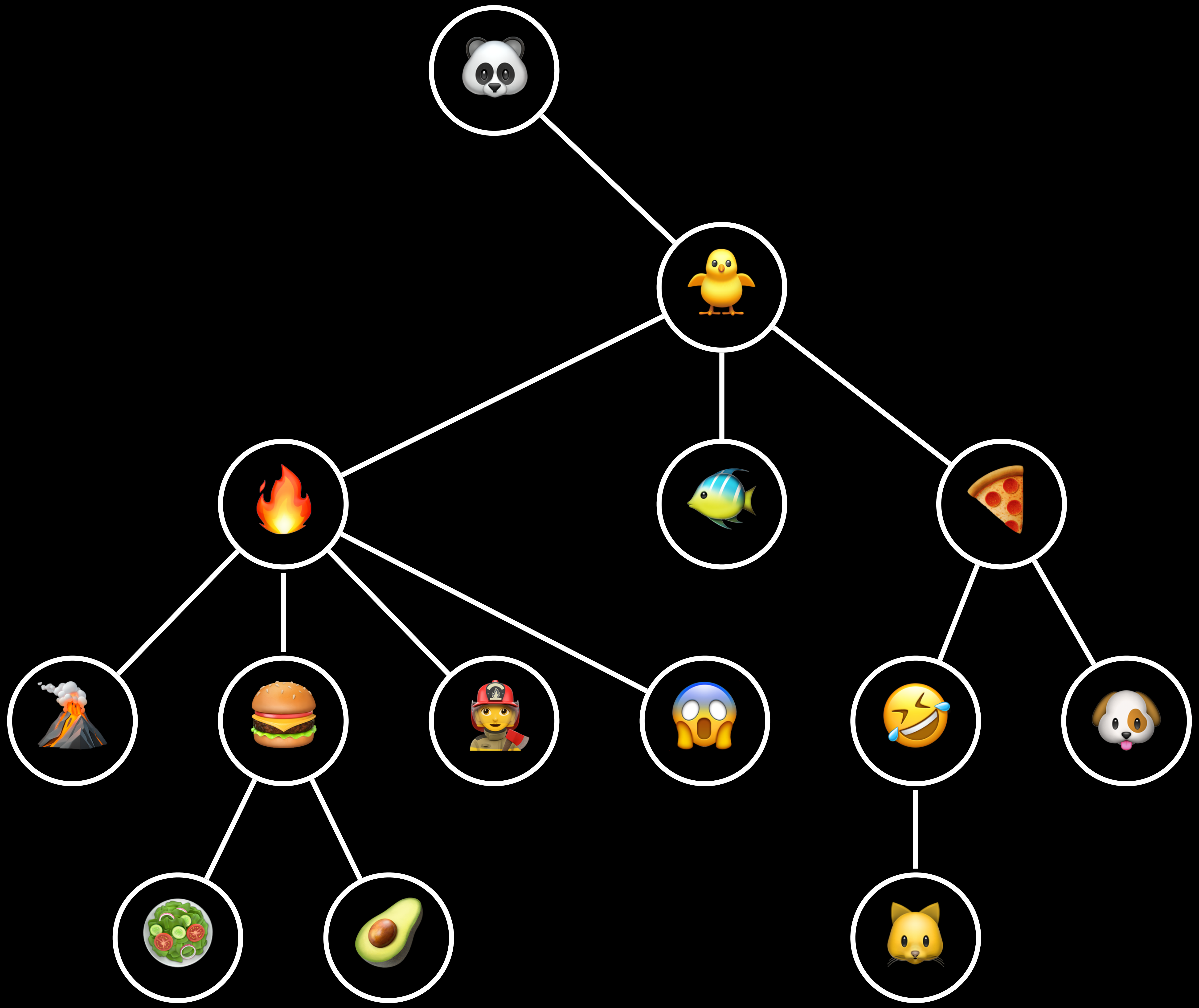
AVL Trees

Graphs

Graph Representations

Graph Traversals

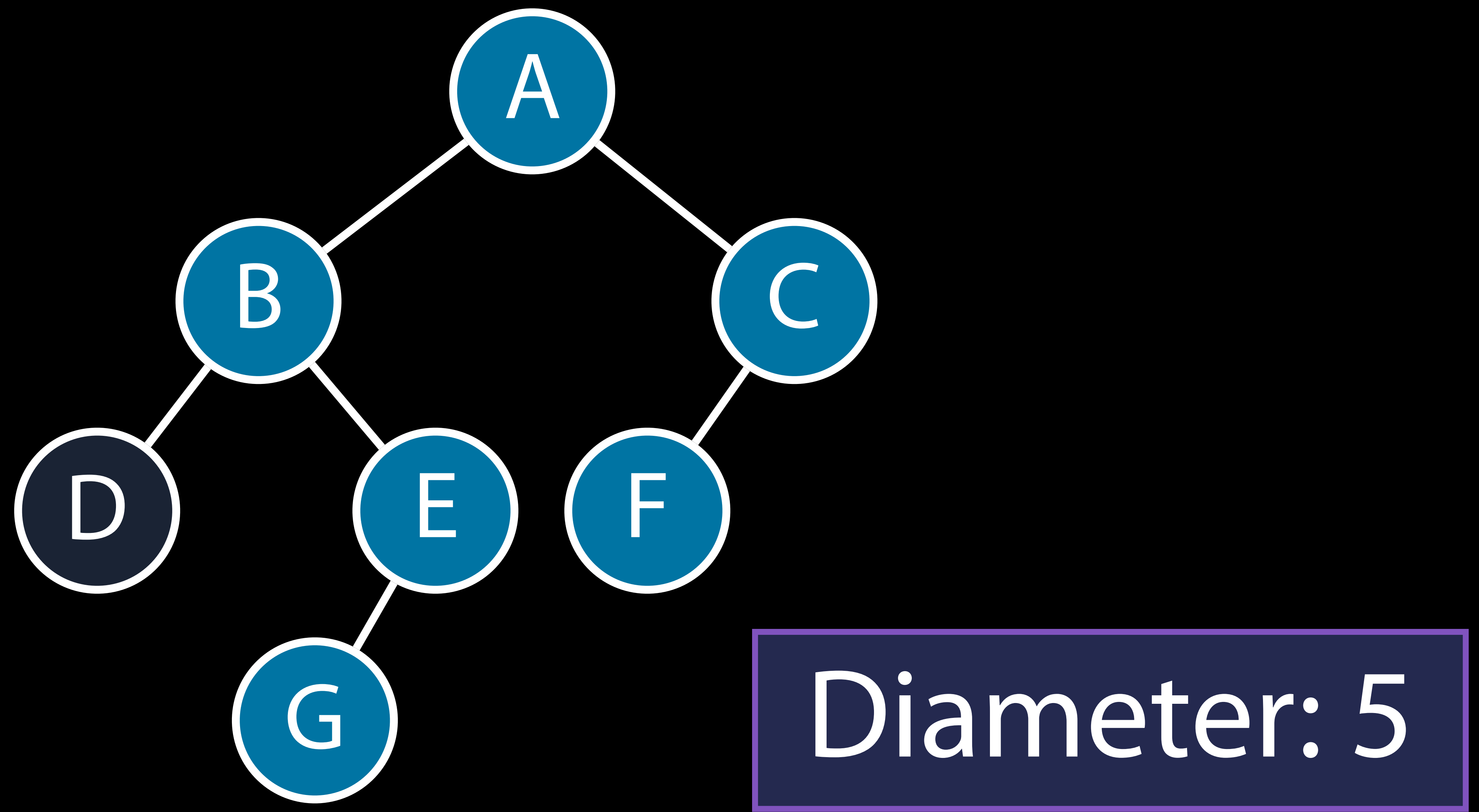
Trees

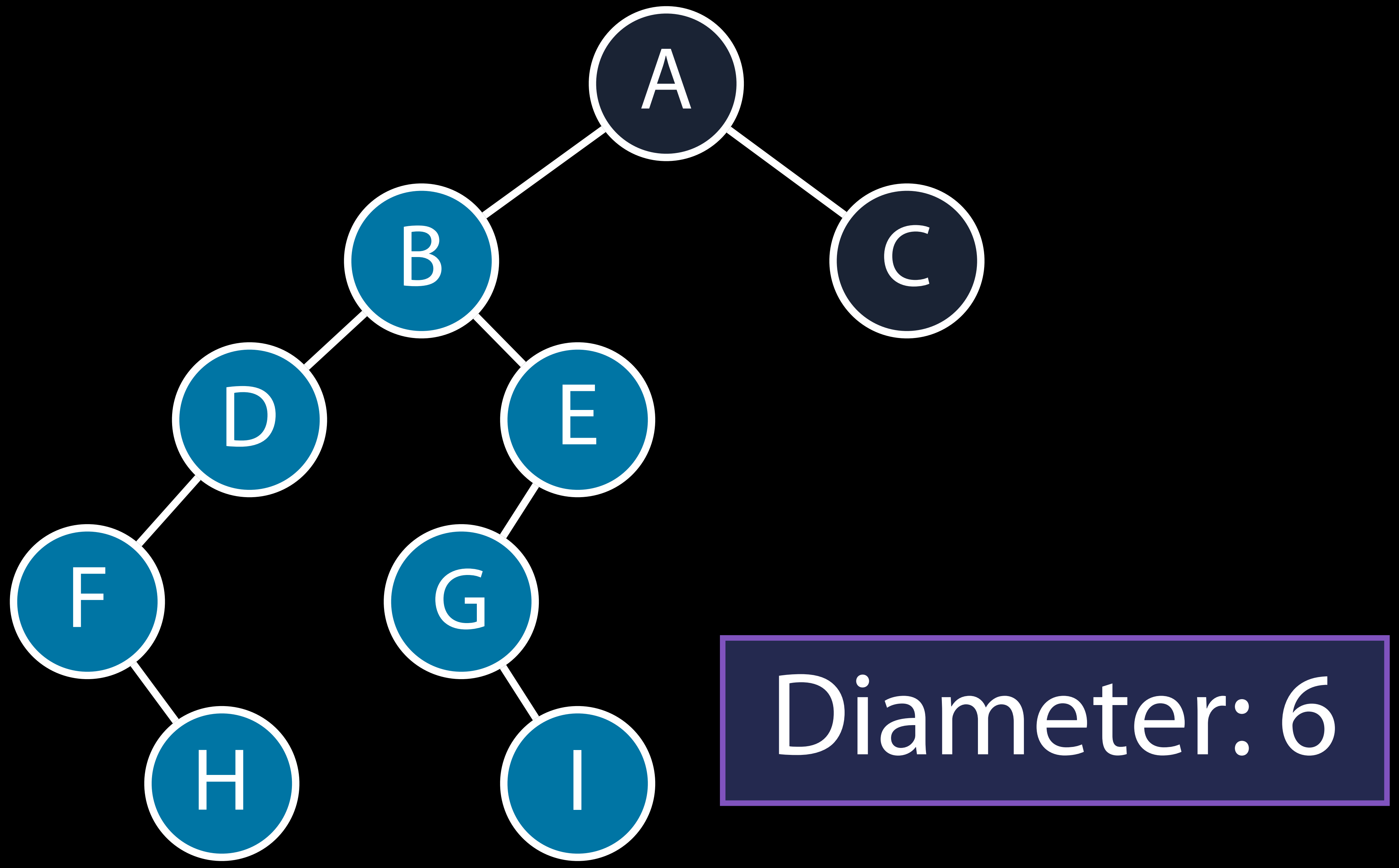


Tree diameter

The **diameter** of a tree is the maximum number of edges on any path connecting two nodes of the tree.

The diameter of an empty tree or a tree with one node is 0.





Tree Diameter

```
int height(const BinaryTreeNode* tree) {
    // represent absent nodes with height -1
    if (tree == nullptr) {
        return -1;
    } else {
        return max(height(tree->left), height(tree->right)) + 1;
    }
}

int diameter(const BinaryTreeNode* tree) {
    if (tree == nullptr) {
        return 0;
    } else {
        // check diameter of subtrees
        int childDiameter = max(diameter(tree->left), diameter(tree->right));
        int nodeDiameter = 2 + height(tree->left) + height(tree->right);
        return max(childDiameter, nodeDiameter);
    }
}
```

Tree diameter

```cpp
class BinaryTreeNode {
public:
    BinaryTreeNode* left = nullptr;
    BinaryTreeNode* right = nullptr;
    int value;
    BinaryTreeNode(int n) : value(n) {}
};
```

Balanced BSTs

Time complexity of find, insert, remove: $O(\log n)$

Balanced BSTs

AVL Trees

B-Trees/2-3-4 Trees

BB[α] Trees

Red-black Trees

Splay-Trees

Skip Lists

Scapegoat Trees

Treaps

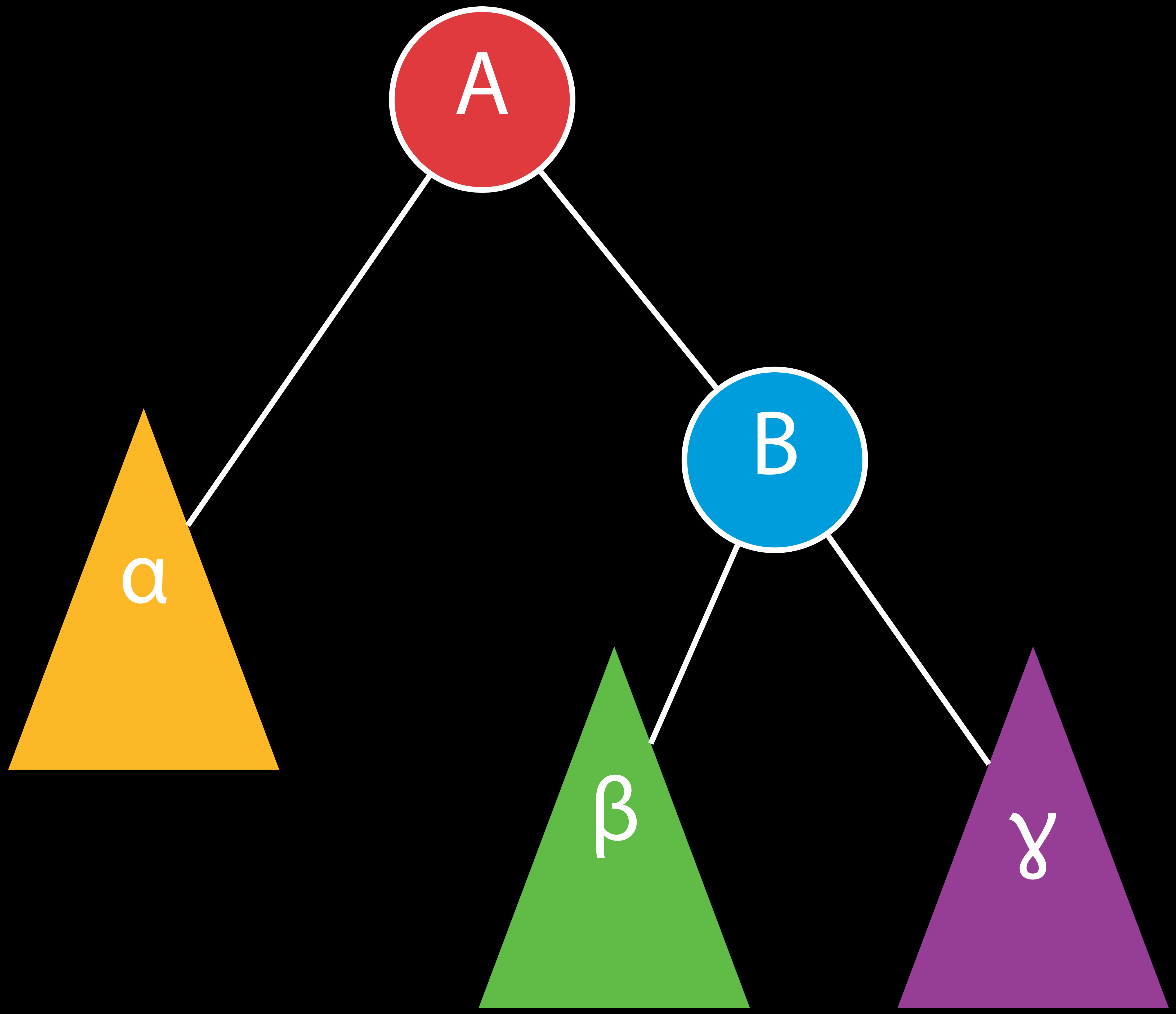
Definitions

Height of a node: max(height of left child, height of right child). Height of a leaf is 0.

