L11. Subset Sum II | Leetcode | Recursion

SDE-PROBLEMS

Day9 (Recursion):

1. **Combination sum-1**
   https://www.youtube.com/watch?v=OyZFFqQtu98&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=49

2. **Combination sum-2**
   https://www.youtube.com/watch?v=G1fRTGRxXU8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=50

3. **Palindrome Partitioning**
   https://www.youtube.com/watch?v=WBgsABoCIE0&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=51

4. **Subset Sums**
   https://www.youtube.com/watch?v=rYkfBRtMJr8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=52

5. **Subset Sum-Ii**

6. K-th permutation Sequence

Day10: (Backtracking)

1. N queens Problem
2. Sudoko
3. M coloring Problem (Graph prob)
4. Rat in a Maze

0:13 / 30:15

L11. Subset Sum II | Leetcode | Recursion

SDE-PROBLEMS

Day9 (Recursion):

1. **Combination sum-1**
   https://www.youtube.com/watch?v=OyZFFqQtu98&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=49

2. **Combination sum-2**
   https://www.youtube.com/watch?v=G1fRTGRxXU8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=50

3. **Palindrome Partitioning**
   https://www.youtube.com/watch?v=WBgsABoCIE0&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=51

4. **Subset Sums**
   https://www.youtube.com/watch?v=rYkfBRtMJr8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=52

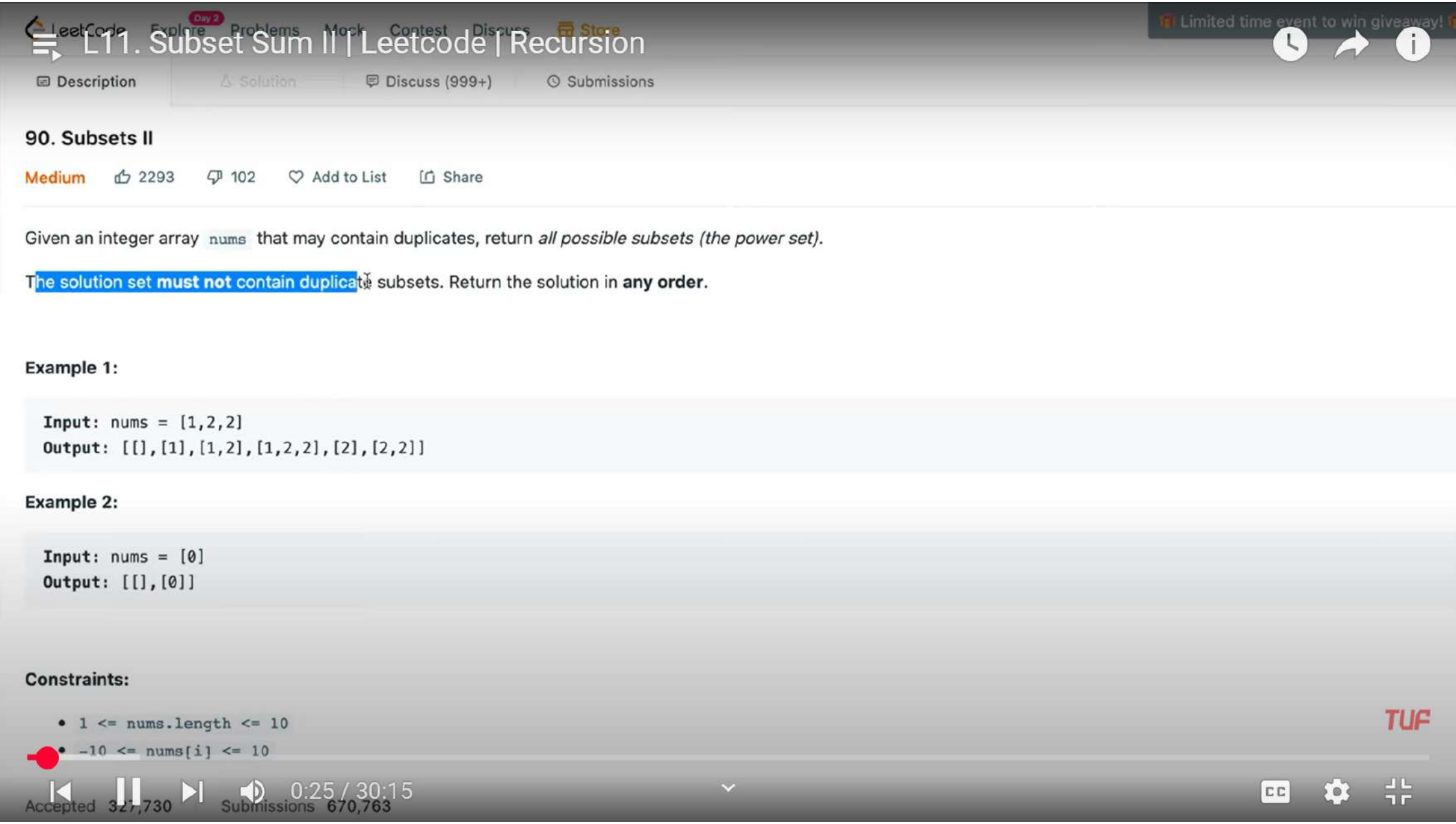
5. **Subset Sum-Ii**

6. K-th permutation Sequence

Day10: (Backtracking)

