1. Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.

You must write an algorithm with O(log n) runtime complexity.

Example 1:

Input: nums = [1,3,5,6], target = 5

Output: 2

Example 2:

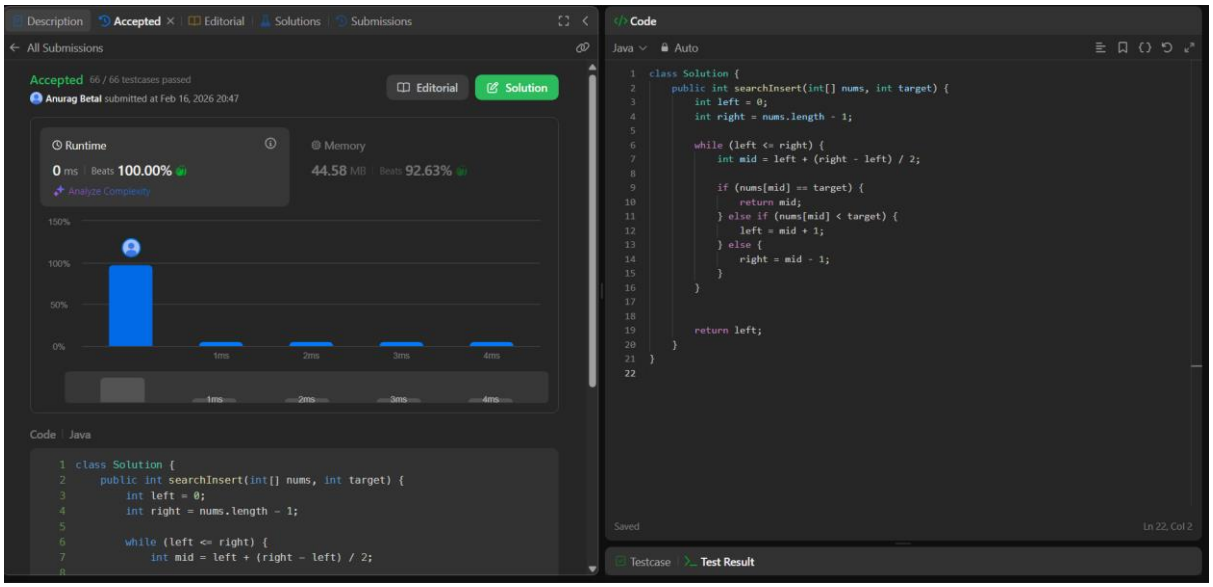
Input: nums = [1,3,5,6], target = 2

Output: 1

Example 3:

Input: nums = [1,3,5,6], target = 7

Output: 4



2. Given an array of distinct integers candidates and a target integer target, return a list of all unique combinations of candidates where the chosen numbers sum to target. You may return the combinations in any order.

The same number may be chosen from candidates an unlimited number of times.

Two combinations are unique if the frequency of at least one of the chosen numbers is different.

The test cases are generated such that the number of unique combinations that sum up to target is less than 150 combinations for the given input.

Example 1:

Input: candidates = [2,3,6,7], target = 7

Output: [[2,2,3],[7]]

Explanation:

2 and 3 are candidates, and 2 + 2 + 3 = 7. Note that 2 can be used multiple times.

7 is a candidate, and 7 = 7.

These are the only two combinations.

Example 2:

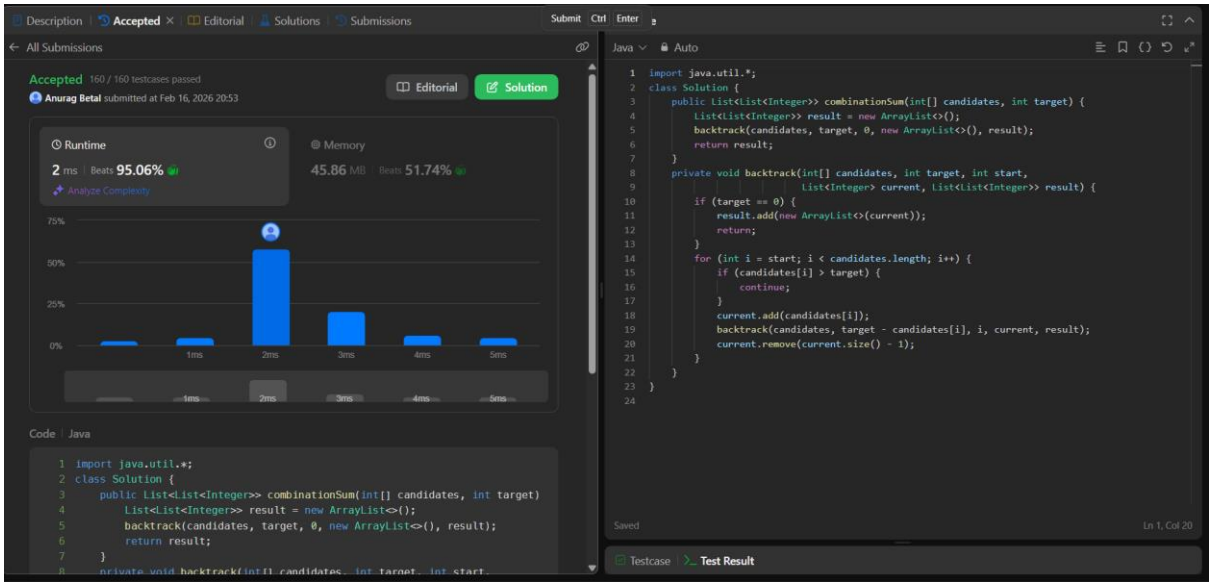
Input: candidates = [2,3,5], target = 8

Output: [[2,2,2,2],[2,3,3],[3,5]]

Example 3:

Input: candidates = [2], target = 1

Output: []



3. Given a collection of candidate numbers (candidates) and a target number (target), find all unique combinations in candidates where the candidate numbers sum to target.

