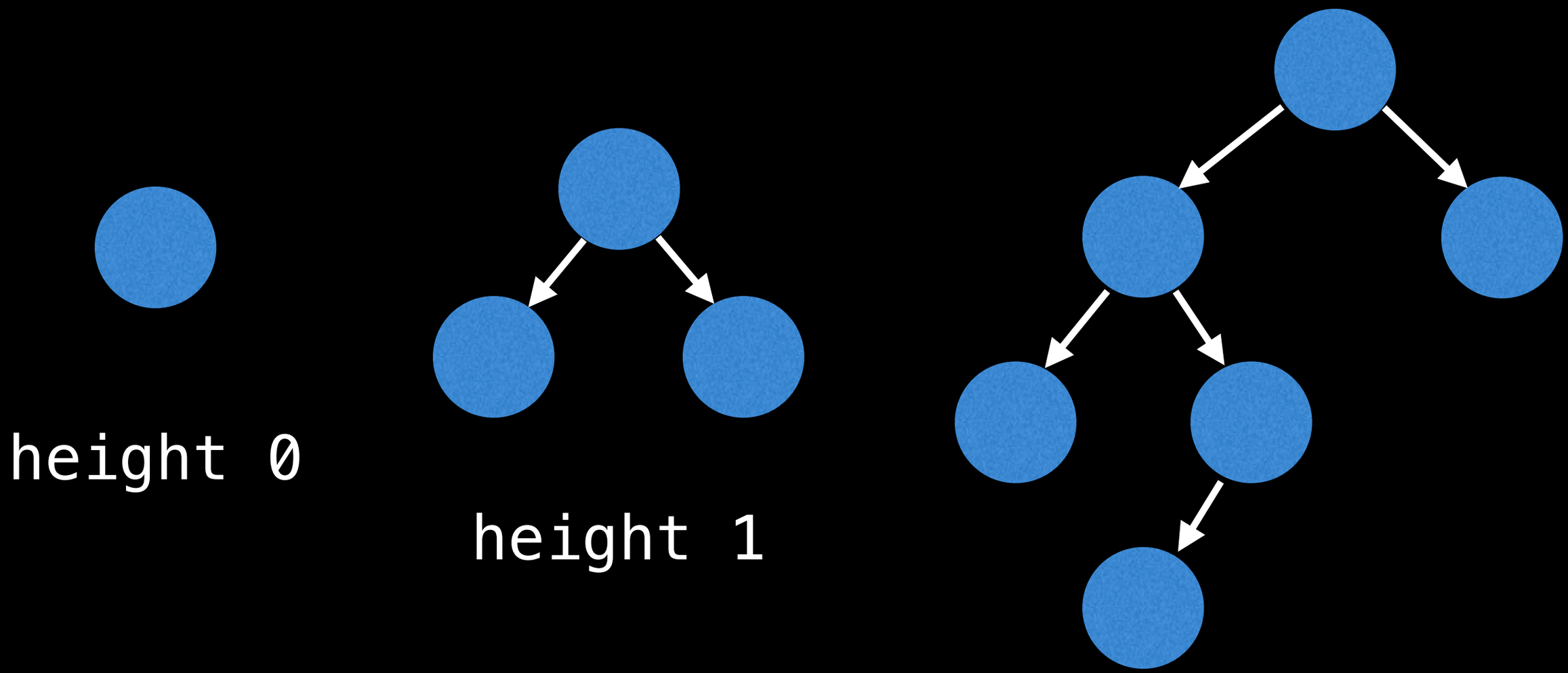
Beginner tree algorithms

William Fiset

Problem 2: Tree Height

Find the **height** of a **binary tree**. The **height** of a tree is the number of edges from the root to the lowest leaf.

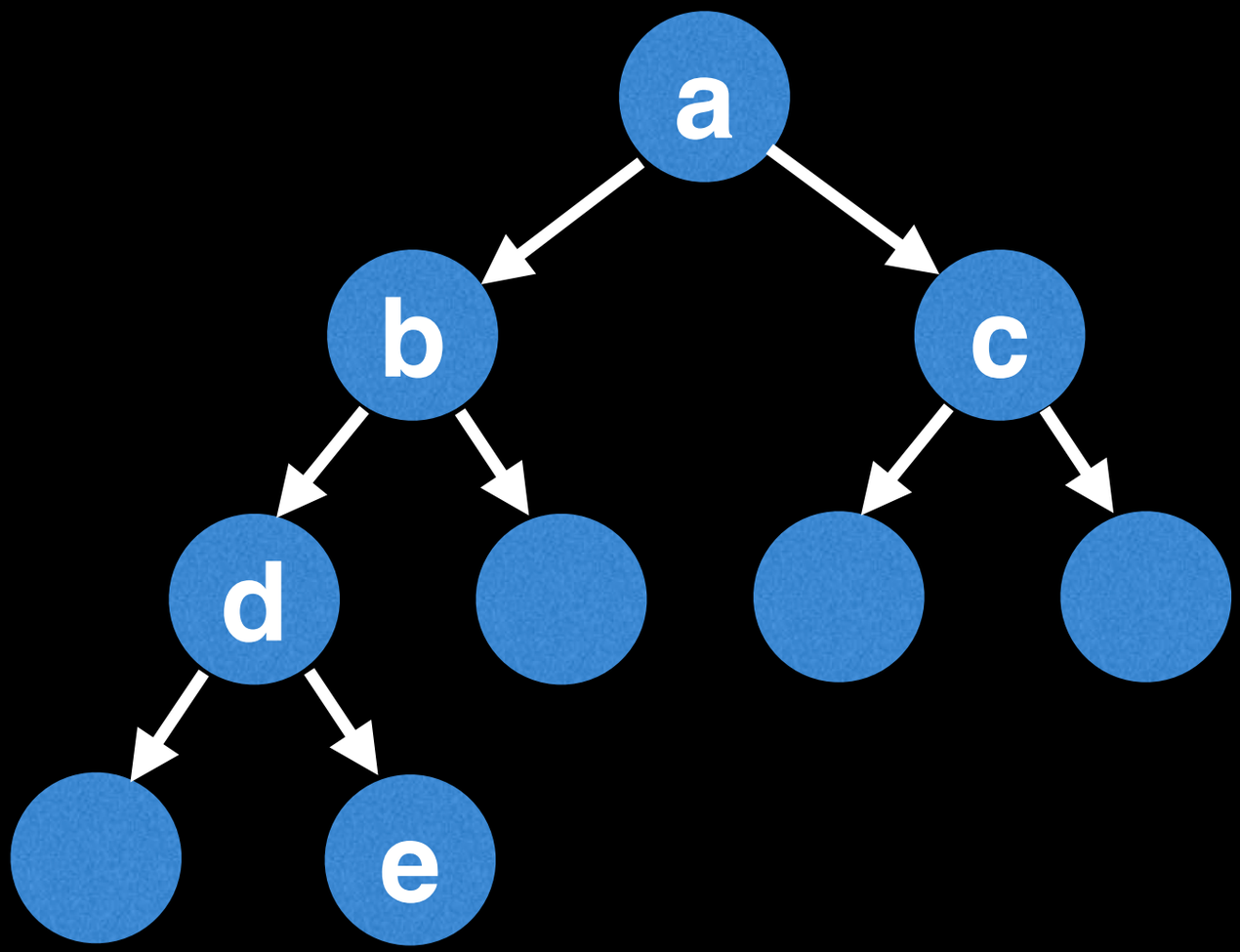


height 0

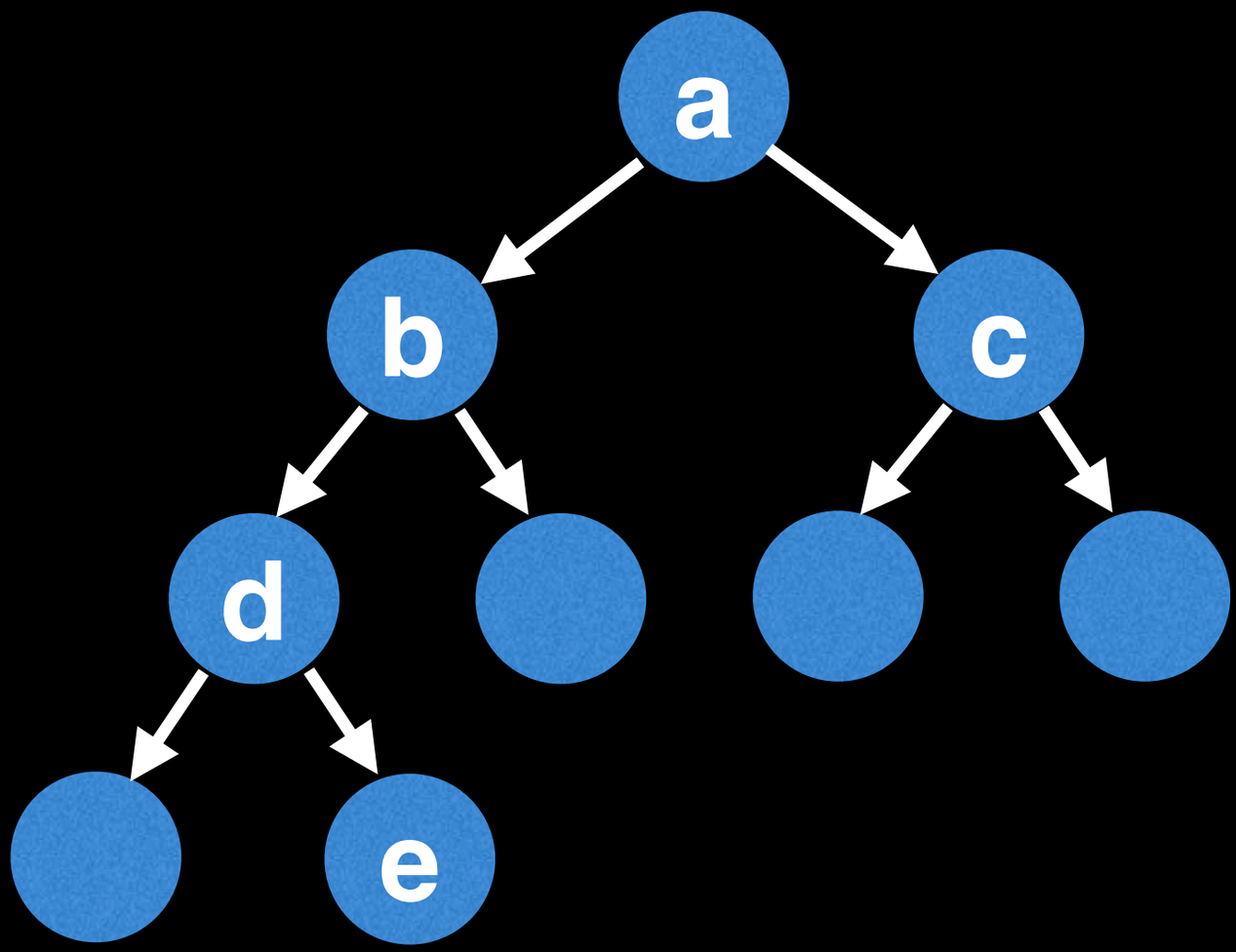
height 1

height 3

Let **h**(x) be the height of the subtree rooted at node x.

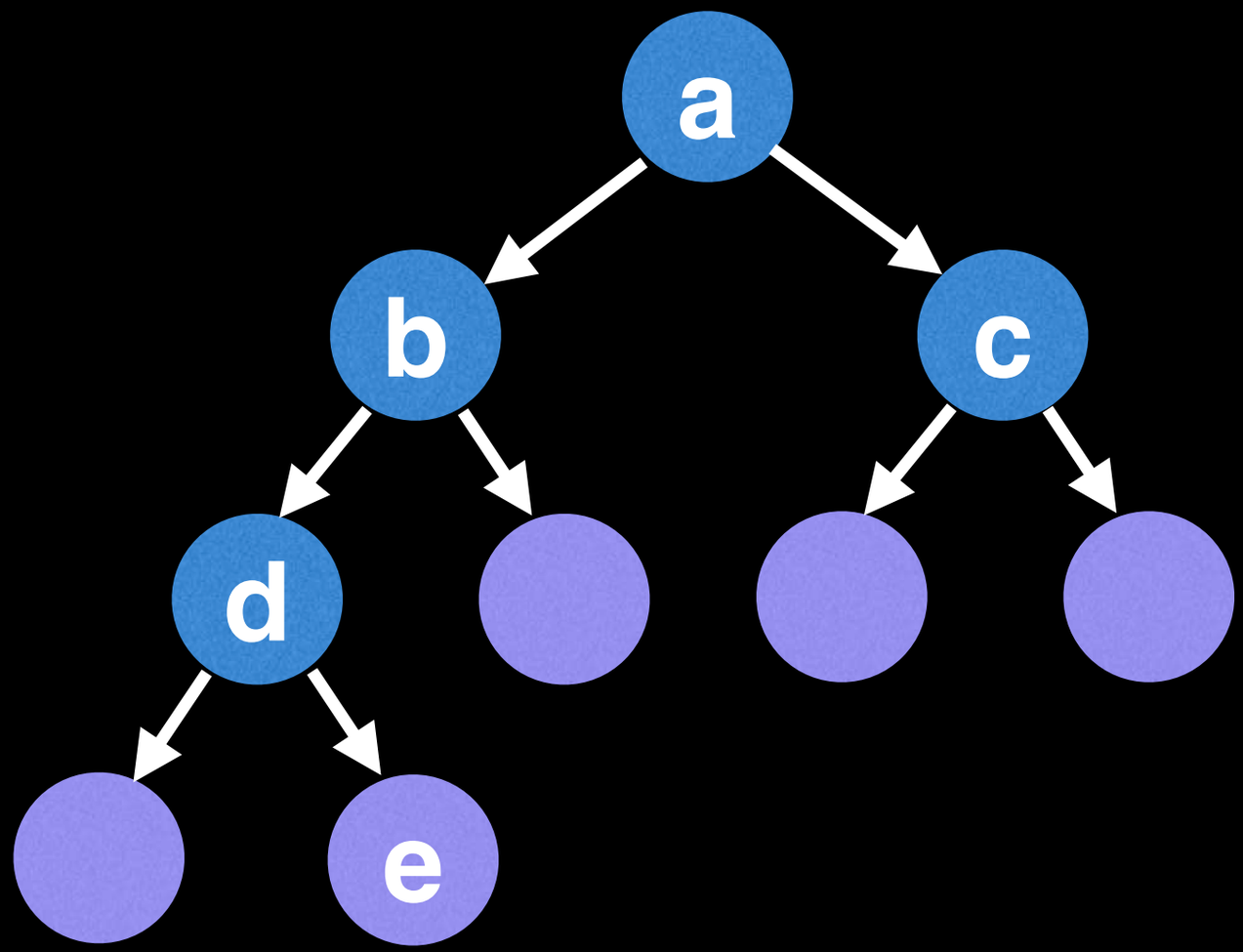


Let **h**(x) be the height of the subtree rooted at node x.



h(a) = 3, **h**(b) = 2, **h**(c) = 1, **h**(d) = 1, **h**(e) = 0

By themselves, leaf nodes such as node **e** don't have children, so they don’t add any additional height to the tree.