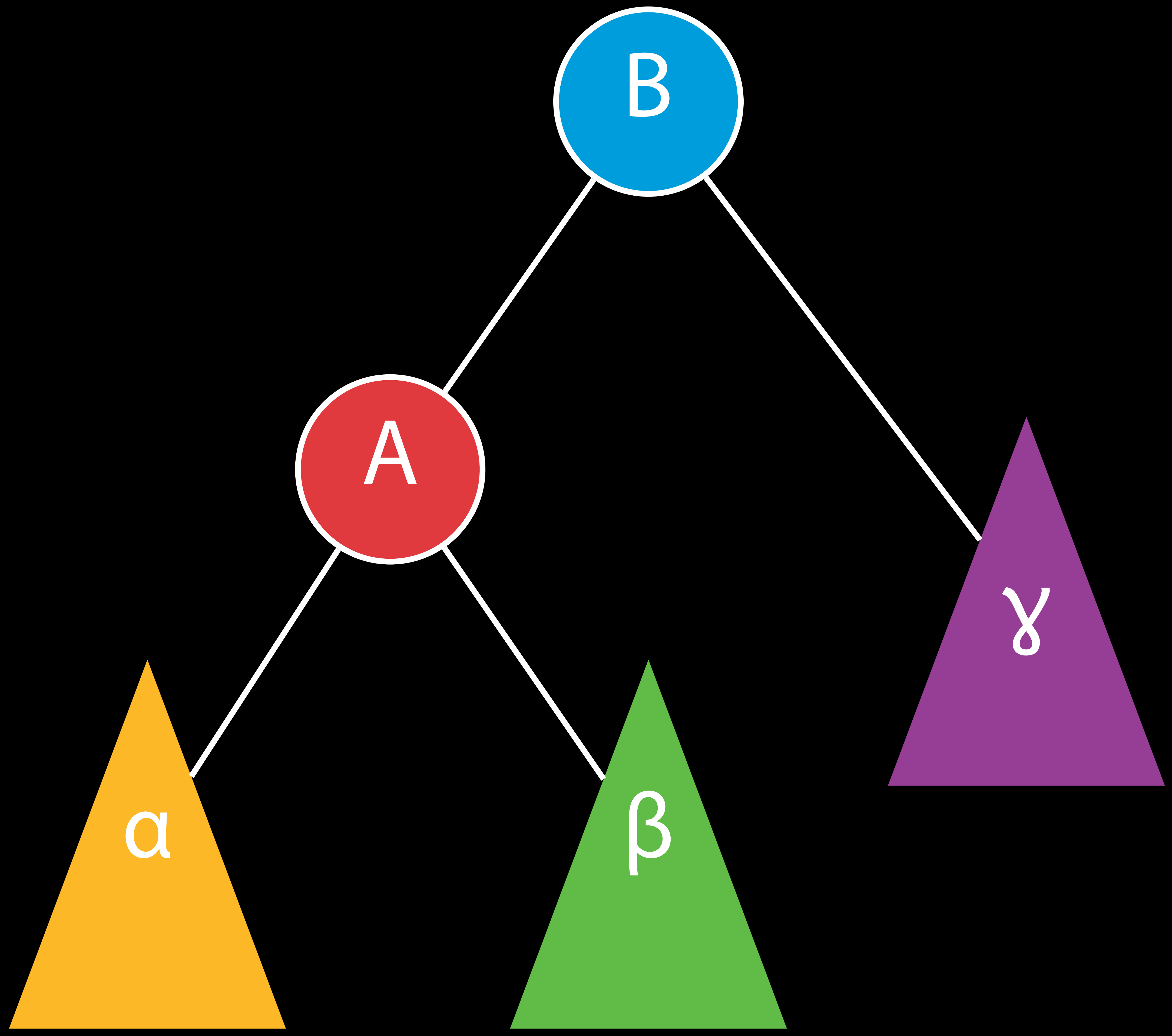
In-order predecessor: go to left subtree and find the right-most node.

In-order successor: go to right subtree and find the left-most node.

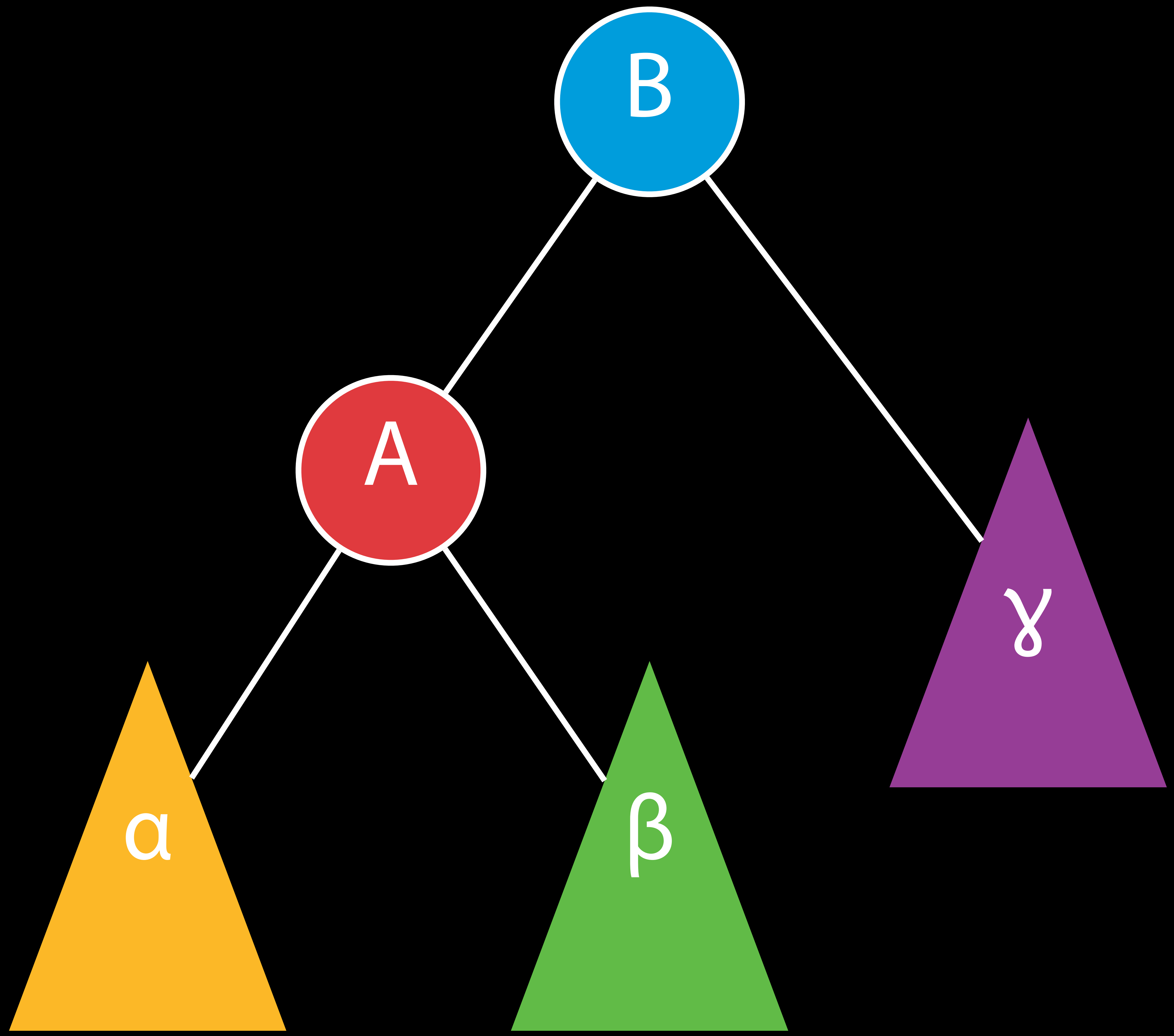
Left Rotation on A



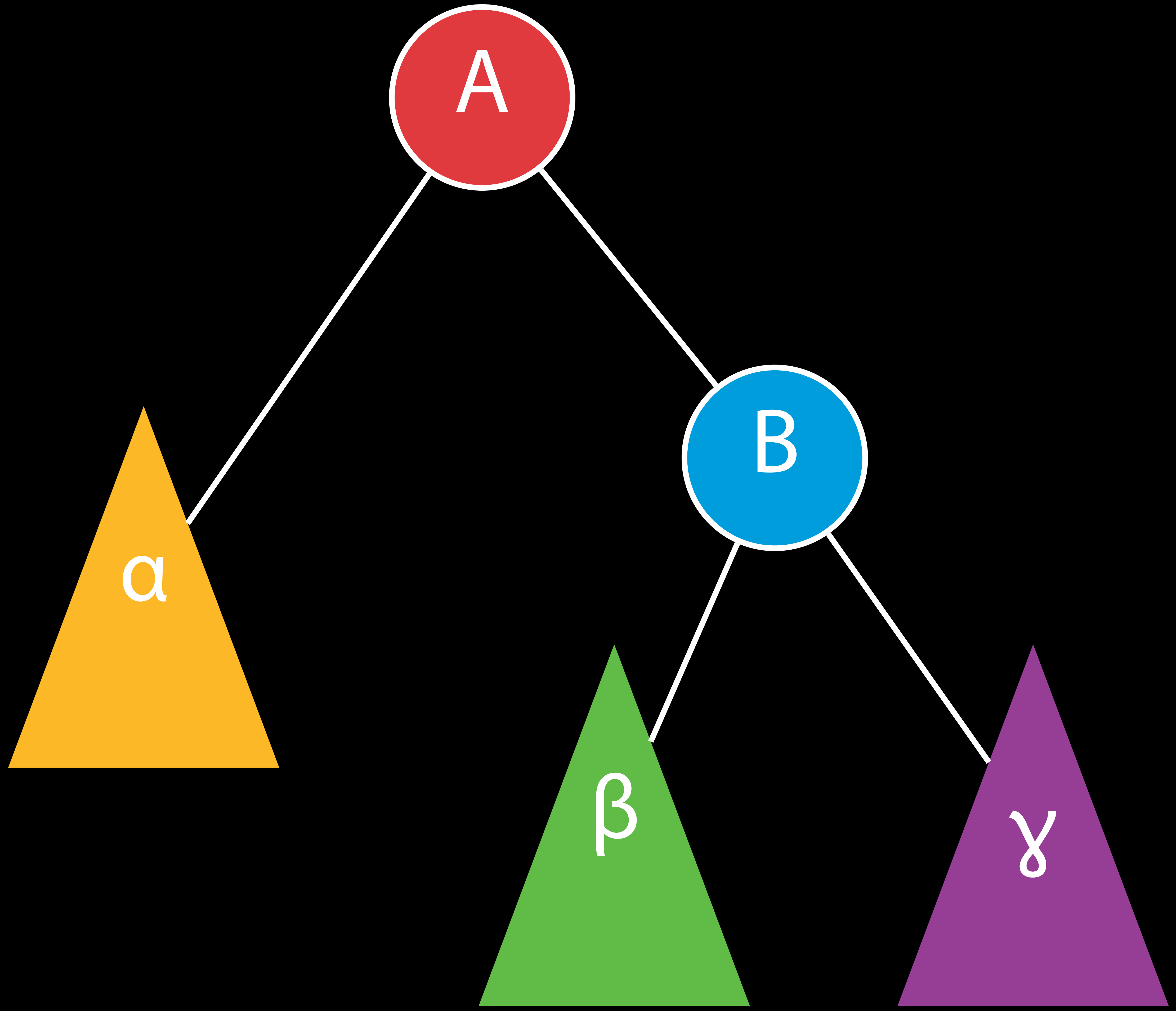
Left Rotation on A



Right Rotation on B

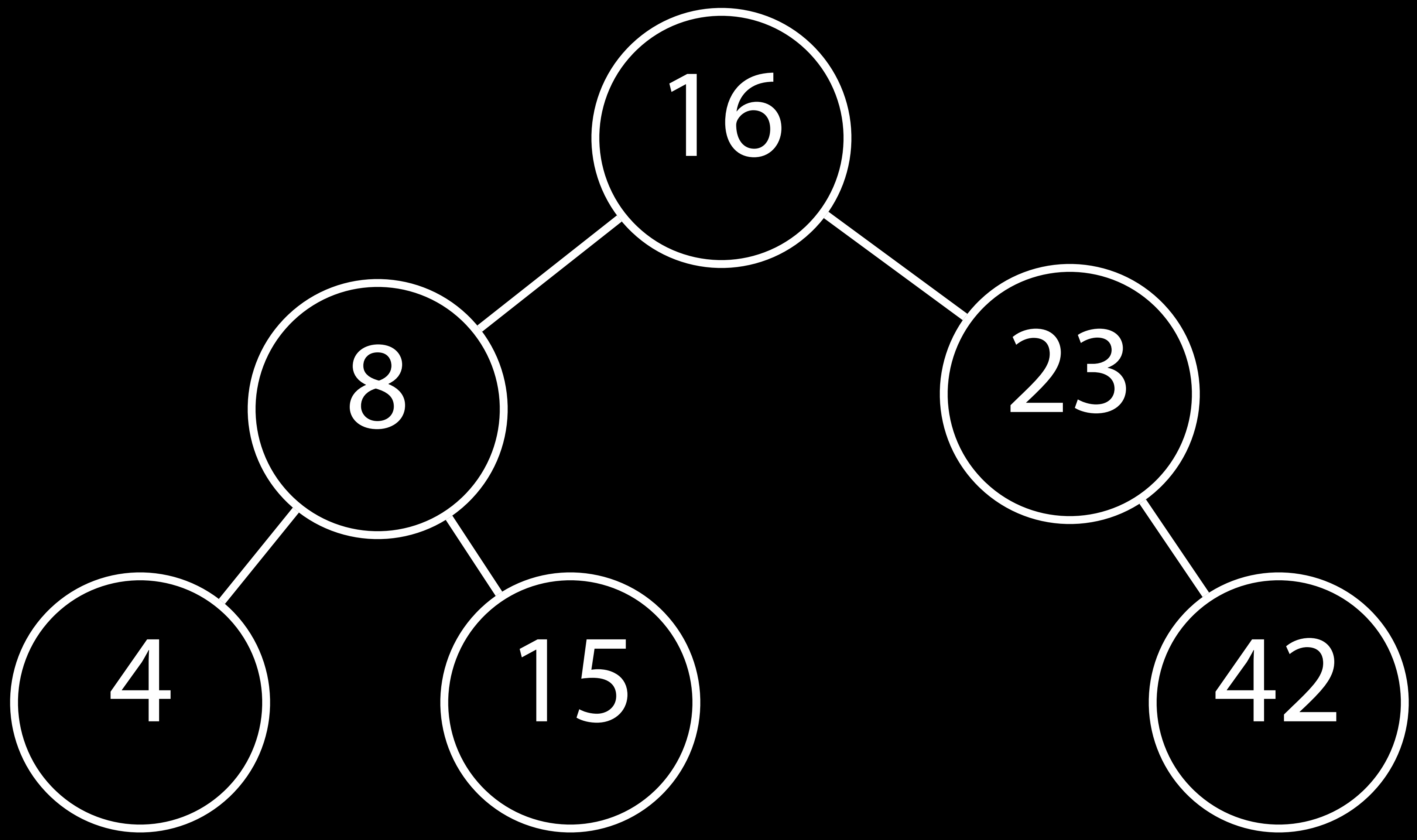


Right Rotation on B



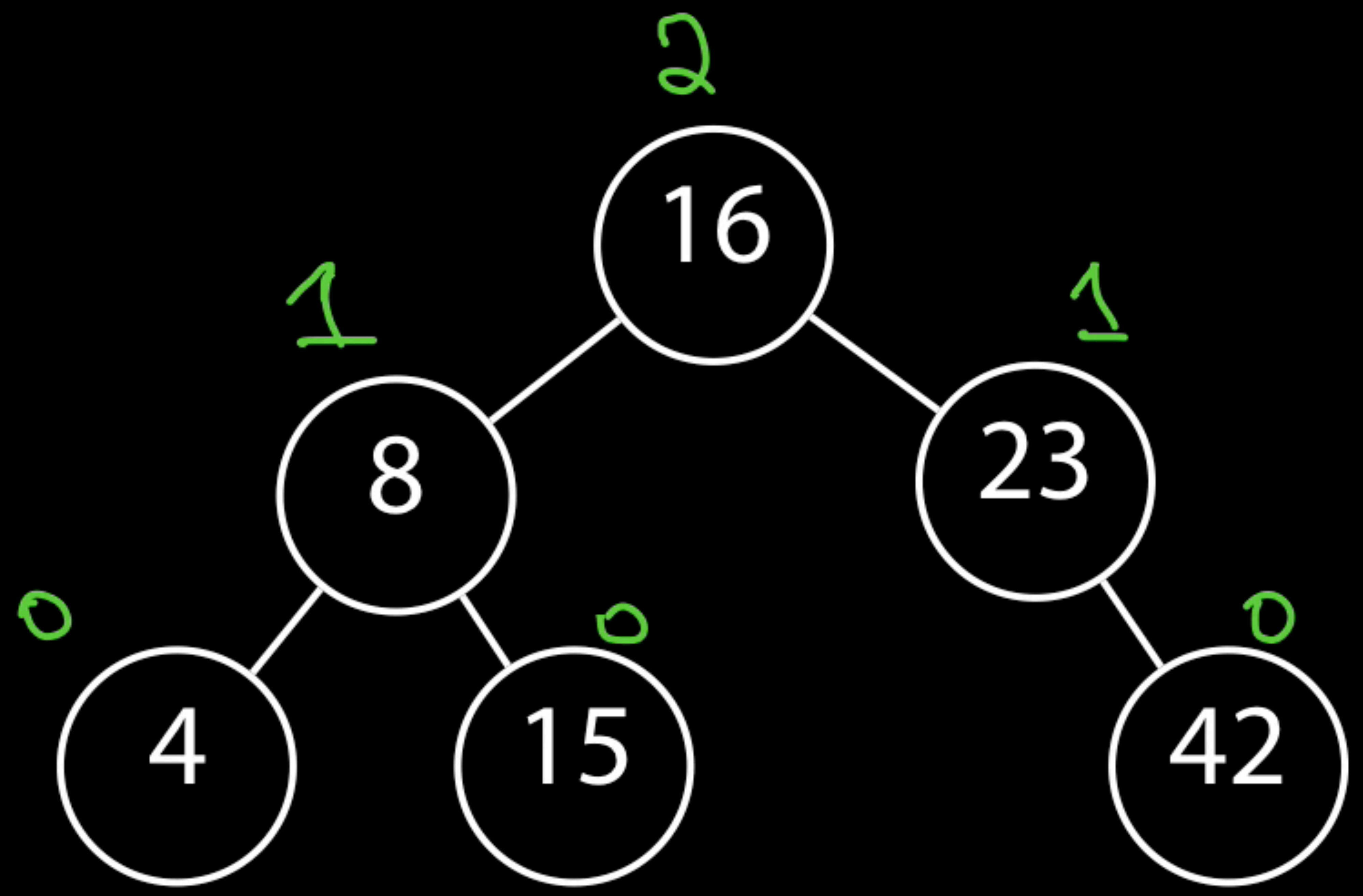
AVL Trees

Binary search tree. For any node, heights of the two child subtrees differ by at most 1.