Each number in candidates may only be used once in the combination.

Note: The solution set must not contain duplicate combinations.

Example 1:

Input: candidates = [10,1,2,7,6,1,5], target = 8

Output:

[

[1,1,6],

[1,2,5],

[1,7],

[2,6]

]

Example 2:

Input: candidates = [2,5,2,1,2], target = 5

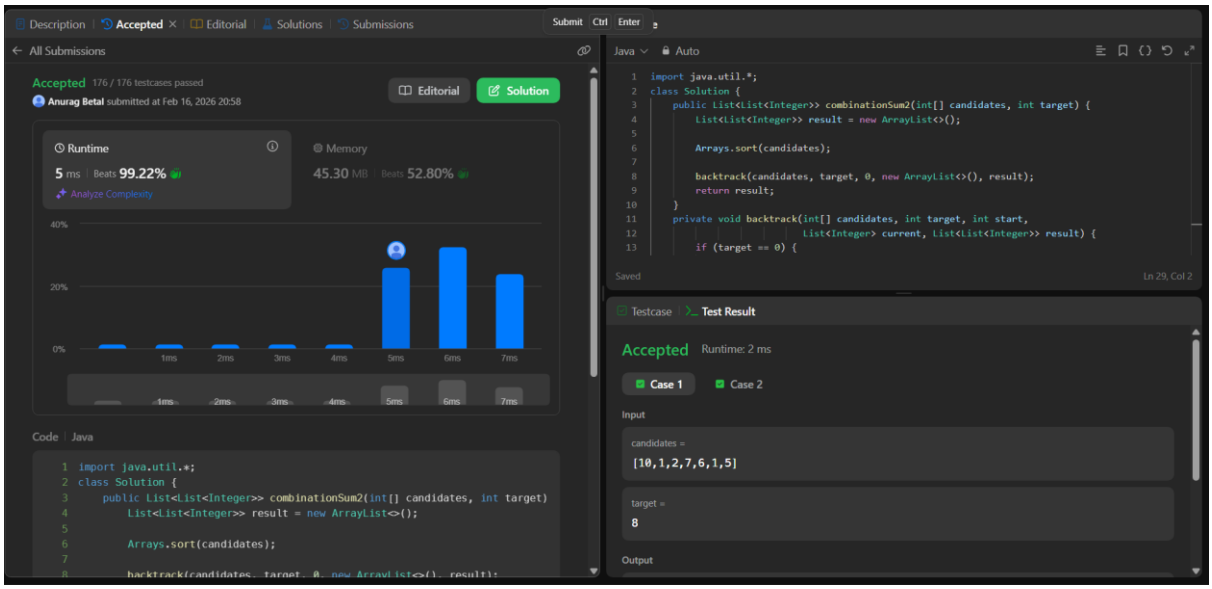
Output:

[

[1,2,2],

[5]

]



4. You are given a 0-indexed array of integers nums of length n. You are initially

positioned at index 0.

Each element nums[i] represents the maximum length of a forward jump from index i.

In other words, if you are at index i, you can jump to any index (i + j) where:

• 0 <= j <= nums[i] and

• i + j < n

Return the minimum number of jumps to reach index n - 1. The test cases are generated

such that you can reach index n - 1.

Example 1:

Input: nums = [2,3,1,1,4]

Output: 2

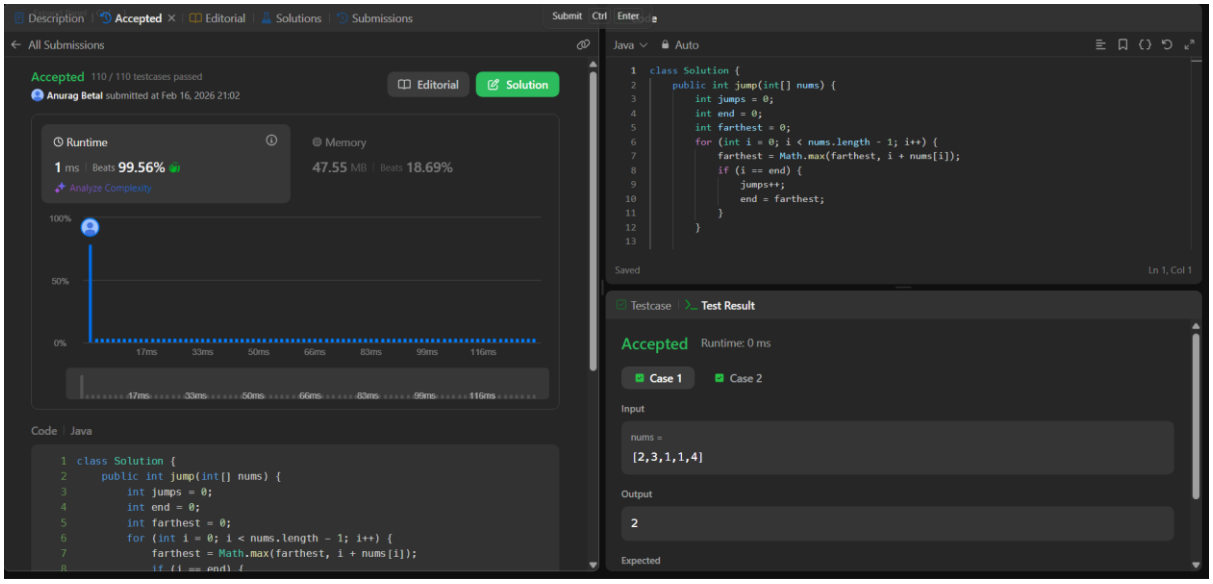
Explanation: The minimum number of jumps to reach the last index is 2. Jump 1 step

from index 0 to 1, then 3 steps to the last index.