1. N queens Problem
2. Sudoko
3. M coloring Problem (Graph prob)
4. Rat in a Maze

0:09 / 30:15


LeetCode
Explore
Day 2
Problems
Mock
Contest
Discuss
Store
Limited time event to win giveaway!
L11. Subset Sum II | Leetcode | Recursion
Description
Solution
Discuss (999+)
Submissions
90. Subsets II
Medium
2293
102
Add to List
Share
Given an integer array nums that may contain duplicates, return all possible subsets (the power set).
The solution set must not contain duplicate subsets. Return the solution in any order.
Example 1:
Input: nums = [1,2,2]
Output: [[],[1],[1,2],[1,2,2],[2],[2,2]]
Example 2:
Input: nums = [0]
Output: [[],[0]]
Constraints:
1 <= nums.length <= 10
-10 <= nums[i] <= 10
TUF
0:25 / 30:15
Accepted 327,730
Submissions 670,763

90. Subsets II

Medium 2293 102 Add to List Share

Given an integer array `nums` that may contain duplicates, return *all possible subsets (the power set)*.

The solution set **must not** contain duplicate subsets. Return the solution in **any order**.

Example 1:

```
Input: nums = [1,2,2]
Output: [[],[1],[1,2],[1,2,2],[2],[2,2]]
```

Example 2:

```
Input: nums = [0]
Output: [[],[0]]
```

Constraints:

- `1 <= nums.length <= 10`
- `-10 <= nums[i] <= 10`

Accepted 327,730 Submissions 670,763

[1, 2, 2, 2, 3, 3]

[1, 2, 2, 2, 3, 3]

