

Duplicate Integer

Easy

Given an integer array `nums`, return `true` if any value appears **more than once** in the array, otherwise return `false`.

Example 1:

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

Output: `true`

Example 2:

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

Output: `false`

Is Anagram

Easy

Given two strings `s` and `t`, return `true` if the two strings are anagrams of each other, otherwise return `false`.

An **anagram** is a string that contains the exact same characters as another string, but the order of the characters can be different.

Example 1:

```
Input: s = "racecar", t = "carrace"
```

```
Output: true
```

Example 2:

```
Input: s = "jar", t = "jam"
```

```
Output: false
```

1. Two Sum

Easy

Topics

Companies

Hint

Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to `target`*.

You may assume that each input would have **exactly one solution**, and you may not use the *same* element twice.

You can return the answer in any order.

Example 1:

Input: `nums = [2,7,11,15]`, `target = 9`

Output: `[0,1]`

Explanation: Because `nums[0] + nums[1] == 9`, we return `[0, 1]`.

Example 2:

Input: `nums = [3,2,4]`, `target = 6`

Output: `[1,2]`

Example 3:

Input: `nums = [3,3]`, `target = 6`

Output: `[0,1]`

49. Group Anagrams

Medium

Topics

Companies

Given an array of strings `strs`, group **the anagrams** together. You can return the answer in **any order**.

An **Anagram** is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

Example 1:

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

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

Example 2:

Input: `strs = [""]`

Output: `[[""]]`

Example 3:

Input: `strs = ["a"]`

Output: `[["a"]]`

347. Top K Frequent Elements

Medium

Topics

Companies

Given an integer array `nums` and an integer `k`, return *the `k` most frequent elements*. You may return the answer in **any order**.

Example 1:

Input: `nums = [1,1,1,2,2,3]`, `k = 2`

Output: `[1,2]`

Example 2:

Input: `nums = [1]`, `k = 1`

Output: `[1]`

Constraints:

- `1 <= nums.length <= 105`
- `-104 <= nums[i] <= 104`
- `k` is in the range `[1, the number of unique elements in the array]`.
- It is **guaranteed** that the answer is **unique**.

238. Product of Array Except Self

Medium

Topics

Companies

Hint

Given an integer array `nums`, return an array `answer` such that `answer[i]` is equal to the product of all the elements of `nums` except `nums[i]`.

The product of any prefix or suffix of `nums` is **guaranteed** to fit in a **32-bit** integer.

You must write an algorithm that runs in $O(n)$ time and without using the division operation.

Example 1:

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

Output: `[24,12,8,6]`

Example 2:

Input: `nums = [-1,1,0,-3,3]`

Output: `[0,0,9,0,0]`

Constraints:

- $2 \leq \text{nums.length} \leq 10^5$
- $-30 \leq \text{nums}[i] \leq 30$
- The product of any prefix or suffix of `nums` is **guaranteed** to fit in a **32-bit** integer.

125. Valid Palindrome

Easy

Topics

Companies

A phrase is a **palindrome** if, after converting all uppercase letters into lowercase letters and removing all non-alphanumeric characters, it reads the same forward and backward. Alphanumeric characters include letters and numbers.

Given a string `s`, return `true` if it is a **palindrome**, or `false` otherwise.

Example 1:

Input: `s = "A man, a plan, a canal: Panama"`

Output: `true`

Explanation: "amanaplanacanalpanama" is a palindrome.

Example 2:

Input: `s = "race a car"`

Output: `false`

Explanation: "raceacar" is not a palindrome.

Example 3:

Input: `s = ""`

Output: `true`

Explanation: `s` is an empty string "" after removing non-alphanumeric characters.

Since an empty string reads the same forward and backward, it is a palindrome.

167. Two Sum II - Input Array Is Sorted

Medium

Topics

Companies

Given a **1-indexed** array of integers `numbers` that is already **sorted in non-decreasing order**, find two numbers such that they add up to a specific `target` number. Let these two numbers be `numbers[index1]` and `numbers[index2]` where $1 \leq \text{index}_1 < \text{index}_2 \leq \text{numbers.length}$.

Return the indices of the two numbers, `index1` and `index2`, **added by one as an integer array** `[index1, index2]` of length 2.

The tests are generated such that there is **exactly one solution**. You **may not** use the same element twice.

Your solution must use only constant extra space.