Example 2:

Input: nums = [2,3,0,1,4]

Output: 2



5. Given an array of strings strs, group the anagrams together. You can return the answer

in any order.

Example 1:

Input: strs = ["eat","tea","tan","ate","nat","bat"]

Output: [["bat"],["nat","tan"],["ate","eat","tea"]]

Explanation:

• There is no string in strs that can be rearranged to form "bat".

• The strings "nat" and "tan" are anagrams as they can be rearranged to form each other.

• The strings "ate", "eat", and "tea" are anagrams as they can be rearranged to form each other.

Example 2:

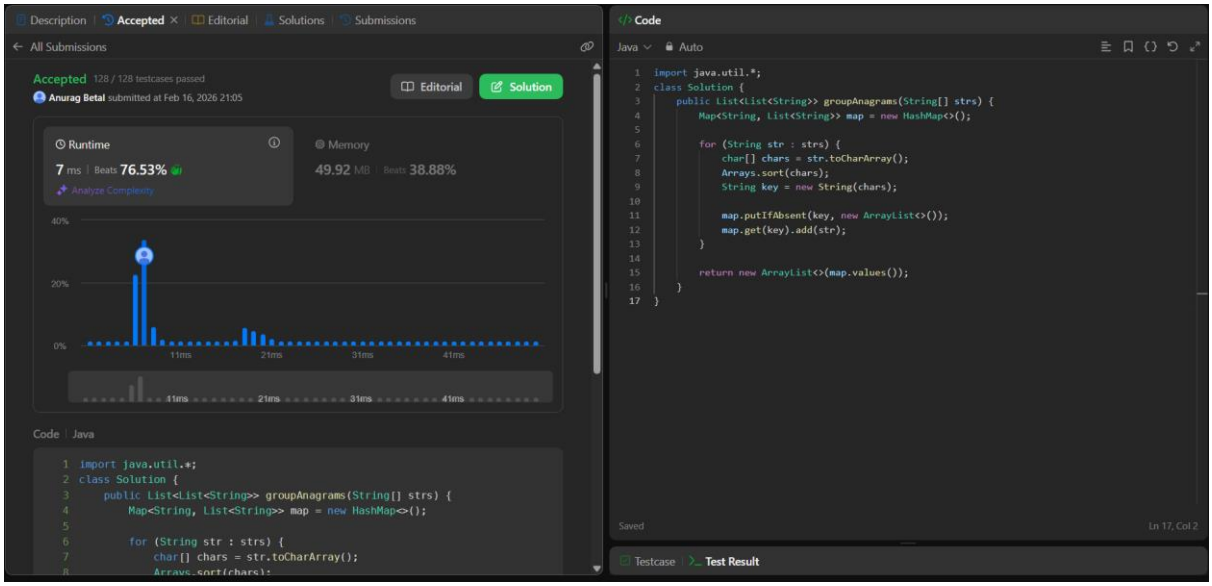
Input: strs = [""]

Output: [[""]]

Example 3:

Input: strs = ["a"]

Output: [["a"]]



6. You are given a large integer represented as an integer array digits, where each digits[i] is the i

th digit of the integer. The digits are ordered from most significant to least significant in left-to-right order. The large integer does not contain any

leading 0's.

Increment the large integer by one and return the resulting array of digits.

Example 1:

Input: digits = [1,2,3]

Output: [1,2,4]

Explanation: The array represents the integer 123.

Incrementing by one gives 123 + 1 = 124.

Thus, the result should be [1,2,4].

Example 2:

Input: digits = [4,3,2,1]

Output: [4,3,2,2]

Explanation: The array represents the integer 4321.

Incrementing by one gives 4321 + 1 = 4322.

Thus, the result should be [4,3,2,2].

Example 3:

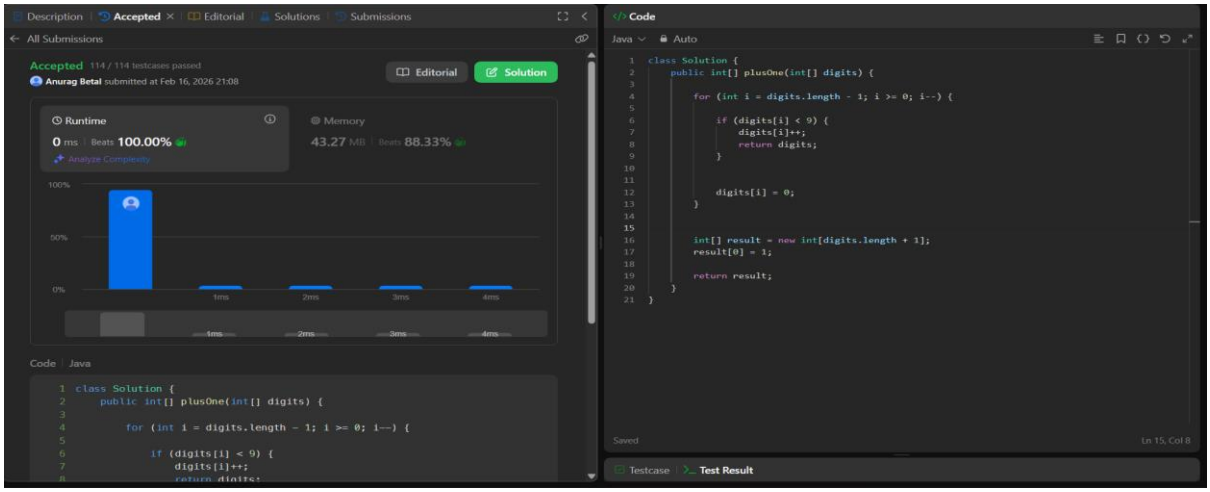
Input: digits = [9]

Output: [1,0]

Explanation: The array represents the integer 9.