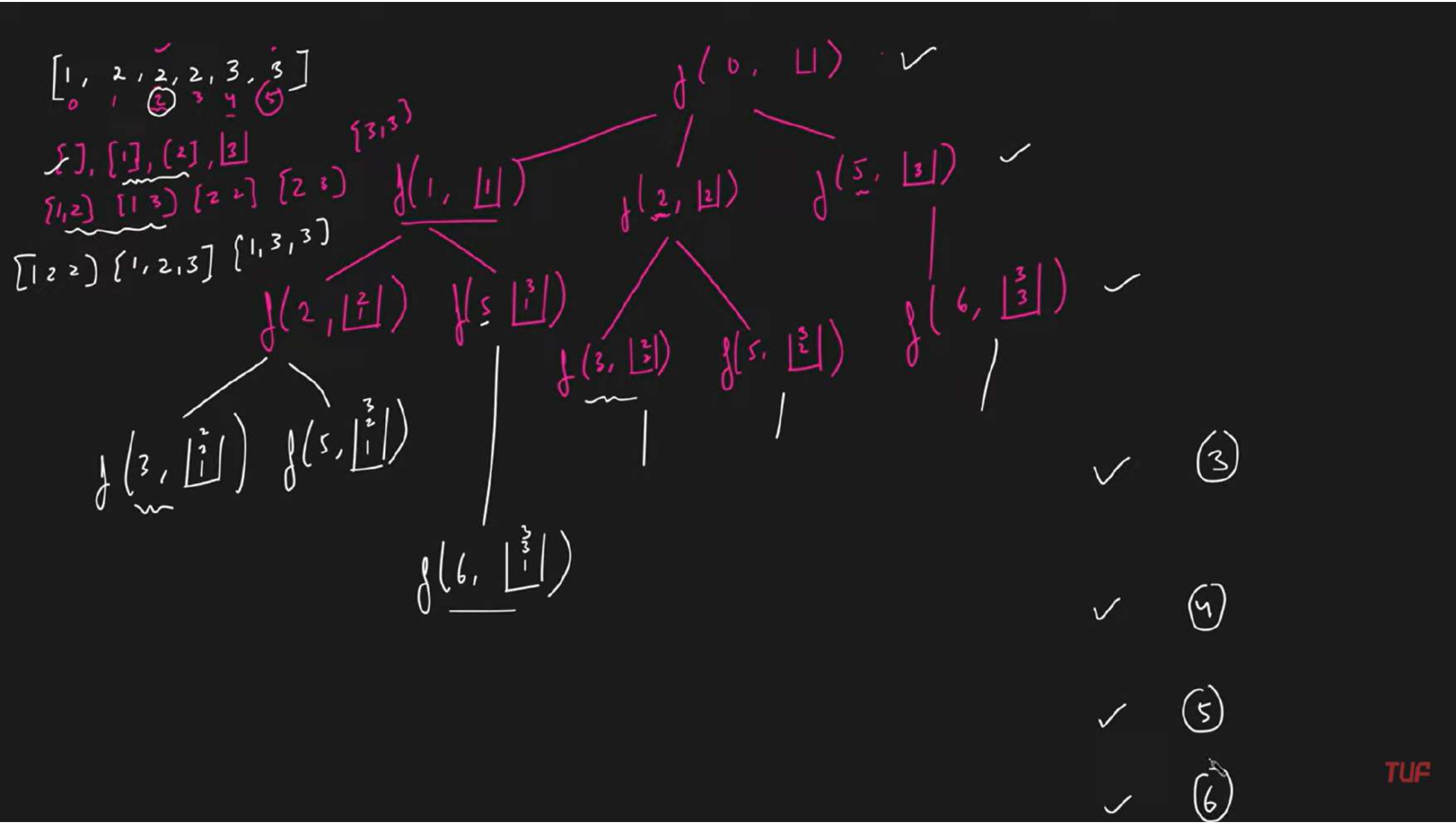
2^n ⟶ set ⟶ list<list>

