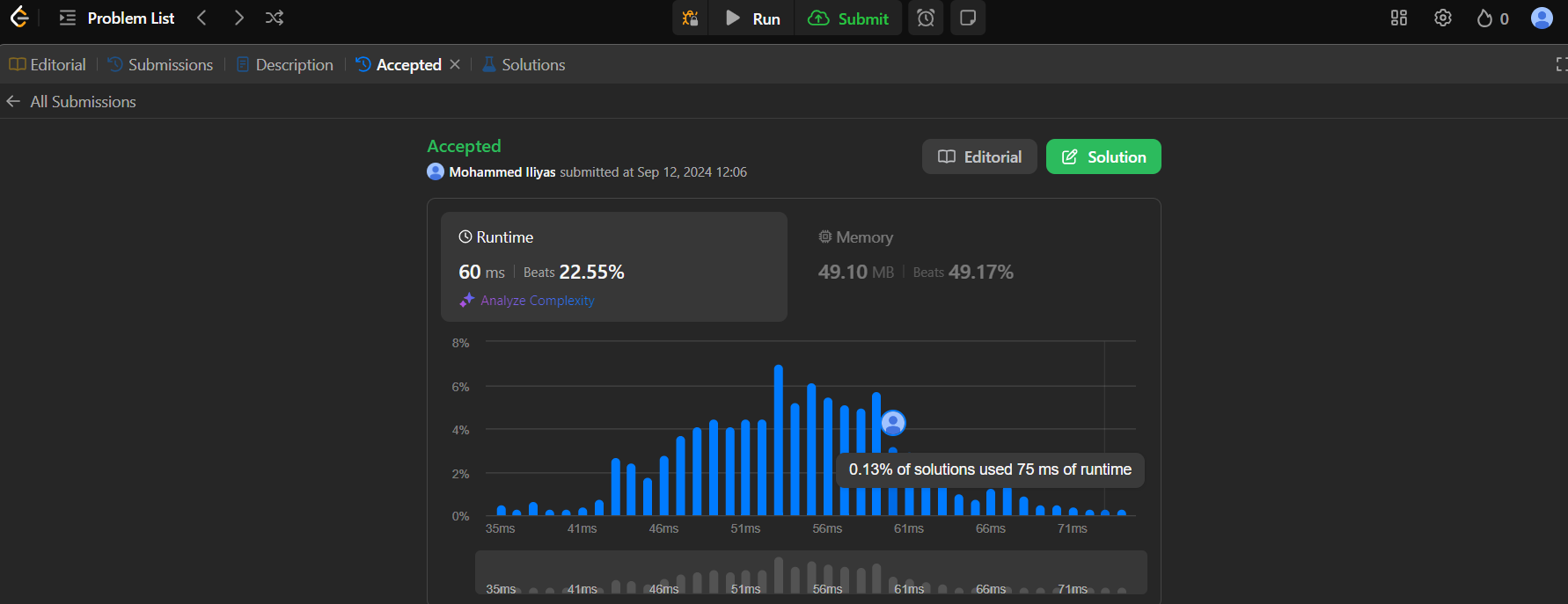
1. Question: <https://leetcode.com/problems/number-of-students-unable-to-eat-lunch/description/>

Solution: <https://leetcode.com/problems/number-of-students-unable-to-eat-lunch/post-solution/?submissionId=1387379566>



Time Complexity: O(n)