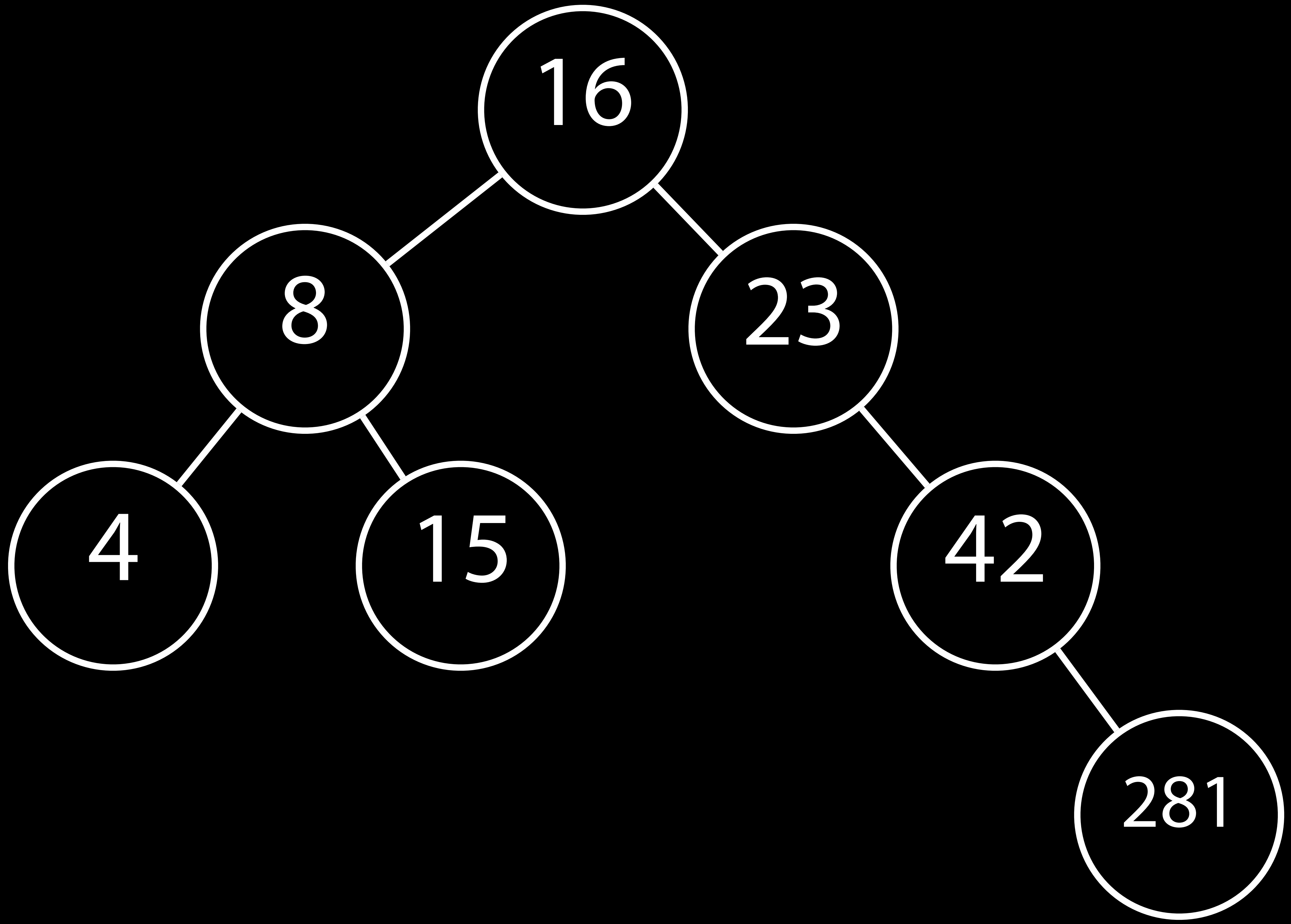


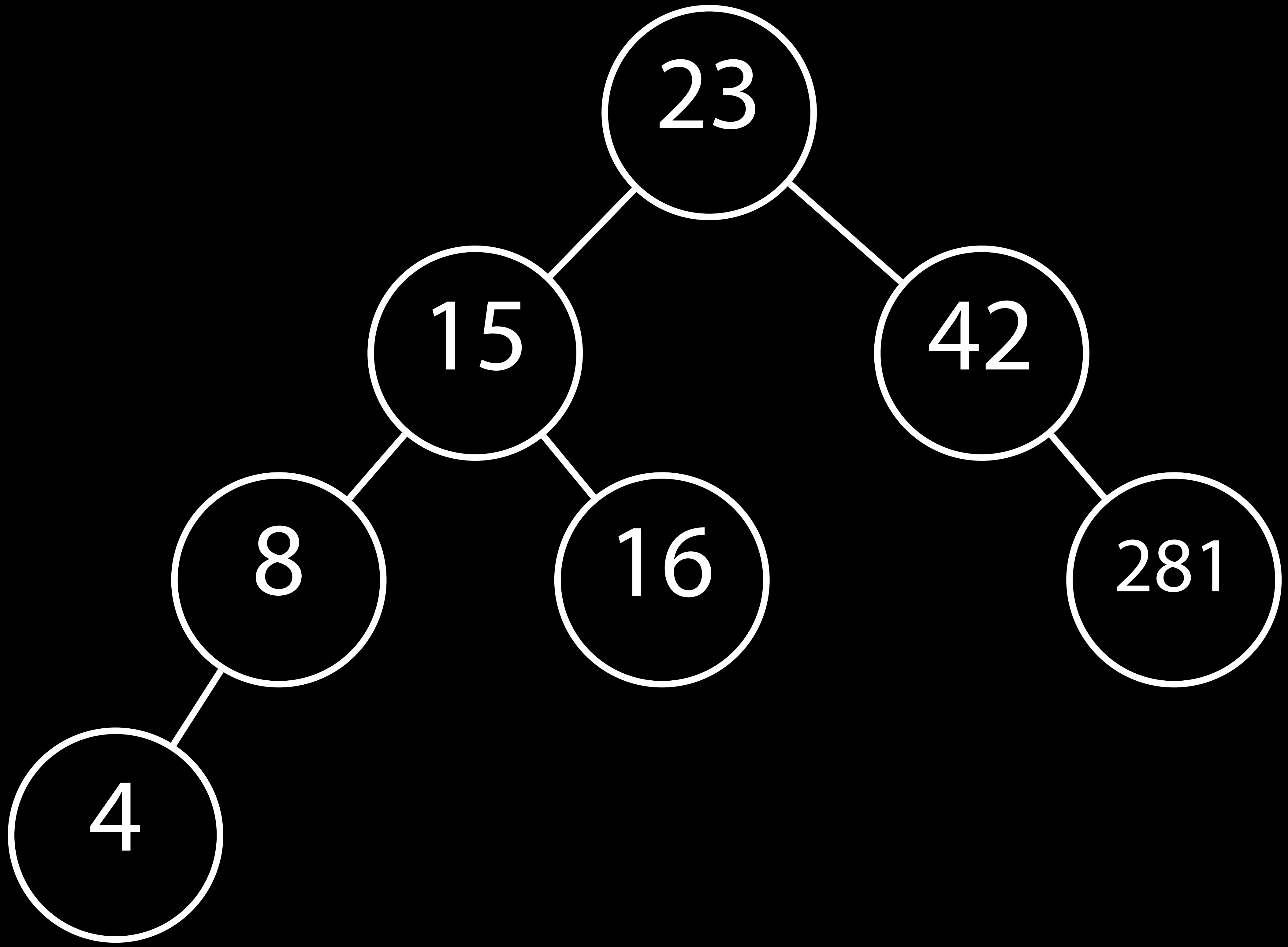
AVL Trees

Binary search tree. For any node, heights of the two child subtrees differ by at most 1.

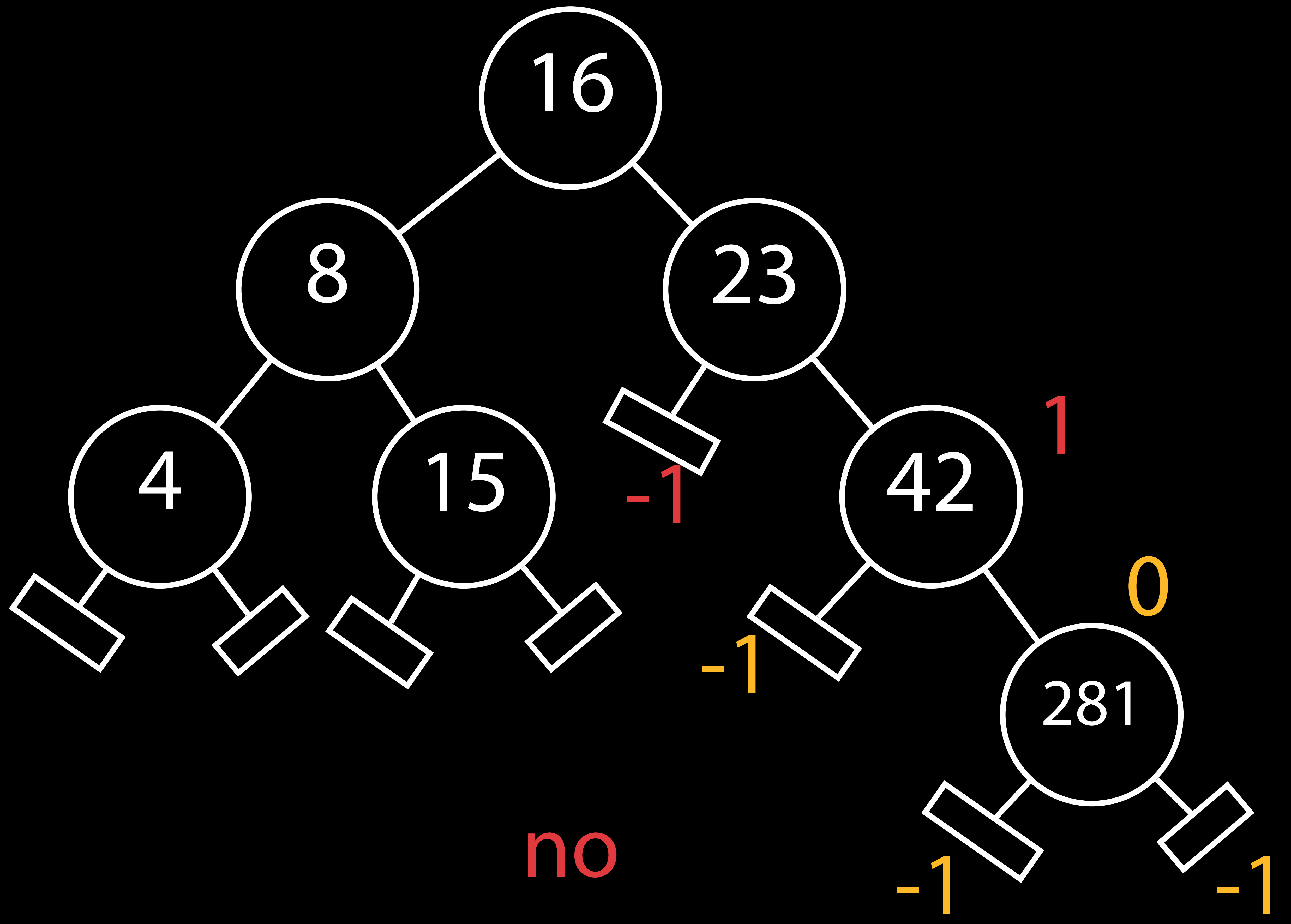


AVL Trees?

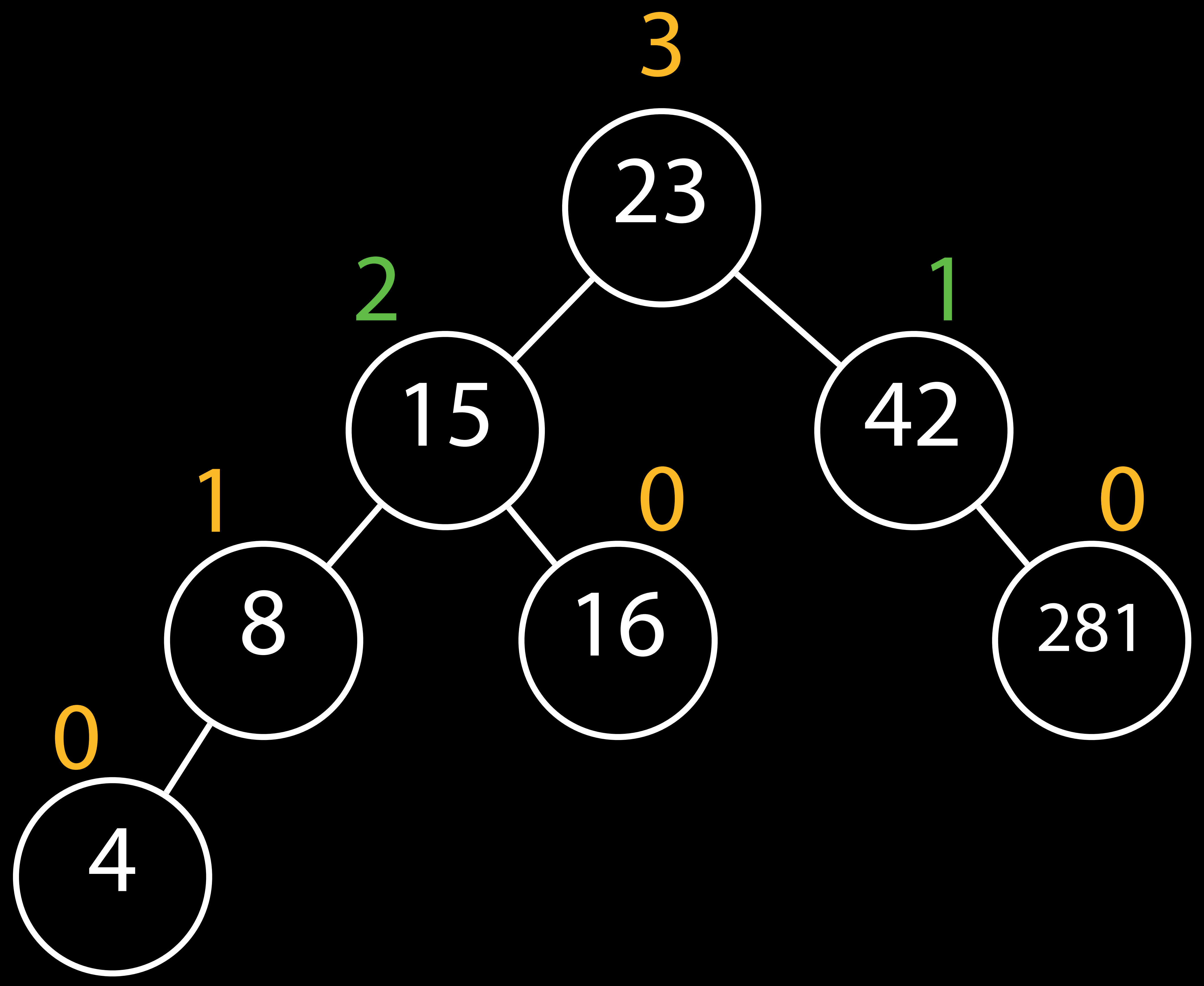




AVL Trees?