Example 1:

Input: `numbers = [2,7,11,15]`, `target = 9`

Output: `[1,2]`

Explanation: The sum of 2 and 7 is 9. Therefore, `index1 = 1`, `index2 = 2`. We return `[1, 2]`.

Example 2:

Input: `numbers = [2,3,4]`, `target = 6`

Output: `[1,3]`

Explanation: The sum of 2 and 4 is 6. Therefore `index1 = 1`, `index2 = 3`. We return `[1, 3]`.

Example 3:

Input: `numbers = [-1,0]`, `target = -1`

Output: `[1,2]`

Explanation: The sum of -1 and 0 is -1. Therefore `index1 = 1`, `index2 = 2`. We return `[1, 2]`.

20. Valid Parentheses

Easy

Topics

Companies

Hint

Given a string `s` containing just the characters `'('`, `')'`, `'{'`, `'}'`, `'['` and `']'`, determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.
3. Every close bracket has a corresponding open bracket of the same type.

Example 1:

Input: `s = "()"`

Output: `true`

Example 2:

Input: `s = "()[]{}"`

Output: `true`

Example 3:

Input: `s = "(]"`

Output: `false`

Example 4:

Input: `s = "([)]"`

Output: `true`

155. Min Stack

Medium

Topics

Companies

Hint

Design a stack that supports push, pop, top, and retrieving the minimum element in constant time.

Implement the `MinStack` class:

- `MinStack()` initializes the stack object.
- `void push(int val)` pushes the element `val` onto the stack.
- `void pop()` removes the element on the top of the stack.
- `int top()` gets the top element of the stack.
- `int getMin()` retrieves the minimum element in the stack.

You must implement a solution with $O(1)$ time complexity for each function.

Example 1:

Input

```
["MinStack","push","push","push","getMin","pop","top","getMin"]  
[[],[-2],[0],[-3],[],[],[],[[]]]
```

Output

```
[null,null,null,null,-3,null,0,-2]
```

Explanation

```
MinStack minStack = new MinStack();  
minStack.push(-2);  
minStack.push(0);  
minStack.push(-3);  
minStack.getMin(); // return -3  
minStack.pop();  
minStack.top();    // return 0  
minStack.getMin(); // return -2
```

704. Binary Search

Easy

Topics

Companies

Given an array of integers `nums` which is sorted in ascending order, and an integer `target`, write a function to search `target` in `nums`. If `target` exists, then return its index. Otherwise, return `-1`.

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

Example 1:

Input: `nums = [-1,0,3,5,9,12]`, `target = 9`

Output: `4`

Explanation: 9 exists in `nums` and its index is 4

Example 2:

Input: `nums = [-1,0,3,5,9,12]`, `target = 2`

Output: `-1`

Explanation: 2 does not exist in `nums` so return `-1`

Constraints:

- $1 \leq \text{nums.length} \leq 10^4$
- $-10^4 < \text{nums}[i], \text{target} < 10^4$
- All the integers in `nums` are **unique**.
- `nums` is sorted in ascending order.

74. Search a 2D Matrix

Medium

Topics

Companies

You are given an $m \times n$ integer matrix `matrix` with the following two properties:

- Each row is sorted in non-decreasing order.
- The first integer of each row is greater than the last integer of the previous row.

Given an integer `target`, return `true` if `target` is in `matrix` or `false` otherwise.

You must write a solution in $O(\log(m * n))$ time complexity.

Example 1:

1	3	5	7
10	11	16	20
23	30	34	60

Input: `matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]]`,
`target = 3`

Output: `true`

Example 2:

1	3	5	7
10	11	16	20
23	30	34	60

Input: `matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]]`,
`target = 13`

Output: `false`

121. Best Time to Buy and Sell Stock

Easy

Topics

Companies

You are given an array `prices` where `prices[i]` is the price of a given stock on the i^{th} day.

You want to maximize your profit by choosing a **single day** to buy one stock and choosing a **different day in the future** to sell that stock.

Return *the maximum profit you can achieve from this transaction*. If you cannot achieve any profit, return `0`.

Example 1:

Input: `prices = [7,1,5,3,6,4]`



Output: `5`

Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.

Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.

Example 2:

Input: `prices = [7,6,4,3,1]`



Output: `0`

Explanation: In this case, no transactions are done and the max profit = 0.

3. Longest Substring Without Repeating Characters

Medium

Topics

Companies

Hint

Given a string `s`, find the length of the **longest substring** without repeating characters.