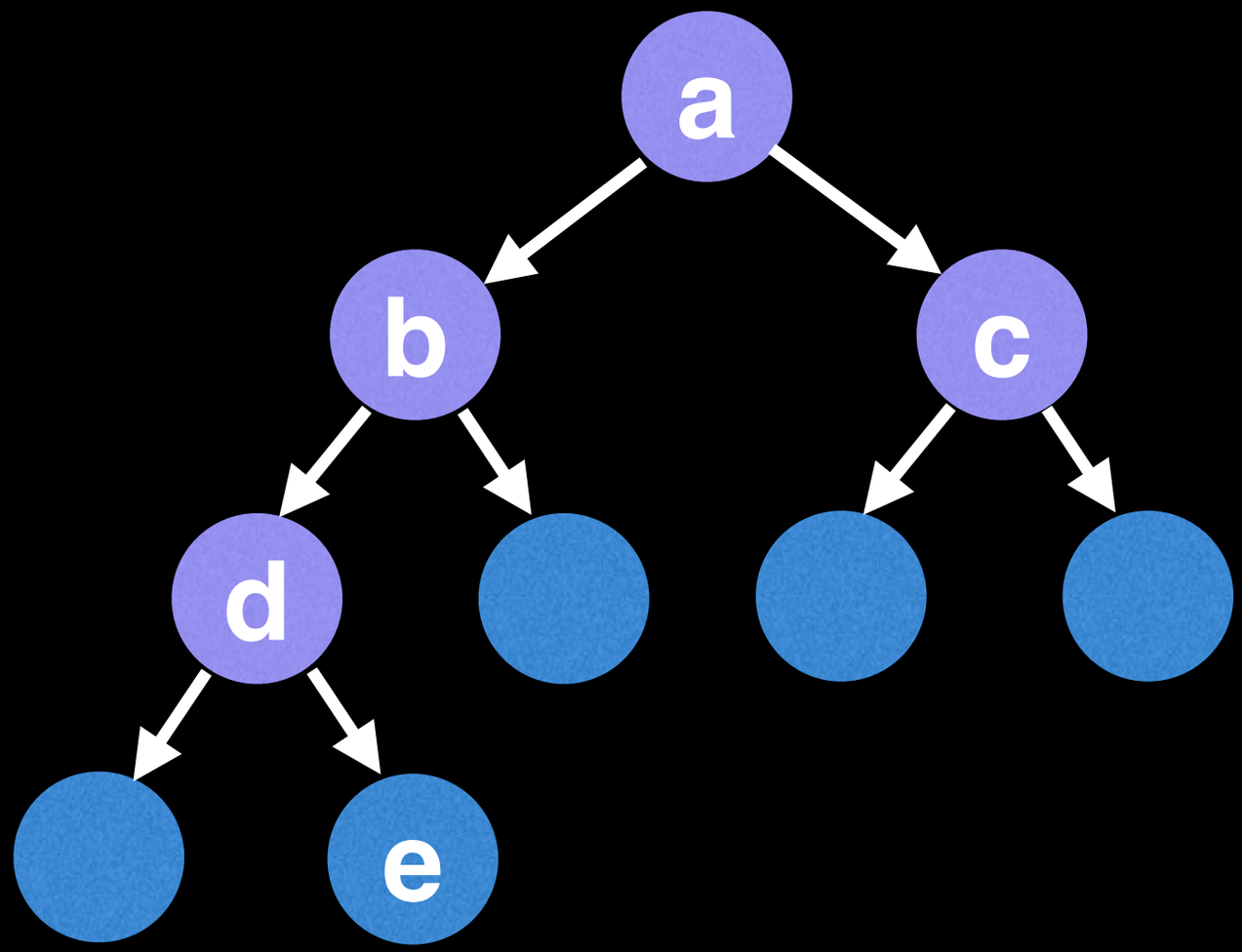


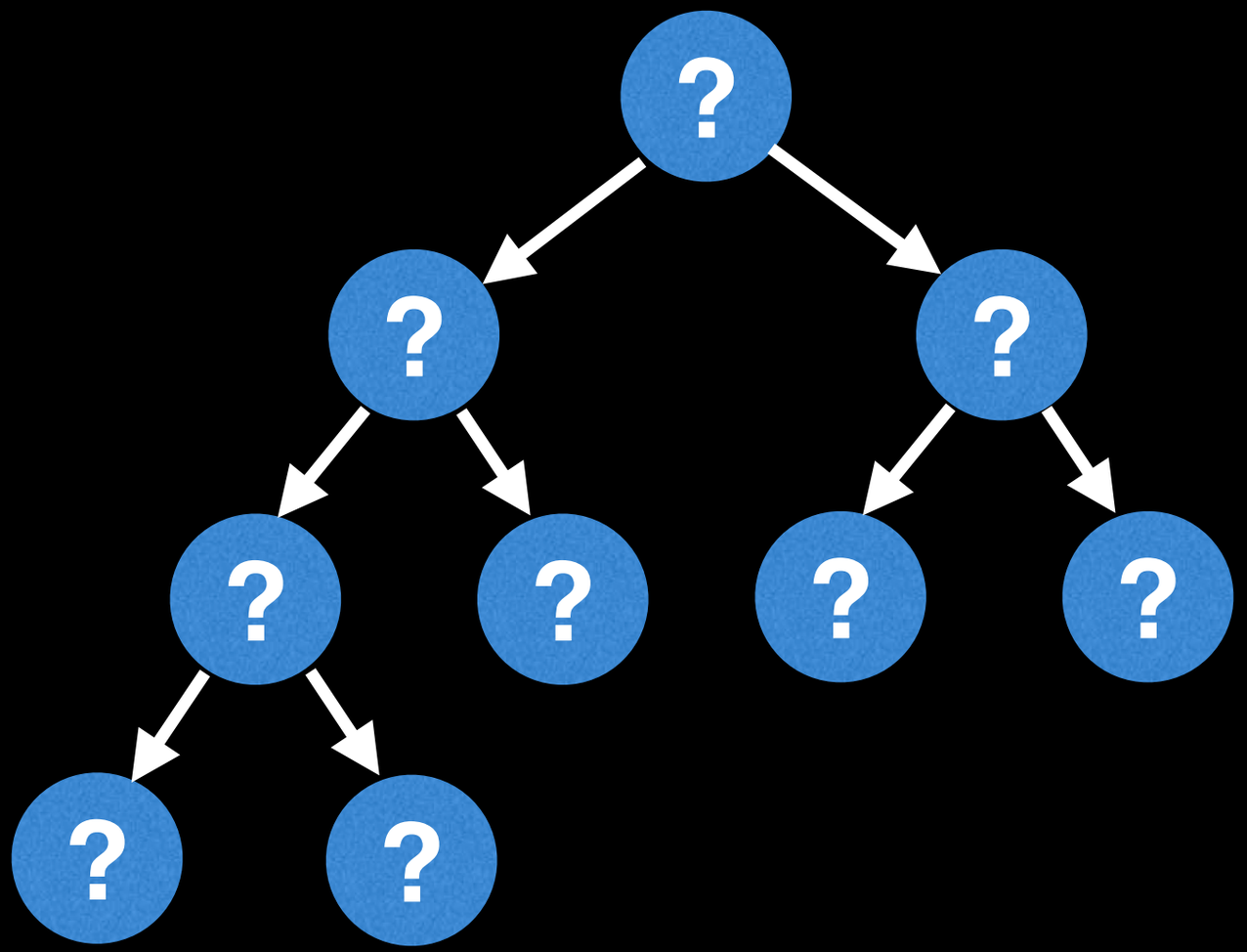
As a base case we can conclude that:

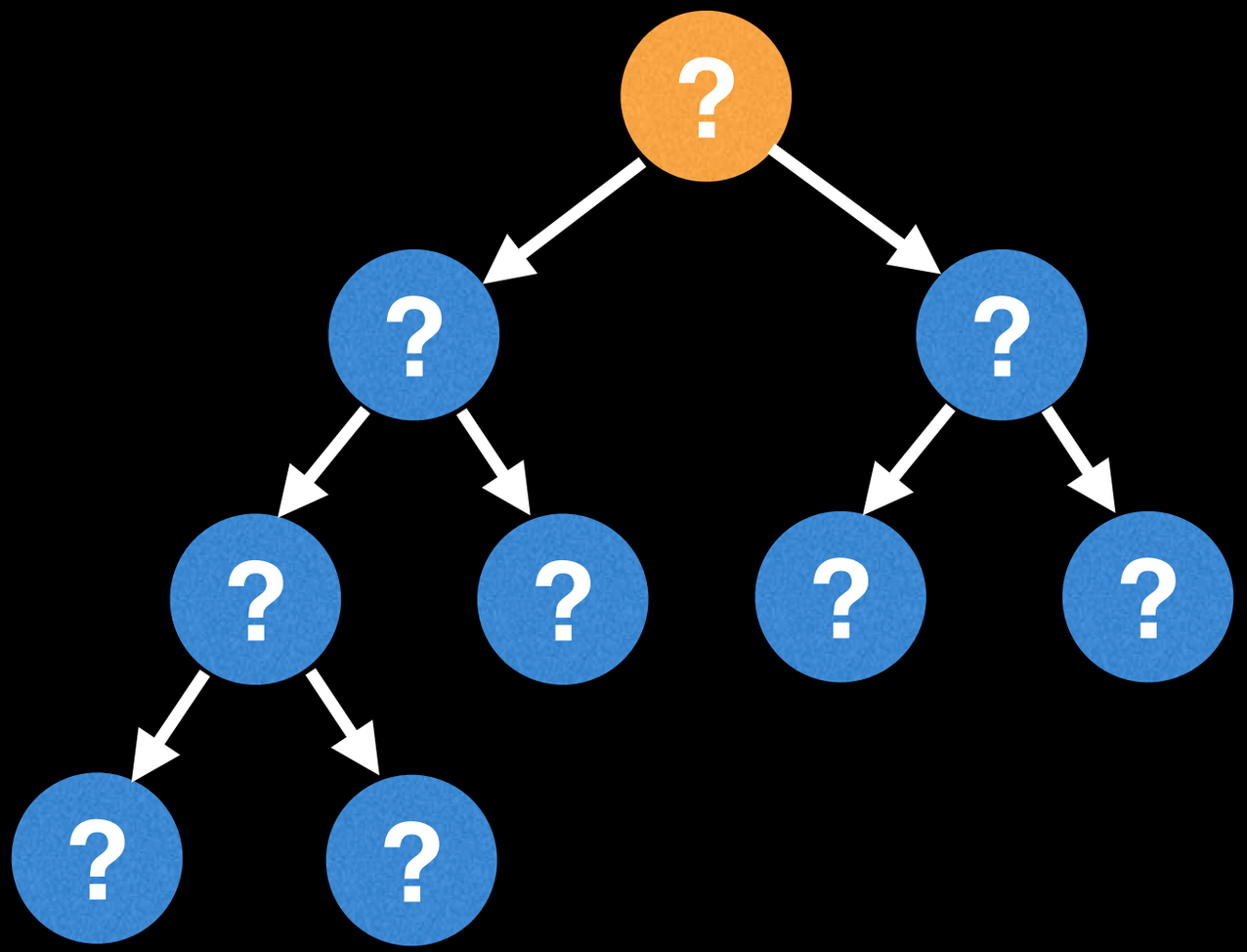
h(leaf node) = 0

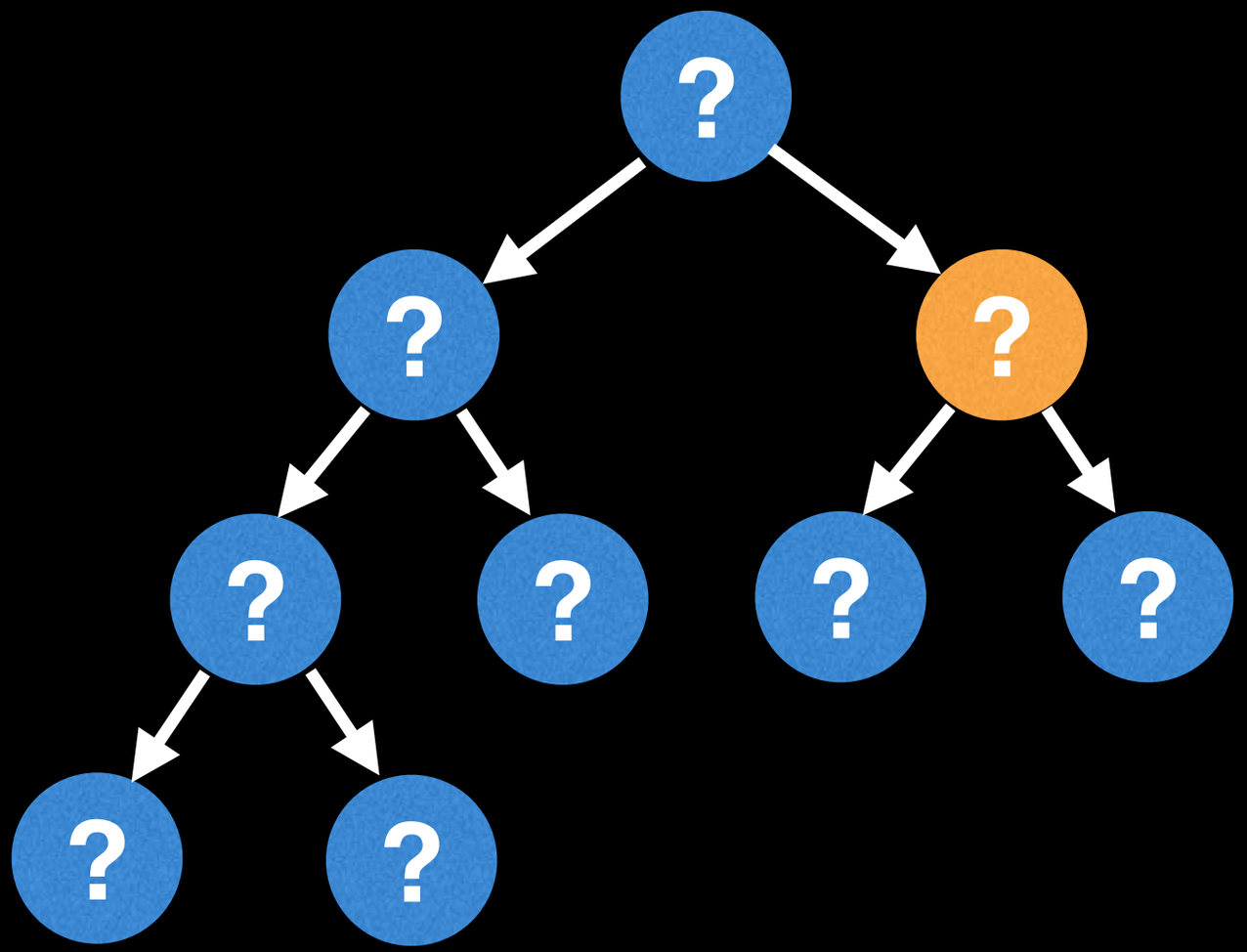
Assuming node x is not a leaf node, we're able to formulate a recurrence for the height:

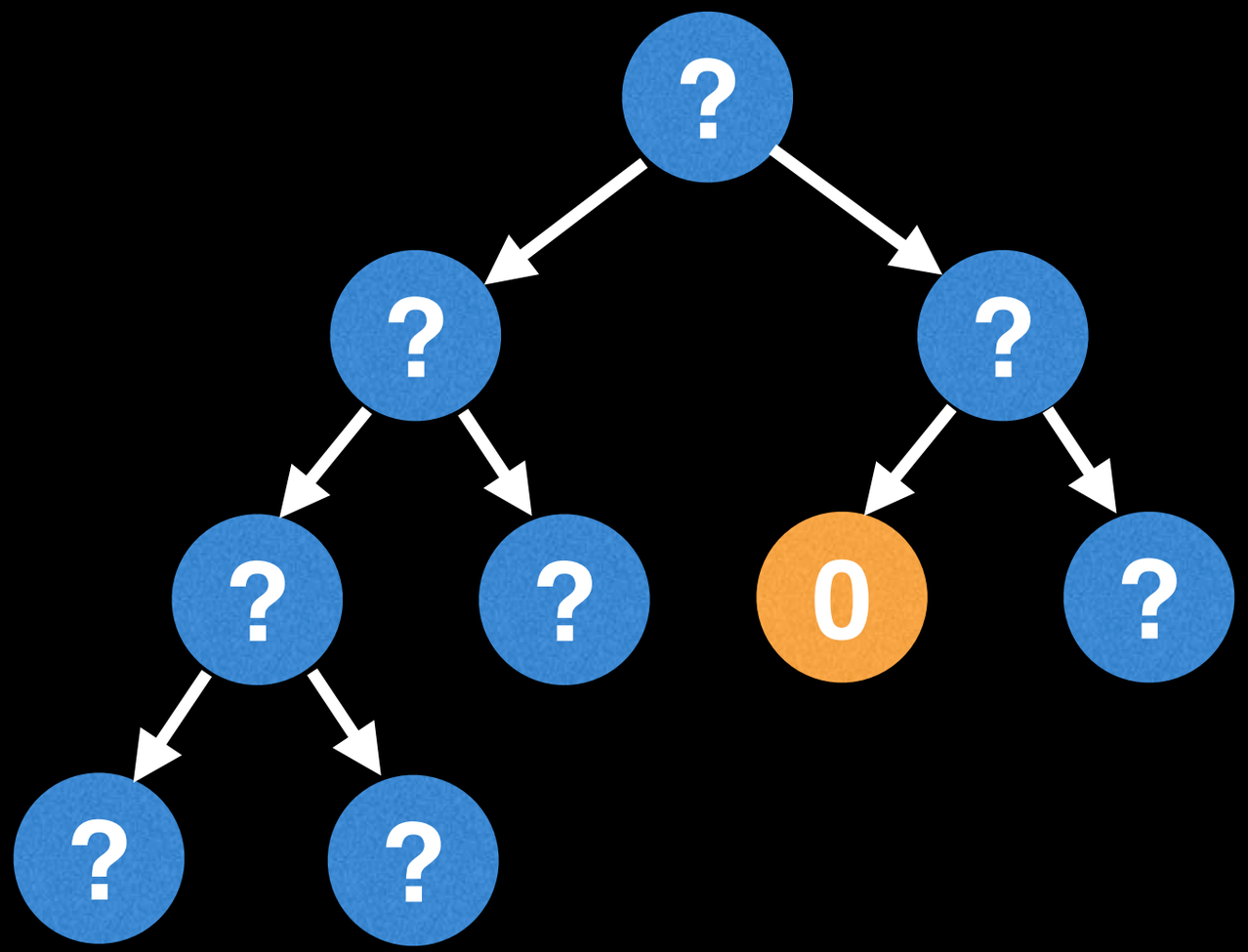
$$h(x) = \max(h(x.\text{left}), h(x.\text{right})) + 1$$