no



yes

AVL insert

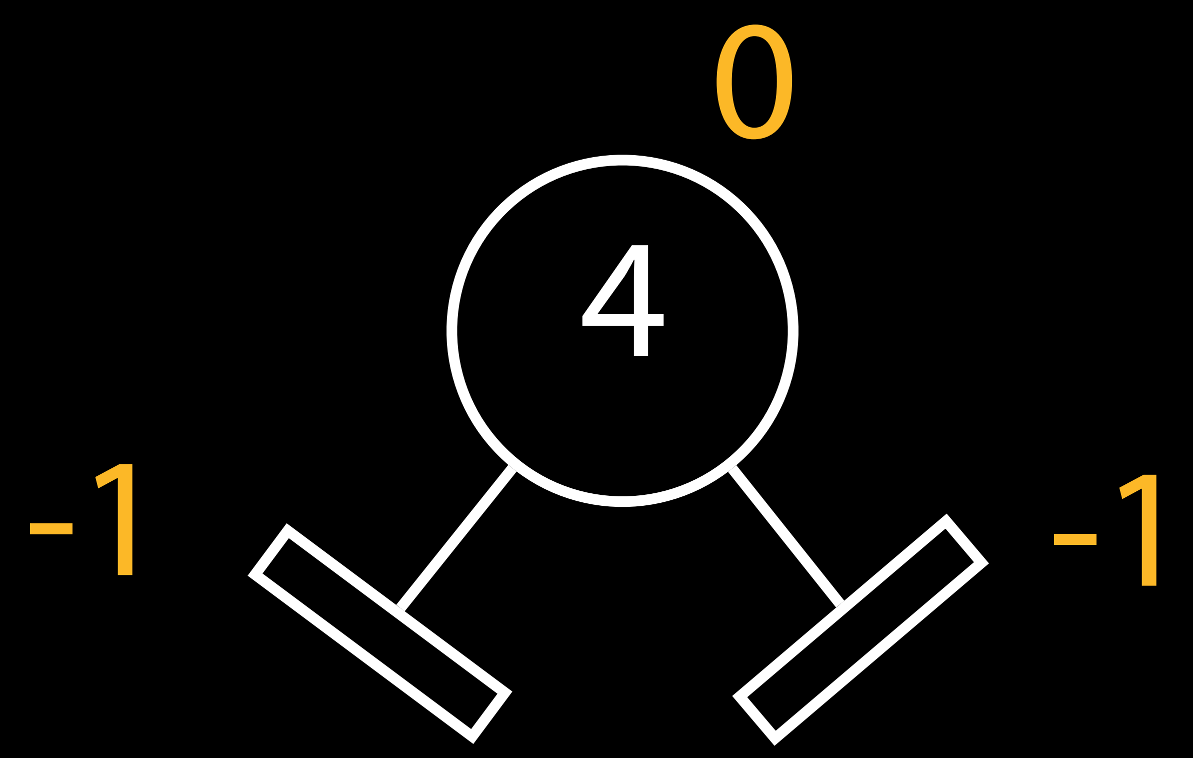
Insert ~~4, 8, 15, 16, 23 and 42~~ 4, 8, 15, 23, 16 and 42.

Insertion

Insert **4**, 8, 15, 23, 16 and 42.

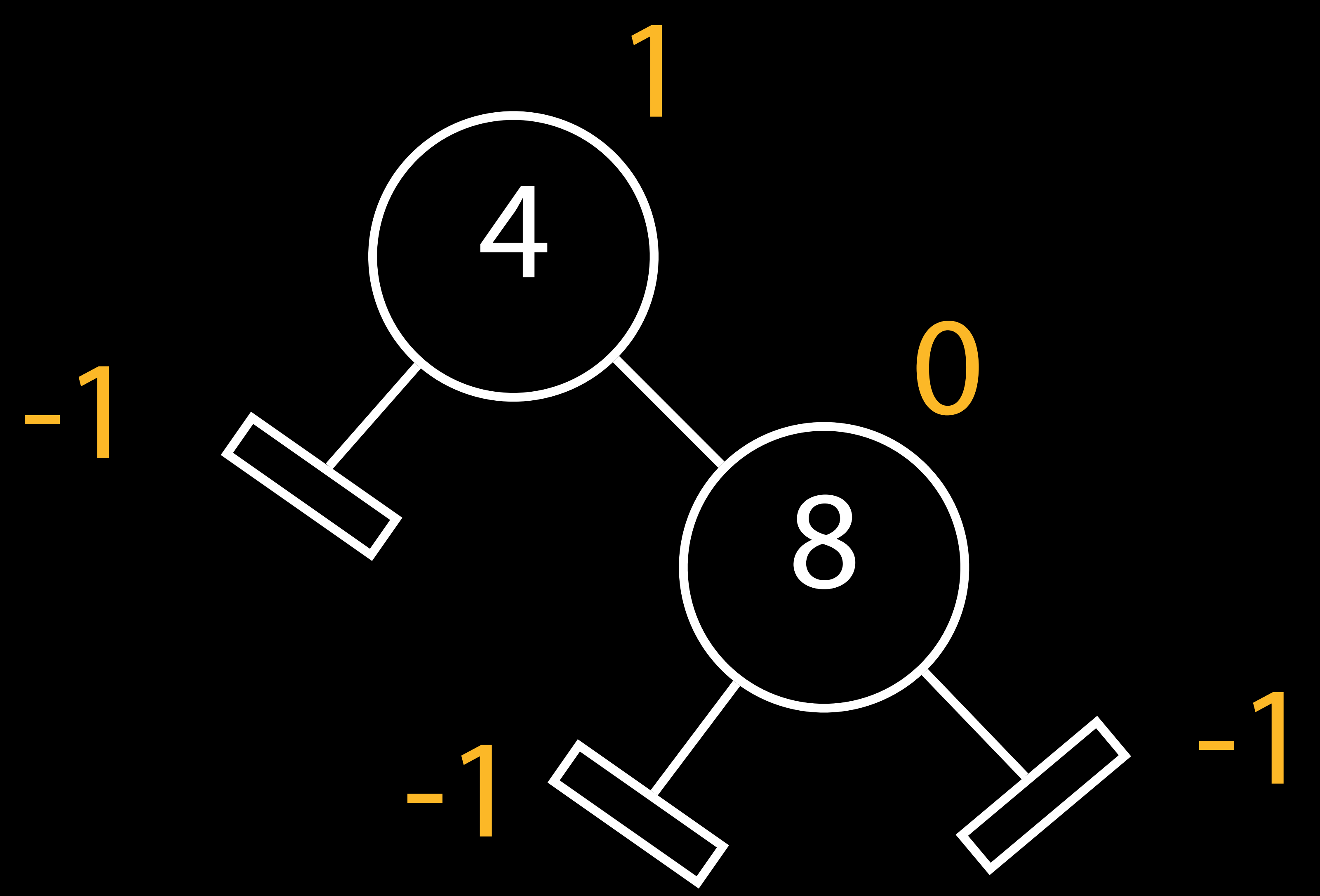
Insertion

Insert 4, 8, 15, 23, 16 and 42.



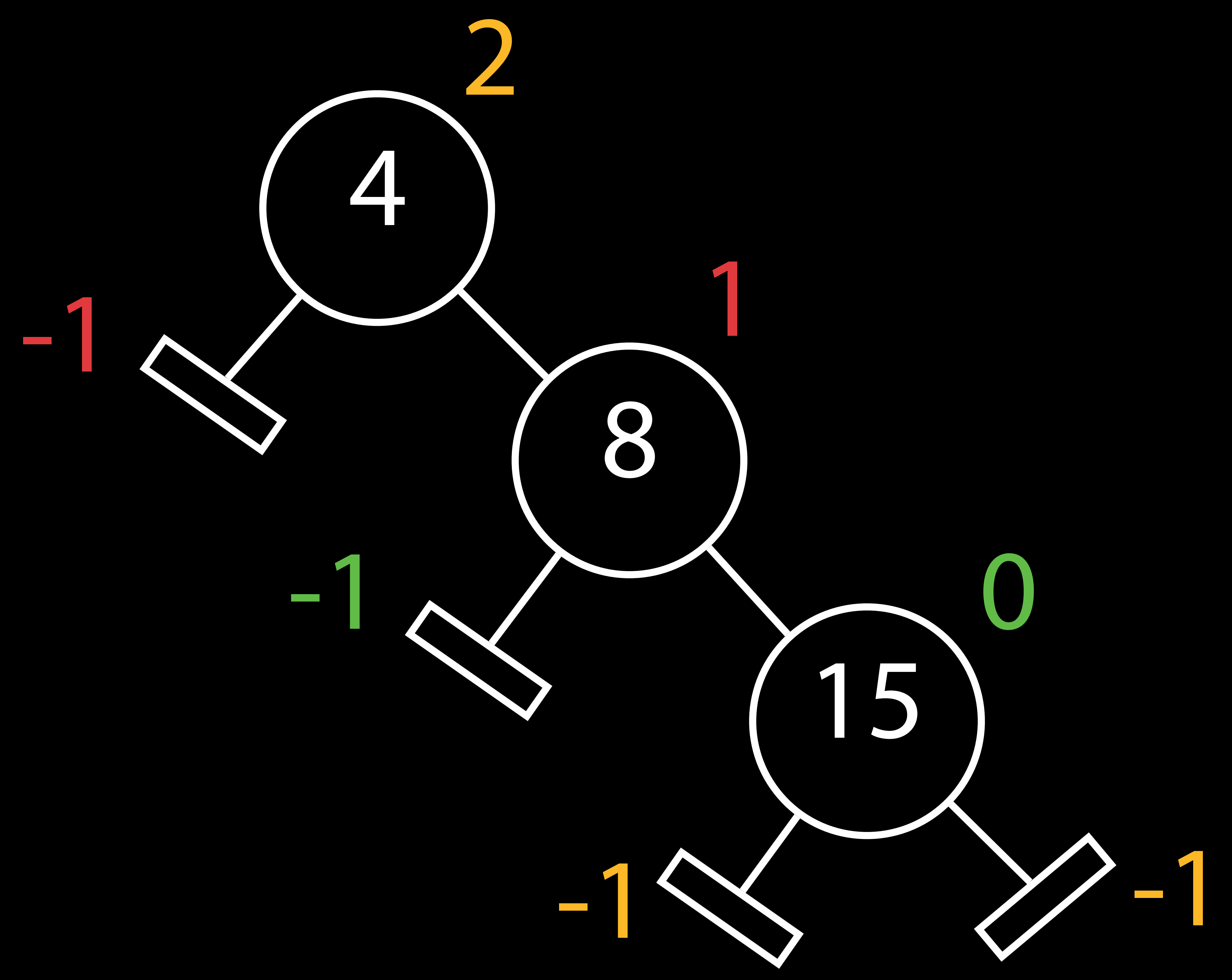
Insertion

Insert 4, 8, 15, 23, 16 and 42.



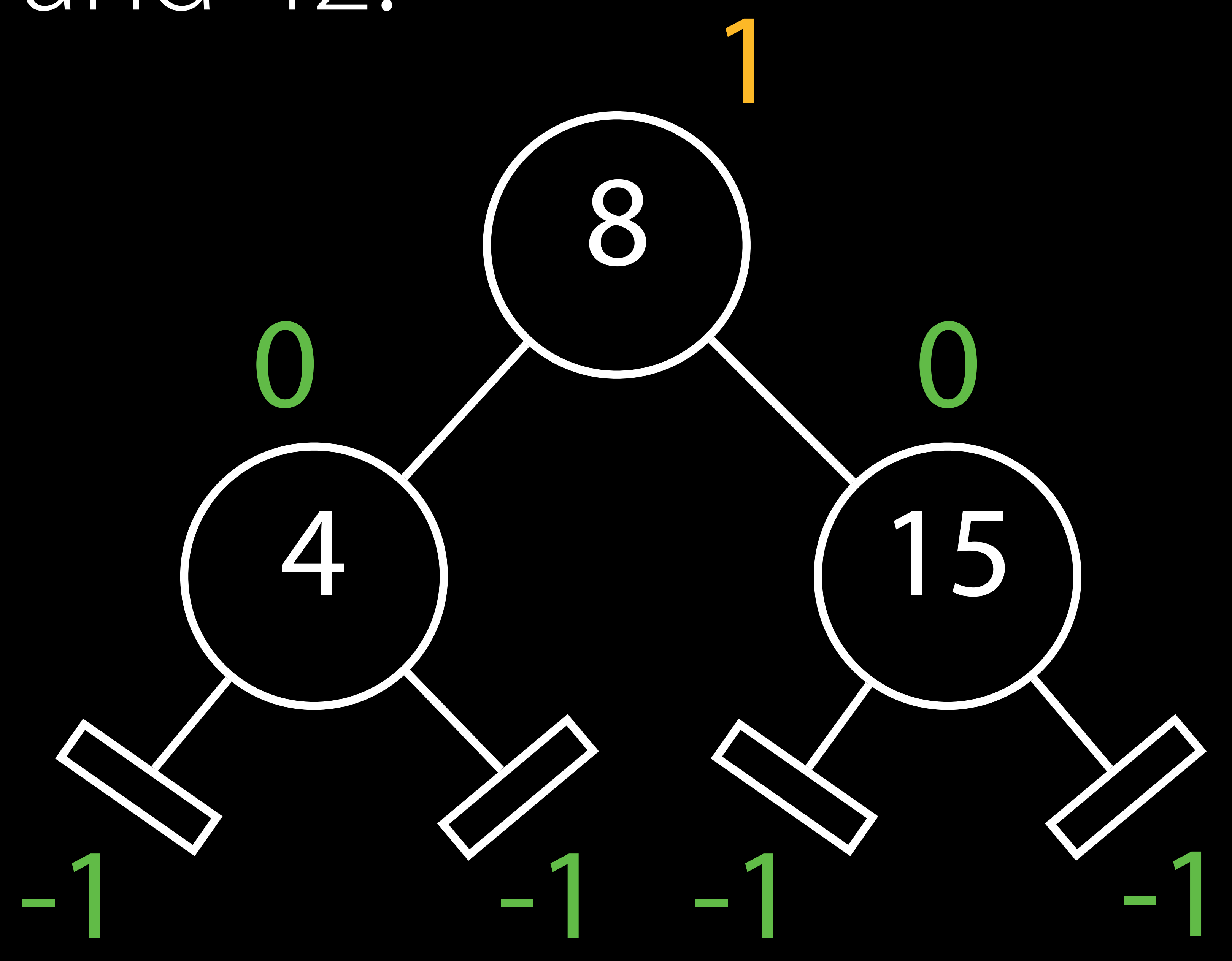
Insertion

Insert 4, 8, 15, 23, 16 and 42.



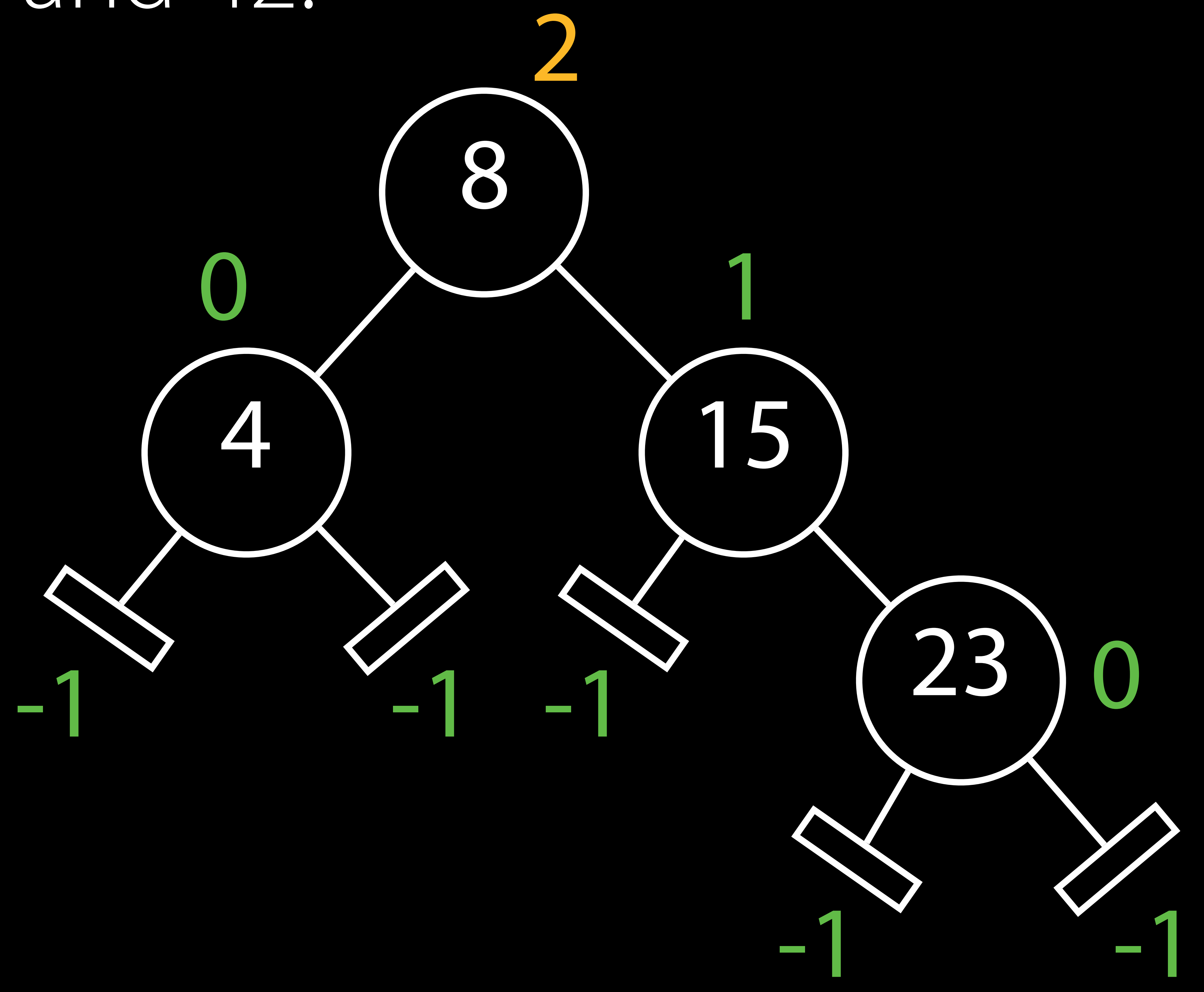
Insertion

Insert 4, 8, 15, 23, 16 and 42.