Incrementing by one gives 9 + 1 = 10.

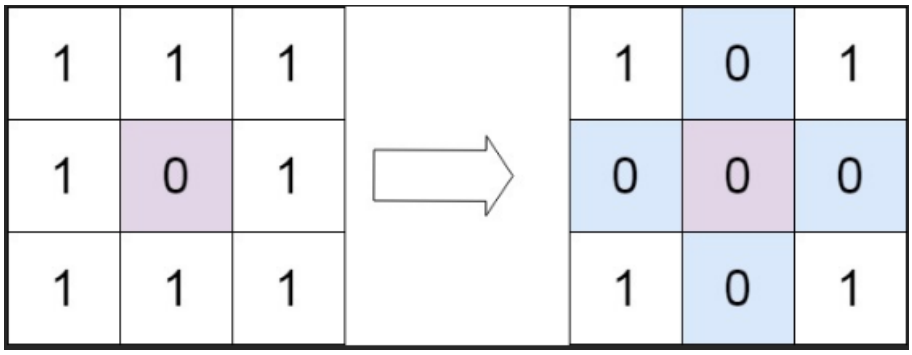
Thus, the result should be [1,0]



7. Given an m x n integer matrix matrix, if an element is 0, set its entire row and column to 0's.

You must do it in place.

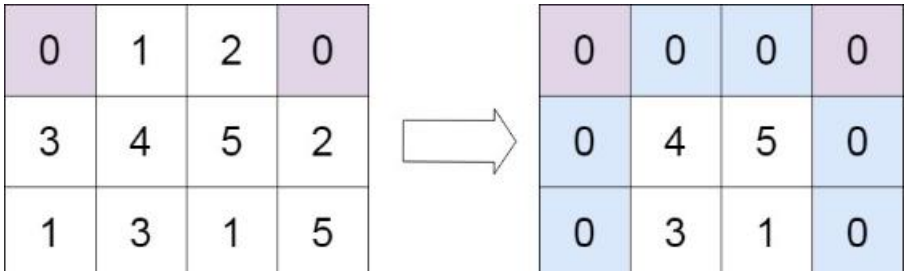
Example 1:



Input: matrix = [[1,1,1],[1,0,1],[1,1,1]]

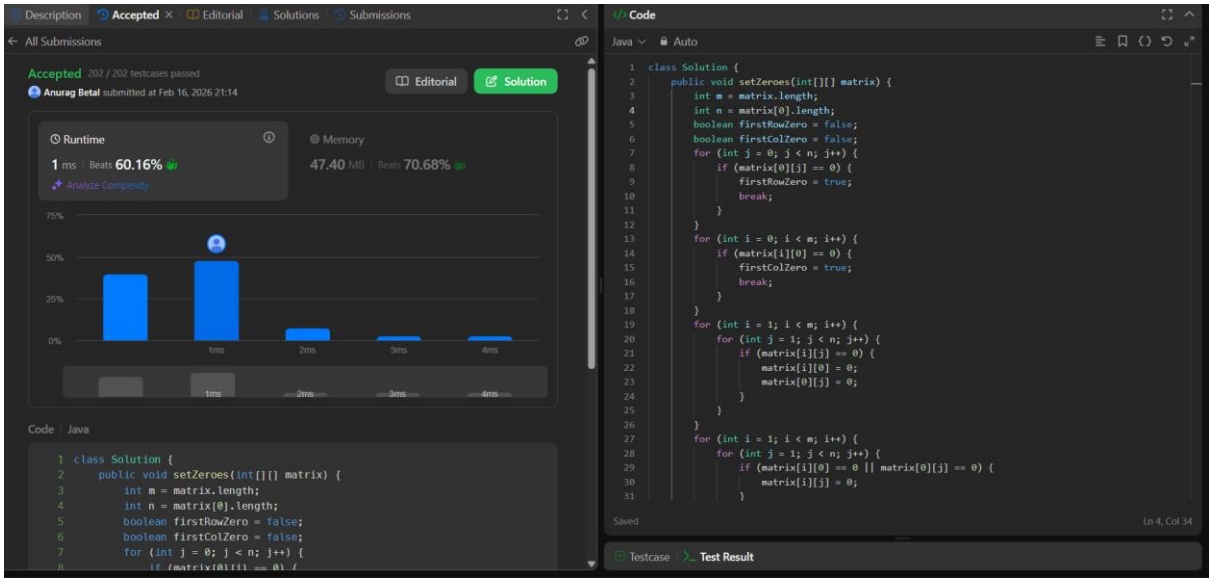
Output: [[1,0,1],[0,0,0],[1,0,1]]