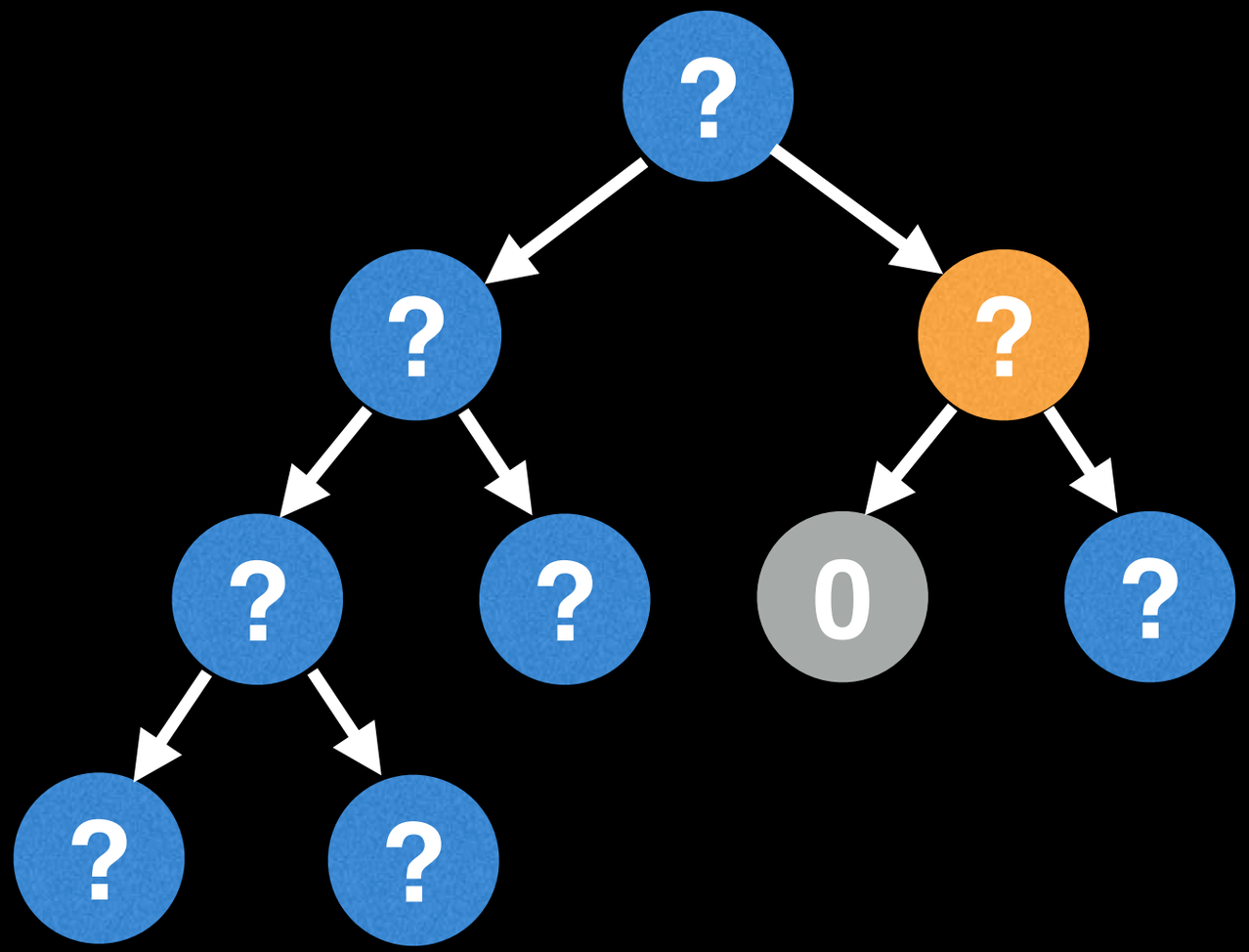


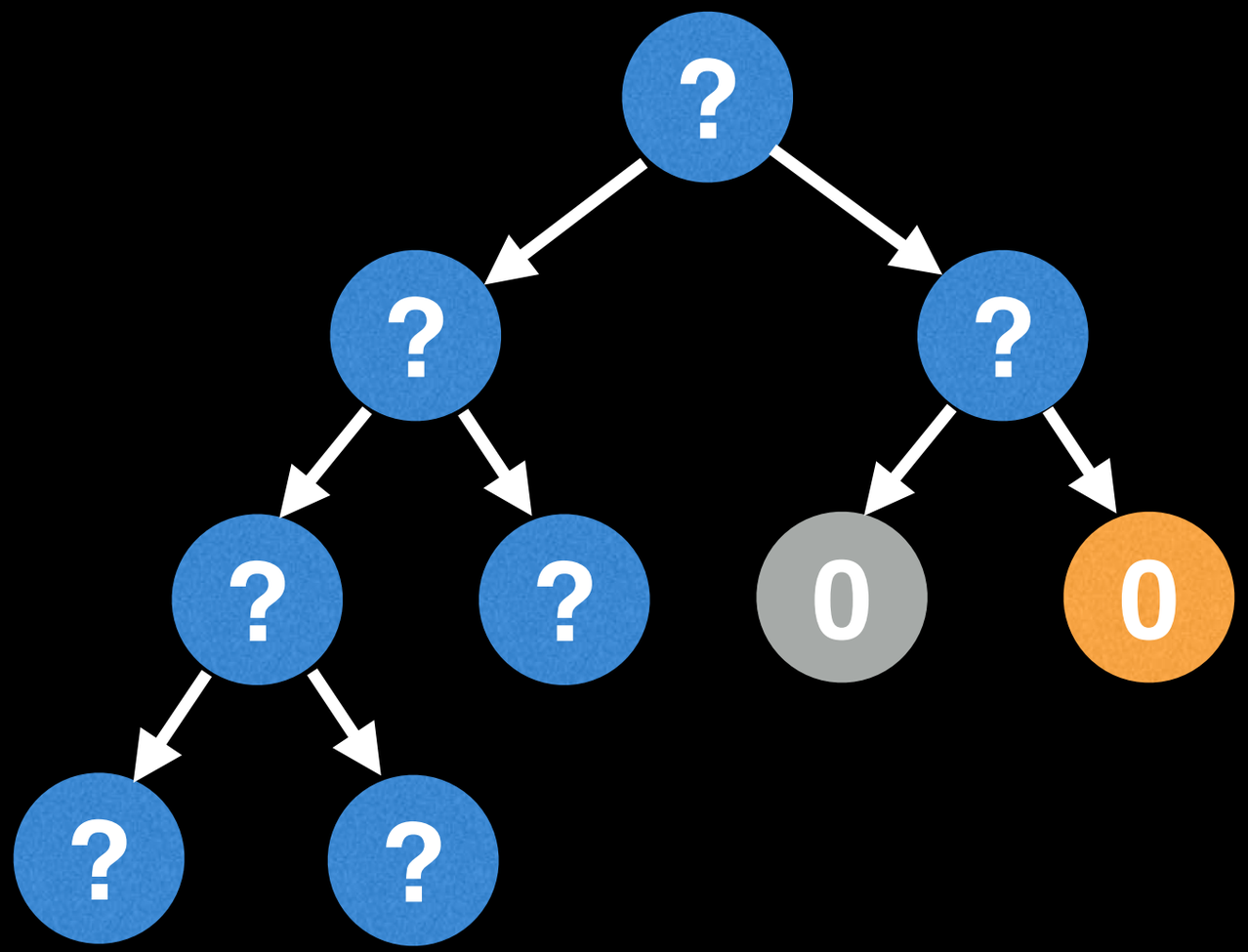




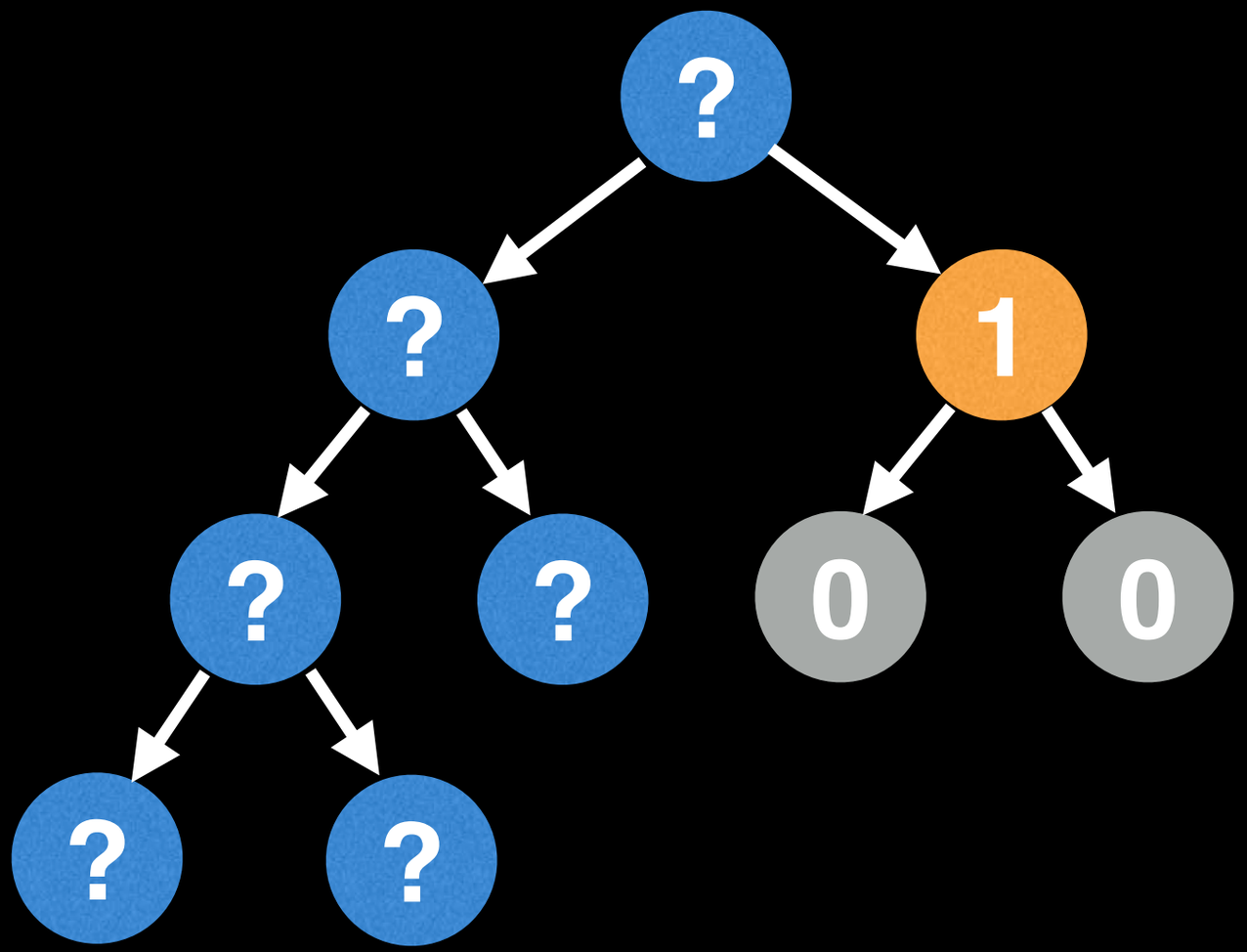


Leaf node has a height of 0

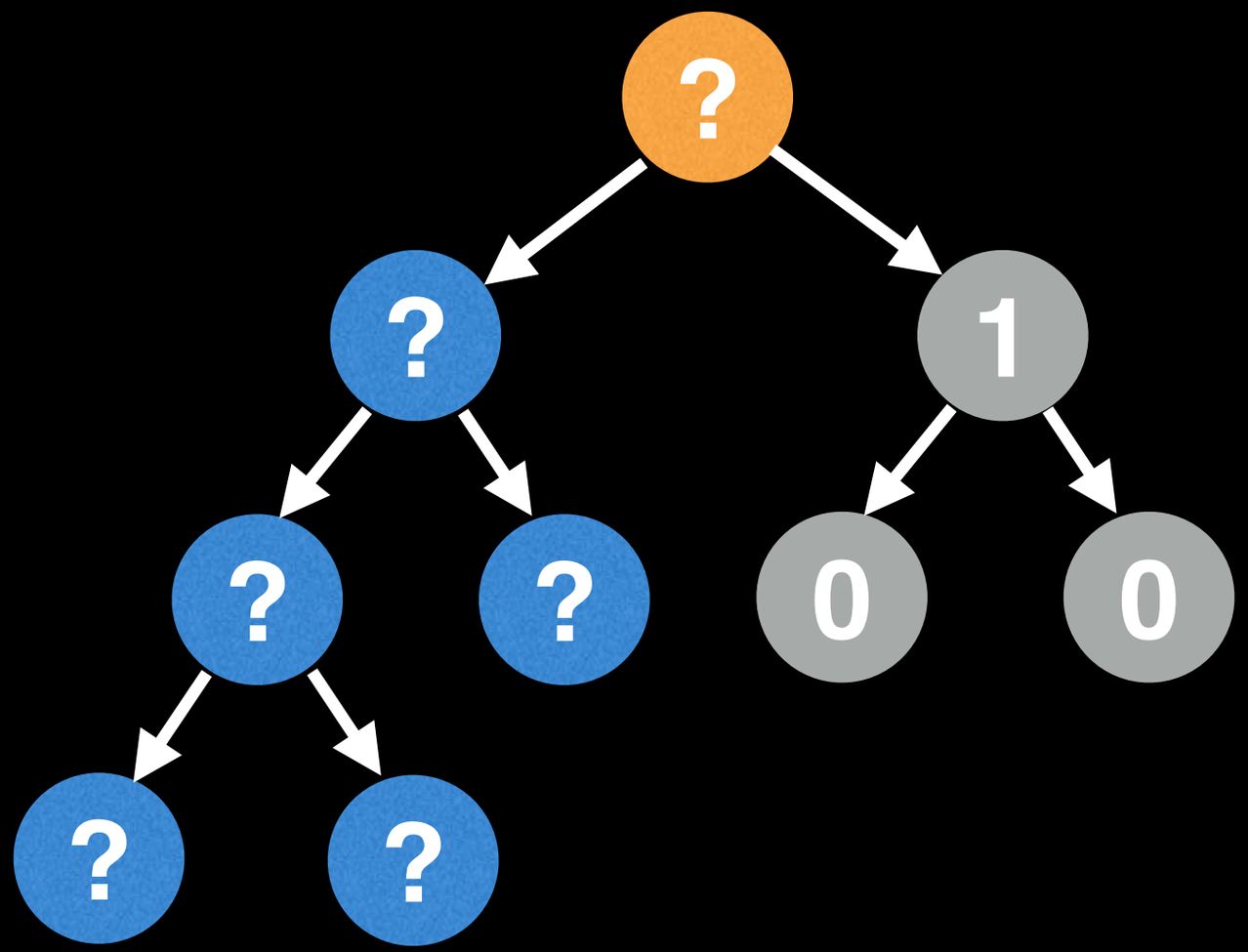

?
?
?
?
?
0
?
?
?