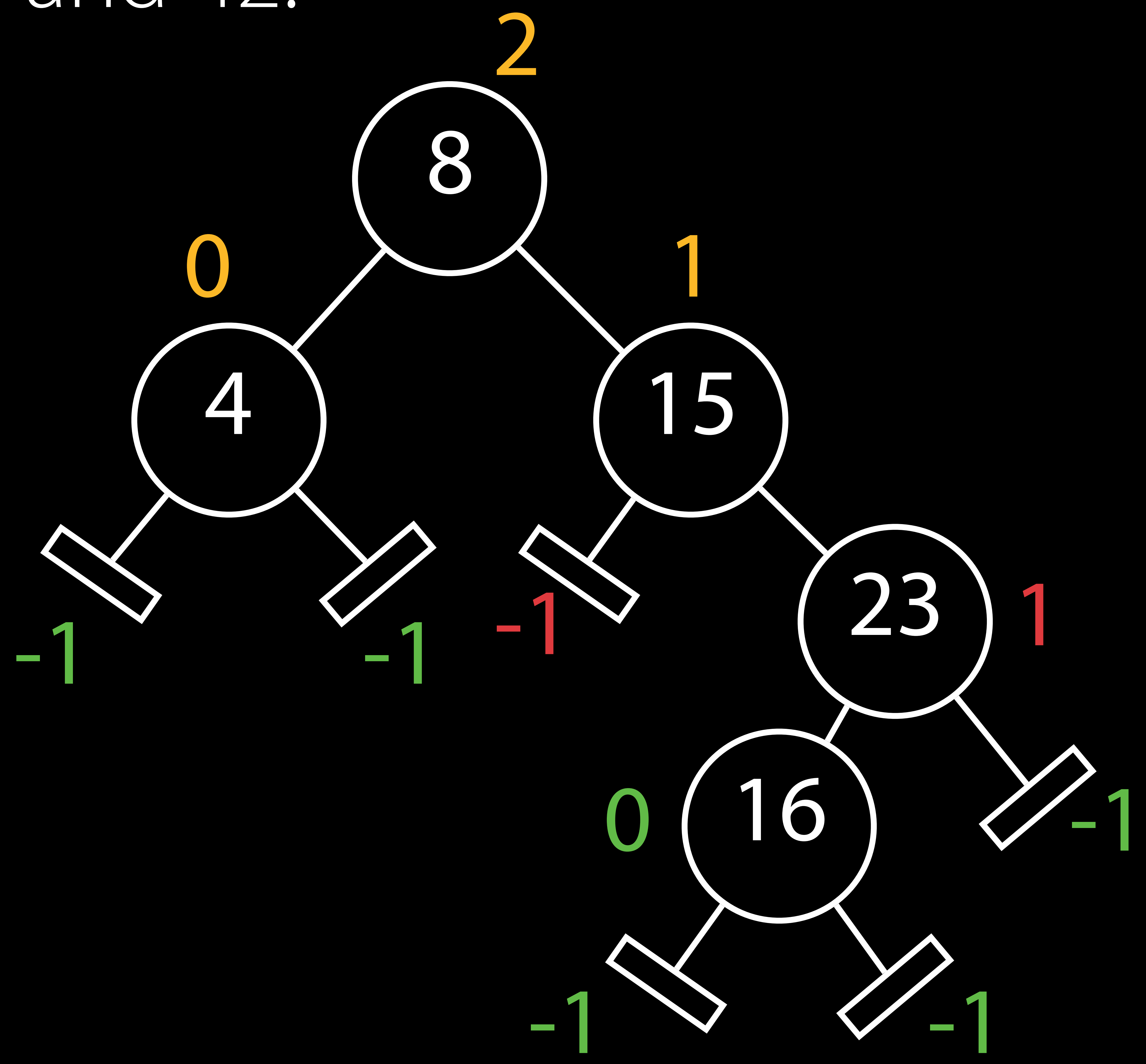
Insertion

Insert 4, 8, 15, 23, 16 and 42.



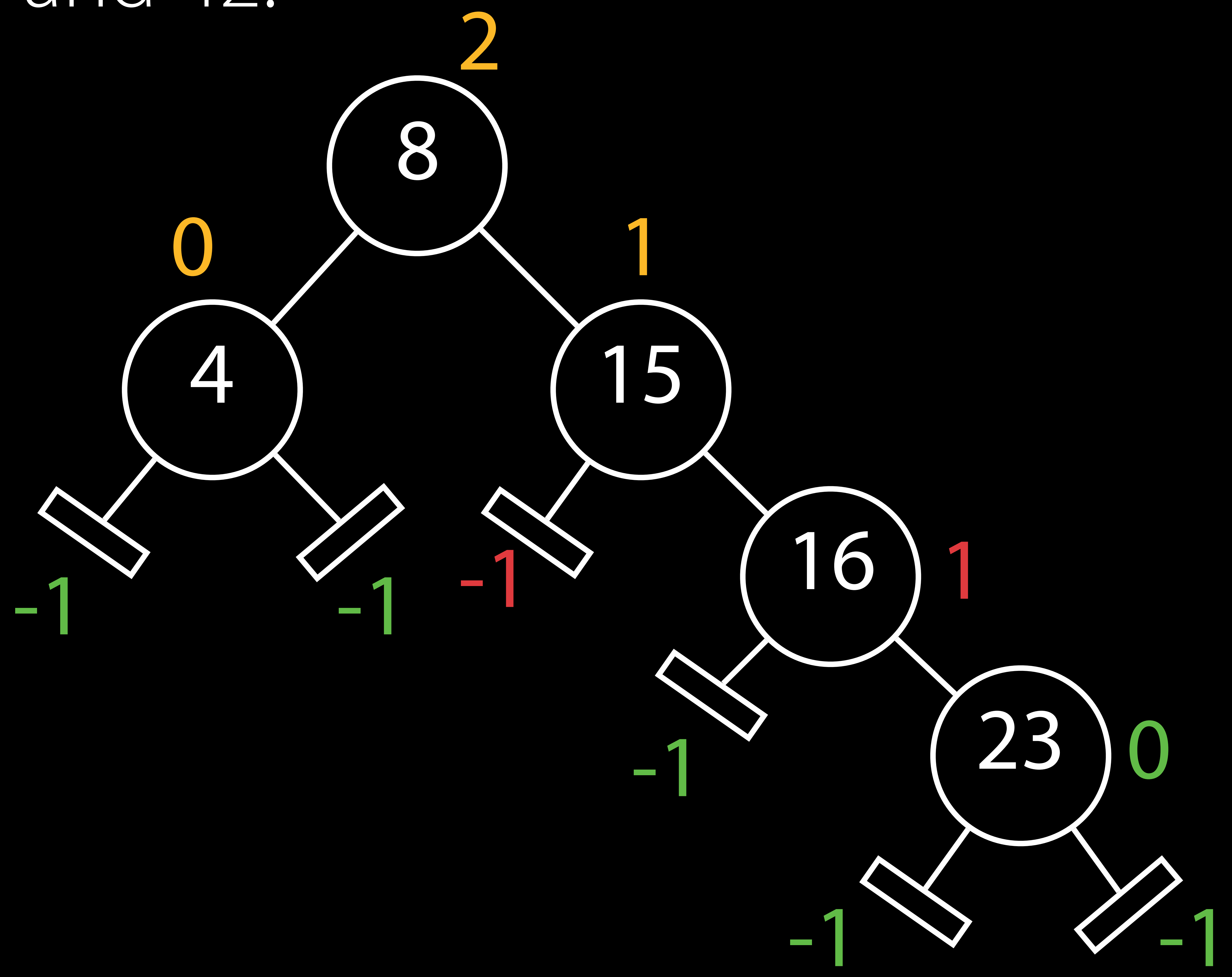
Insertion

Insert 4, 8, 15, 23, 16 and 42.



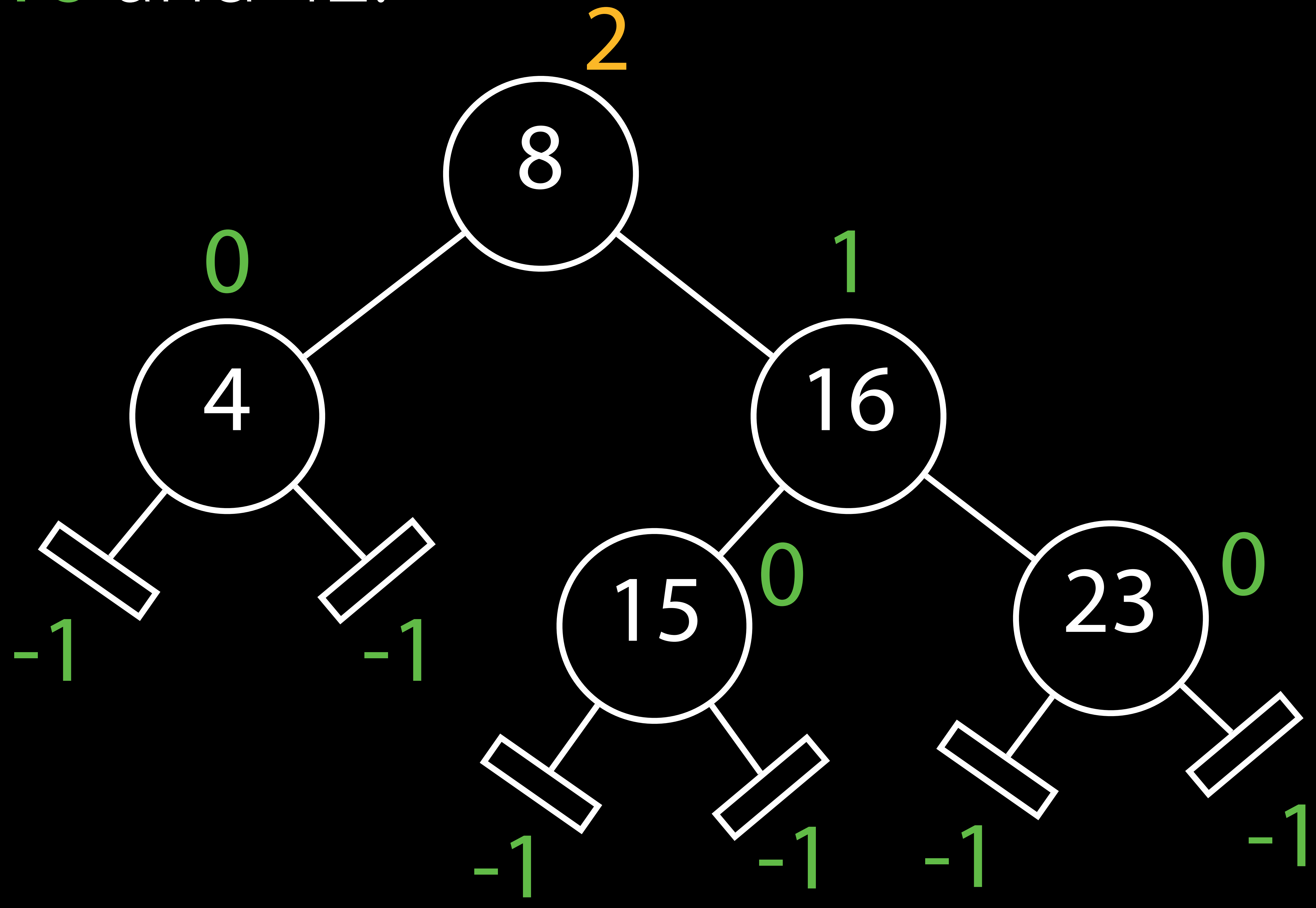
Insertion

Insert 4, 8, 15, 23, 16 and 42.



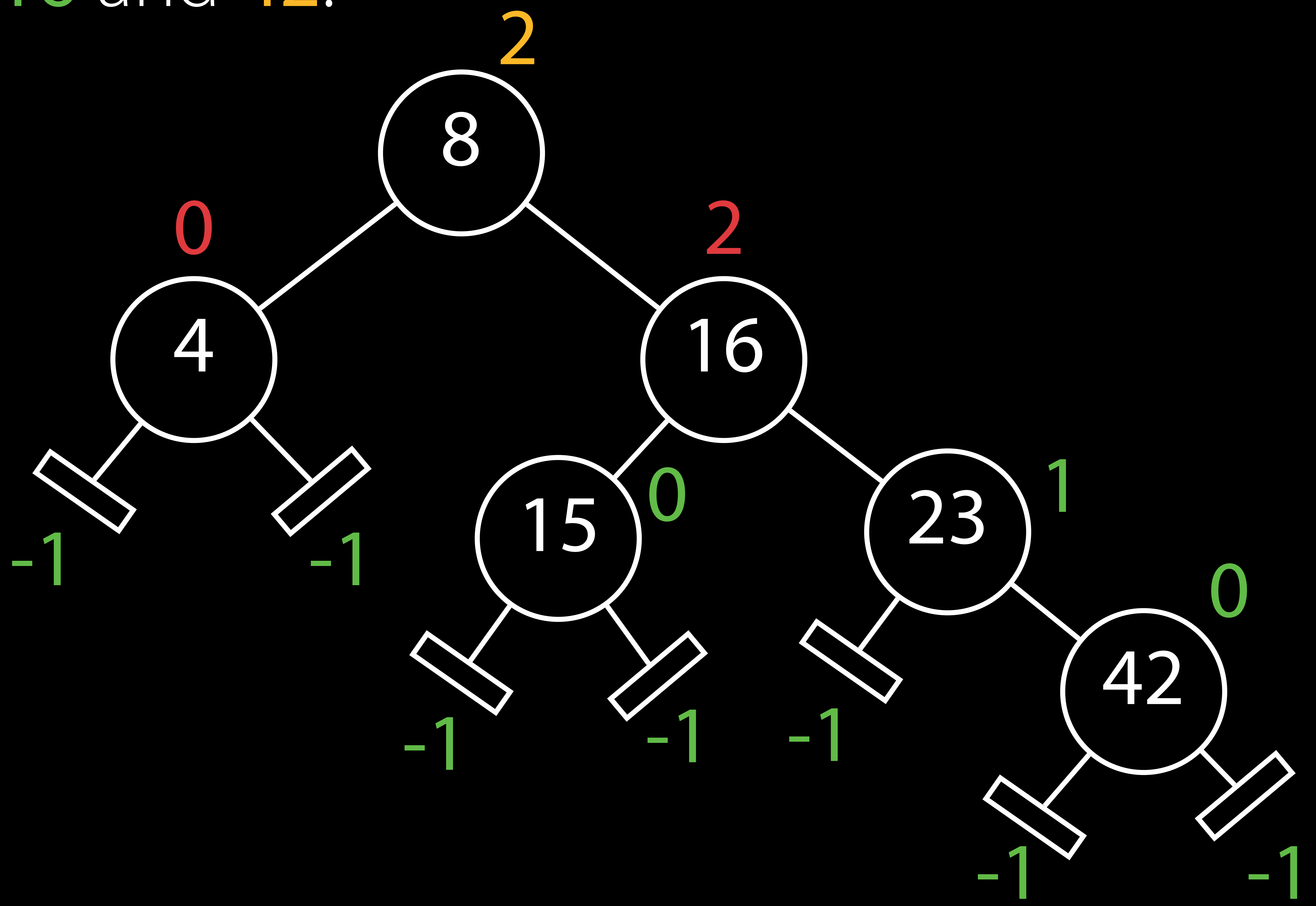
Insertion

Insert 4, 8, 15, 23, 16 and 42.