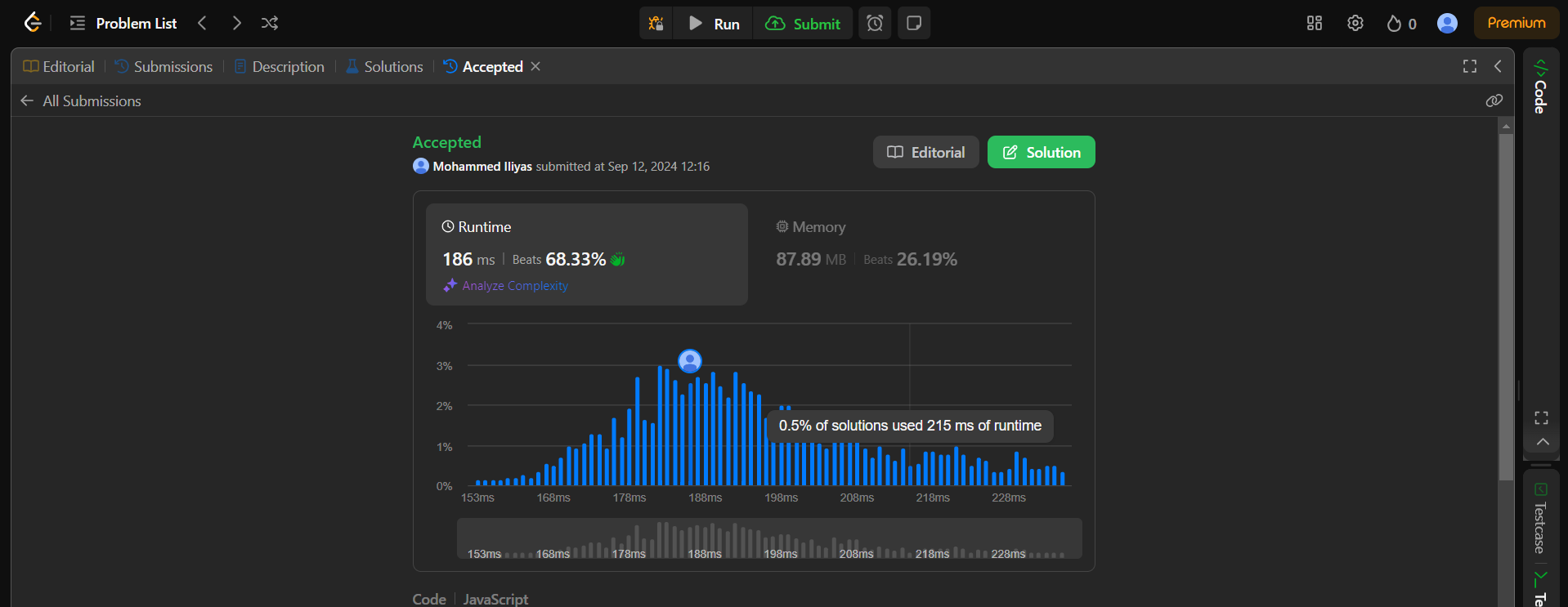
The time complexity of this function is *O*(*n*), where n is the total number of students or sandwiches. This is because the function iterates over each sandwich exactly once.

Space Complexity: O (1)

The space required does not scale with the size of the input, leading to constant space complexity

1. Question: <https://leetcode.com/problems/minimum-depth-of-binary-tree/description/>

Solution: <https://leetcode.com/problems/minimum-depth-of-binary-tree/post-solution/?submissionId=1387388043>



Time Complexity: O(n)

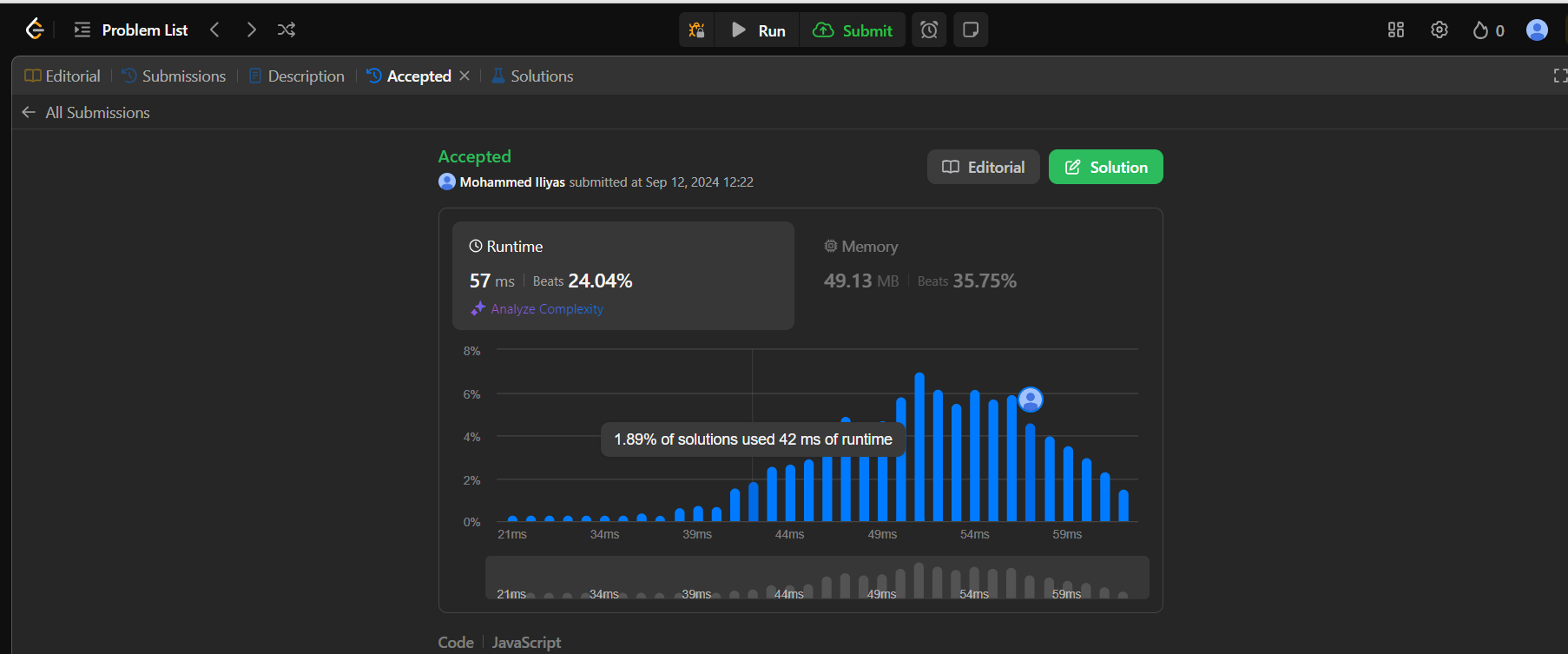
The time complexity is O(n), where n is the number of nodes in the tree. In the worst case, all nodes are visited once.

