

Description | Accepted X | Editorial | Solutions | Submissions

All Submissions

Accepted 88 / 88 testcases passed

Ayushsharma96 submitted at Feb 15, 2026 23:07

Runtime 0 ms | Beats 100.00% Memory 48.16 MB | Beats 70.74%

Analyze Complexity

Code

Java Auto

```
24     last = mid;
25     l=mid+1;
26   }
27   else if(nums[mid]>target) r=mid-1;
28   else l= mid+1;
29 }
30
31 return new int[]{first,last};
32 }
33 }
```

Saved

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

nums = [5,7,7,8,8,10]

target = 8

```
1 class Solution {
2     public int[] searchRange(int[] nums, int target) {
```

Description Accepted | Editorial | Solutions | Submissions

Accepted 196 / 196 testcases passed  
Ayusharma96 submitted at Feb 15, 2026 23:08

Runtime: 0 ms | Beats 100.00%  
Memory: 43.94 MB | Beats 44.98%

Analyze Complexity

Code

```
Java v Auto
21 | | | start = mid + 1; // move right
22 | | | } else {
23 | | | | end = mid - 1; // move left
24 | | |
25 | | }
26 | |
27 | |
28 | |
29 | |
30 }
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

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

target = 0

```
1 class Solution {
2     public int search(int[] nums, int target) {
3         int start = 0;
4         int end = nums.length - 1;
5         while (start <= end) {
6             int mid = start + (end - start) / 2;
7             if (nums[mid] == target) {
8                 return mid;
9             } else if (nums[mid] < target) {
10                 start = mid + 1;
11             } else {
12                 end = mid - 1;
13             }
14         }
15         return -1;
16     }
17 }
```

Accepted 294 / 294 testcases passed  
Ayushsharma96 submitted at Feb 15, 2026 23:09

Editorial

Solution

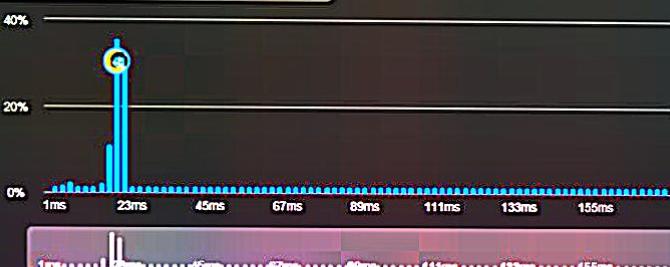
Runtime

20 ms | Beats 67.07%

Analyze Complexity

Memory

45.94 MB | Beats 46.65%



Code | Java

```
1 class Solution {  
2     public List<List<Integer>> fourSum(int[] nums, int target) {  
3         if (nums == null || nums.length < 4) {  
4             return new ArrayList<List<Integer>>();  
5         }  
6         Arrays.sort(nums);  
7         List<List<Integer>> result = new ArrayList<List<Integer>>();  
8         for (int i = 0; i < nums.length - 3; i++) {  
9             for (int j = i + 1; j < nums.length - 2; j++) {  
10                 int left = j + 1;  
11                 int right = nums.length - 1;  
12                 while (left < right) {  
13                     int sum = nums[i] + nums[j] + nums[left] + nums[right];  
14                     if (sum == target) {  
15                         result.add(Arrays.asList(nums[i], nums[j],  
16                                nums[left], nums[right]));  
17                         left++;  
18                         right--;  
19                     } else if (sum < target) {  
20                         left++;  
21                     } else {  
22                         right--;  
23                     }  
24                 }  
25             }  
26         }  
27         return result;  
28     }  
29 }
```

```
31     }  
32     }  
33     }  
34     }  
35     }  
36     }  
37     }  
38     }  
39 }
```

Saved

Ln 39, Col 2

Testcase | Test Result

Accepted Runtime: 2 ms

Case 1 Case 2

Input

nums =  
[1, 0, -1, 0, -2, 2]

target =

0

Description | Editorial | Solutions | Accepted X | Submissions

All Submissions

Accepted 88 / 88 testcases passed

Ayusharma96 submitted at Feb 15, 2026 23:12

Runtime: 155 ms | Beats 34.16% | Analyze Complexity

Memory: 43.00 MB | Beats 76.72%

Java v Auto