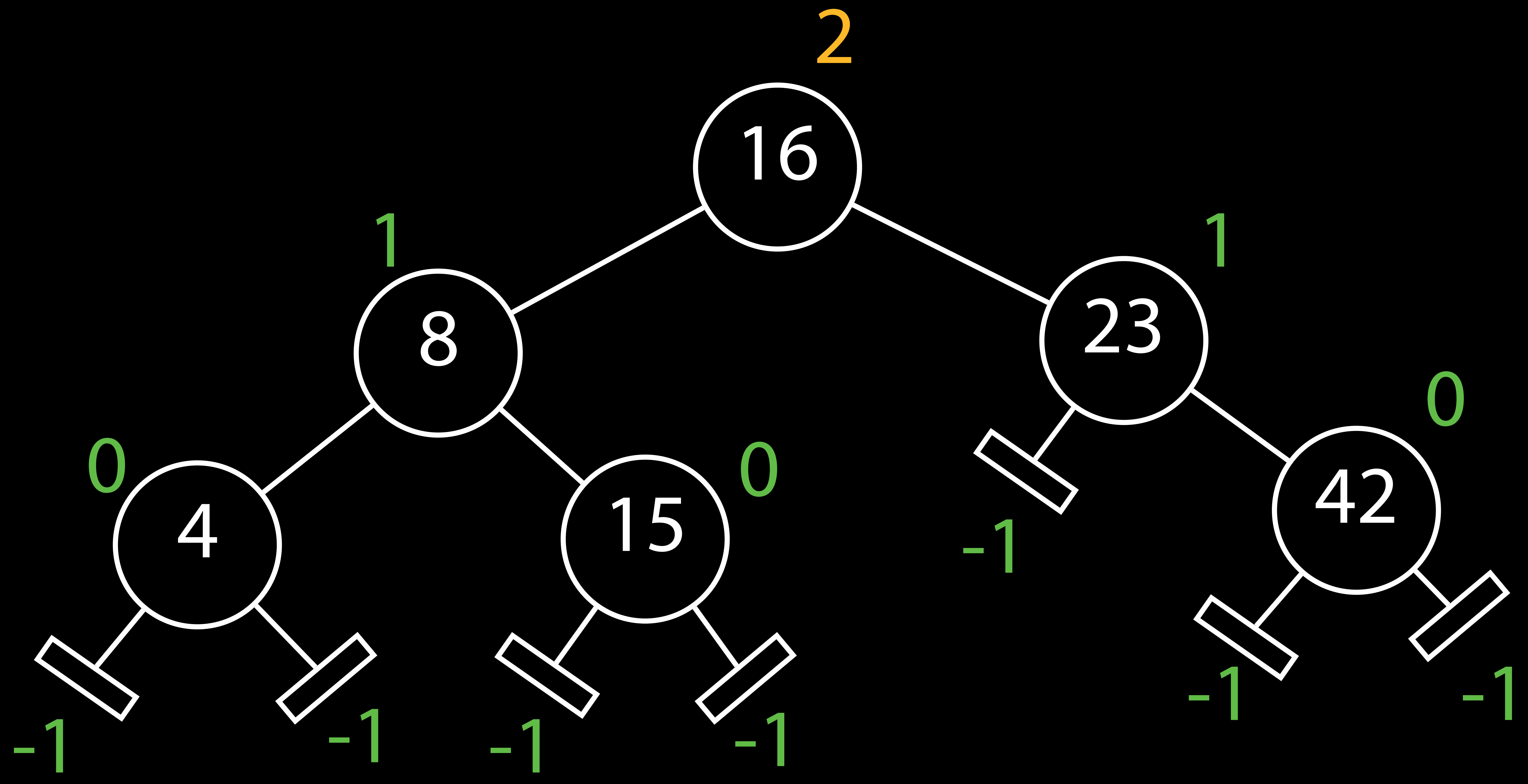
Insertion

Insert 4, 8, 15, 23, 16 and 42.



Insertion

Insert 4, 8, 15, 23, 16 and 42.



AVL remove

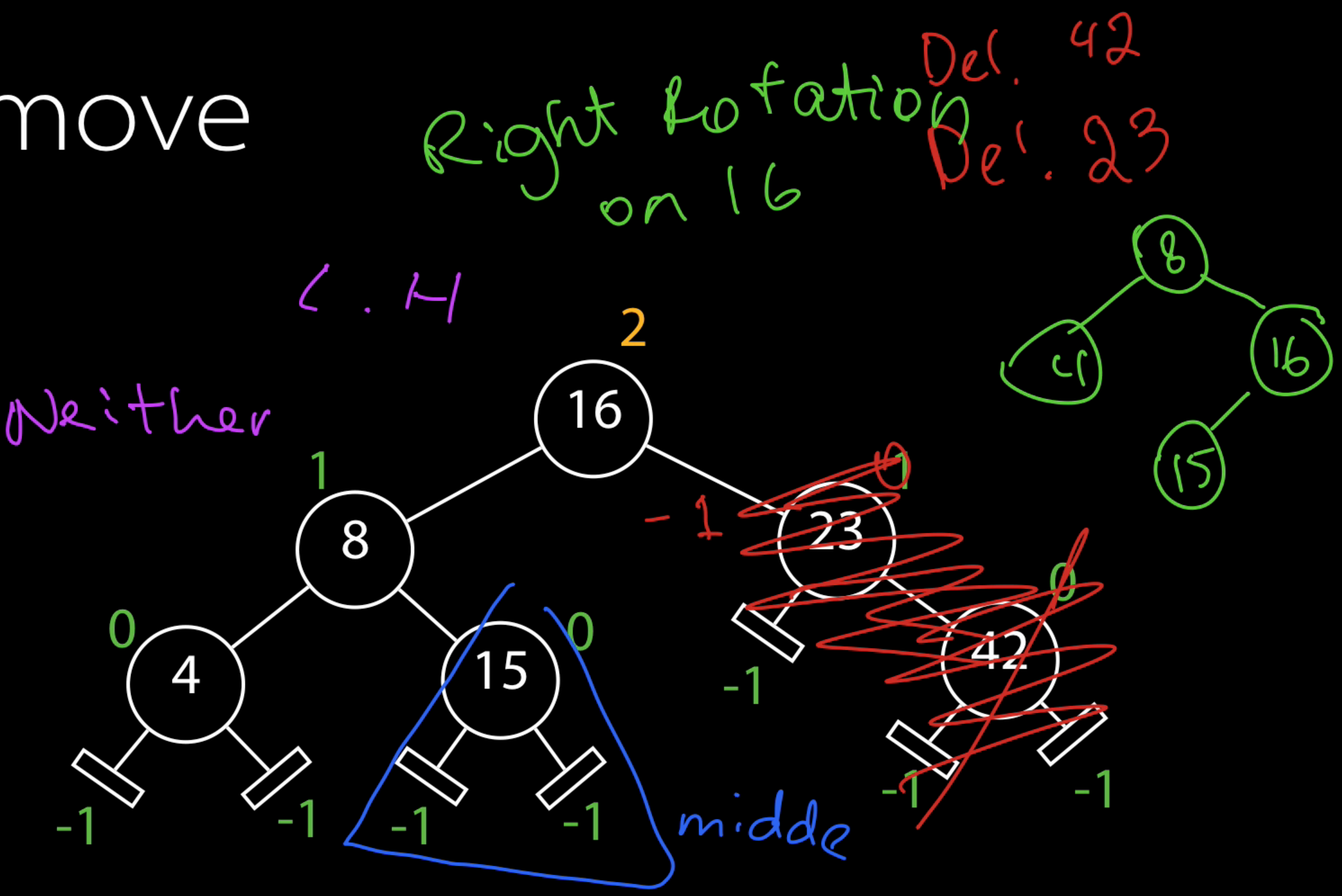
Deleting a leaf node is easy

Deleting a node with one child is easy

To delete a node with two children, replace with in-order predecessor or with in-order successor

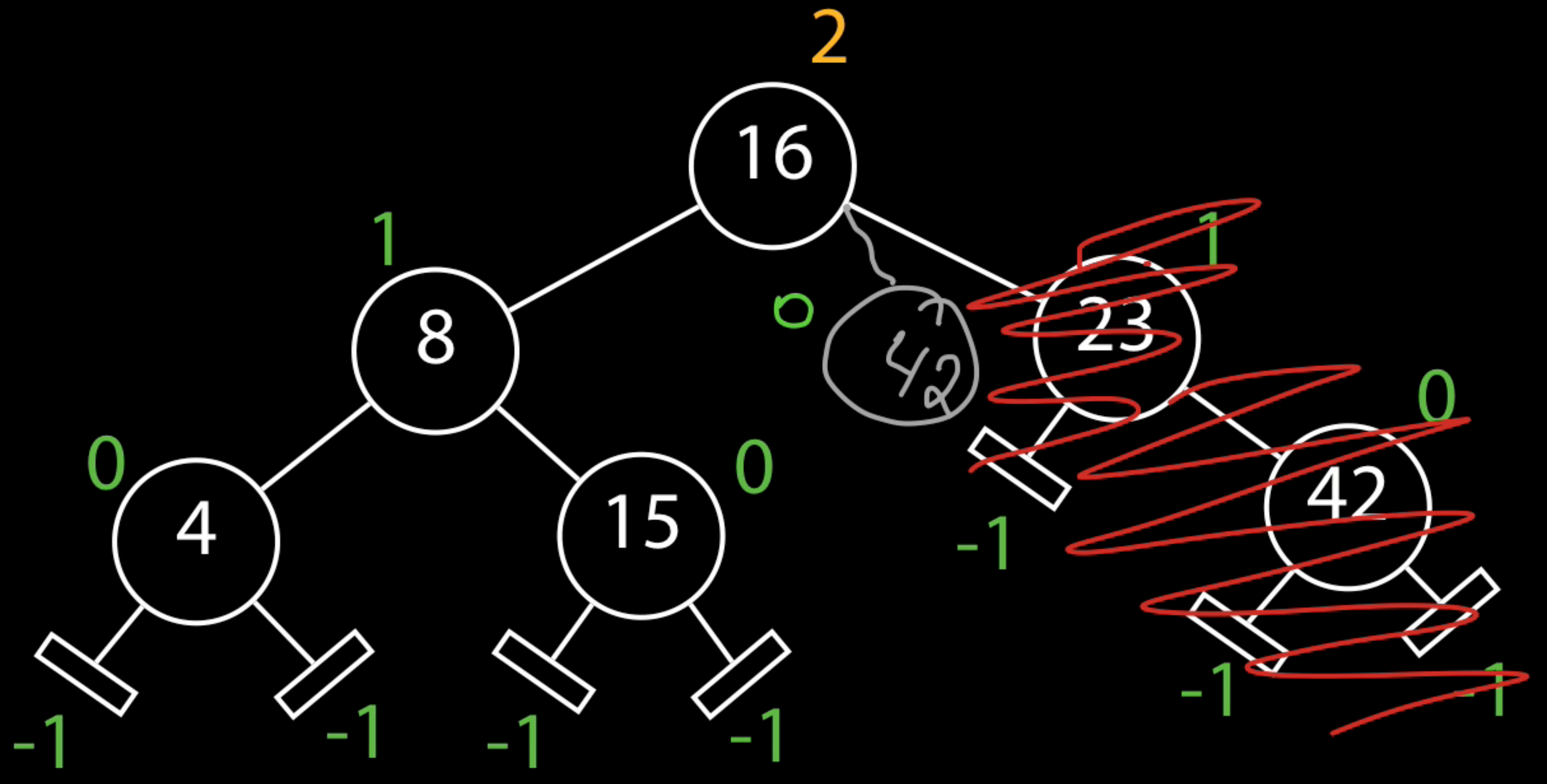
Might need to rebalance after deletion

Remove

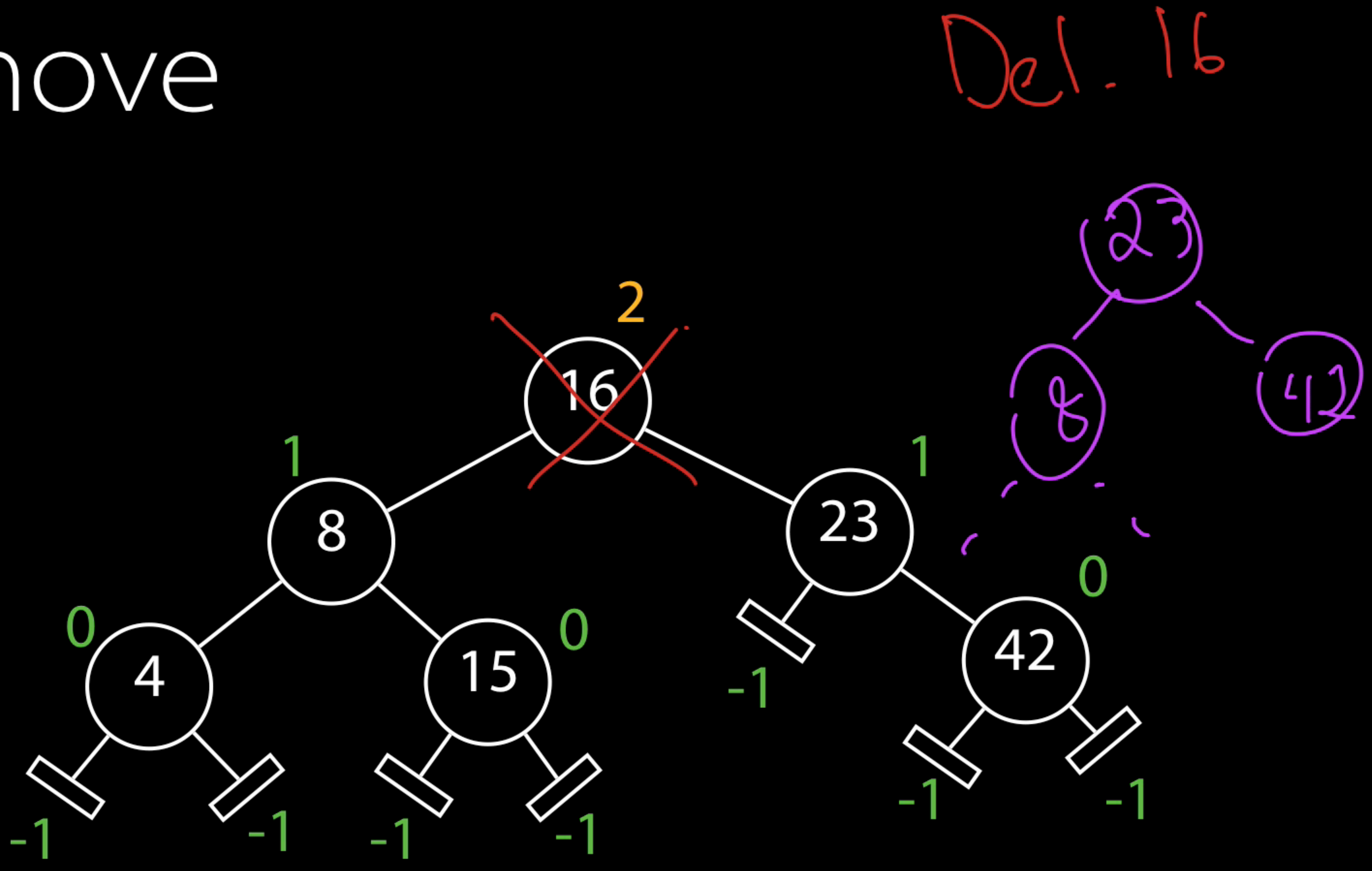

Right Rotation on 16
Del. 42
Del. 23
L. H
Neither
2
16
1
8
0
4
15
23
42
-1
middle

Remove

Del. 23



Remove



AVL tree performance

Searching: $O(\log N)$

Insertion: $O(\log N)$

Deletion: $O(\log N)$

The path from the root to the deepest leaf in an AVL tree is at most $\sim 1.44 \log(N + 2)$

AVL Trees

Given a node in an AVL binary search tree, what is the worst-case complexity of finding the node with the next smaller key?