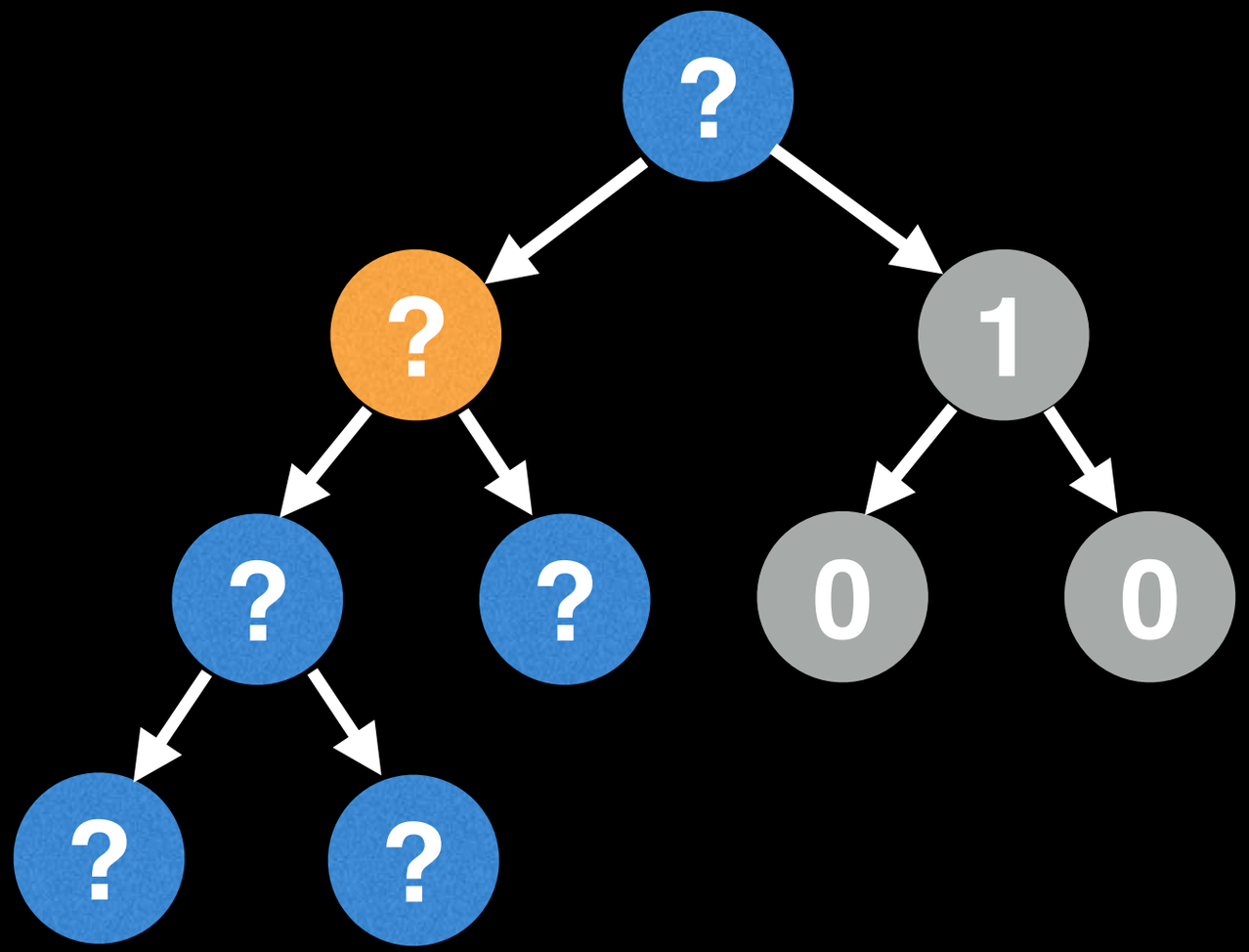


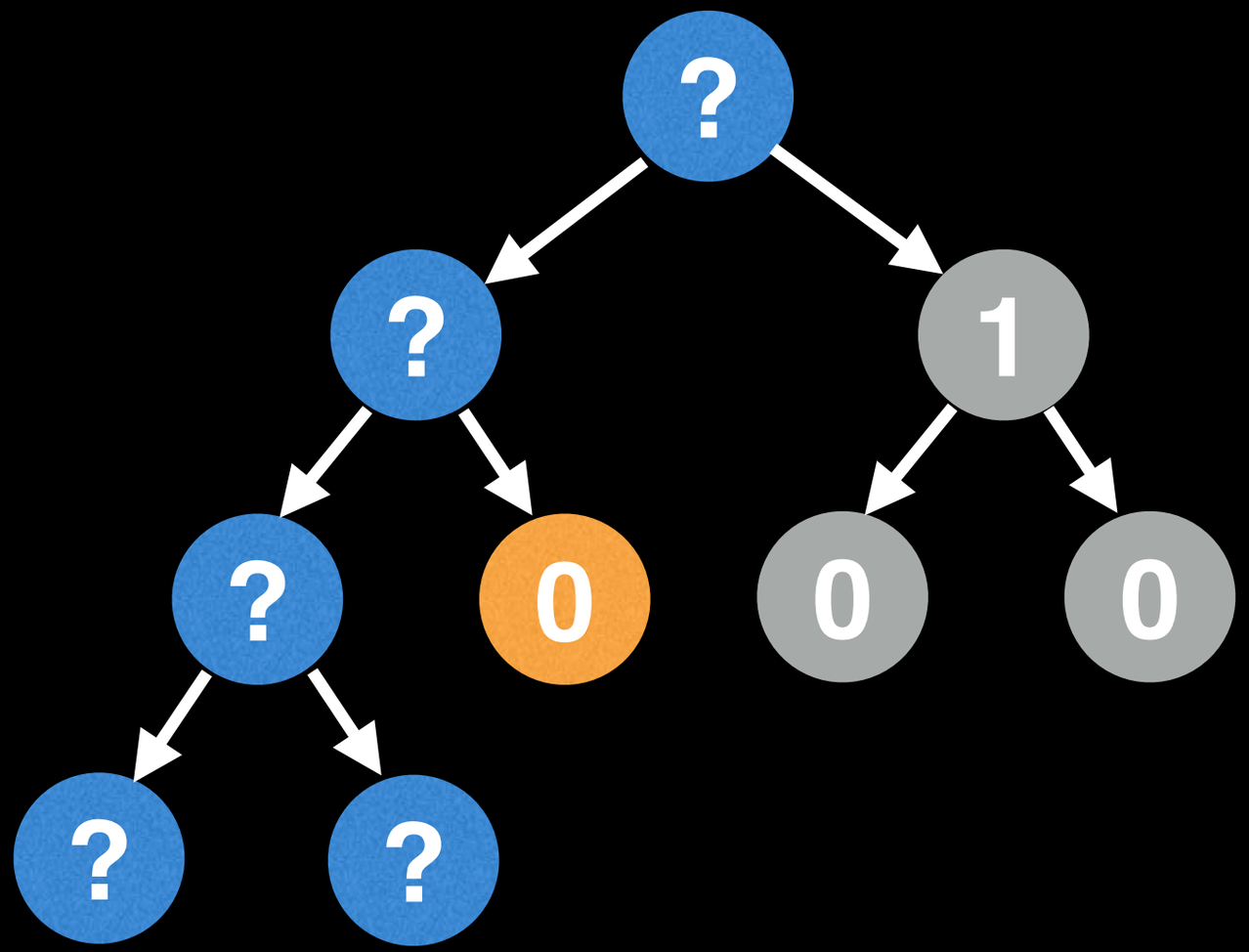
Leaf node has a height of 0



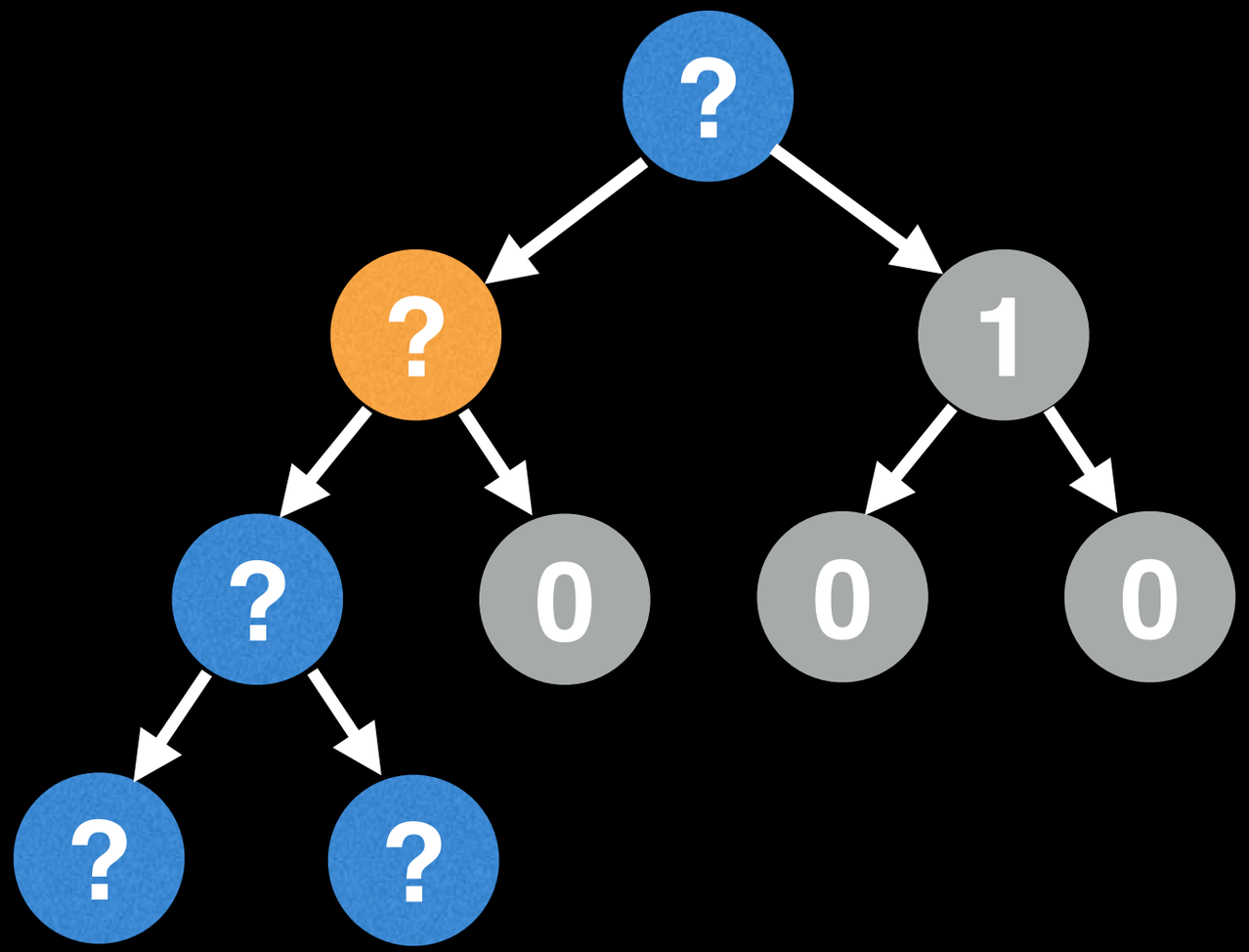

```
height = max(0, 0) + 1 = 1
```

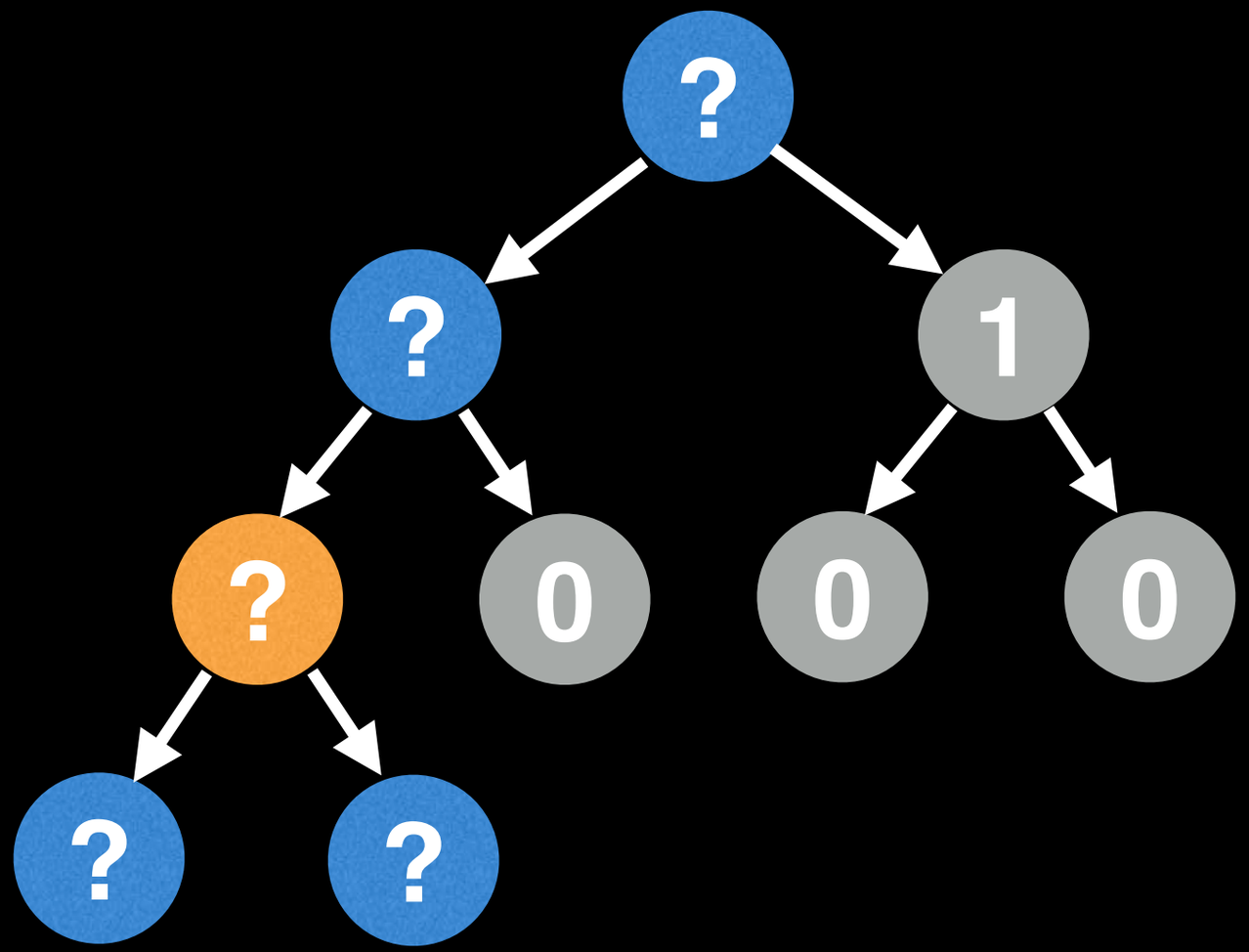

?
?
1
?
?
0
0
?
?

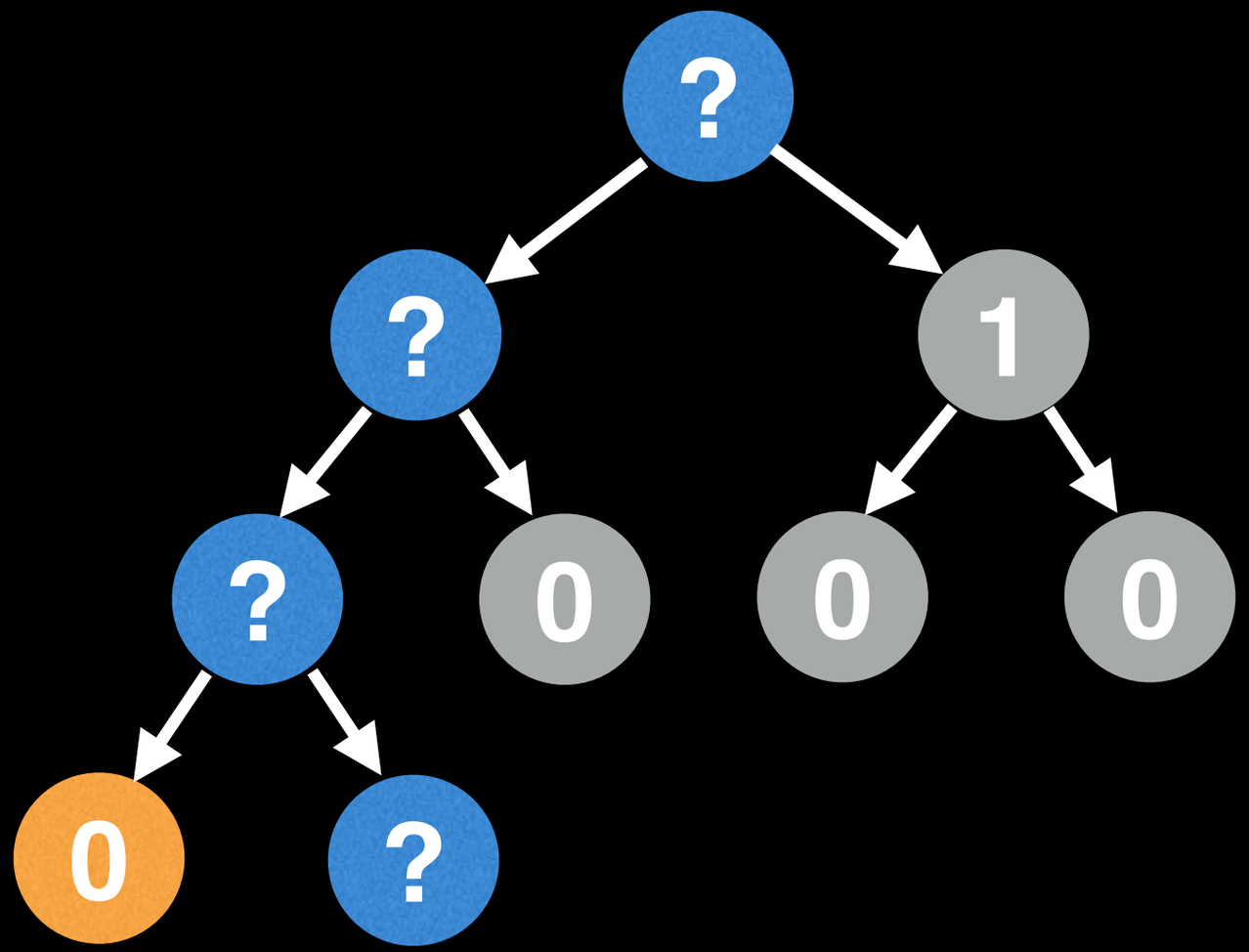

?
?
1
?
?
0
0
?
?