```
if (backtrack(board, word, visited, i + 1, j, index + 1) ||
    backtrack(board, word, visited, i - 1, j, index + 1) ||
    backtrack(board, word, visited, i, j + 1, index + 1) ||
    backtrack(board, word, visited, i, j - 1, index + 1)) {
    return true;
}
visited[i][j] = false;
return false;
}
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

board =

```
[["A","B","C","E"], ["S","F","C","S"], ["A","D","E","E"]]
```

word =

```
"ABCCED"
```

Code | Java

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

## Code

Java ▾ Auto

≡ ⌂ ⌃ ⌄ ⌅

```
14     // 1. Include the element
15     tempSets.add(nums[i]);
16
17     // 2. Move to the next element
18     backtrack(resultSets, tempSets, nums, i + 1);
19
20     // 3. Backtrack: Remove the element to try the next one
21     tempSets.remove(tempSets.size() - 1);
22
23 }
24 }
```

Saved

Ln 24, Col 2

Testcase  Test Result

Accepted Runtime: 0 ms

## </> Code

Java ▾ Auto



```
13 } high--;
14 }
15 }
16 }
17
18 private void swap(int[] nums, int i, int j) {
19     int temp = nums[i];
20     nums[i] = nums[j];
21     nums[j] = temp;
22 }
23 }
```

Saved

Ln 23, Col 2

## Testcase | Test Result

Accepted Runtime: 0 ms

Case 1

Case 2

Input

nums =

[2,0,2,1,1,0]

Output

[0,0,1,1,2,2]

← All Submissions

Accepted 128 / 128 testcases passed

Ayusharma96 submitted at Feb 15, 2026 23:17

Editorial Solution

Runtime 9 ms | Beats 36.58% Memory 50.30 MB | Beats 22.40%

Analyze Complexity

0% 20% 40%

0% 11ms 21ms 31ms 41ms

Code | Java

```
1
2
3 class Solution {
4     public String getSignature(String s) {
5         int[] count = new int[26];
```

Java Auto

```
1
2
3 class Solution {
4     public String getSignature(String s) {
5         int[] count = new int[26];
6         for (char c : s.toCharArray()) {
7             count[c - 'a']++;
8         }
9
10        StringBuilder sb = new StringBuilder();
11        for (int i = 0; i < 26; i++) {
12            if (count[i] != 0) {
```

Saved

Ln 1, Col 1

Testcase Test Result

You must run your code first

## Code

Java Auto

```
1 class Solution {
2     public int[] plusOne(int[] digits) {
3         for (int i = digits.length - 1; i >= 0; i--) {
4             if (digits[i] < 9) {
5                 digits[i]++;
6                 return digits;
7             }
8             digits[i] = 0;
9         }
10    int[] res = new int[digits.length + 1];
11    res[0] = 1;
12    return res;
```

Saved

Ln 1, Col 1

Testcase |  Test Result

You must run your code first

Java ▾ Auto

```
20     if(fr) {
21         for(int j = 0; j < matrix[0].length; j++) {
22             matrix[0][j] = 0;
23         }
24     }
25     if(fc) {
26         for(int i = 0; i < matrix.length; i++) {
27             matrix[i][0] = 0;
28         }
29     }
30 }
```

Saved

Ln 30, Col 5

Testcase |  Test Result

Accepted Runtime: 0 ms

Case 1

Case 2

Input

matrix =

[[1,1,1],[1,0,1],[1,1,1]]

Output

[[1,0,1],[0,0,0],[1,0,1]]

Description | Accepted X | Editorial | Solutions | Submissions

All Submissions

Accepted 133 / 133 testcases passed

Ayushsharma96 submitted at Feb 15, 2026 23:20

Runtime: 0 ms | Beats 100.00% Memory: 43.95 MB | Beats 61.79%

Analyze Complexity

Runtime Performance: 100% (0ms)