AVL Trees

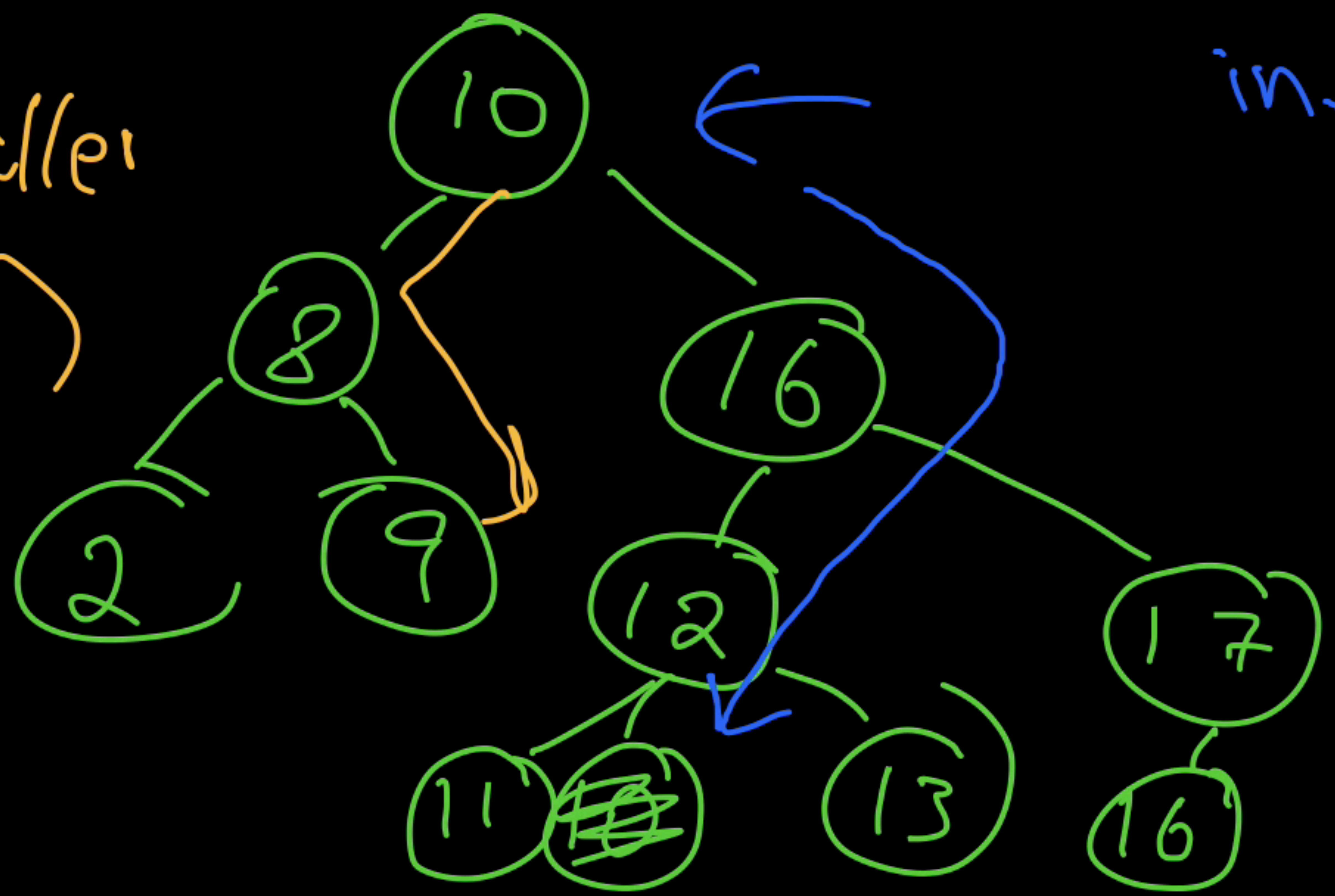
Given a node in an AVL binary search tree, what is the worst-case complexity of finding the node with the next larger key?

AVL Trees

Given a node in an AVL binary search tree, what is the worst-case complexity of finding the node with the next ~~smaller~~ larger key?

next smaller

O(log n)



in-order successor

O(log N)

AVL Trees

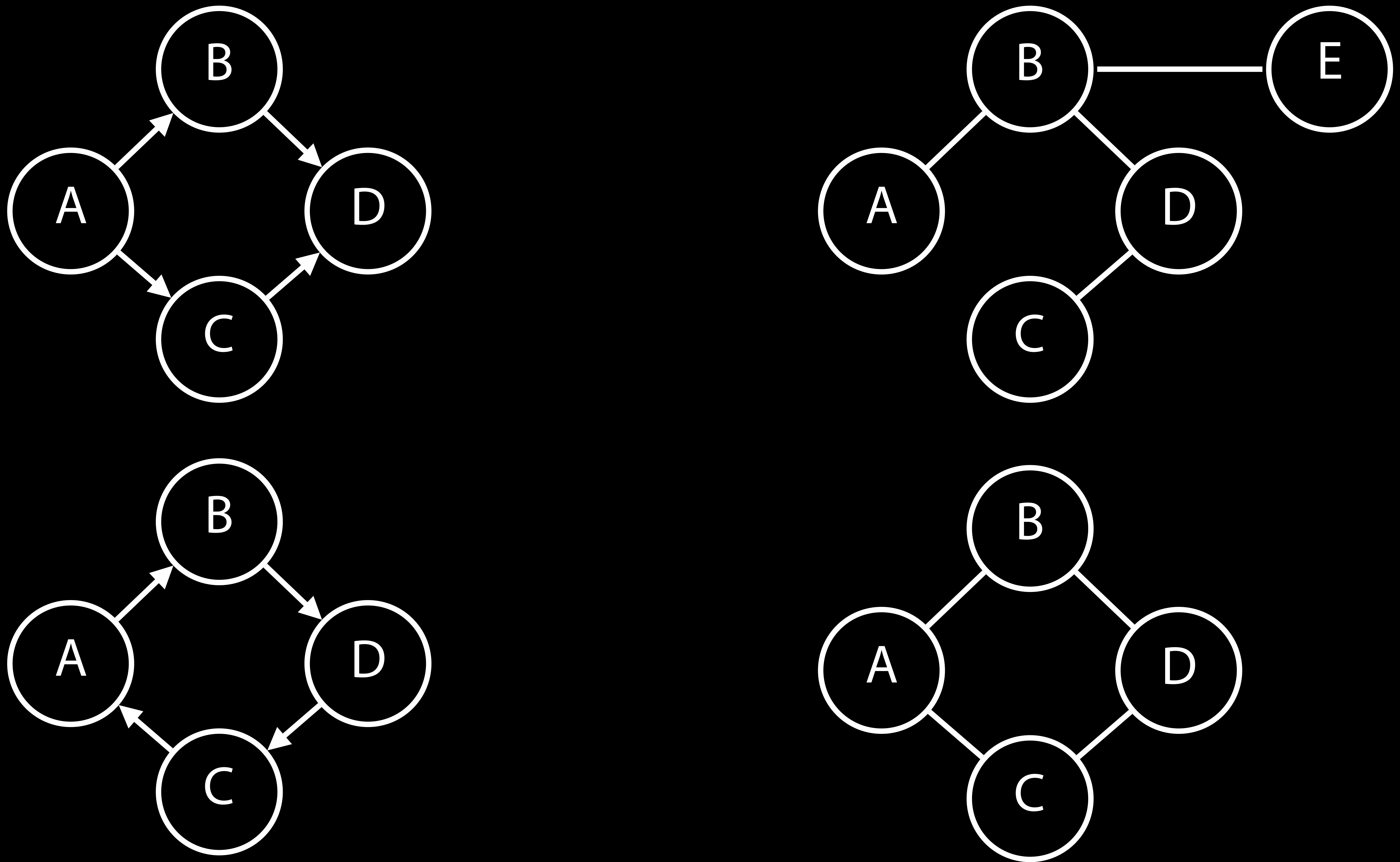
True / false: Deletion of a key from an AVL tree, immediately followed by insertion of the same key always results in the initial AVL tree.

AVL Trees

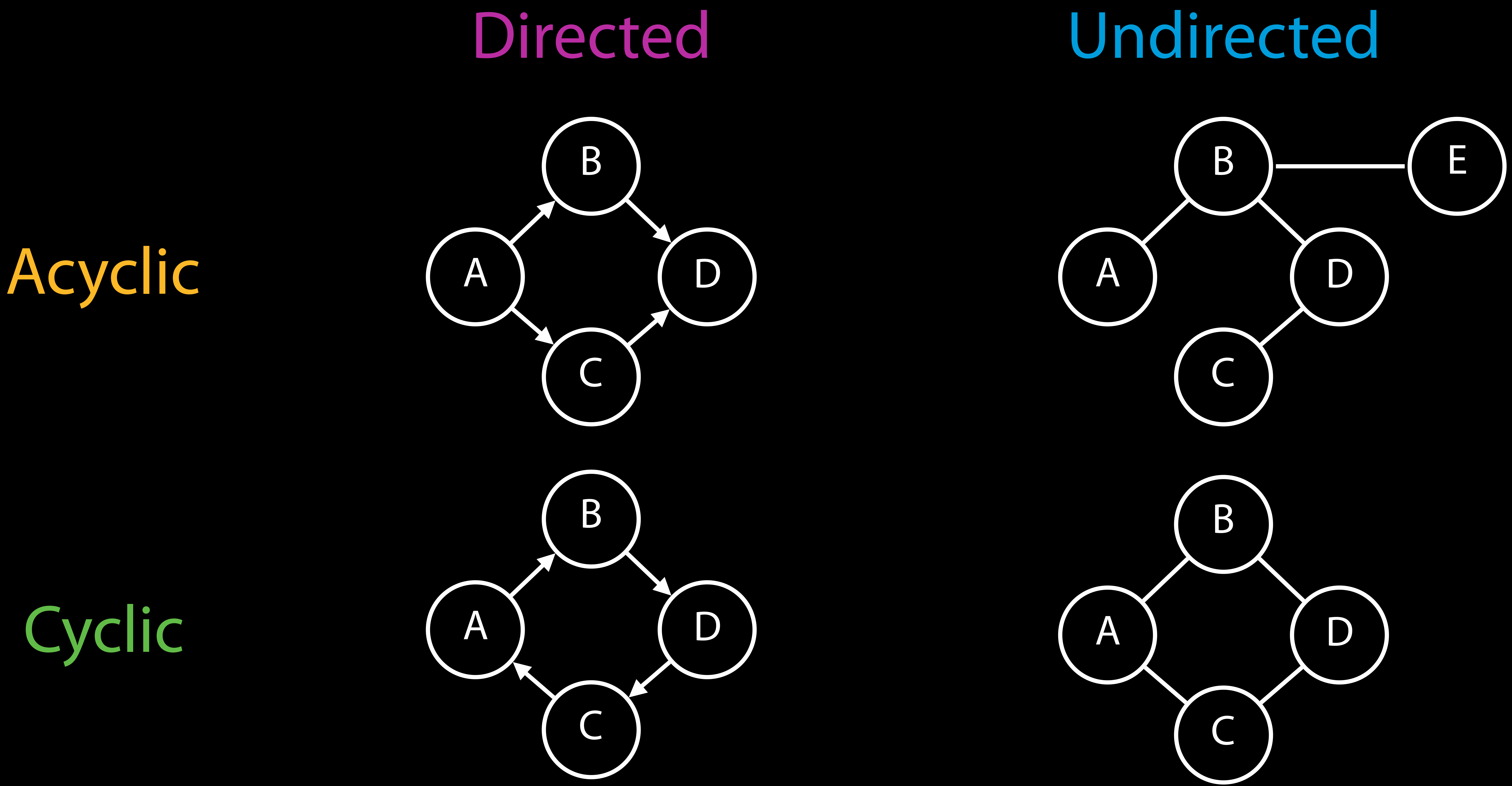
True / false: Deletion of a key from an AVL tree, immediately followed by insertion of the same key always results in the initial AVL tree.

Graphs

Graph Types



Graph Types



Graph terminology

Graph: $G = (V, E)$

Set of **vertices**, a.k.a. **nodes**.

Set of **edges**: Pairs of vertices.

Vertices with an edge between are **adjacent**.

Vertices or edges may have **labels** (or **weights**).

Graph terminology

A path is a sequence of vertices connected by edges.

A cycle is a path whose first and last vertices are the same.

A graph with a cycle is cyclic. A graph without a cycle is acyclic.

Two vertices are connected if there is a path between them. If all vertices are connected, we say the graph is connected.

Graph terminology

A **path** is a sequence of vertices connected by edges.

A **cycle** is a path whose first and last vertices are the same.

A graph with a cycle is **cyclic**. A graph without a cycle is **acyclic**.

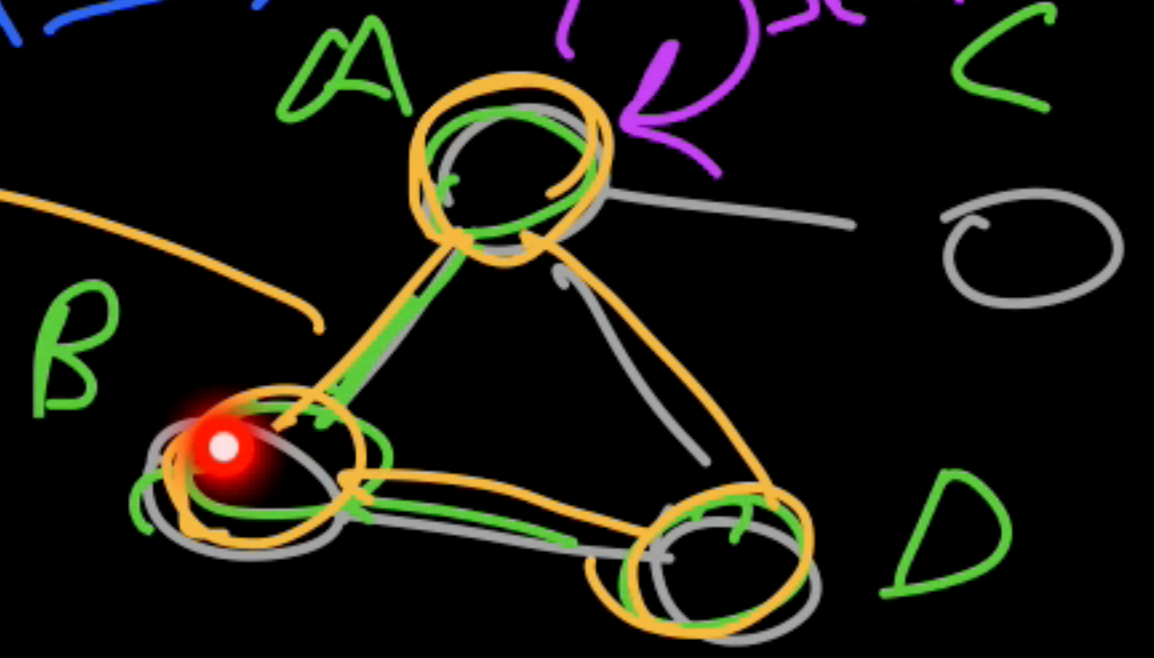
Two vertices are connected if there is a path between them. If all vertices are connected, we say the graph is **connected**.

undirected

self-edge

directed →
strongly connected

cycle

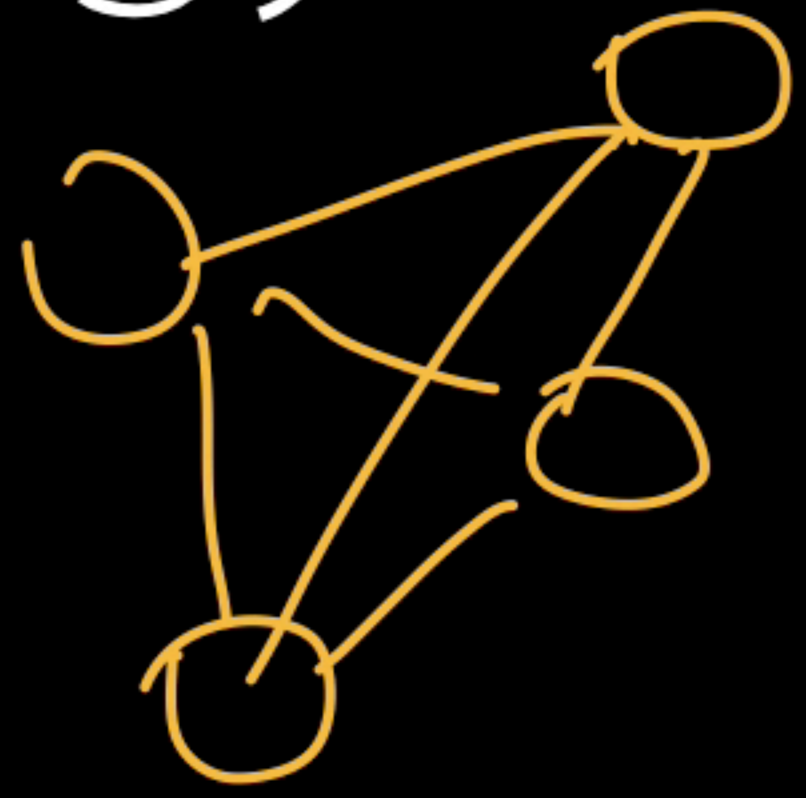


(A, B)
(B, D)