Space Complexity: O(n)

The space complexity is O(n), which is the maximum number of nodes that can be stored in the queue at any time. In the worst case, this could be proportional to the number of nodes at the bottom level of the tree, which is approximately O(n) for a complete binary tree.

1. Question: <https://leetcode.com/problems/binary-tree-postorder-traversal/description/>

Solution: <https://leetcode.com/problems/binary-tree-postorder-traversal/post-solution/?submissionId=1387393439>



Time Complexity: O(n)

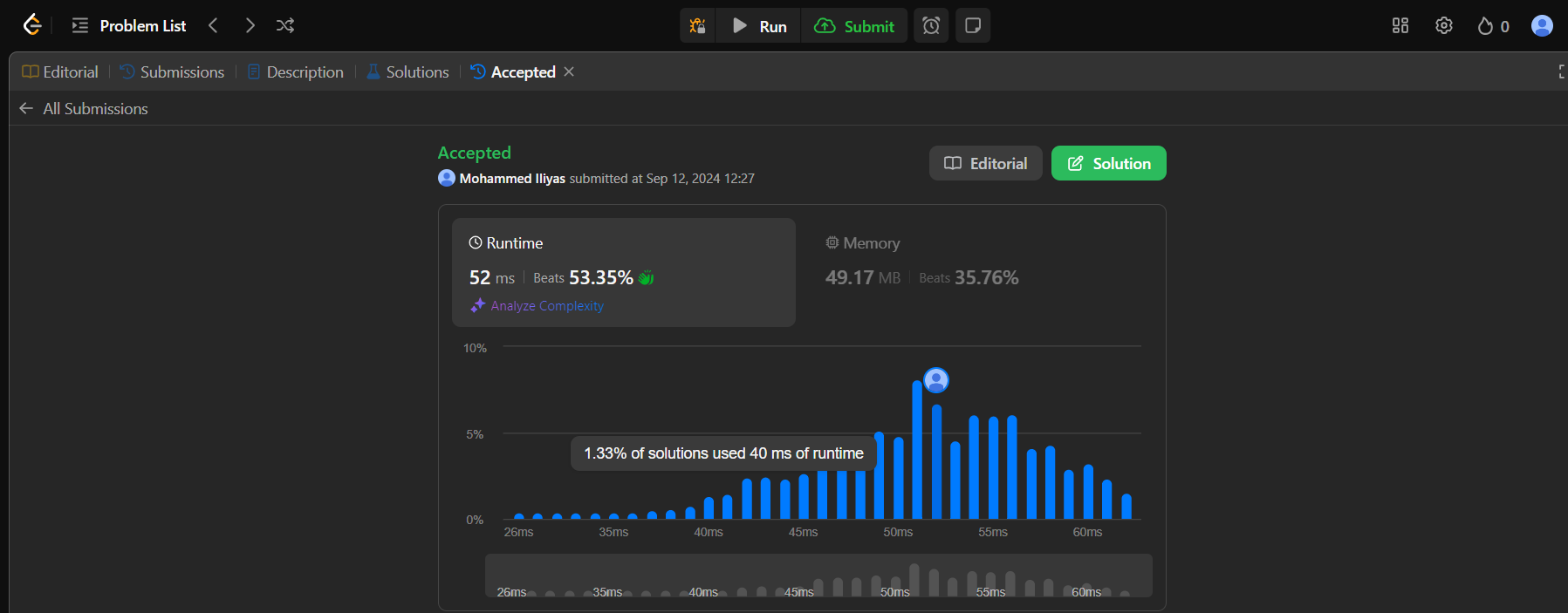
Each node in the tree is visited exactly once, making the time complexity proportional to the number of nodes, N

Space Complexity: O(n)

n the worst case (for an unbalanced tree), the recursive call stack can hold up to N function calls, where N is the depth of the tree. Additionally, the space required for the result array is also O(N), as it stores the values of all the nodes.