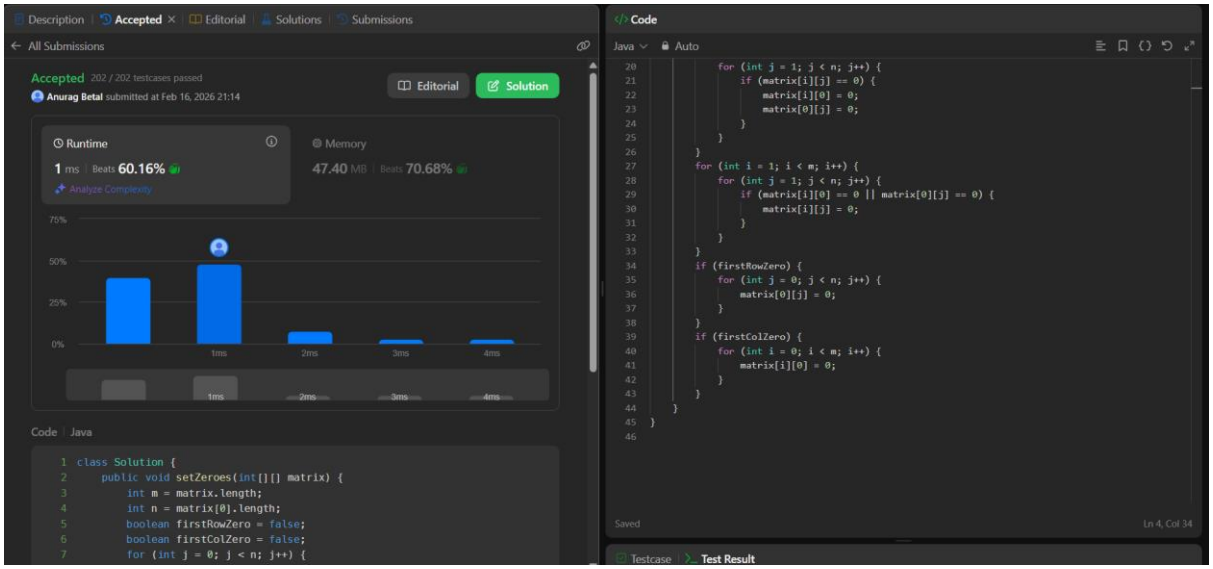
Example 2:



Input: matrix = [[0,1,2,0],[3,4,5,2],[1,3,1,5]]

Output: [[0,0,0,0],[0,4,5,0],[0,3,1,0]]





8. Given an array nums with n objects colored red, white, or blue, sort them in-place so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively.

You must solve this problem without using the library's sort function.

Example 1:

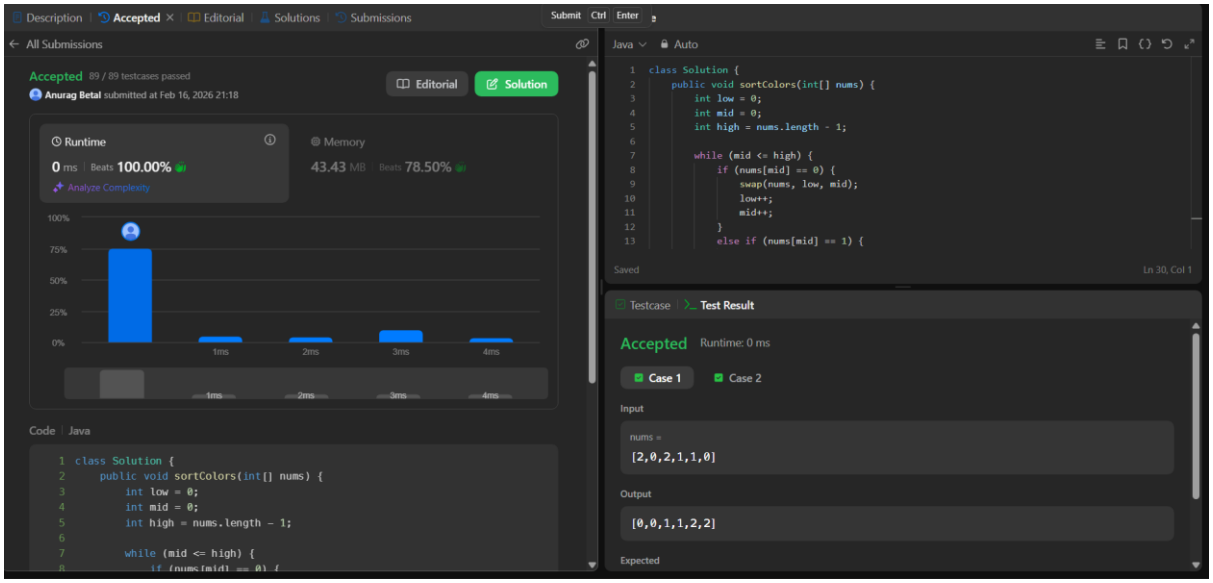
Input: nums = [2,0,2,1,1,0]

Output: [0,0,1,1,2,2]

Example 2:

Input: nums = [2,0,1]

Output: [0,1,2]



9. Given an integer array nums of unique elements, return all possible subsets (the power set).

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

Example 1:

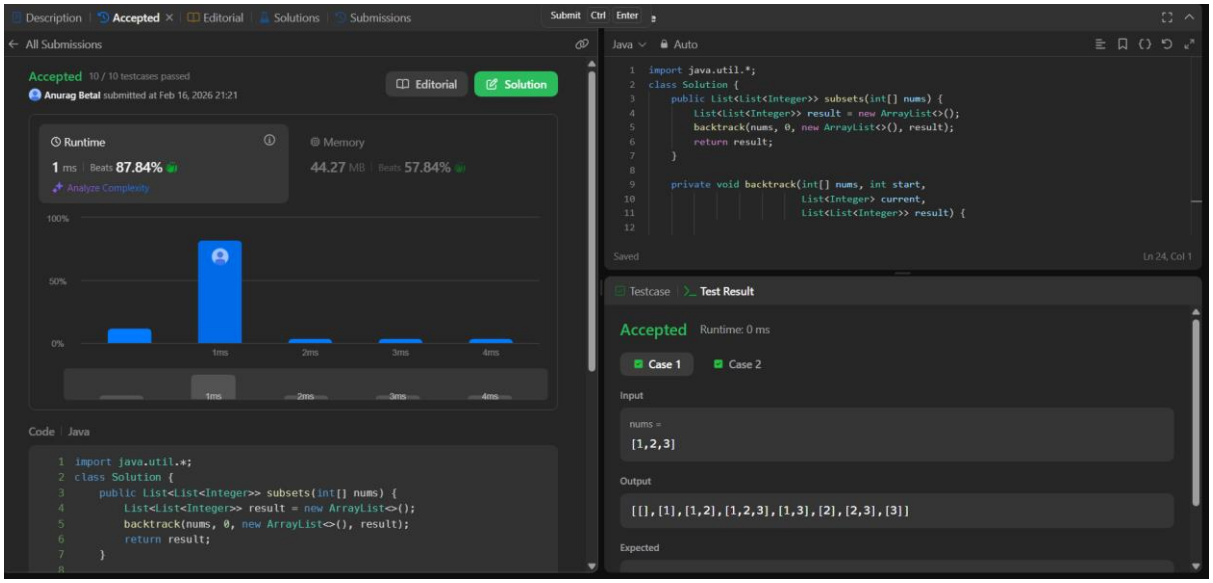
Input: nums = [1,2,3]