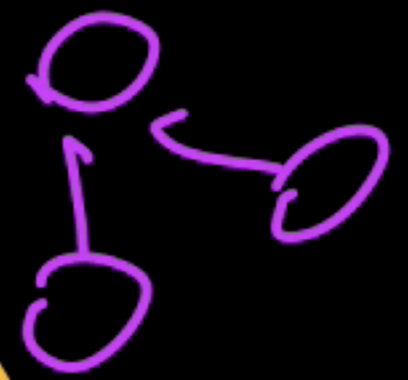
Graph terminology

DAG

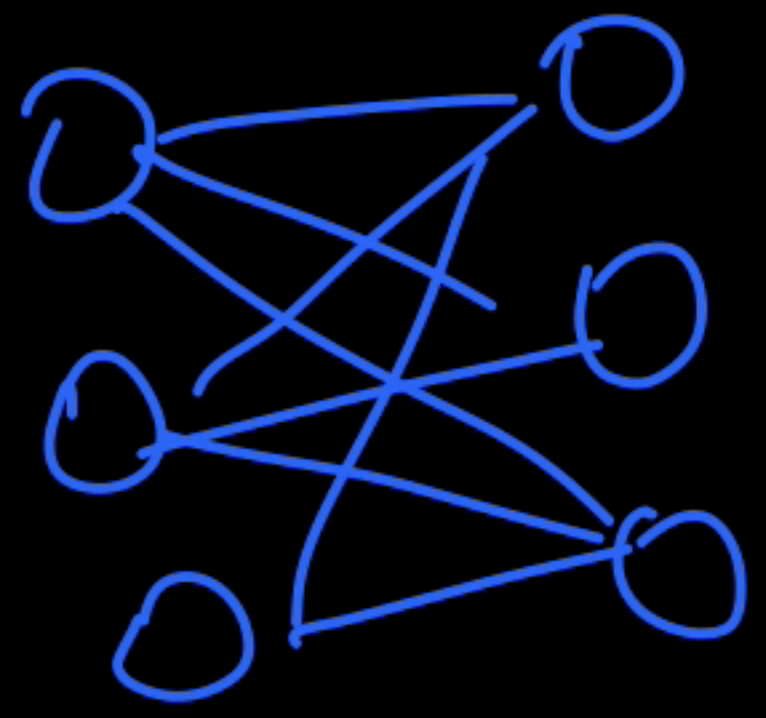
Complete graph



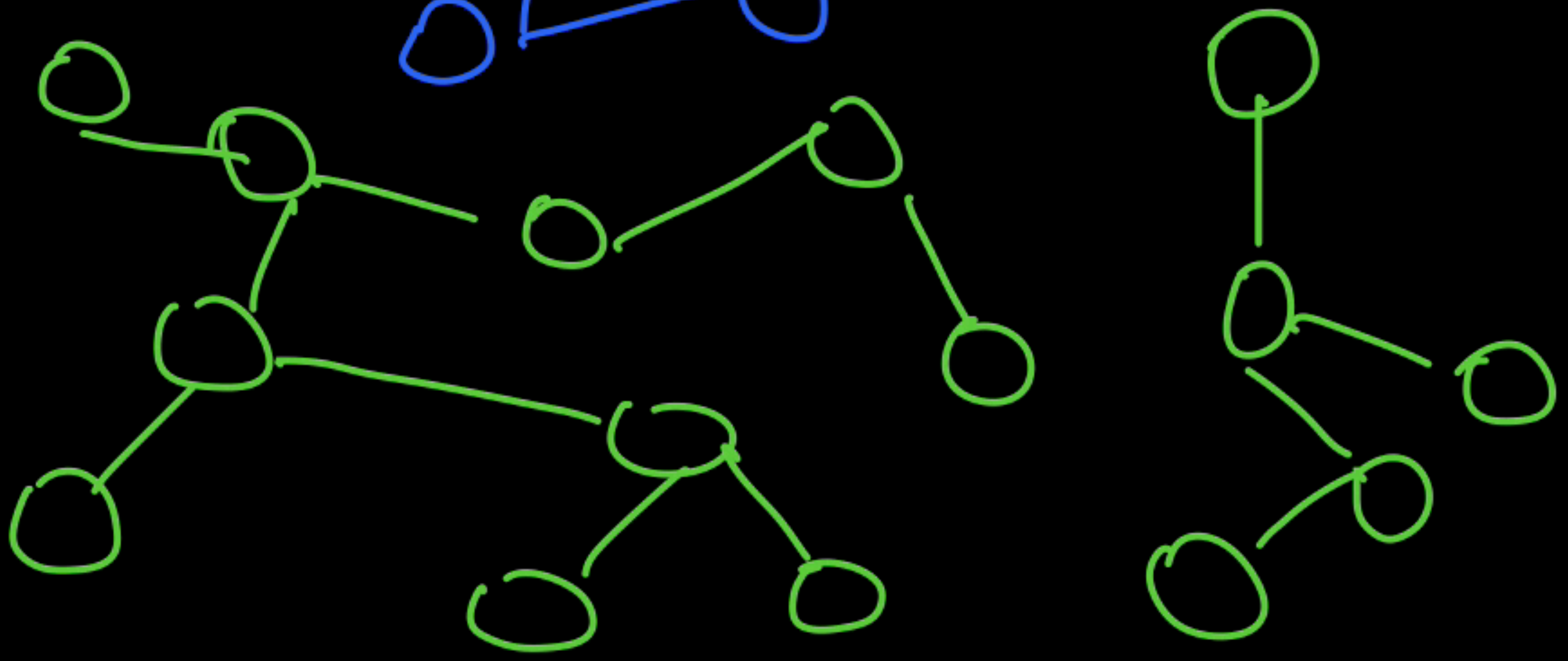
Rooted tree



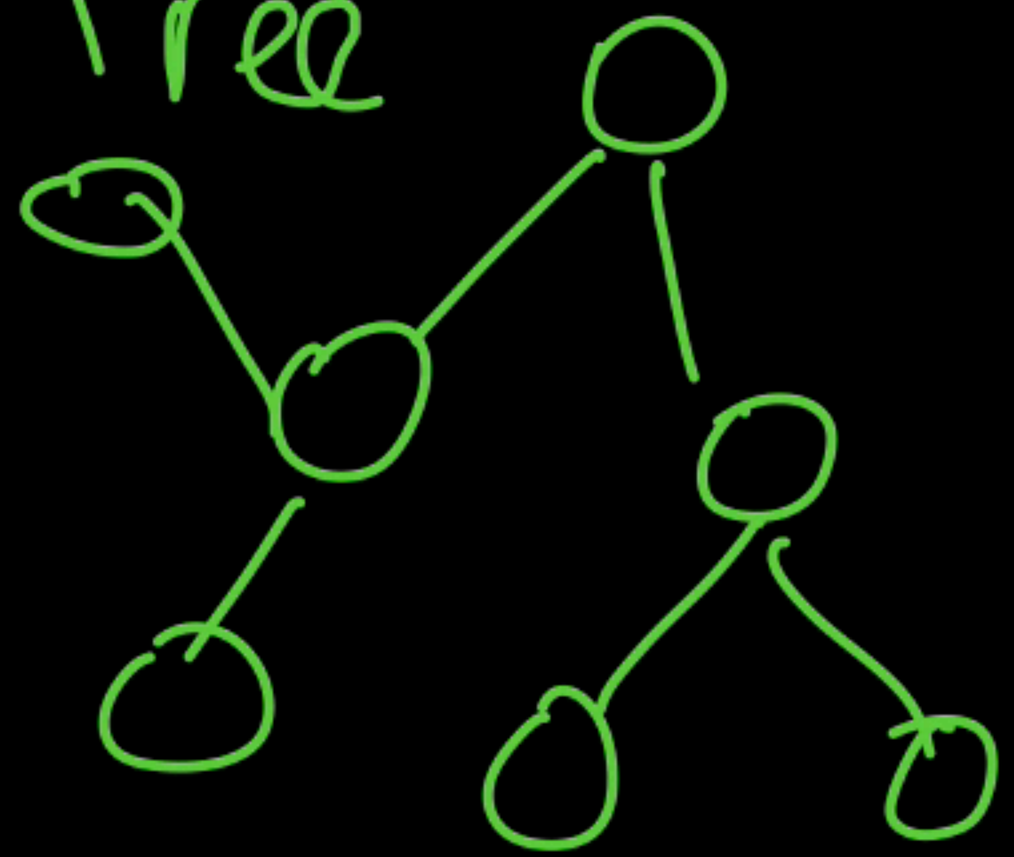
Bipartite



Forest



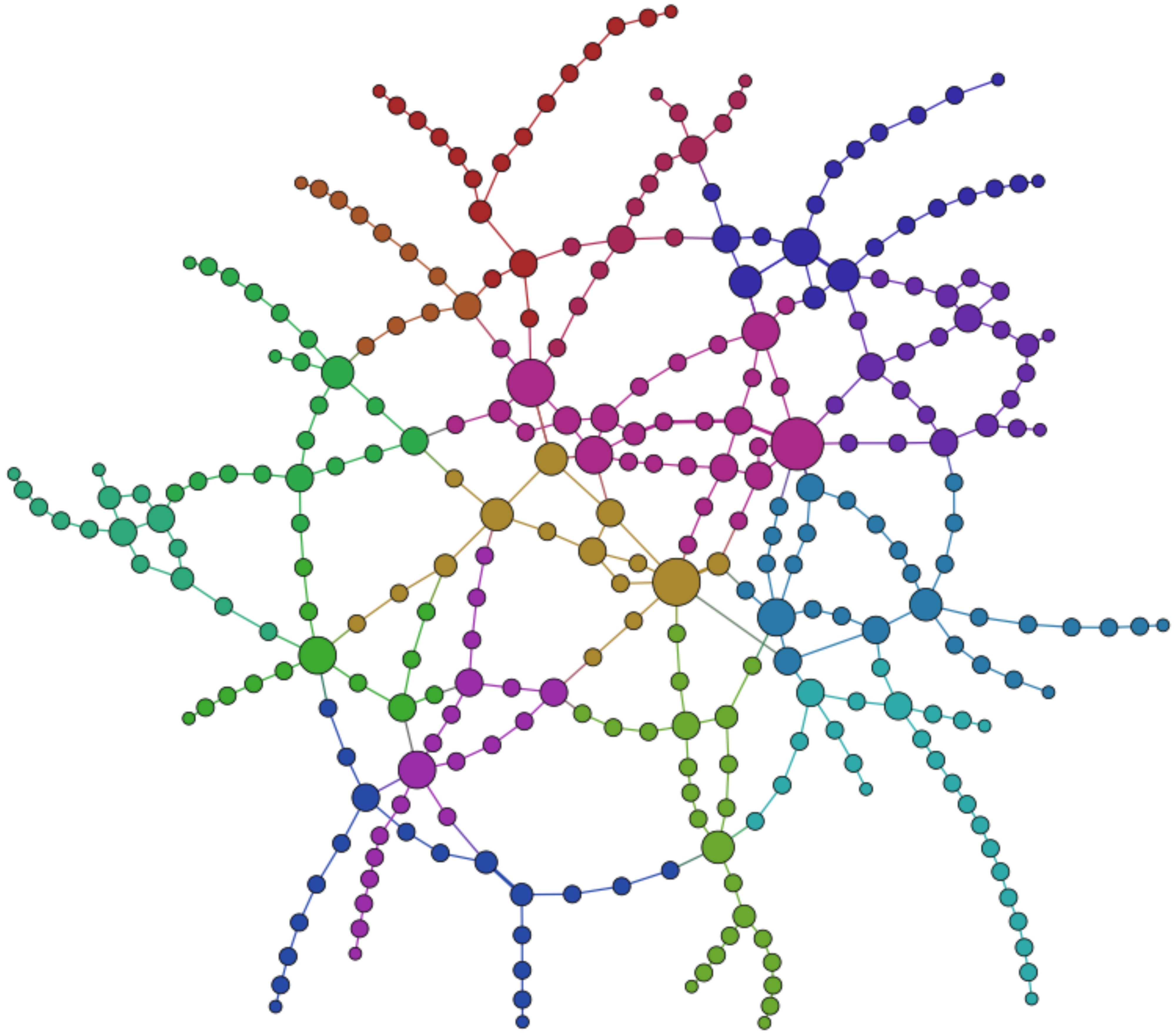
(free) Tree



Graph Example

Introduction to Network Visualization with GEPHI – Martin Grandjean

Examples




facebook
December 2010

Graph representations

$G = (V, E)$

$|V|$ # of vertices

$|E|$ # of edges

sparse: $|E| << |V|^2$

dense: $|E| \approx |V|^2$

Complete graph

$|E| = \Theta(|V|^2)$

Planar graph

(edges cannot cross)

$|V| \geq \Theta(|E|)$

Graph representations

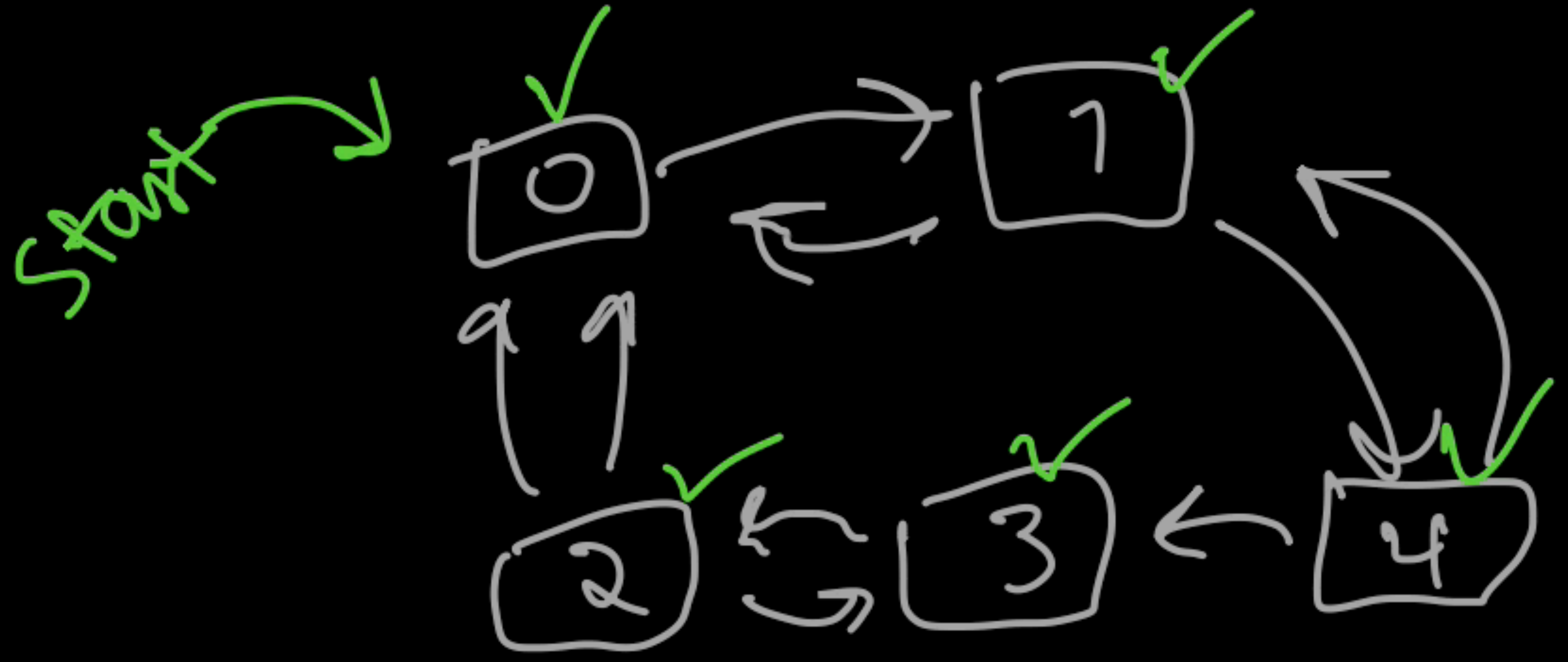
Graph traversal

Pre-order depth-first traversal

Post-order depth-first traversal

Breadth-first traversal

Graph traversal



Stack

1. Pre-order depth-first traversal

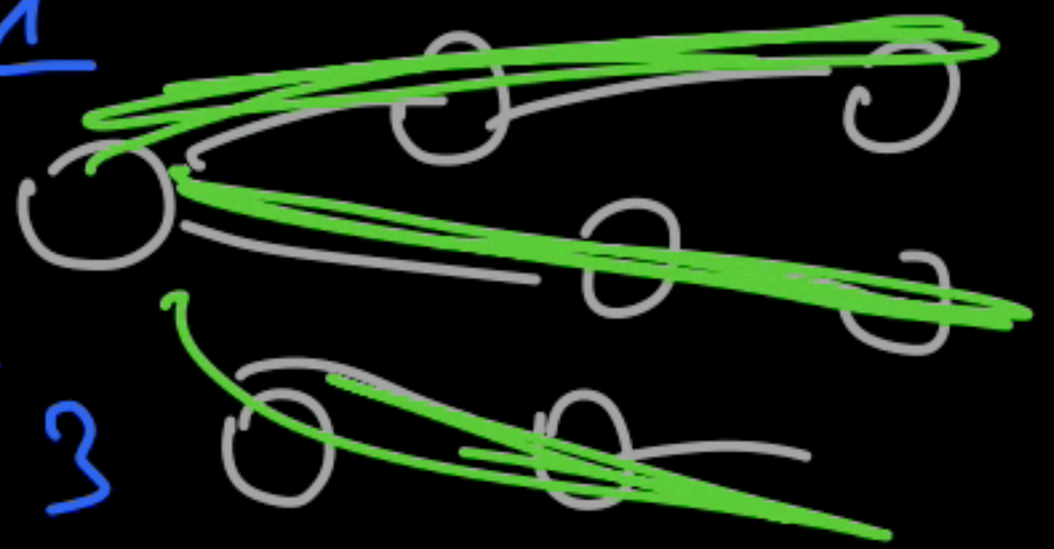
Post-order depth-first traversal

Breadth-first traversal Queue

Level-order

0 1 2 4 3

visit everything at level 1
level 2
level 3



visit 0
visit 1
visit 4
visit 3
visit 2

dfs(0)
dfs(1)
dfs(2)
dfs(4)
dfs(3)
dfs(2)

Silicon Valley

http://www.hbo.com/silicon-valley

Q&A