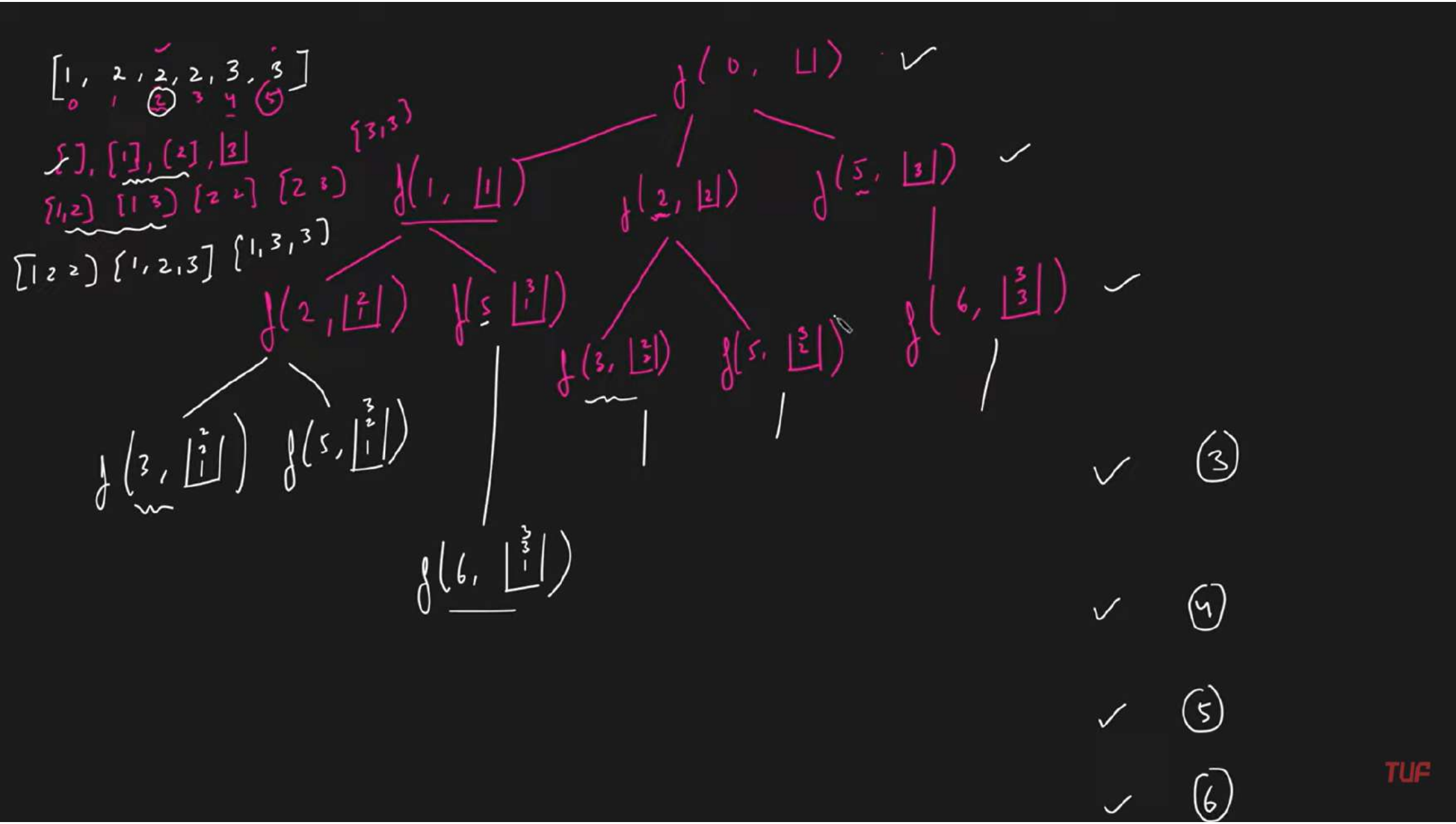
[1, 2, 2, 2, 3, 3]