Example 1:

Input: `s = "abcabcbb"`

Output: `3`

Explanation: The answer is "abc", with the length of 3.

Example 2:

Input: `s = "bbbbbb"`

Output: `1`

Explanation: The answer is "b", with the length of 1.

Example 3:

Input: `s = "pwwkew"`

Output: `3`

Explanation: The answer is "wke", with the length of 3.

Notice that the answer must be a substring, "pwke" is a subsequence and not a substring.

424. Longest Repeating Character Replacement

Medium

Topics

Companies

You are given a string `s` and an integer `k`. You can choose any character of the string and change it to any other uppercase English character. You can perform this operation at most `k` times.

Return the length of the longest substring containing the same letter you can get after performing the above operations.

Example 1:

Input: `s = "ABAB", k = 2`

Output: 4

Explanation: Replace the two 'A's with two 'B's or vice versa.

Example 2:

Input: `s = "AABABBA", k = 1`

Output: 4

Explanation: Replace the one 'A' in the middle with 'B' and form "AABBBBA".

The substring "BBBB" has the longest repeating letters, which is 4. There may exists other ways to achieve this answer too.

206. Reverse Linked List

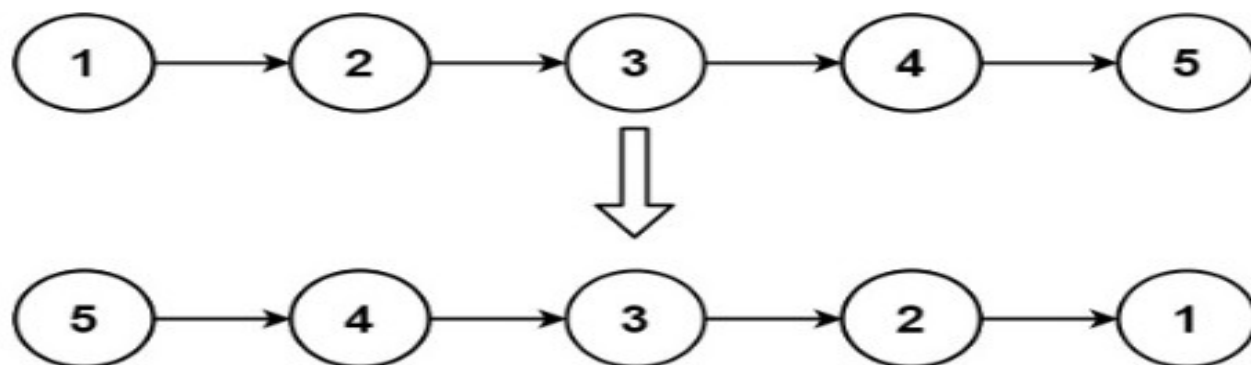
Easy

Topics

Companies

Given the `head` of a singly linked list, reverse the list, and return *the reversed list*.

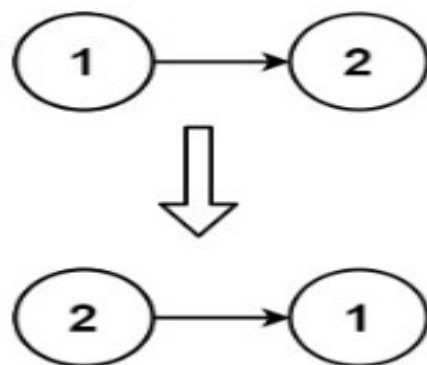
Example 1:



Input: `head = [1,2,3,4,5]`

Output: `[5,4,3,2,1]`

Example 2:



Input: `head = [1,2]`

Output: `[2,1]`

Example 3:

Input: `head = []`

Output: `[]`

21. Merge Two Sorted Lists

Easy

Topics

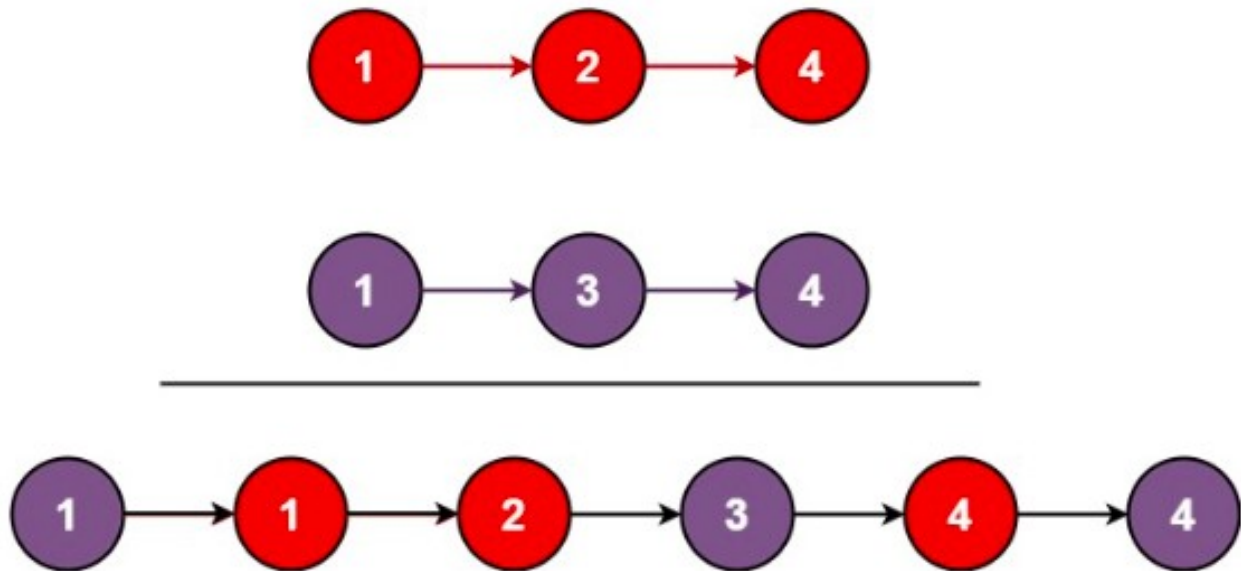
Companies

You are given the heads of two sorted linked lists `list1` and `list2`.

Merge the two lists into one **sorted** list. The list should be made by splicing together the nodes of the first two lists.

Return the head of the merged linked list.

Example 1:



Input: `list1 = [1,2,4]`, `list2 = [1,3,4]`

Output: `[1,1,2,3,4,4]`

Example 2:

Input: `list1 = []`, `list2 = []`

Output: `[]`

Example 3:

Input: `list1 = []`, `list2 = [0]`

Output: `[0]`

143. Reorder List

Medium

Topics

Companies

You are given the head of a singly linked-list. The list can be represented as:

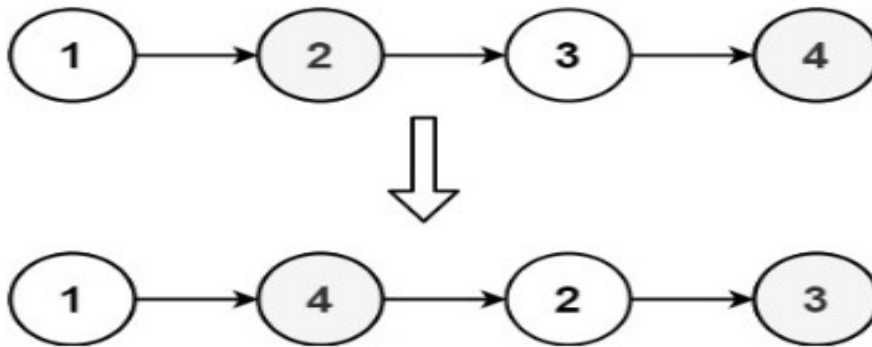
$L_0 \rightarrow L_1 \rightarrow \dots \rightarrow L_{n-1} \rightarrow L_n$

Reorder the list to be on the following form:

$L_0 \rightarrow L_n \rightarrow L_1 \rightarrow L_{n-1} \rightarrow L_2 \rightarrow L_{n-2} \rightarrow \dots$

You may not modify the values in the list's nodes. Only nodes themselves may be changed.