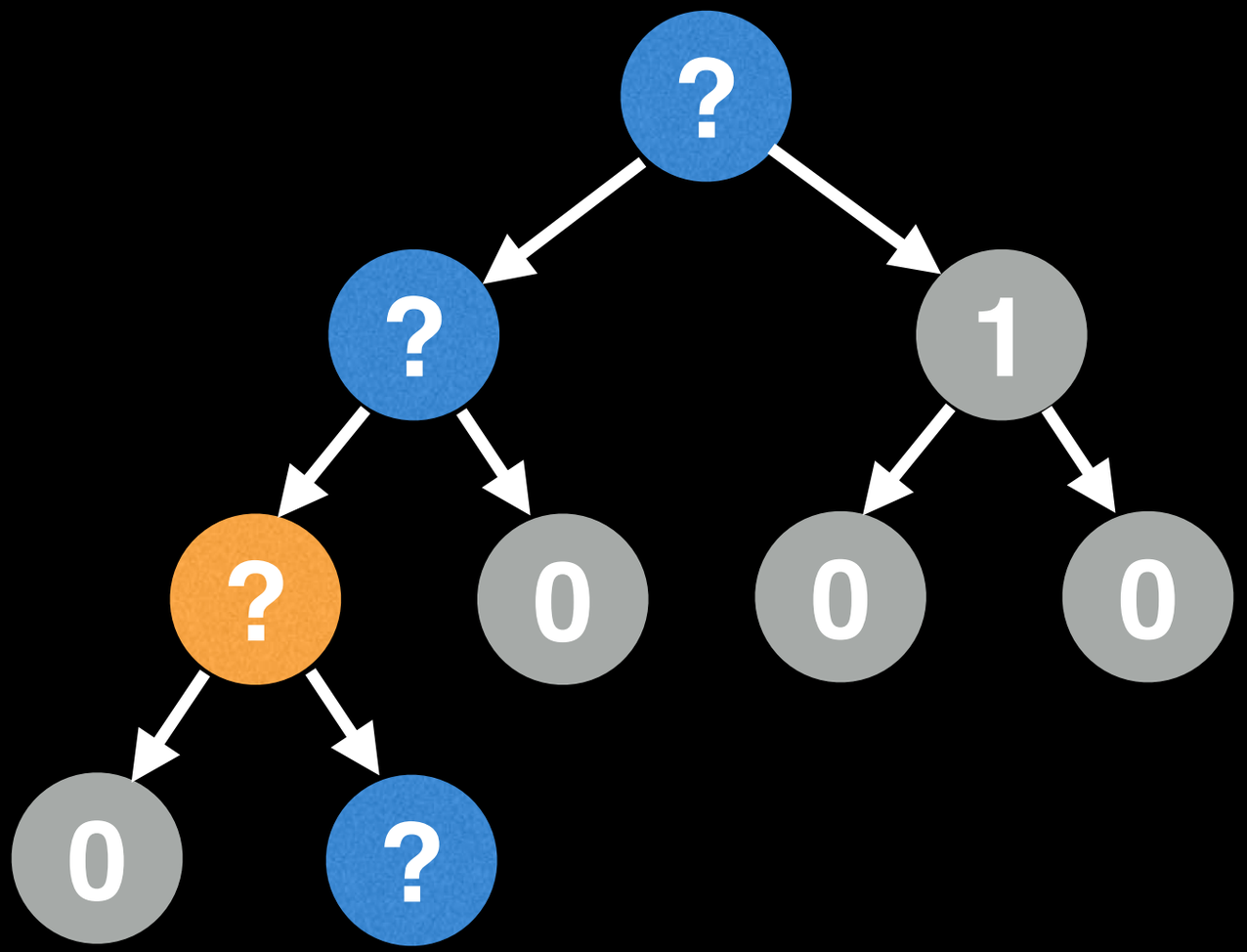
Leaf node has a height of 0

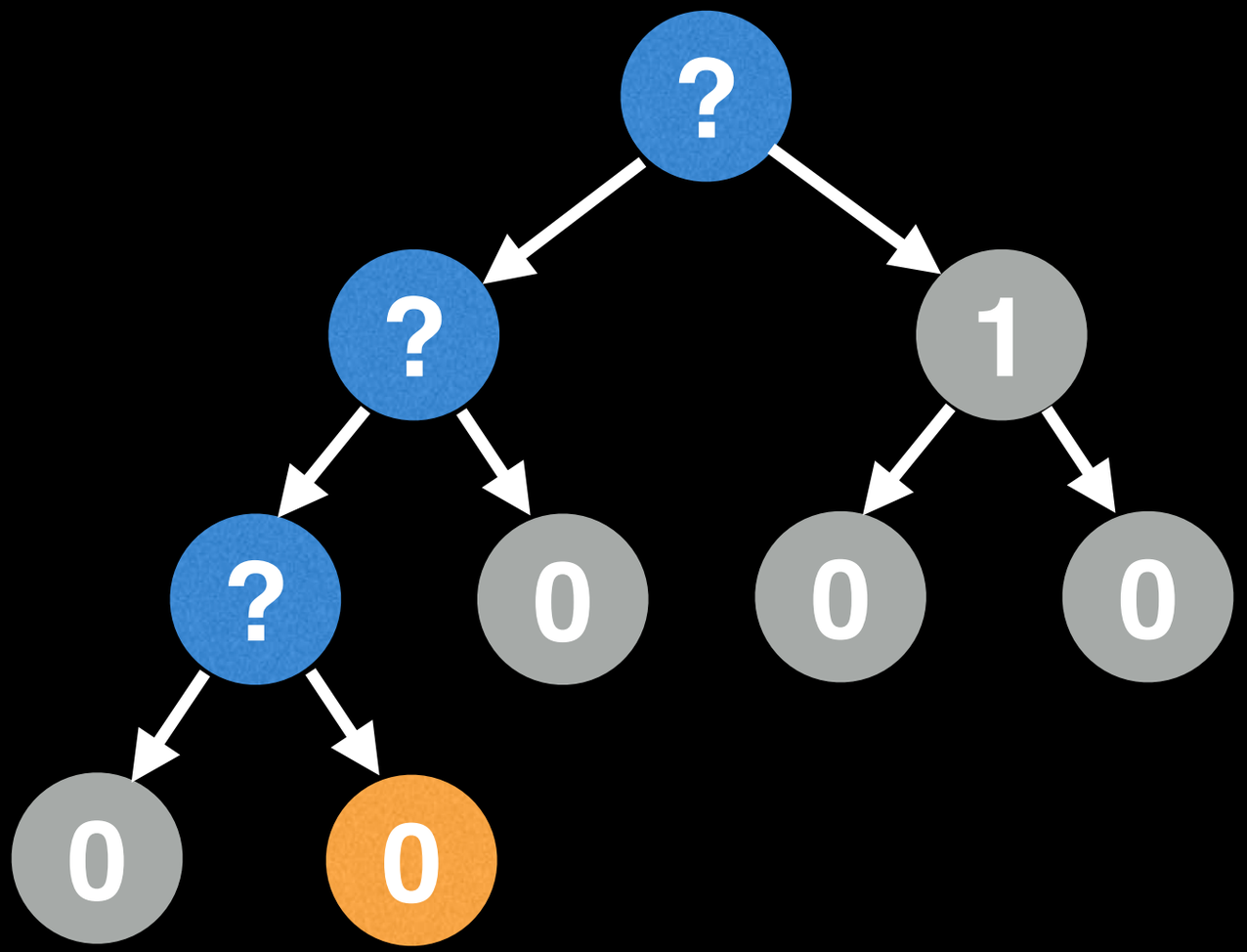

?
?
1
?
0
0
0
?
?


?
?
1
?
0
0
0
?
?

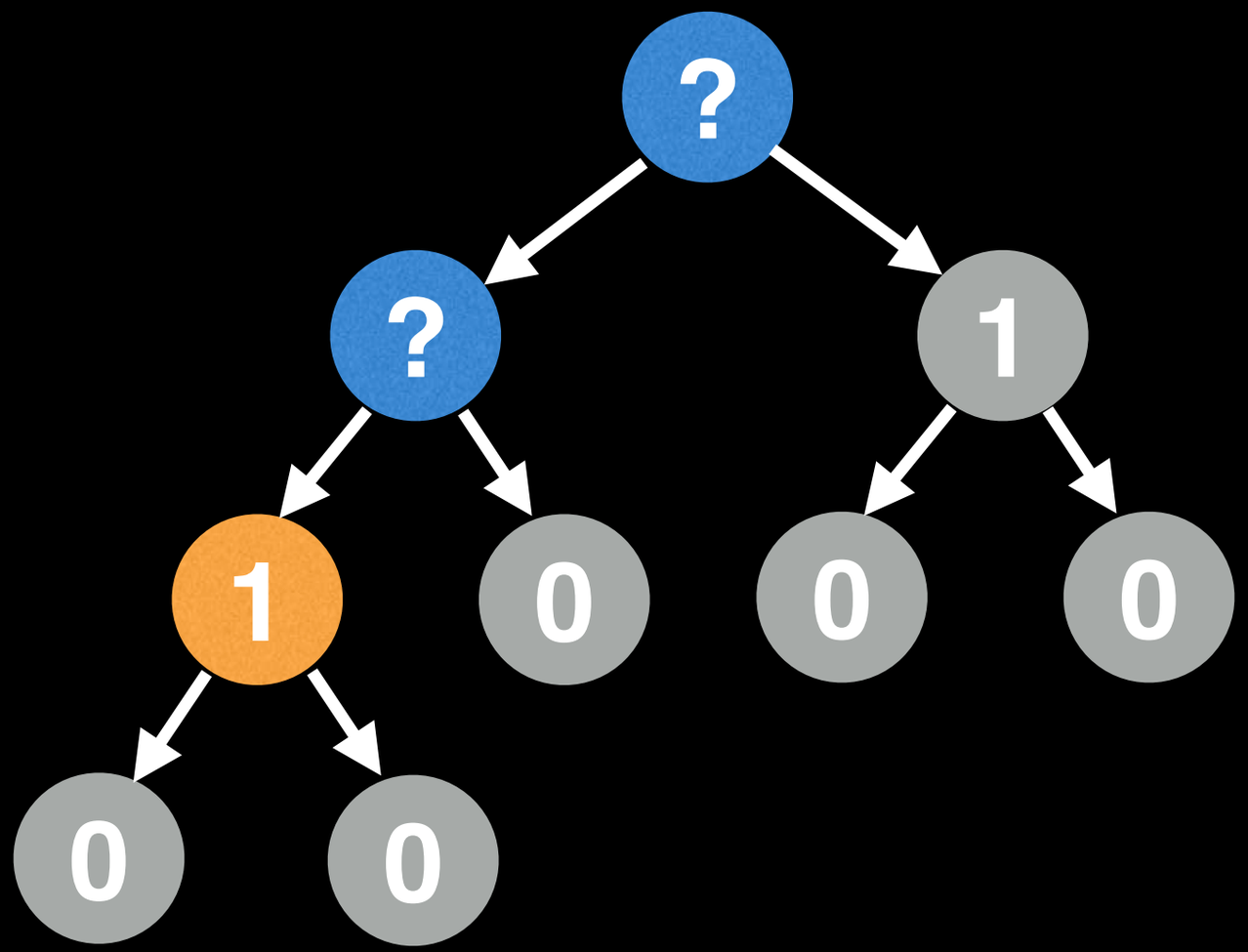


Leaf node has a height of 0

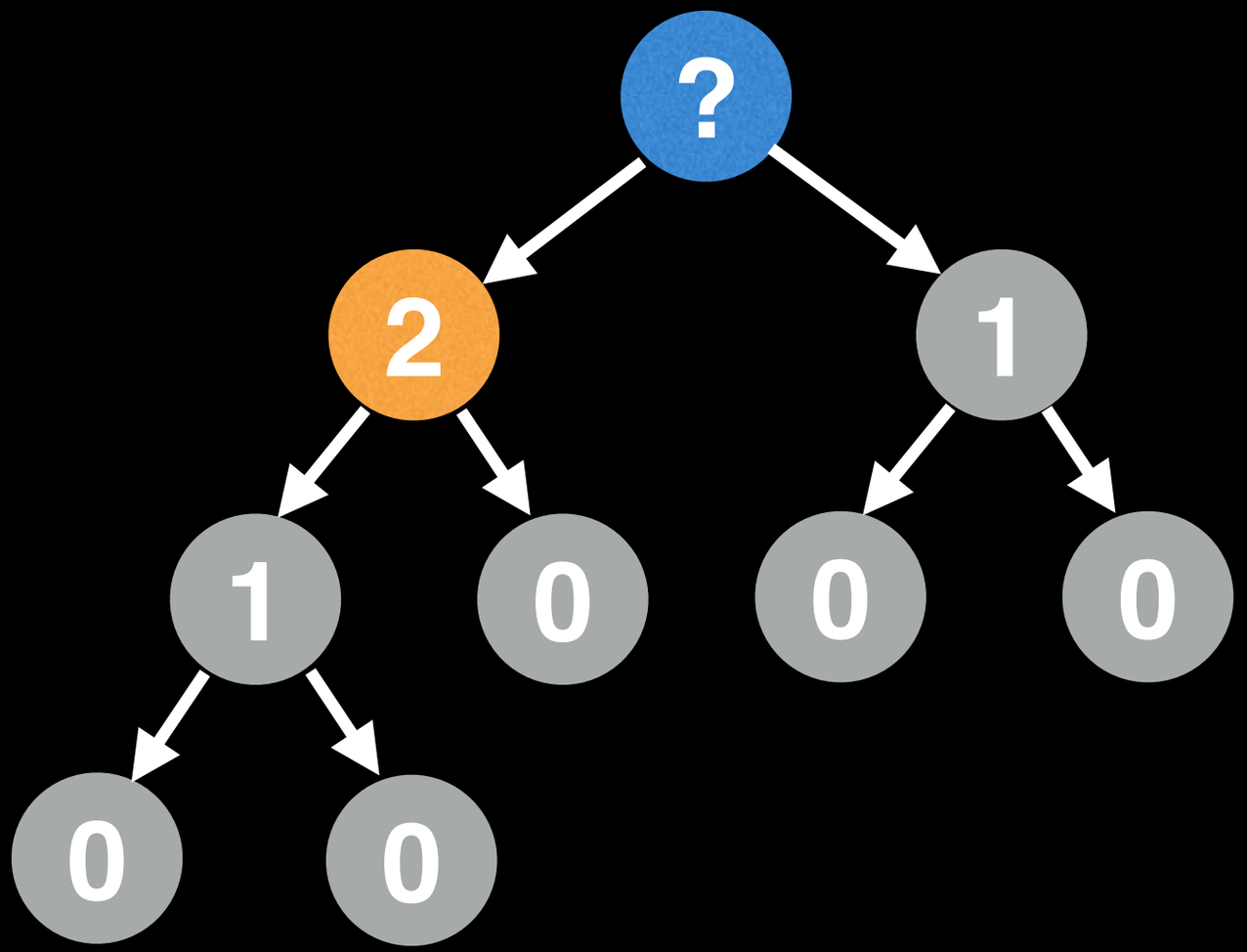

?
?
1
?
0
0
0
0
?