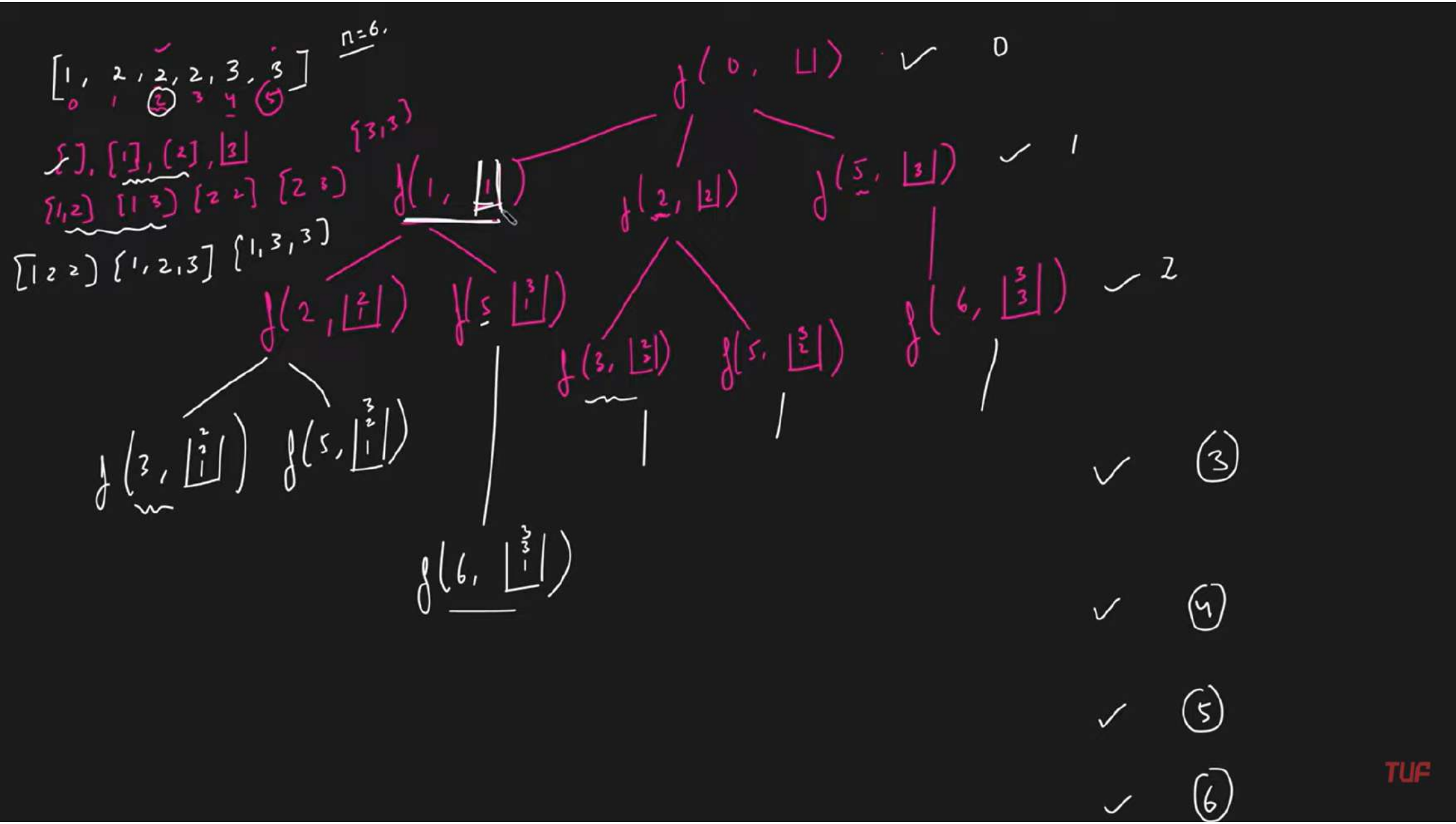
$m \times \log m$

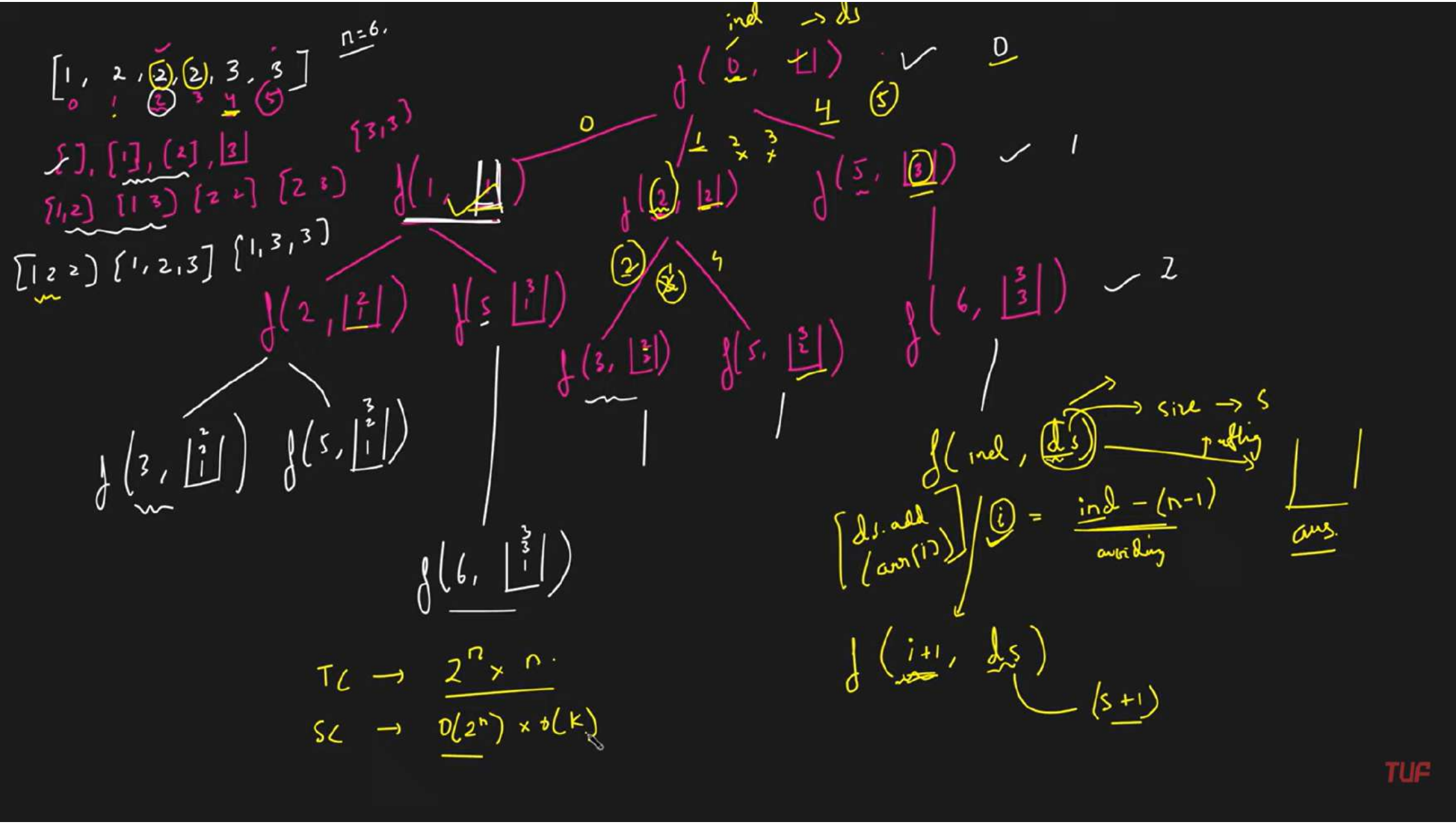