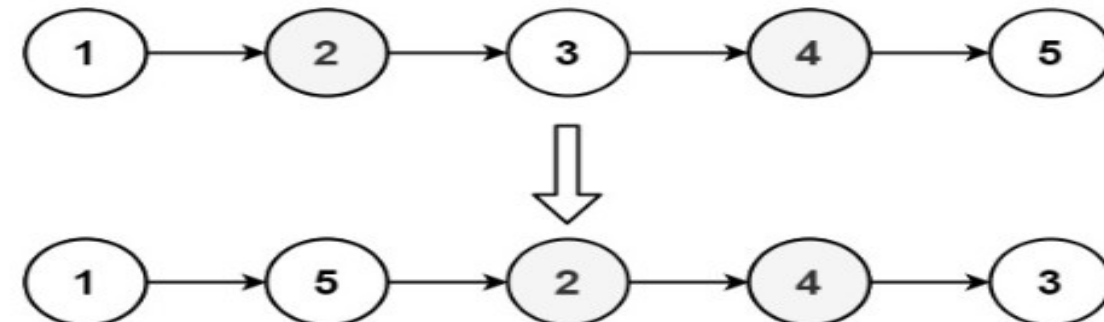
Example 1:



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

Output: [1,4,2,3]

Example 2:



Input: head = [1,2,3,4,5]

Output: [1,5,2,4,3]

19. Remove Nth Node From End of List

Medium

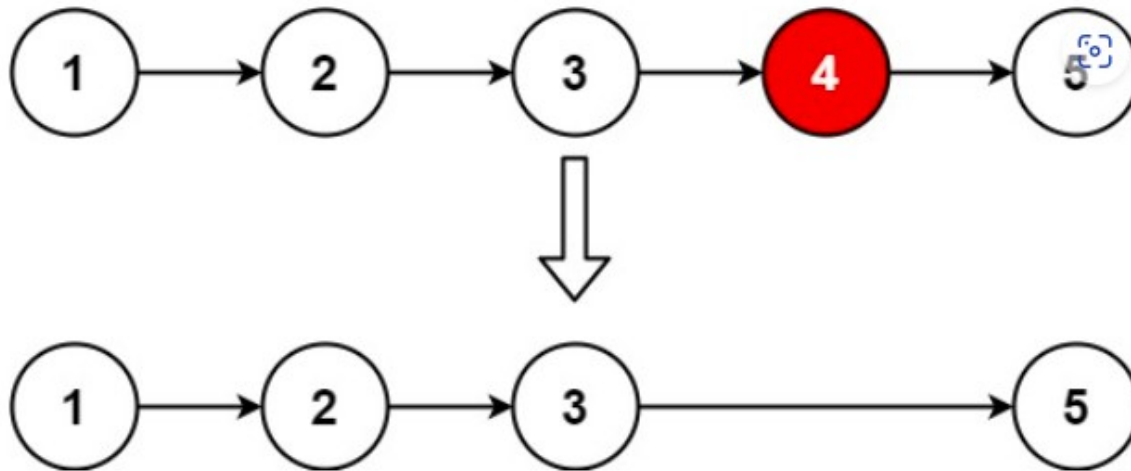
Topics

Companies

Hint

Given the `head` of a linked list, remove the n^{th} node from the end of the list and return its head.

Example 1:



Input: `head = [1,2,3,4,5]`, `n = 2`

Output: `[1,2,3,5]`

Example 2:

Input: `head = [1]`, `n = 1`

Output: `[]`

Example 3:

Input: `head = [1,2]`, `n = 1`

Output: `[1]`

226. Invert Binary Tree

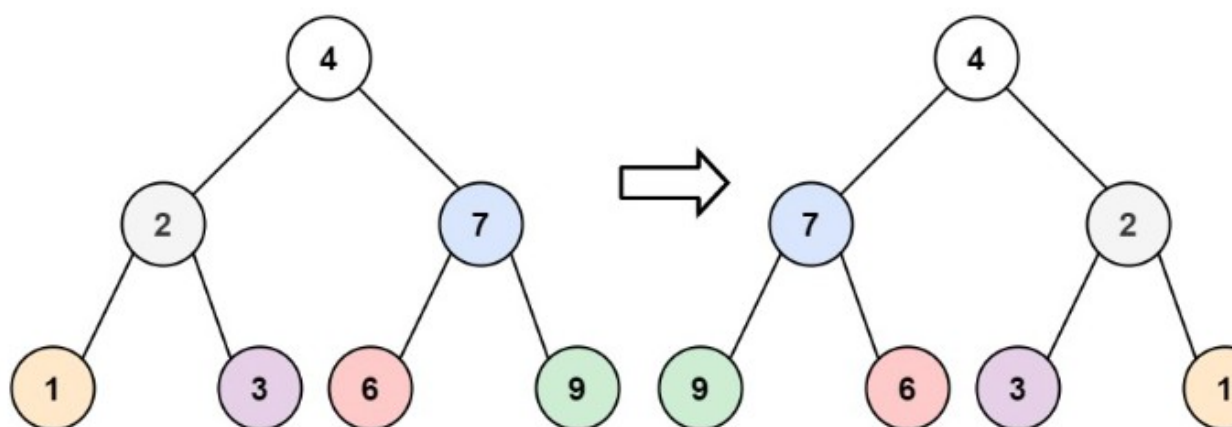
Easy

Topics

Companies

Given the `root` of a binary tree, invert the tree, and return *its root*.