Memory Performance: 100% (43.95 MB)

Code | Java

```
1 class Solution {  
2     public boolean searchMatrix(int[][] matrix, int target) {  
3         int m = matrix.length;  
4         int n = matrix[0].length;  
5  
6         int left = 0;  
7         int right = m * n - 1;  
8  
9         while (left <= right) {  
10             int mid = left + (right - left) / 2;  
11             int row = mid / n;  
12             int col = mid % n;
```

Code | Auto

Java | Auto

Testcase | Test Result

You must run your code first

## 35. Search Insert Position

Solved

[Easy](#) [Topics](#) [Companies](#)

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

**Constraints:**[18.6K](#) [100](#) [D 424](#) [★](#) [🔗](#) [🕒](#)

Code

Java v Auto

```
1 //class Solution {  
2     public int searchInsert(int[] nums, int target) {  
3         int start = 0;  
4         int end = nums.length-1;  
5  
6         while (start <= end) {  
7             int mid = start + (end-start)/2;  
8             if (nums[mid] == target) return mid;  
9             else if (nums[mid] > target) end = mid-1;  
10            else start = mid+1;  
11        }  
12    }
```

Saved

[Testcase](#) | [Test Result](#)

You must run your code first

244 Online

## 39. Combination Sum

Medium [Topics](#) [Companies](#)

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]]

20.8K 222

Solved

Code

Java Auto

```
1 class Solution {
2     List<List<Integer>> ans = new ArrayList<>();
3     ArrayList<Integer> ls = new ArrayList<>();
4
5     public List<List<Integer>> combinationSum(int[] c, int target) {
6         cum(c, target, 0);
7         return ans;
8     }
9
10    public void cum(int[] c, int target, int start) {
11        if (target == 0) {
12            ans.add(new ArrayList<>(ls));
13        }
14    }
15}
```

Restored from local Upgrade to Cloud Saving

Ln 1, Col 1

Testcase  Test Result

You must run your code first

292 Online

← All Submissions

Accepted 176 / 176 testcases passed  
Aayusharma96 submitted at Feb 15, 2026 23:30

Runtime: 9 ms | Beats 16.65% | Analyze Complexity

Memory: 45.26 MB | Beats 67.41%

Java v Auto

```
1 class Solution {  
2     public List<List<Integer>> combinationSum2(int[] candidates, int target) {  
3         Arrays.sort(candidates);  
4         Set<List<Integer>> set = new HashSet<>();  
5         solve(0, candidates, target, set, new ArrayList<>());  
6         return new ArrayList<>(set);  
7     }  
8     public void solve(int idx, int[] nums, int target,  
9                         Set<List<Integer>> set,  
10                        List<Integer> temp) {  
11    }  
12}
```

Saved

Testcase | Test Result

You must run your code first

```
1 class Solution {  
2     public List<List<Integer>> combinationSum2(int[] candidates, int target)  
3     {  
4         Arrays.sort(candidates);  
5         Set<List<Integer>> set = new HashSet<>();  
6         solve(0, candidates, target, set, new ArrayList<>());  
7     }  
8     public void solve(int idx, int[] nums, int target,  
9                         Set<List<Integer>> set,  
10                        List<Integer> temp) {  
11    }  
12}
```

## Code

Java ▾ Auto



```
1 class Solution {
2     public int jump(int[] nums) {
3         int ans = 0; // number of minimum jumps taken
4         int end = 0; // end of the current jump range
5         int farthest = 0; // farthest index we can reach from current level
6
7         // we stop at nums.length - 1 because once we reach or pass it, we're done
8
9         for(int i = 0; i < nums.length - 1; ++i){
10             // update the farthest reachable index from the current position
11             farthest = Math.max(farthest, i + nums[i]);
12     }
```

Saved

Ln 1, Col 1

Testcase |  Test Result

You must run your code first