$\downarrow$

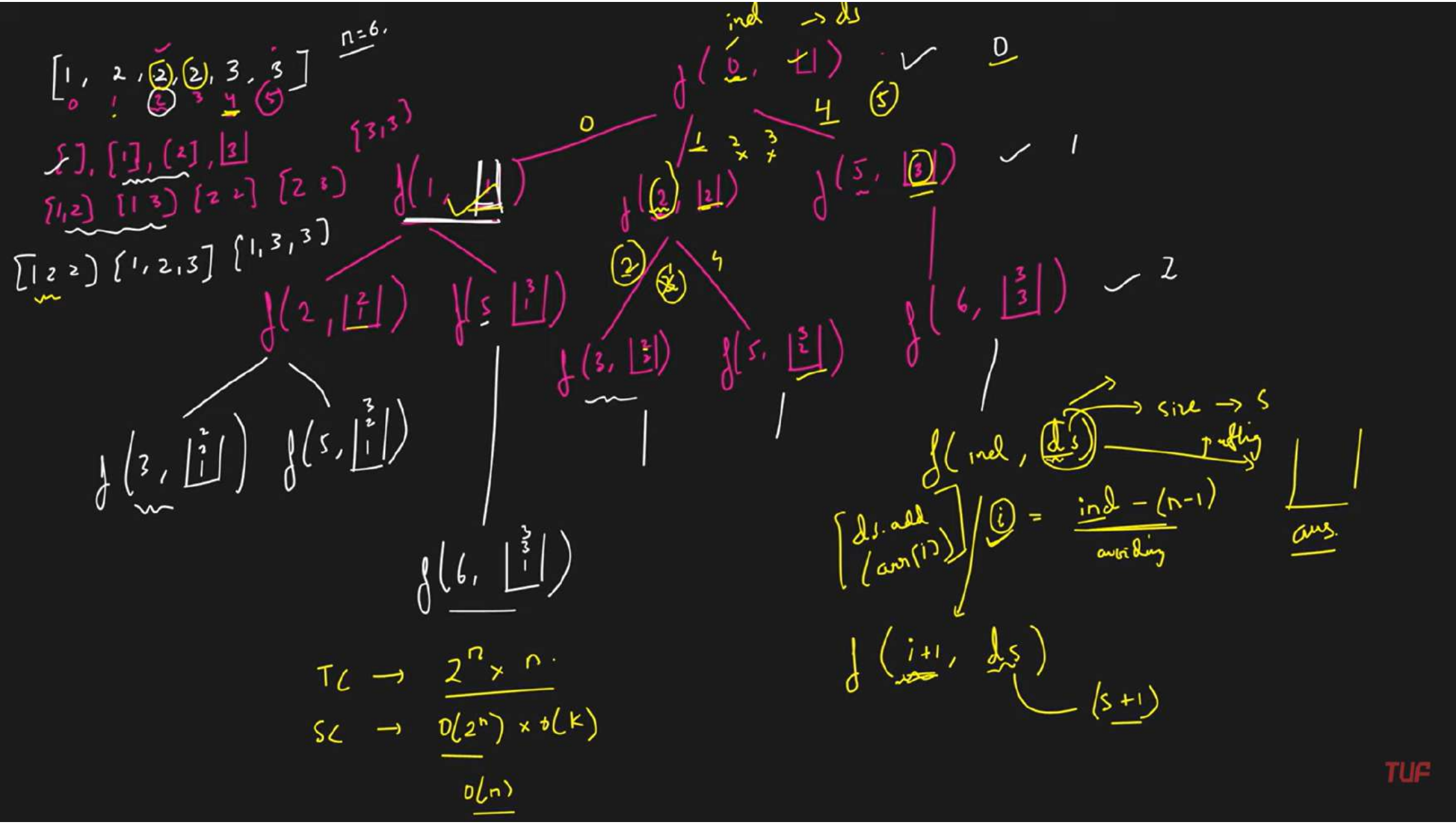
(2^n)