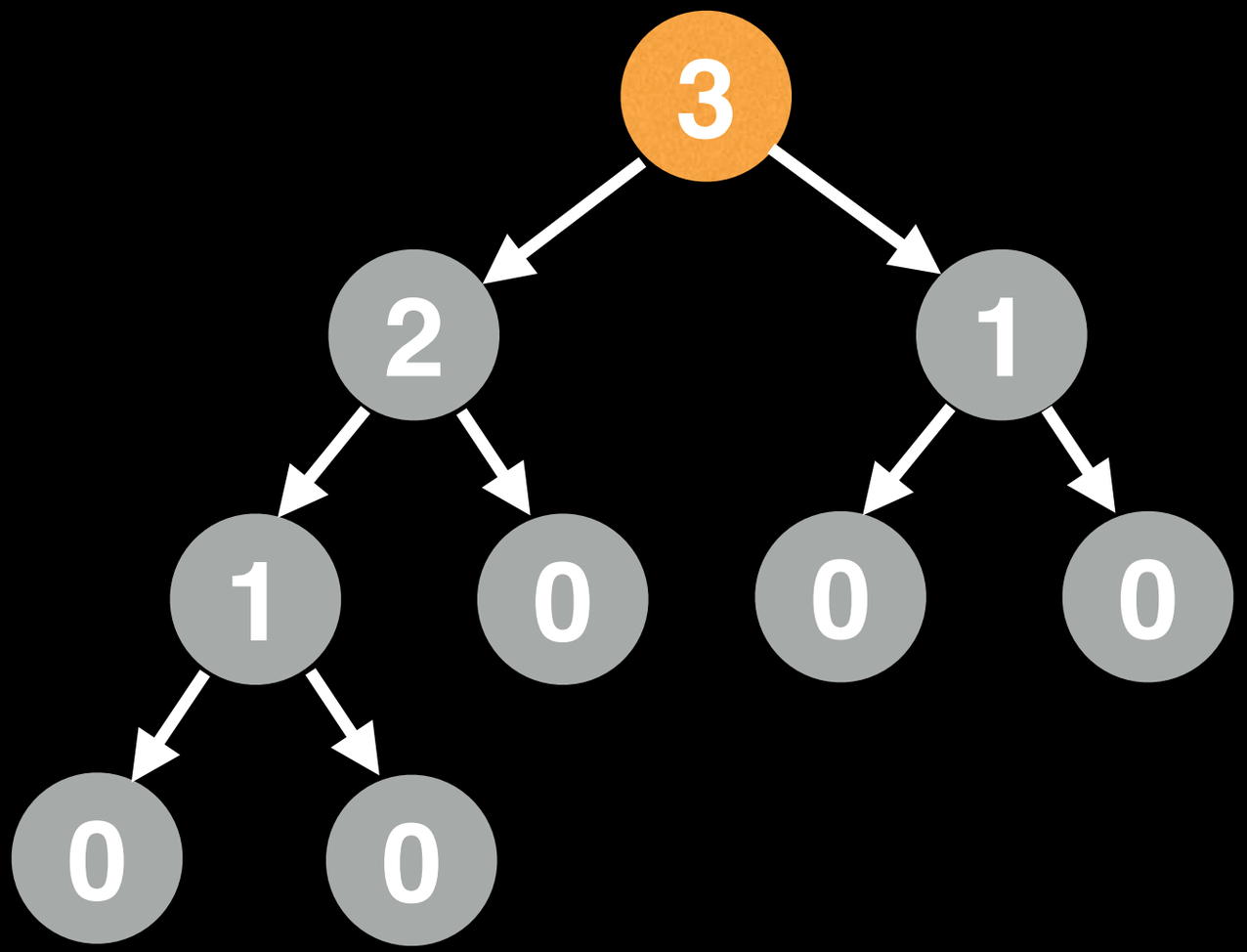
Leaf node has a height of 0



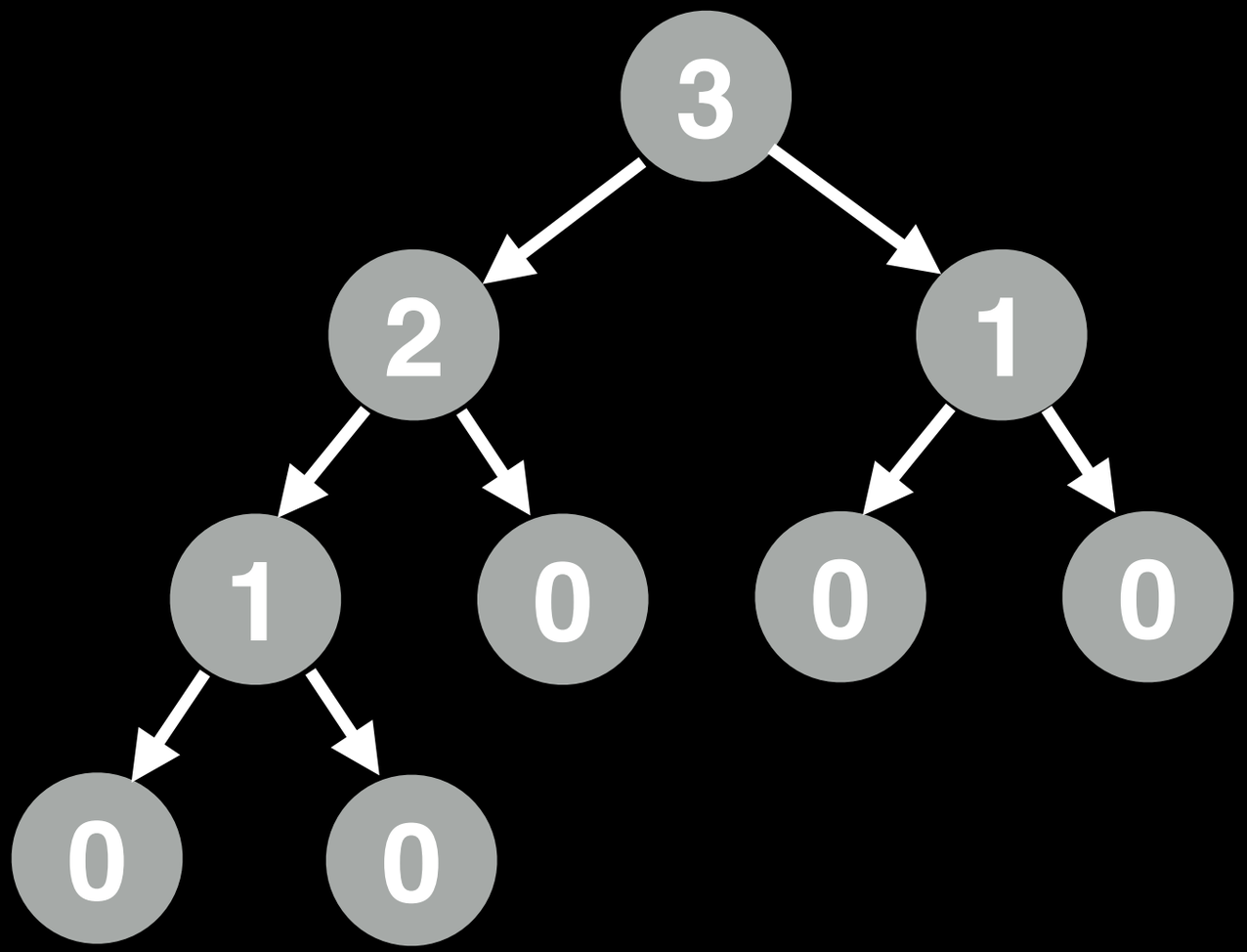

```
height = max(0, 0) + 1 = 1
```




```
height = max(1, 0) + 1 = 2
```




```
height = max(2, 1) + 1 = 3
```


3
2
1
1
0
0
0
0
0

```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
  # Handle empty tree case
  if node == null:
     return -1
  # Identify leaf nodes and return zero
  if node.left == null and node.right == null:
    return 0

  return max(treeHeight(node.left),
             treeHeight(node.right)) + 1
```

```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
    # Handle empty tree case
    if node == null:
        return -1
    # Identify leaf nodes and return zero
    if node.left == null and node.right == null:
        return 0

    return max(treeHeight(node.left),
               treeHeight(node.right)) + 1
```

```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
    # Handle empty tree case
    if node == null:
        return -1
    # Identify leaf nodes and return zero
    if node.left == null and node.right == null:
        return 0

    return max(treeHeight(node.left),
               treeHeight(node.right)) + 1
```

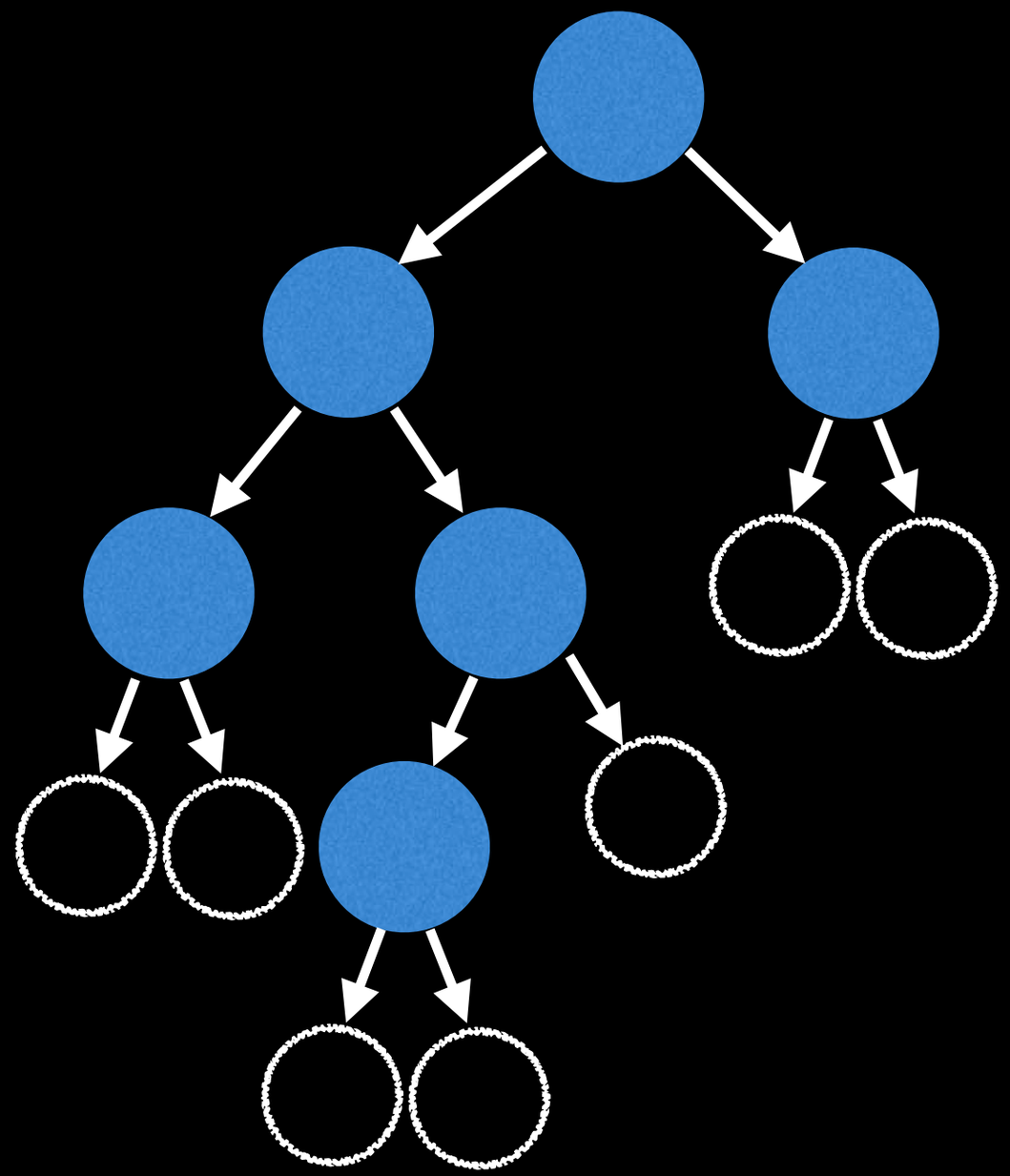
```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
  # Handle empty tree case
  if node == null:
     return -1
# Identify leaf nodes and return zero
if node.left == null and node.right == null:
  return 0

return max(treeHeight(node.left),
           treeHeight(node.right)) + 1
```

```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
  # Handle empty tree case
  if node == null:
     return -1
  # Identify leaf nodes and return zero
  if node.left == null and node.right == null:
    return 0

  return max(treeHeight(node.left),
             treeHeight(node.right)) + 1
```

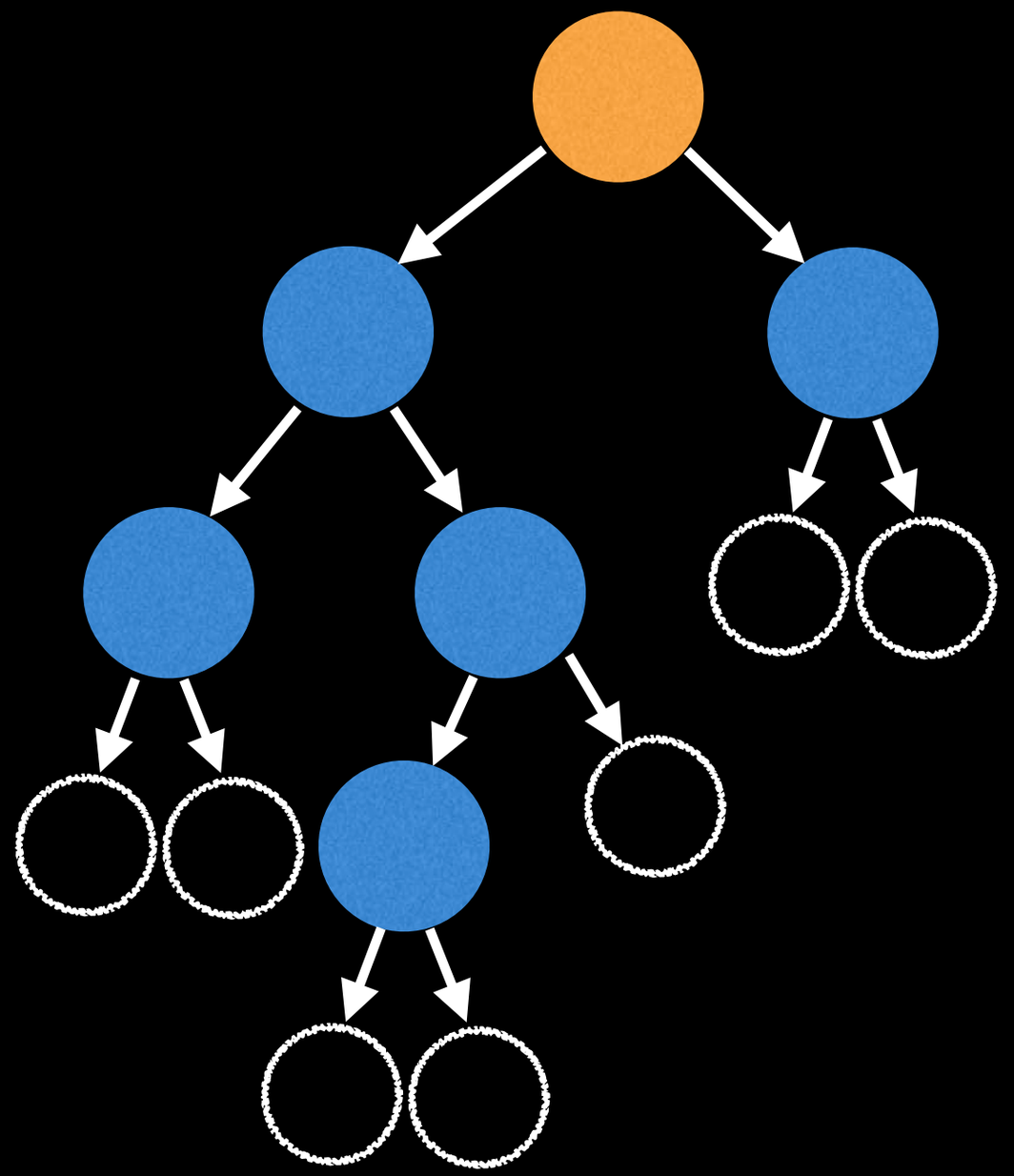
```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
  # Handle empty tree case
  if node == null:
     return -1
  # Identify leaf nodes and return zero
  if node.left == null and node.right == null:
    return 0

  return max(treeHeight(node.left),
             treeHeight(node.right)) + 1
```

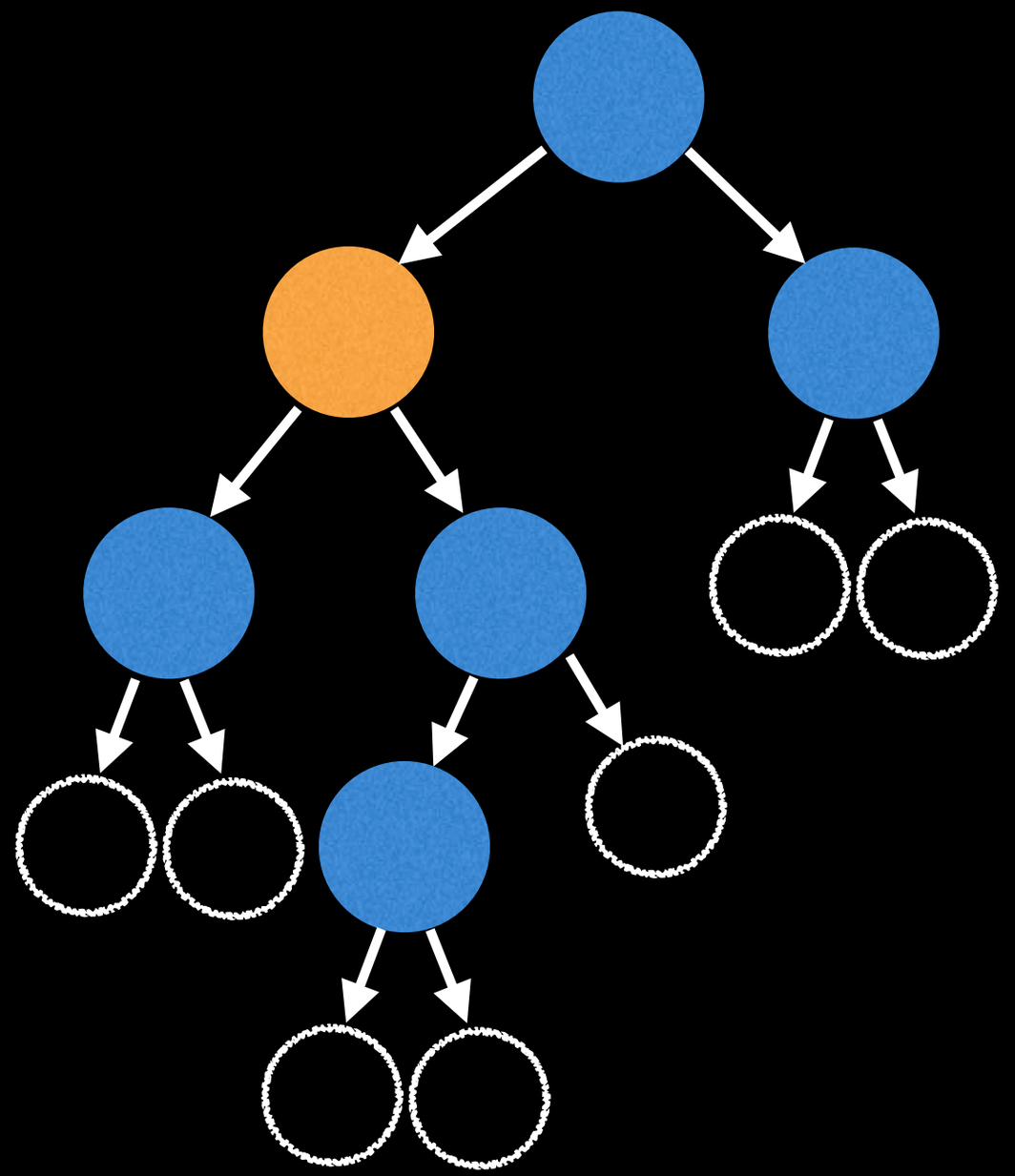
```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
  # Return -1 when we hit a null node
  # to correct for the right height.
   if node == null:
     return -1

  return max(treeHeight(node.left),
             treeHeight(node.right)) + 1
```