1. Question: <https://leetcode.com/problems/binary-tree-preorder-traversal/description/>

Solution: <https://leetcode.com/problems/binary-tree-preorder-traversal/post-solution/?submissionId=1387397838>



Time Complexity: O(n)

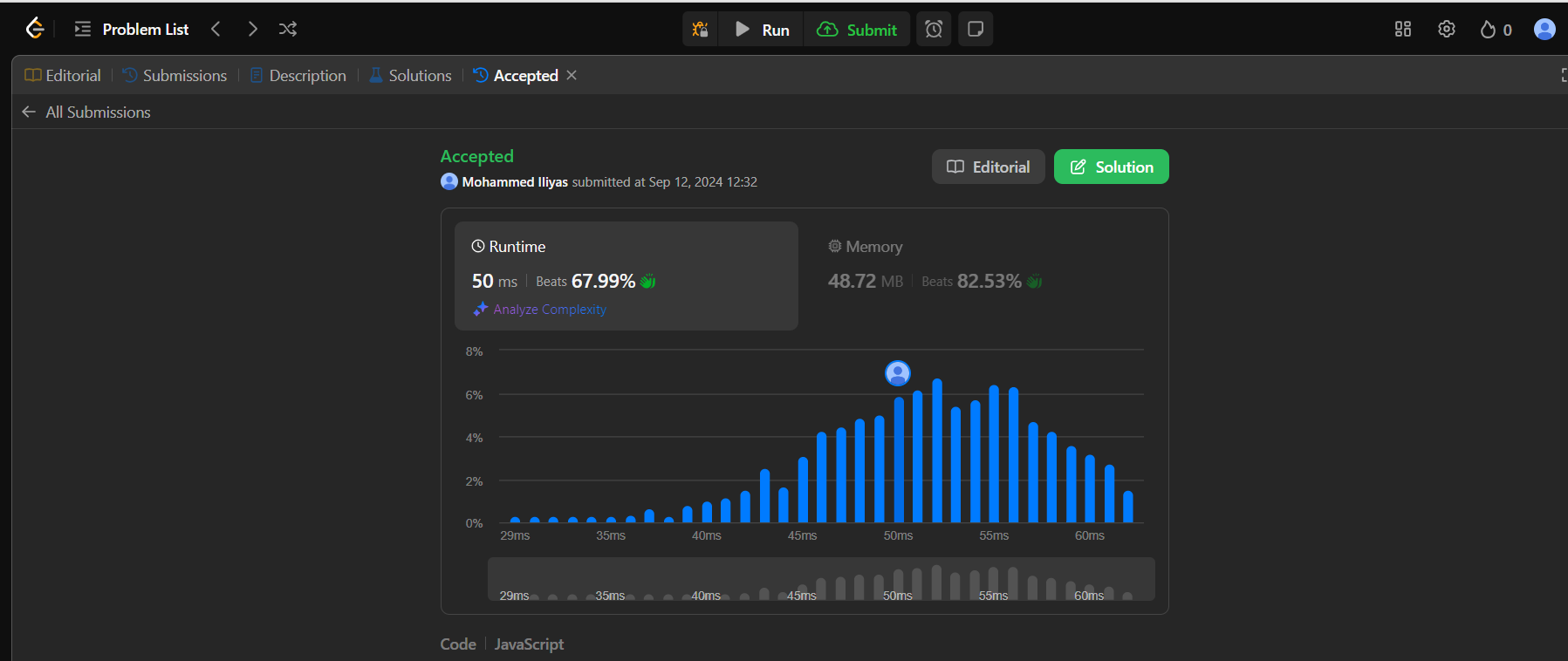
O(n), where (N) is the number of nodes in the tree. Each node is visited exactly once during the traversal.

Space Complexity: O(n)

O(n) for storing the result list res, which contains the node values

1. Question: <https://leetcode.com/problems/binary-tree-inorder-traversal/description/>

Solution: <https://leetcode.com/problems/binary-tree-inorder-traversal/post-solution/?submissionId=1387402359>



Time Complexity: O(n)

 O(n), where n is the number of nodes in the binary tree. We visit each node once.

Space Complexity: O (log n)

O(h), where h is the height of the binary tree. The space is used for the recursive call stack, and in the worst case (skewed tree), it is O(n). In the average case (balanced tree), it is

O (logn).