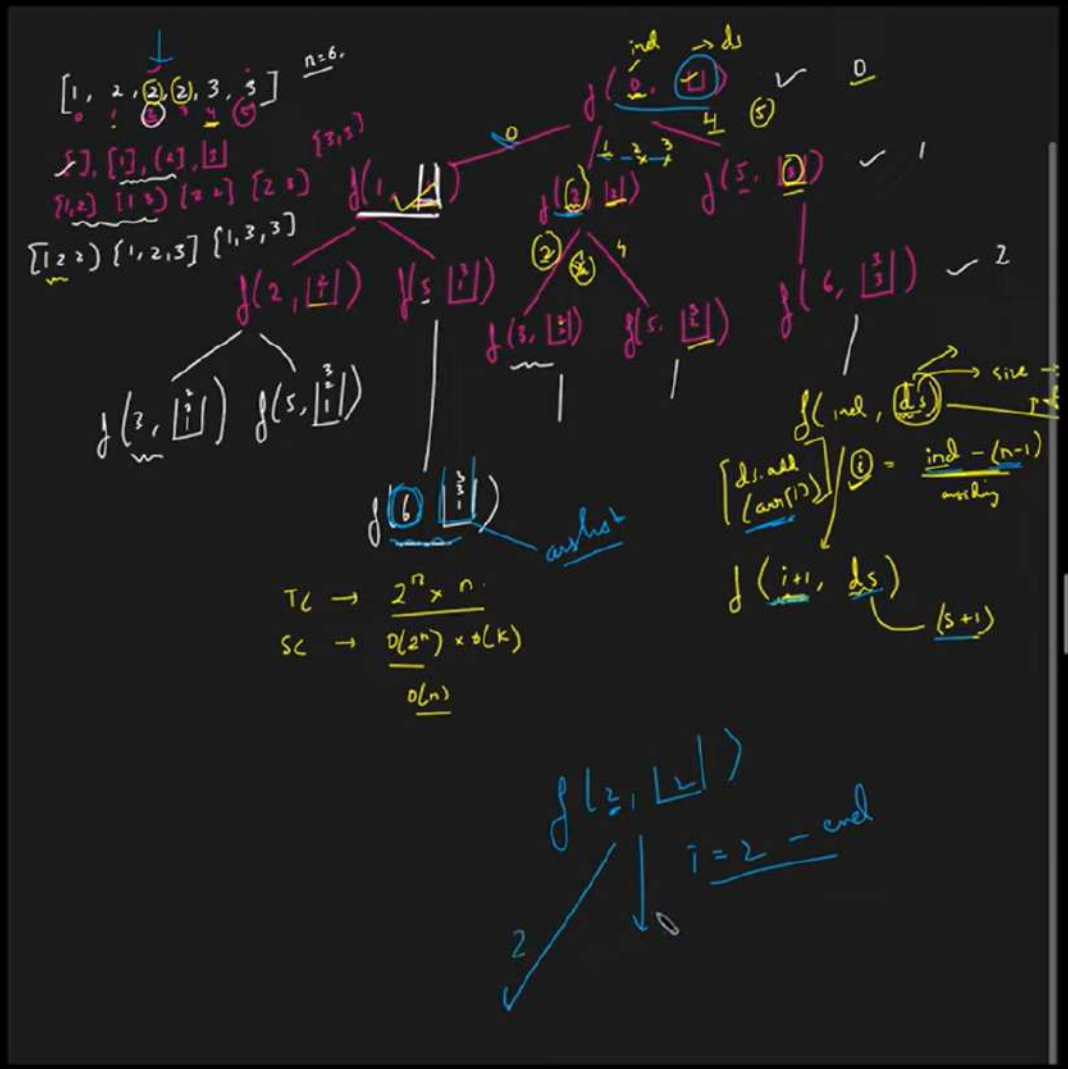

[1, 2, 2, 2, 3, 3]
0 1 2 3 4 5
[], [1], [2], [3]
[1,2] [1,3] [2,2] [2,3]
[1,2,2] [1,2,3] [1,3,3]
[3,3]
f(0, [])
f(1, [1])
f(2, [2])
f(5, [3])
f(2, [2,1])
f(5, [3,1])
f(6, [3,3])
f(3, [2,2])
f(5, [3,2])
f(3, [2,1])
f(5, [3,1,2])
f(6, [3,1,3])
(3)
(4)
(5)
(6)
TUF


[1, 2, 2, 2, 3, 3]
0 1 2 3 4 5
[], [1], [2], [3]
[1,2] [1,3] [2,2] [2,3] [3,3]
[1,2,2] [1,2,3] [1,3,3]
f(0, [])
f(1, [1])
f(2, [2])
f(5, [3])
f(2, [2,1])
f(5, [3,1])
f(3, [3])
f(5, [3,2])
f(6, [3,3])
f(3, [2,1])
f(5, [3,2,1])
f(6, [3,3,1])
(3)
(4)
(5)
(6)
TUF


[1, 2, 2, 2, 3, 3]
n=6.
[], [1], [2], [3]
[1,2] [1,3] [2,2] [2,3] [3,3]
[1,2,2] [1,2,3] [1,3,3]
f(0, [])
f(1, [1])
f(2, [2])
f(3, [3])
f(2, [1,1])
f(3, [1,2])
f(5, [1,3])
f(3, [2,1])
f(5, [2,2])
f(6, [2,3])
f(5, [1,3])
f(6, [3,3])
f(6, [1,1,3])
D
1
2
3
4
5
6


n=6.
[1, 2, 2, 2, 3, 3]
[], [1], [2], [3]
[1,2] [1,3] [2,2] [2,3] [3,3]
[1,2,2] [1,2,3] [1,3,3]
ind → ds
f(0, -1)
f(1, 1)
f(2, 2)
f(5, 3)
f(2, 1)
f(5, 1)
f(3, 2)
f(5, 2)
f(3, 1)
f(5, 1)
f(6, 3)
f(6, 1)
TC → 2^n × n.
SC → O(2^n) × O(k)
f(ind, ds)
size → s
ds.add(arr[i])
i = ind - (n-1)
ans.
f(i+1, ds)
(s+1)


n=6.
[1, 2, 2, 2, 3, 3]
[], [1], [2], [3]
[1,2] [1,3] [2,2] [2,3] [3,3]
[1,2,2] [1,2,3] [1,3,3]
ind → ds
f(0, -1)
f(1, [1])
f(2, [2])
f(5, [3])
f(2, [1,1])
f(5, [1,3])
f(3, [2,1])
f(5, [2,2])
f(6, [3,3])
f(3, [1,1,1])
f(5, [1,2,2])
f(6, [1,3,3])
f(6, [2,2,2])
TC → 2^n × n
SC → O(2^n) × O(k)
O(n)
f(ind, ds)
ds.add(arr[i])
i = ind - (n-1)
size → s
nothing
ans.
f(i+1, ds)
(s+1)