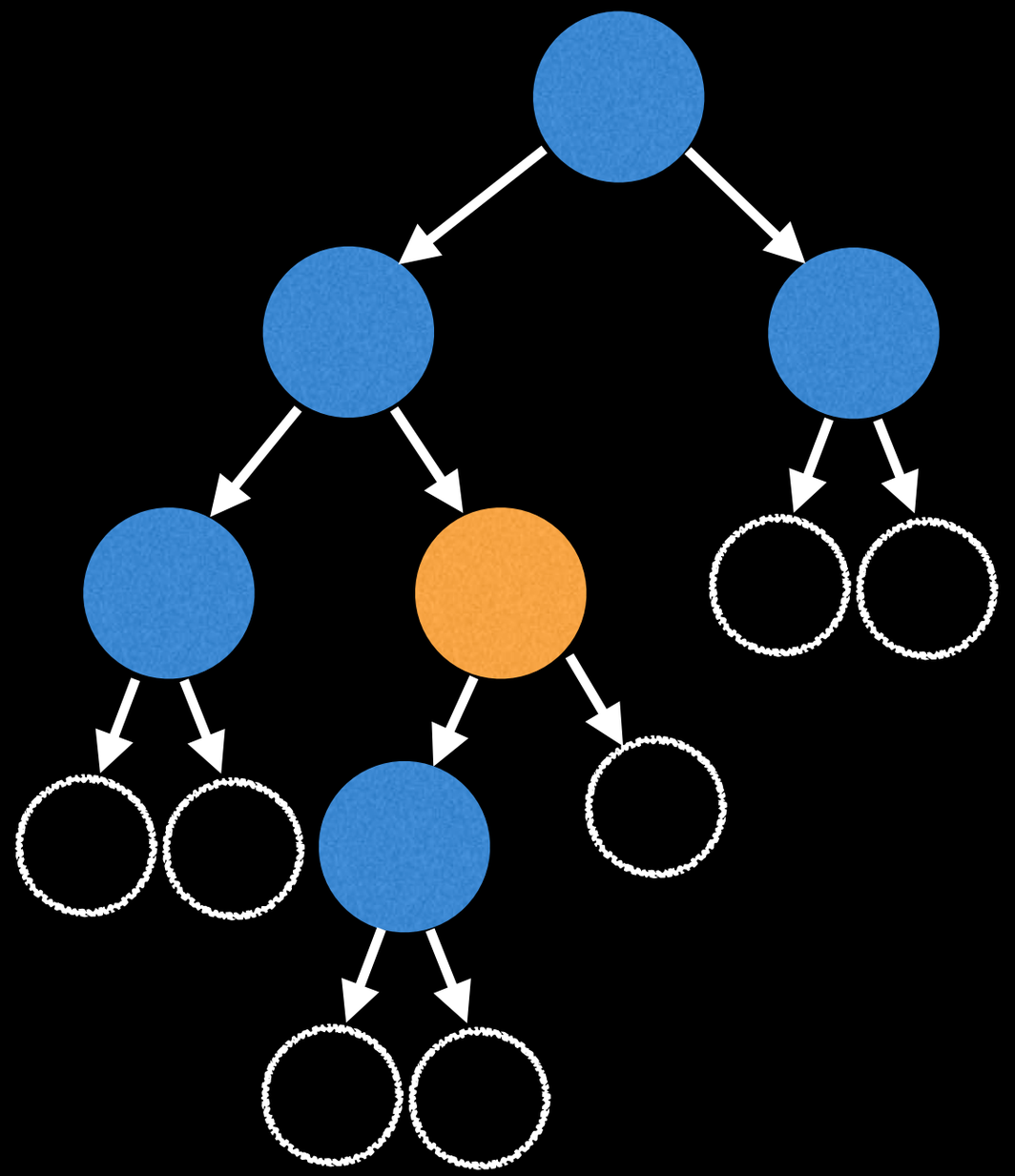
Notice that if we visit the null nodes
our tree is one unit taller.

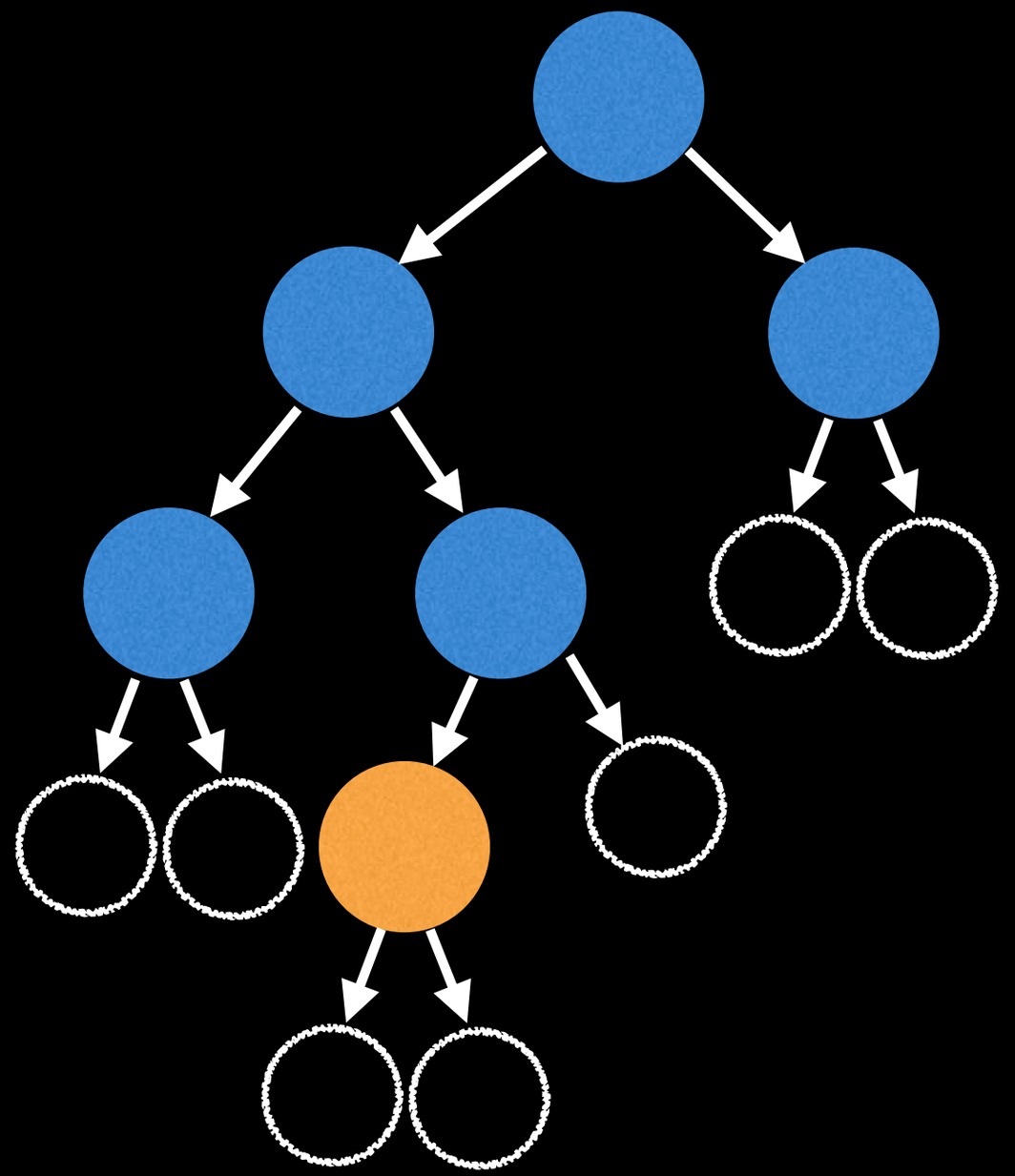


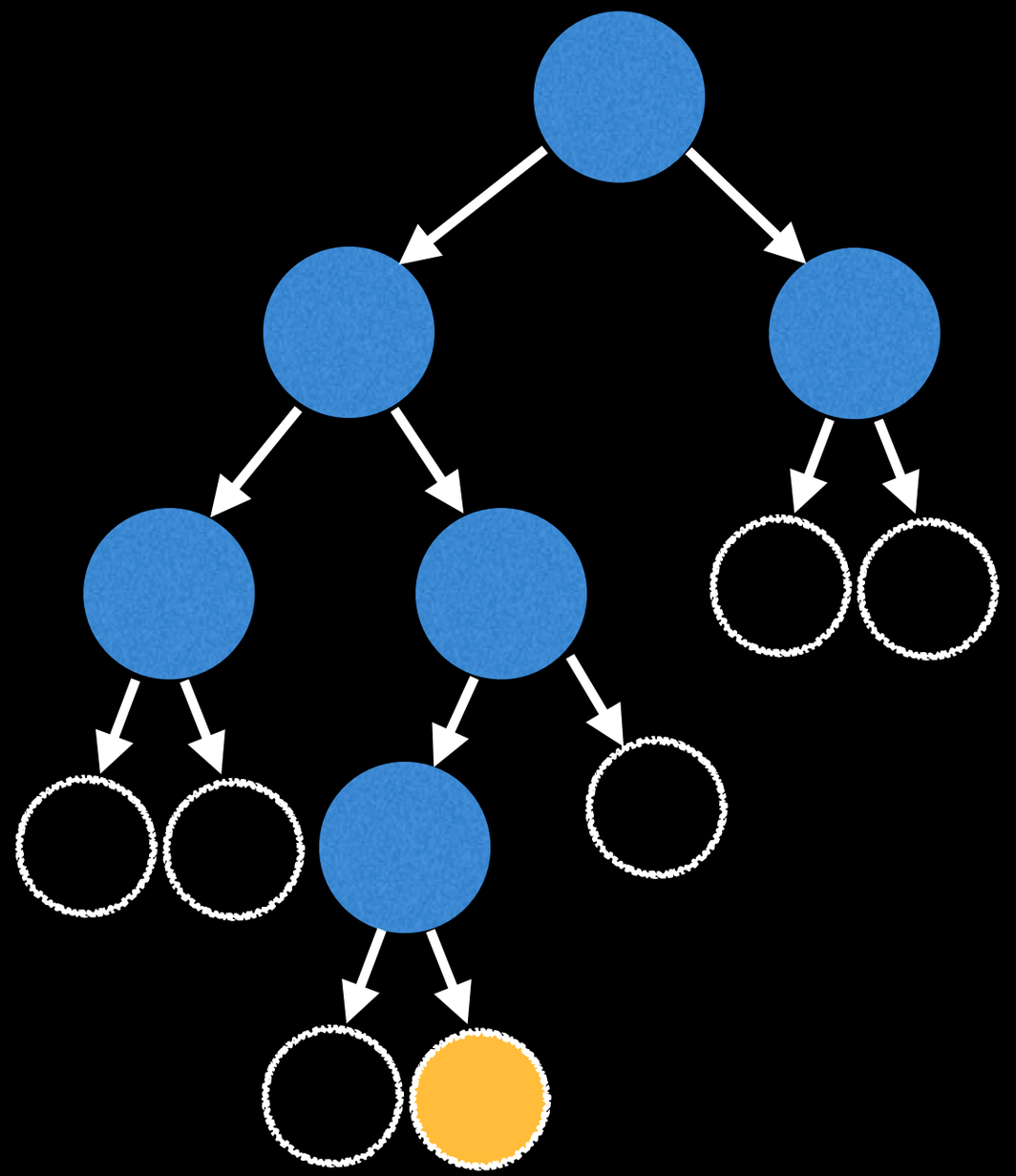
When we go down the tree we need to correct for the height added by the null nodes.