Output: [[],[1],[2],[1,2],[3],[1,3],[2,3],[1,2,3]]

Example 2:

Input: nums = [0]

Output: [[],[0]]

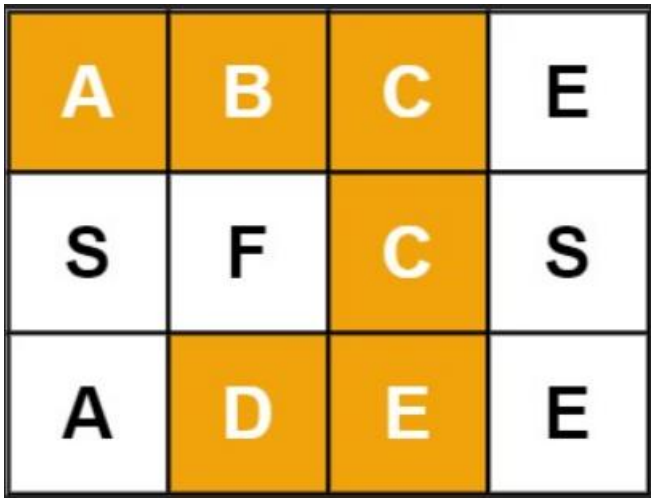


10. Given an m x n grid of characters board and a string word, return true if word exists in

the grid.

The word can be constructed from letters of sequentially adjacent cells, where adjacent cells are horizontally or vertically neighboring. The same letter cell may not be used more than once.

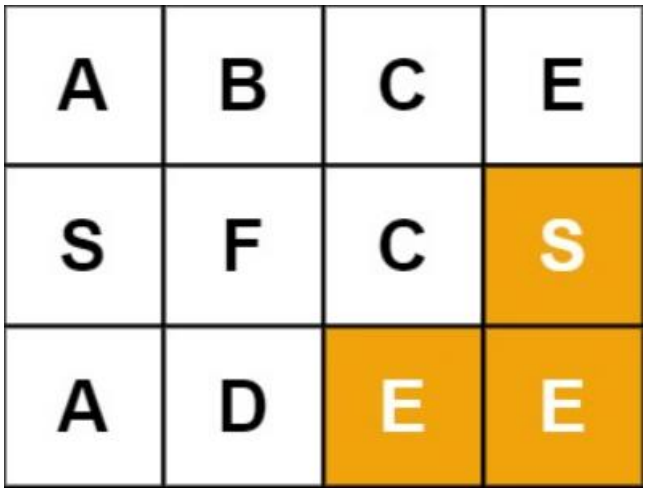
Example 1:



Input: board = [["A","B","C","E"],["S","F","C","S"],["A","D","E","E"]], word = "ABCCED"

Output: true

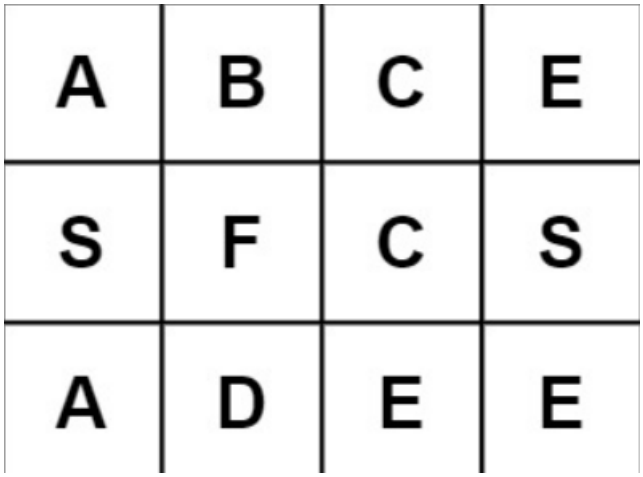
Example 2:



Input: board = [["A","B","C","E"],["S","F","C","S"],["A","D","E","E"]], word = "SEE"