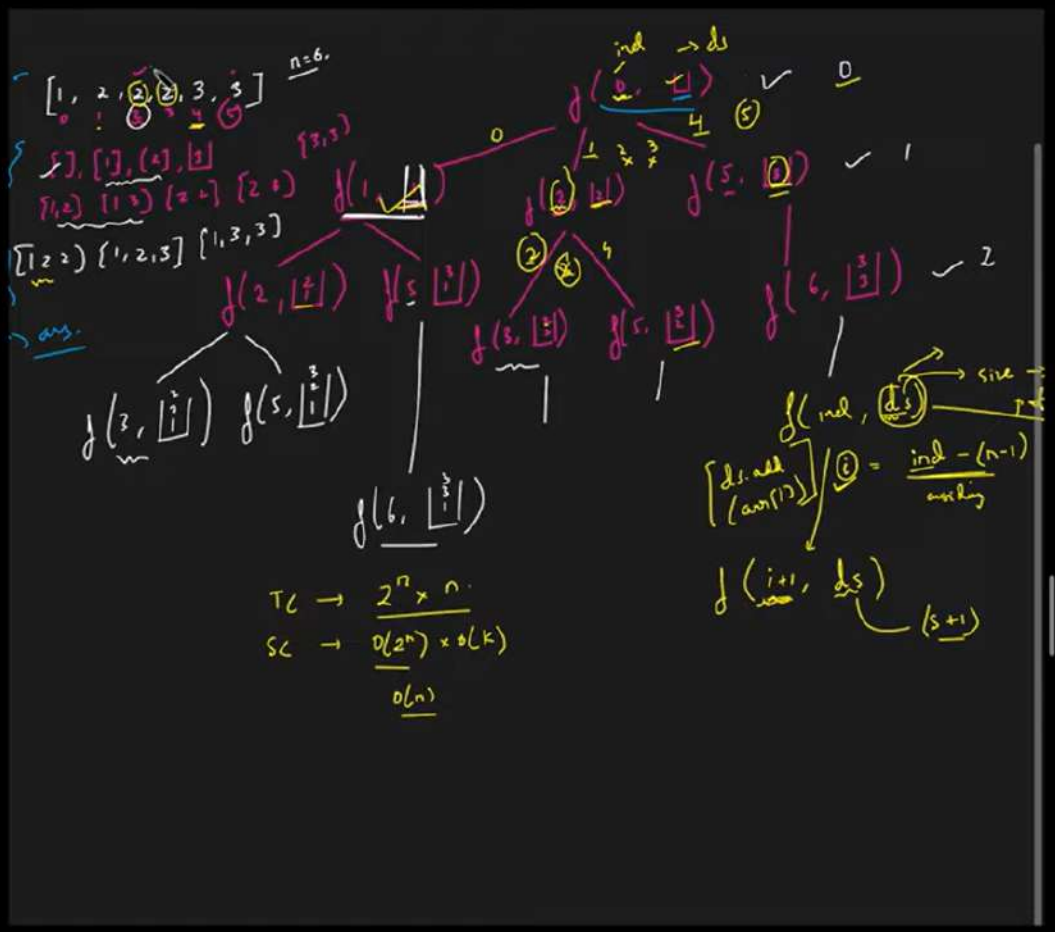
Java

Autocomplete

```java
class Solution {

    public void findSubsets(int ind, int[] nums, List<Integer> ds, List<List<Integer>
        ansList.add(new ArrayList<>(ds));
        for(int i = ind;i<nums.length;i++) {
            if(i!=ind && nums[i] == nums[i-1]) continue;
            ds.add(nums[i]);
            findSubsets(i+1, nums, ds, ansList);
            ds.remove(ds.size() - 1);
        }

    }
    public List<List<Integer>> subsetsWithDup(int[] nums) {
        Arrays.sort(nums);
        List<List<Integer>> ansList = new ArrayList<>();
        findSubsets(0, nums, new ArrayList<>(), ansList);
        return ansList;
    }
}
```



C++ Autocomplete

```cpp
class Solution {
private:
    void findSubsets(int ind, vector<int> &nums, vector<int> &ds, vector<vector<int>>
        ans.push_back(ds);
        for(int i = ind;i<nums.size();i++) {
            if(i!=ind && nums[i] == nums[i-1]) continue;
            ds.push_back(nums[i]);
            findSubsets(i+1, nums, ds, ans);
            ds.pop_back();
        }
    }
public:
    vector<vector<int>> subsetsWithDup(vector<int>& nums) {
        vector<vector<int>> ans;
        vector<int> ds;
        sort(nums.begin(), nums.end());
        findSubsets(0, nums, ds, ans);
        return ans;
    }
};
```