

# LeetCode Bootcamp Week 6

```
class Solution:
    def lowestCommonAncestor(self, root: 'TreeNode', p: 'TreeNode', q:
'TreeNode') -> 'TreeNode':
        if root == p or root == q:
            return root

        left = right = None
        # else look in left and right child
        if root.left:
            left = self.lowestCommonAncestor(root.left, p, q)
        if root.right:
            right = self.lowestCommonAncestor(root.right, p, q)

        if left and right:
            return root
        else:
            return left or right
```

The screenshot displays the LeetCode submission interface for the 'Lowest Common Ancestor of a Binary Tree' problem. The interface is divided into several sections:

- Problem List:** Shows the problem name and a 'Run' button.
- Code Editor:** Contains the Python code for the solution, which is a recursive function to find the lowest common ancestor.
- Testcase:** Shows the input for the test case: `root = [3,5,1,6,2,0,8,null,null,7,4]`, `p = 5`, and `q = 1`.
- Test Result:** Shows the result of the test case: 'Accepted' with a runtime of 40 ms.
- Runtime and Memory:** Shows the runtime as 39 ms (Beats 98.99%) and memory as 22.20 MB (Beats 21.61%).
- Graph:** A bar chart showing the distribution of runtime times for other submissions.

The code in the editor is as follows:

```
1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def lowestCommonAncestor(self, root: 'TreeNode', p: 'TreeNode', q: 'TreeNode') -> 'TreeNode':
10         if root == p or root == q:
11             return root
12
13         left = right = None
14         # else look in left and right child
15         if root.left:
```

```

class Solution:
    def topKFrequent(self, nums: List[int], k: int) -> List[int]:
        count = Counter(nums)
        sort_count_keys = sorted(count.keys(), key=lambda x: count[x])
        return sort_count_keys[-k:]

```

Description

Accepted

Editorial

Solutions

Submissions

All Submissions

Accepted 32 / 32 testcases passed

XoITACHIxo submitted at Apr 29, 2025 11:59

Editorial

Solution

Runtime

39 ms | Beat: 98.99%

Analyze Complexity

Memory

22.20 MB | Beat: 21.61%

Code | Python3

```

# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, x):
#         self.val = x
#         self.left = None
#         self.right = None

```

Code

Python3

Auto

```

1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, x):
4 #         self.val = x
5 #         self.left = None
6 #         self.right = None
7
8 class Solution:
9     def lowestCommonAncestor(self, root: 'TreeNode', p: 'TreeNode', q: 'TreeNode') -> 'TreeNode':
10         if root == p or root == q:
11             return root
12
13         left = right = None
14         # else look in left and right child
15         if root.left:

```

Testcase

Test Result

Accepted Runtime: 40 ms

Case 1 Case 2 Case 3

Input

root =

[3,5,1,6,2,0,8,null,null,7,4]

p =

5

q =

1

Output