

Binary Tree Path Sum (easy)

We'll cover the following



- Problem Statement
- Try it yourself
- Solution
- Code
 - Time complexity
 - Space complexity

Problem Statement#

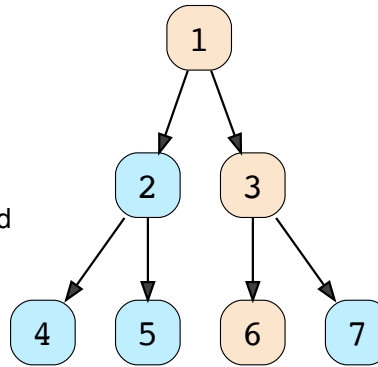
Given a binary tree and a number 'S', find if the tree has a path from root-to-leaf such that the sum of all the node values of that path equals 'S'.

Example 1:

S: 10

Output: true

Explanation: The path with sum '10' is highlighted

**Example 2:**

S: 23

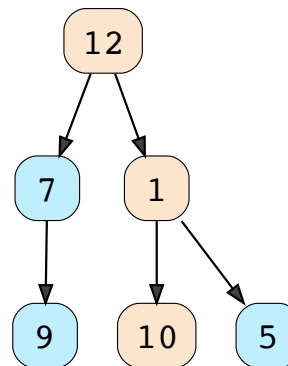
Output: true

Explanation: The path with sum '23' is highlighted

S: 16

Output: false

Explanation: There is no root-to-leaf path with sum '16'.



Try it yourself#

Try solving this question here:

Java

Python3

JS

C++

```
1 class TreeNode {
2
3     constructor(value) {
4         this.value = value;
5         this.left = null;
```



```
6      this.right = null;
7    }
8  };
9
10
11  const has_path = function(root, sum) {
12    // TODO: Write your code here
13    return false;
14  };
15
16
17  var root = new TreeNode(12)
18  root.left = new TreeNode(7)
19  root.right = new TreeNode(1)
20  root.left.left = new TreeNode(9)
21  root.right.left = new TreeNode(10)
22  root.right.right = new TreeNode(5)
23  console.log(`Tree has path: ${has_path(root, 23)}`)
24  console.log(`Tree has path: ${has_path(root, 16)}`)
25
```

RunSaveReset

Solution#

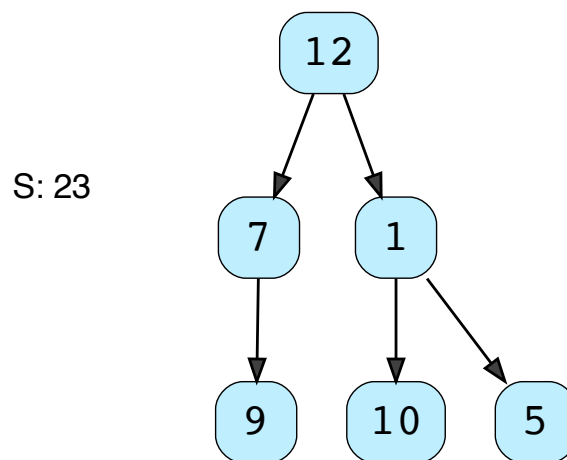
As we are trying to search for a root-to-leaf path, we can use the **Depth First Search (DFS)** technique to solve this problem.

To recursively traverse a binary tree in a DFS fashion, we can start from the root and at every step, make two recursive calls one for the left and one for the right child.

Here are the steps for our Binary Tree Path Sum problem:

1. Start DFS with the root of the tree.
2. If the current node is not a leaf node, do two things:
 - Subtract the value of the current node from the given number to get a new sum => $S = S - \text{node.value}$
 - Make two recursive calls for both the children of the current node with the new number calculated in the previous step.
3. At every step, see if the current node being visited is a leaf node and if its value is equal to the given number 'S'. If both these conditions are true, we have found the required root-to-leaf path, therefore return true.
4. If the current node is a leaf but its value is not equal to the given number 'S', return false.


Let's take the example-2 mentioned above to visually see our algorithm:



Let's start with the root node

1 of 10



 Codecademy (/learn)
9101419520&cid=5671464854355968&pid=5642684278505472)



Here is what our algorithm will look like:

 Java

 Python3

 C++

 JS

```
1 class TreeNode {
2     constructor(val, left = null, right = null) {
3         this.val = val;
4         this.left = left;
5         this.right = right;
6     }
7 }
8
9
10 function hasPath(root, sum) {
11     if (root === null) {
12         return false;
13     }
14
15     // if the current node is a leaf and its value is equal to the sum, we've
16     if (root.val === sum && root.left === null && root.right === null) {
17         return true;
18     }
19
20     // recursively call to traverse the left and right sub-tree
21     // return true if any of the two recursive call return true
22     return hasPath(root.left, sum - root.val) || hasPath(root.right, sum - root.val);
23 }
24
25
26 const root = new TreeNode(12);
27 root.left = new TreeNode(7);
28 root.right = new TreeNode(1);
```

[Run](#)[Save](#)[Reset](#)

Time complexity#

The time complexity of the above algorithm is $O(N)$, where 'N' is the total number of nodes in the tree. This is due to the fact that we traverse each node once.

Space complexity#

The space complexity of the above algorithm will be $O(N)$ in the worst case. This space will be used to store the recursion stack. The worst case will happen when the given tree is a linked list (i.e., every node has only one child).

Interviewing soon? We've partnered with Hired so that companies apply to [utm_source=educative&utm_medium=lesson&utm_location=US&utm_campaign=hired](https://www.educative.io/courses/grokking-the-coding-interview/RMIGwgpoKKY?utm_source=educative&utm_medium=lesson&utm_location=US&utm_campaign=hired)

[← Back](#)[Introduction](#)[Next →](#)[All Paths for a Sum \(medium\)](#)[Mark as Completed](#)[Report an Issue](#)