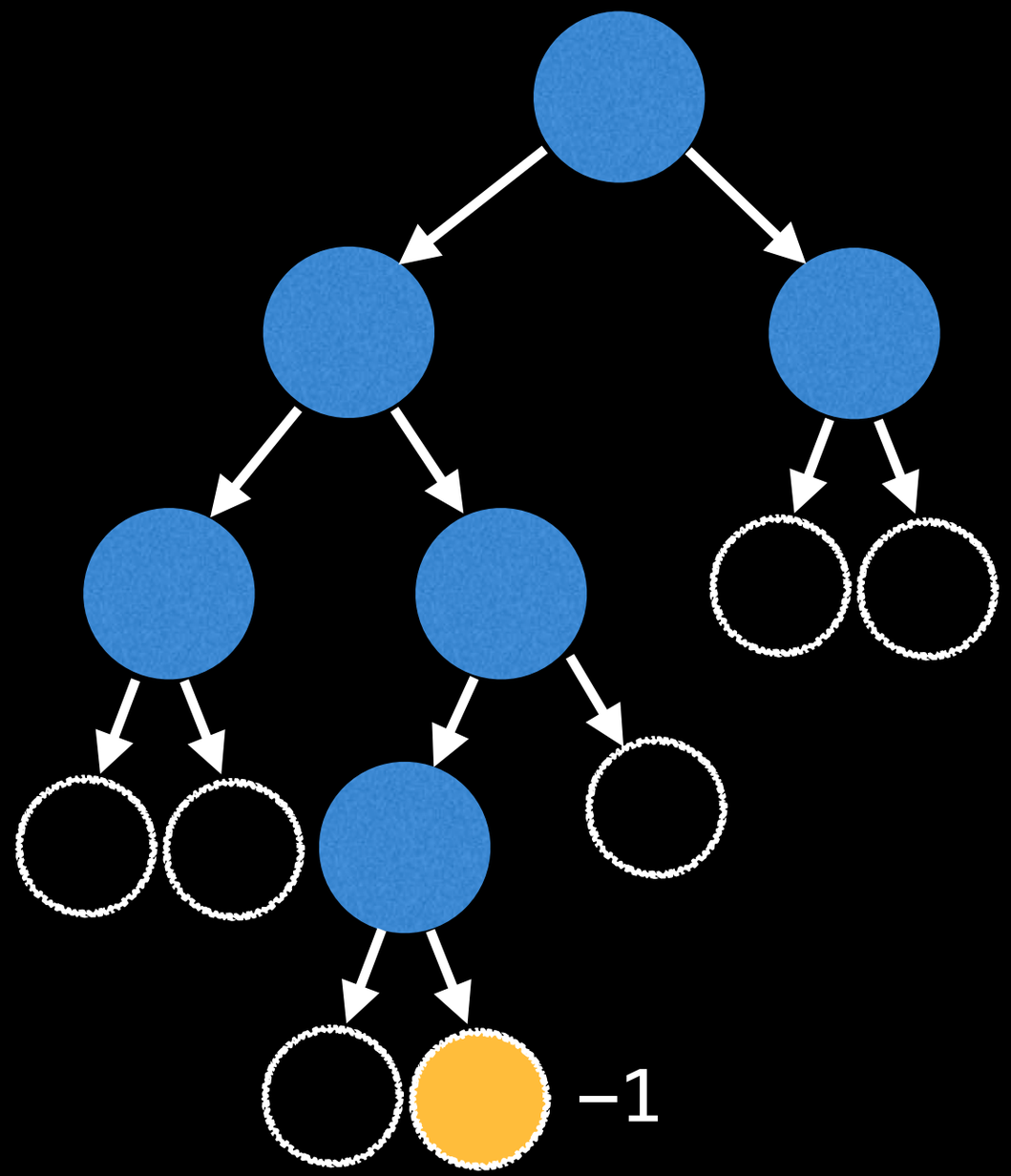


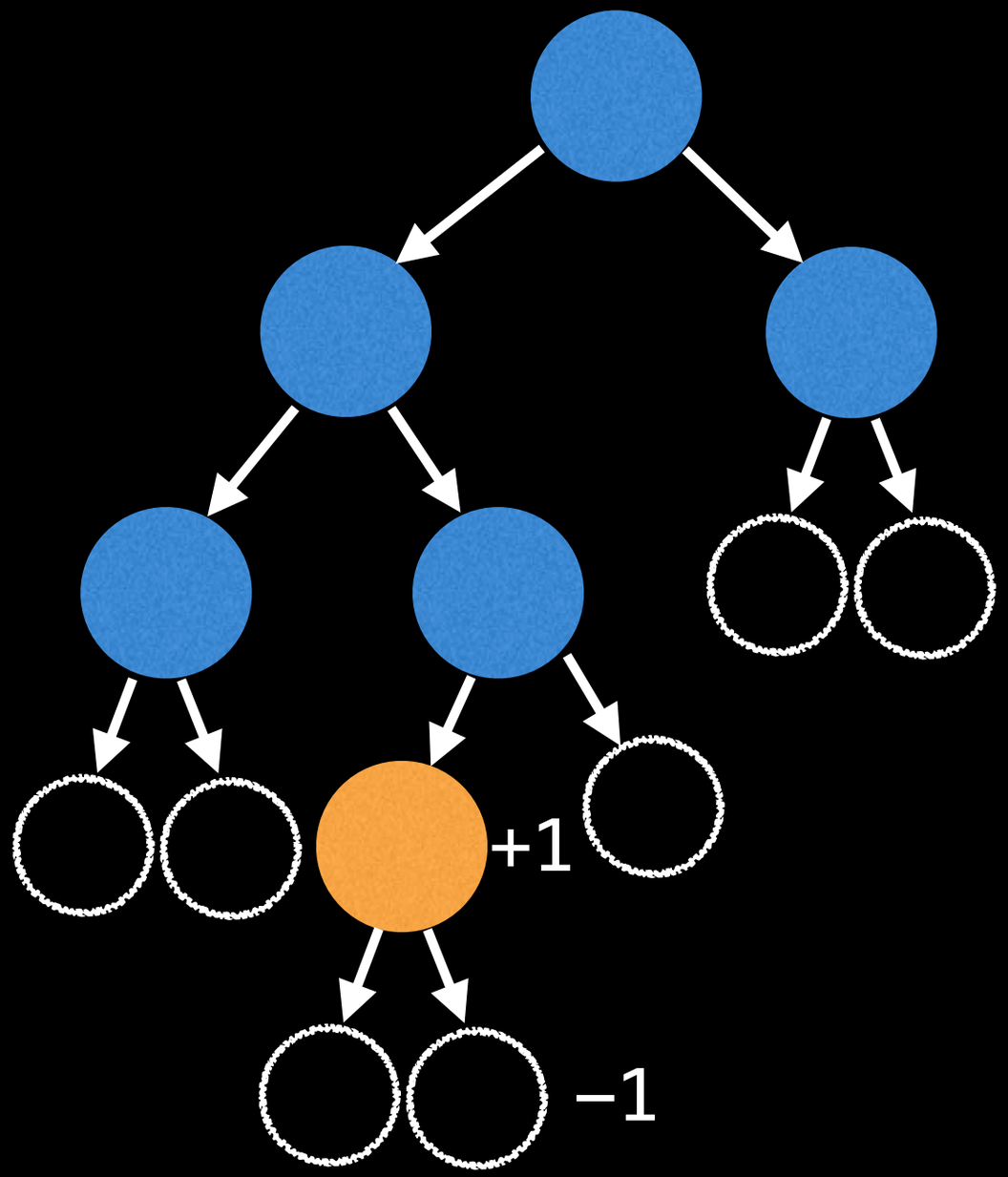


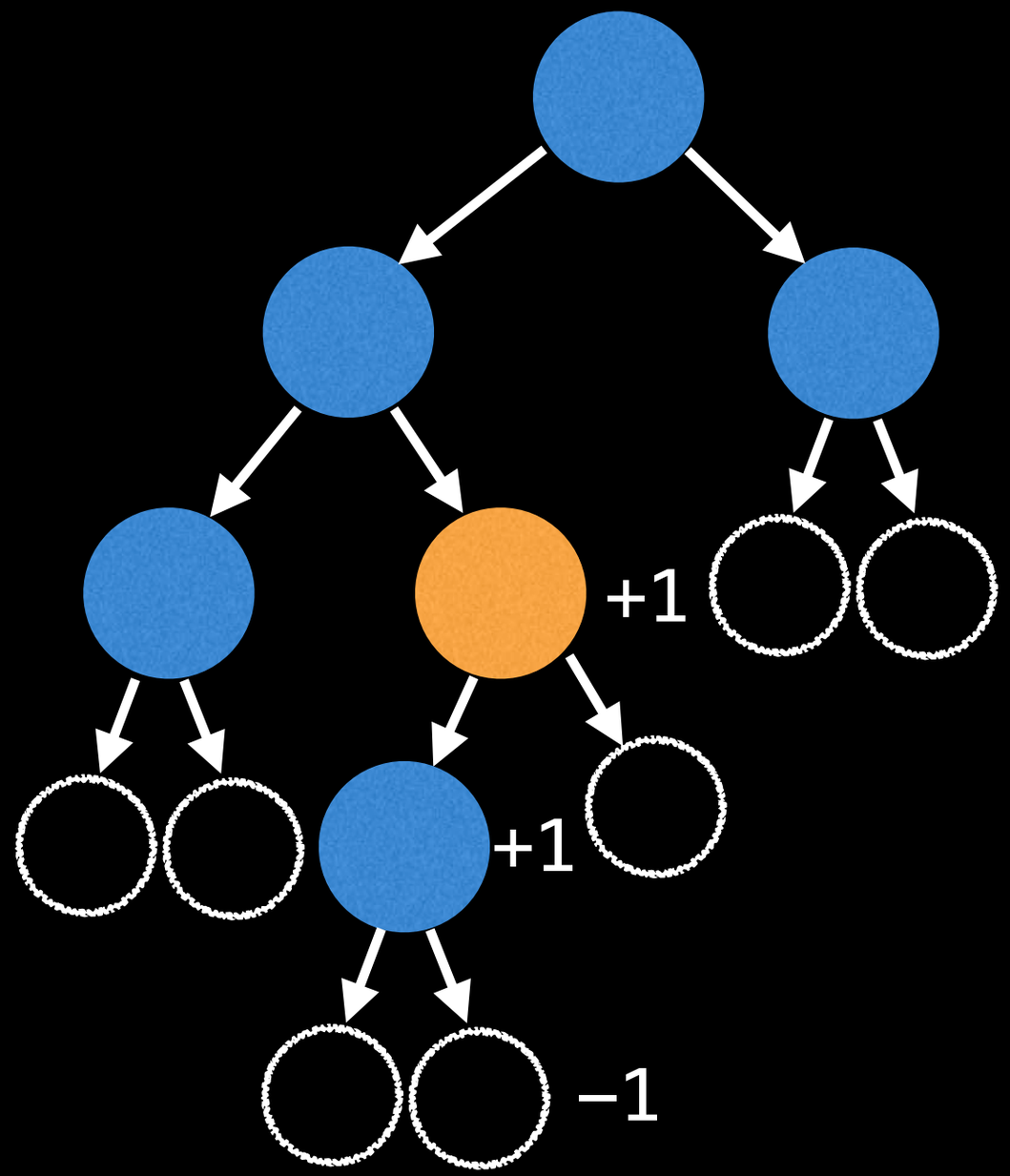


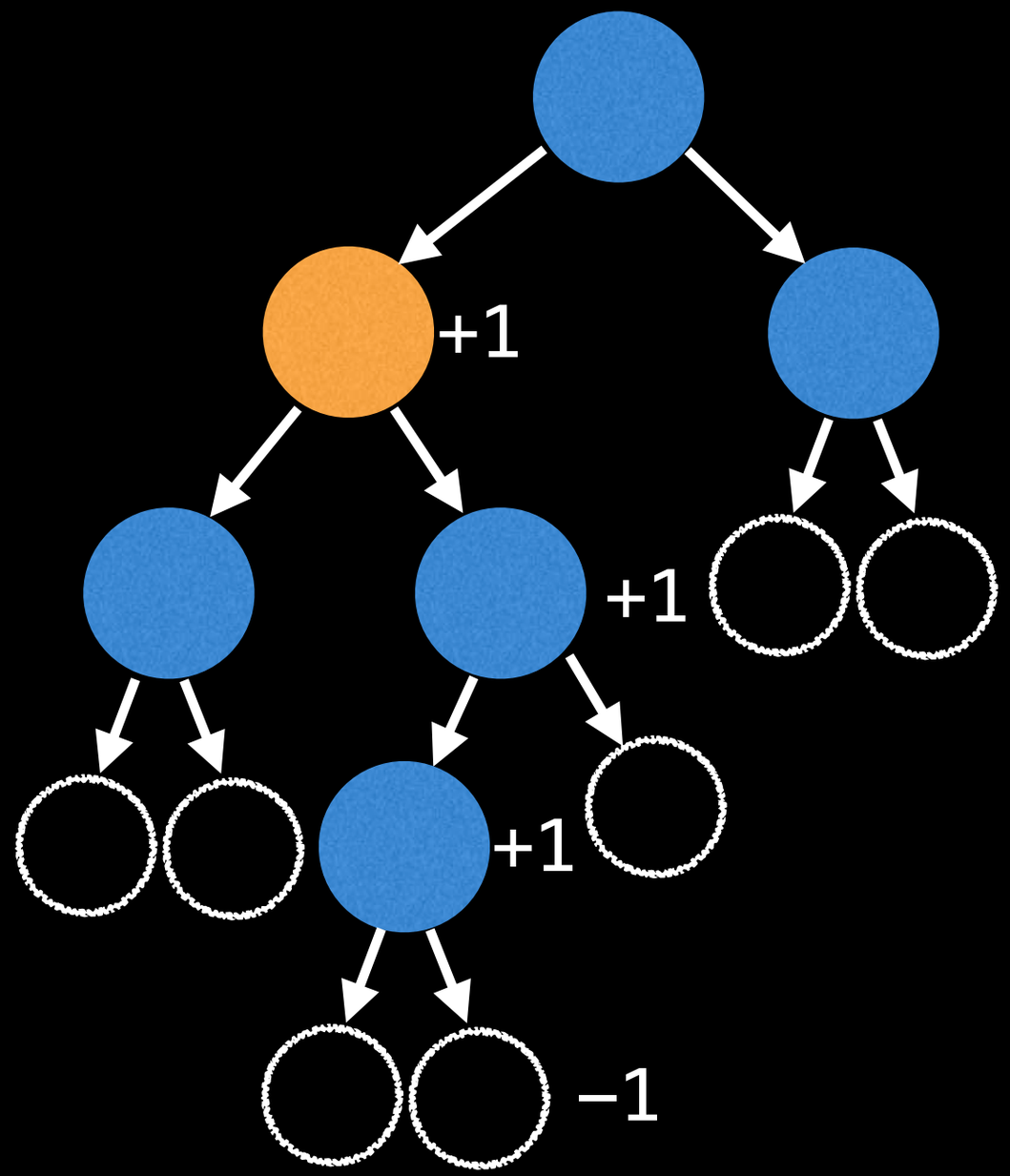


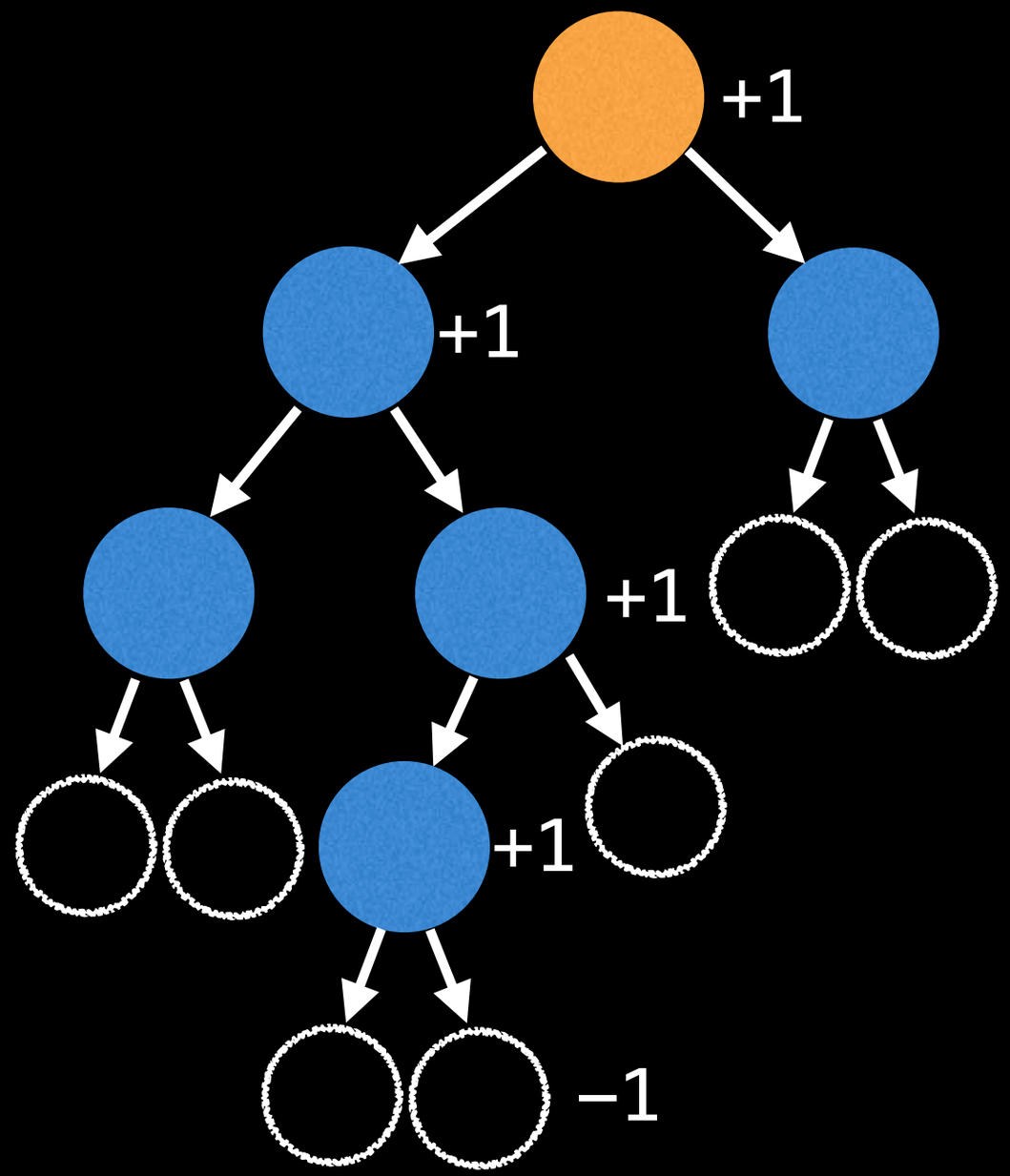


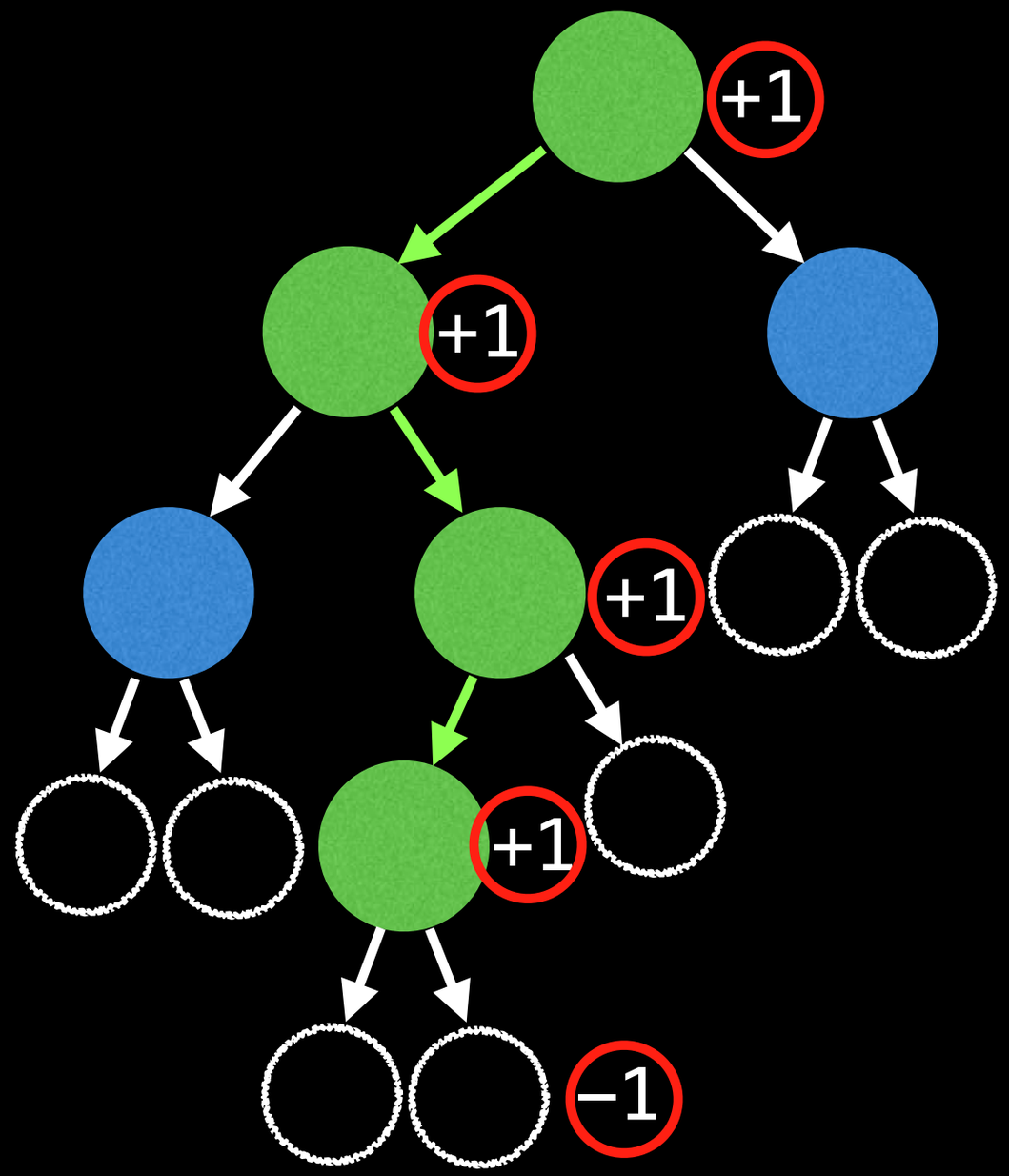

−1


+1
−1


+1
+1
−1


+1
+1
+1
−1


+1
+1
+1
+1
−1



1 + 1 + 1 + 1 − 1 = 3

```
# The height of a tree is the number of
# edges from the root to the lowest leaf.
function treeHeight(node):
  # Return -1 when we hit a null node
  # to correct for the right height.
   if node == null:
     return -1

  return max(treeHeight(node.left),
             treeHeight(node.right)) + 1
```