Output: true

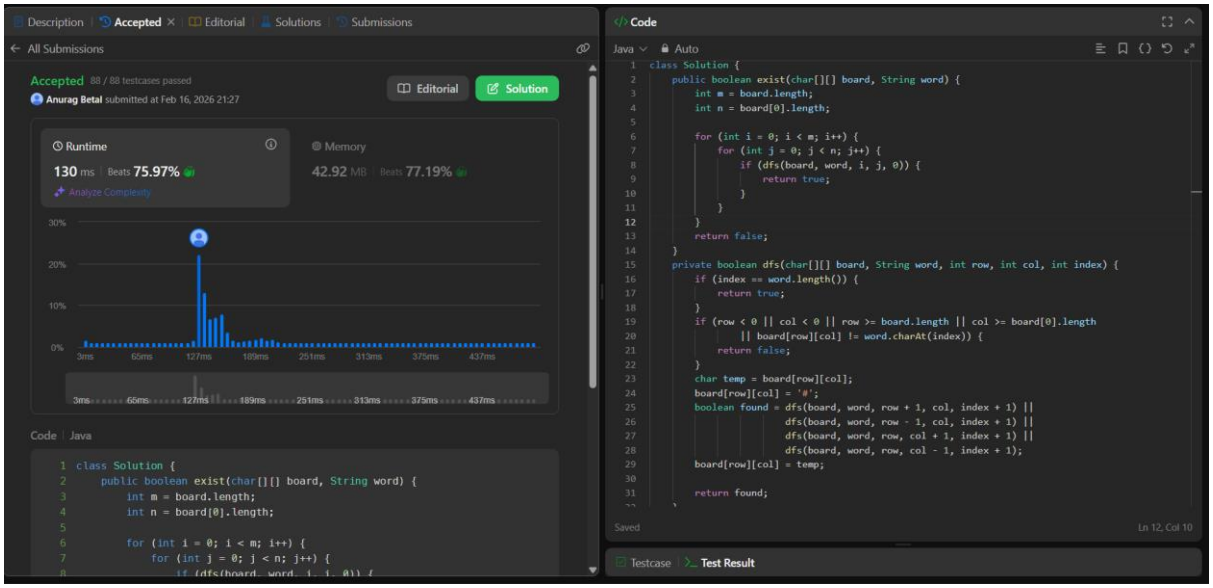
Example 3:



Input: board = [["A","B","C","E"],["S","F","C","S"],["A","D","E","E"]], word =

"ABCB"

Output: false




```java
class Solution {
    public boolean exist(char[][] board, String word) {
        int m = board.length;
        int n = board[0].length;

        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                if (dfs(board, word, i, j, 0)) {
                    return true;
                }
            }
        }
        return false;
    }
    private boolean dfs(char[][] board, String word, int row, int col, int index) {
        if (index == word.length()) {
            return true;
        }
        if (row < 0 || col < 0 || row >= board.length || col >= board[0].length
                || board[row][col] != word.charAt(index)) {
            return false;
        }
        char temp = board[row][col];
        board[row][col] = '#';
        boolean found = dfs(board, word, row + 1, col, index + 1) ||
                        dfs(board, word, row - 1, col, index + 1) ||
                        dfs(board, word, row, col + 1, index + 1) ||
                        dfs(board, word, row, col - 1, index + 1);
        board[row][col] = temp;

        return found;
    }
```

11. Given an array nums of n integers, return an array of all

the unique quadruplets [nums[a], nums[b], nums[c], nums[d]] such that:

• 0 <= a, b, c, d < n

• a, b, c, and d are distinct.

• nums[a] + nums[b] + nums[c] + nums[d] == target

You may return the answer in any order.

Example 1:

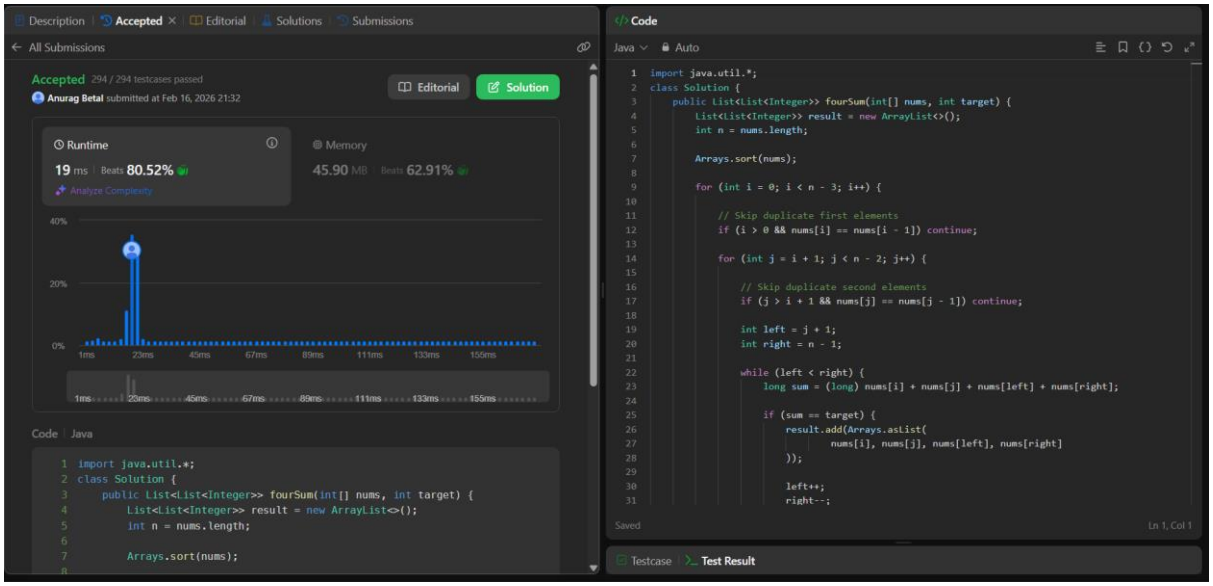
Input: nums = [1,0,-1,0,-2,2], target = 0

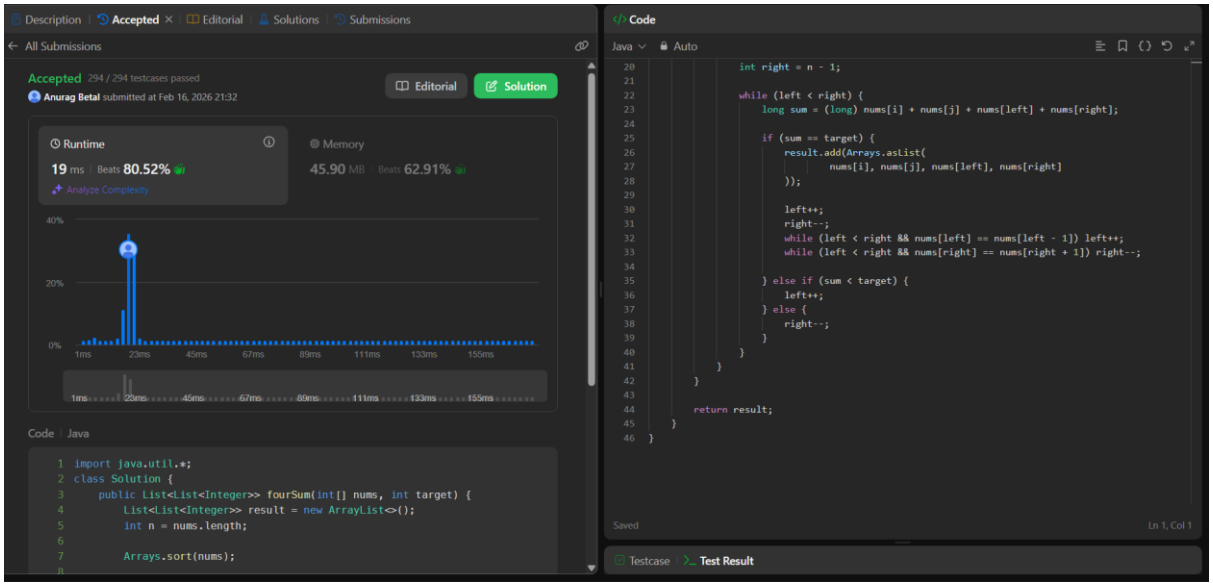
Output: [[-2,-1,1,2],[-2,0,0,2],[-1,0,0,1]]

Example 2:

Input: nums = [2,2,2,2,2], target = 8

Output: [[2,2,2,2]]





12. There is an integer array nums sorted in ascending order (with distinct values).

Prior to being passed to your function, nums is possibly left rotated at an unknown

index k (1 <= k < nums.length) such that the resulting array is [nums[k], nums[k+1], ...,

nums[n-1], nums[0], nums[1], ..., nums[k-1]] (0-indexed). For

example, [0,1,2,4,5,6,7] might be left rotated by 3 indices and become [4,5,6,7,0,1,2].

Given the array nums after the possible rotation and an integer target, return the index of target if it is in nums, or -1 if it is not in nums.

You must write an algorithm with O(log n) runtime complexity.

Example 1:

Input: nums = [4,5,6,7,0,1,2], target = 0

Output: 4

Example 2:

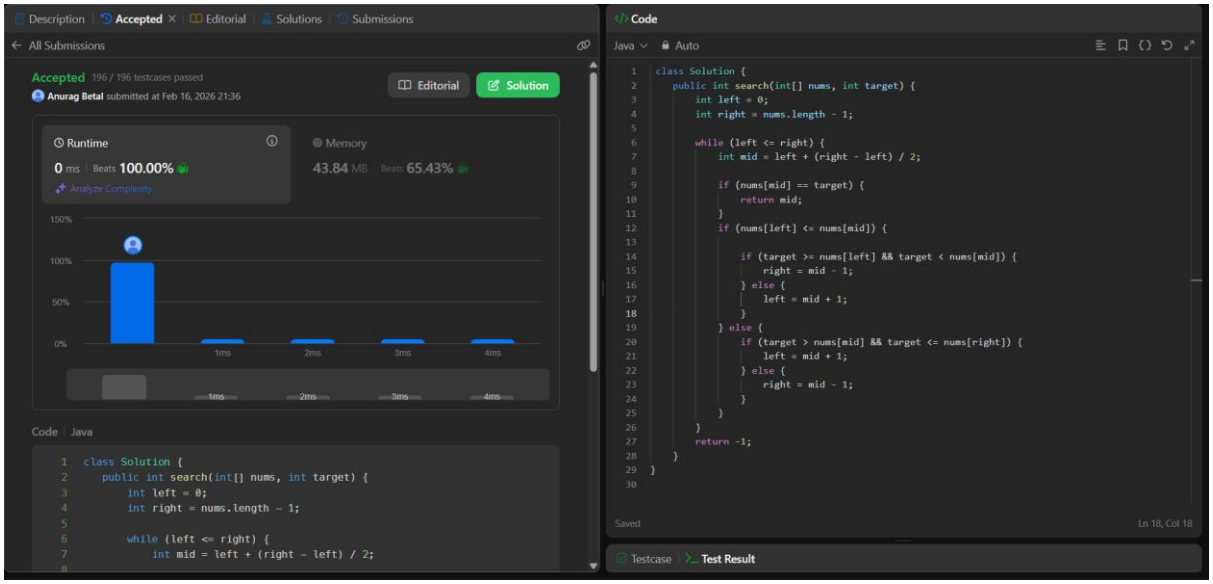
Input: nums = [4,5,6,7,0,1,2], target = 3

Output: -1

Example 3:

Input: nums = [1], target = 0

Output: -1



13. Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.

If target is not found in the array, return [-1, -1].

You must write an algorithm with O(log n) runtime complexity.

Example 1:

Input: nums = [5,7,7,8,8,10], target = 8

Output: [3,4]

Example 2:

Input: nums = [5,7,7,8,8,10], target = 6

Output: [-1,-1]

Example 3:

Input: nums = [], target = 0

Output: [-1,-1]