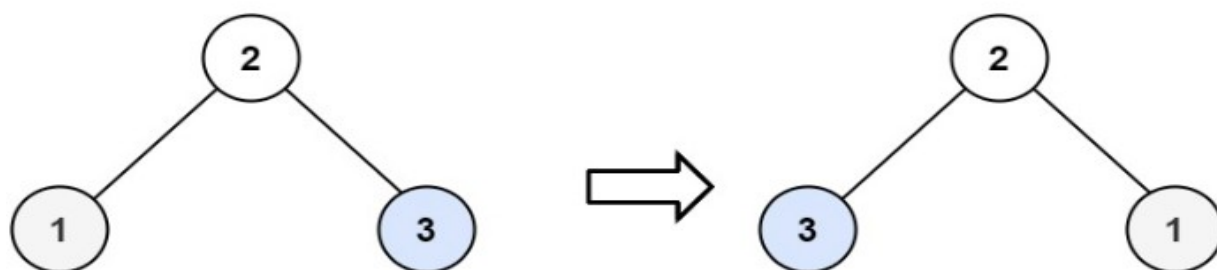
Example 1:



Input: `root = [4,2,7,1,3,6,9]`

Output: `[4,7,2,9,6,3,1]`

Example 2:



Input: `root = [2,1,3]`

Output: `[2,3,1]`

Example 3:

Input: `root = []`

Output: `[]`

104. Maximum Depth of Binary Tree

Easy

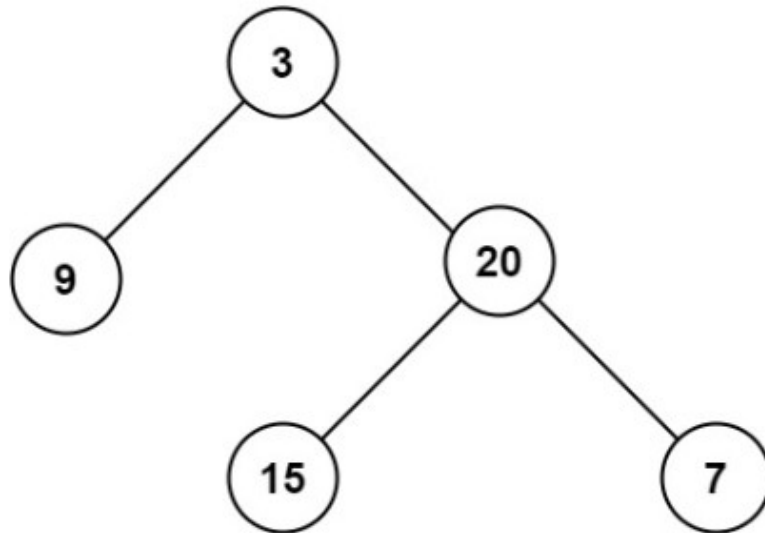
Topics

Companies

Given the `root` of a binary tree, return *its maximum depth*.

A binary tree's **maximum depth** is the number of nodes along the longest path from the root node down to the farthest leaf node.

Example 1:



Input: `root = [3,9,20,null,null,15,7]`

Output: 3

Example 2:

Input: `root = [1,null,2]`

Output: 2

543. Diameter of Binary Tree

Easy

Topics

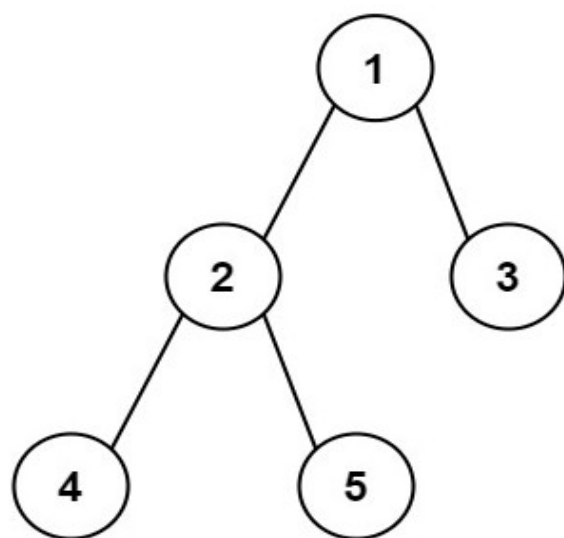
Companies

Given the `root` of a binary tree, return *the length of the **diameter** of the tree*.

The **diameter** of a binary tree is the **length** of the longest path between any two nodes in a tree. This path may or may not pass through the `root`.

The **length** of a path between two nodes is represented by the number of edges between them.

Example 1:



Input: `root = [1,2,3,4,5]`

Output: 3

Explanation: 3 is the length of the path [4,2,1,3] or [5,2,1,3].

Example 2:

Input: `root = [1,2]`

Output: 1

110. Balanced Binary Tree

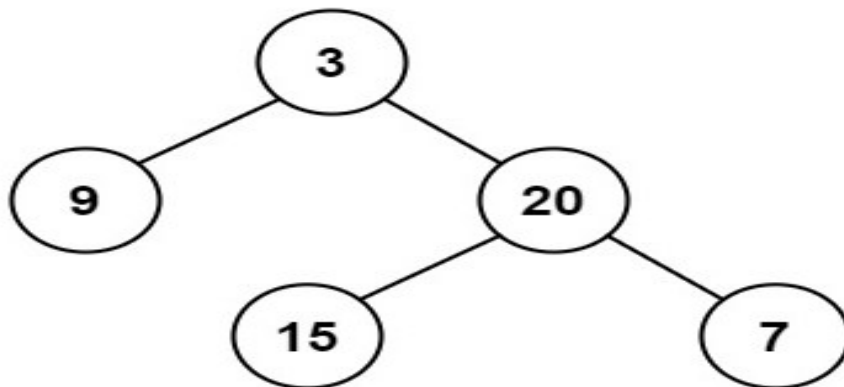
Easy

Topics

Companies

Given a binary tree, determine if it is **height-balanced**.

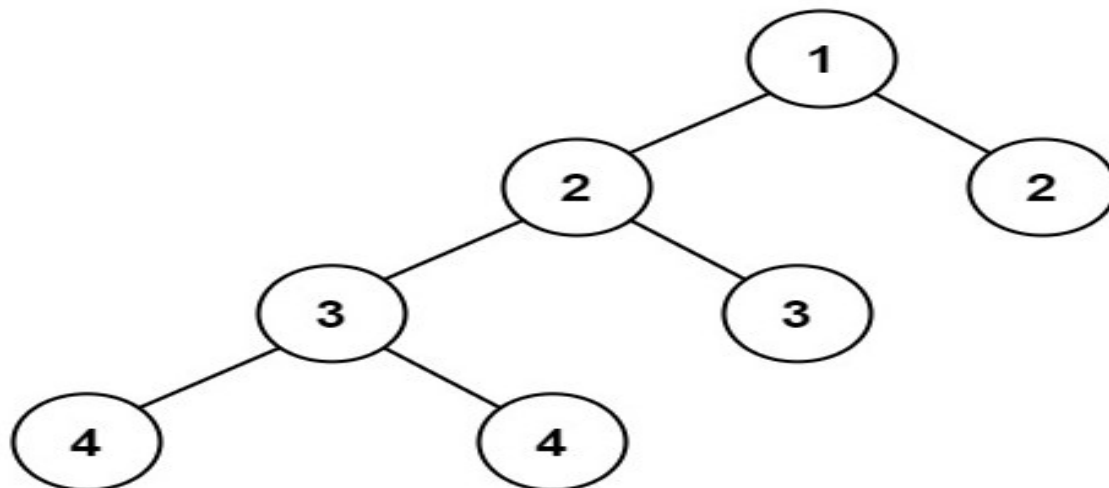
Example 1:



Input: root = [3,9,20,null,null,15,7]

Output: true

Example 2:



Input: root = [1,2,2,3,3,null,null,4,4]

Output: false

Example 3:

Input: root = []

Output: true

100. Same Tree

Easy

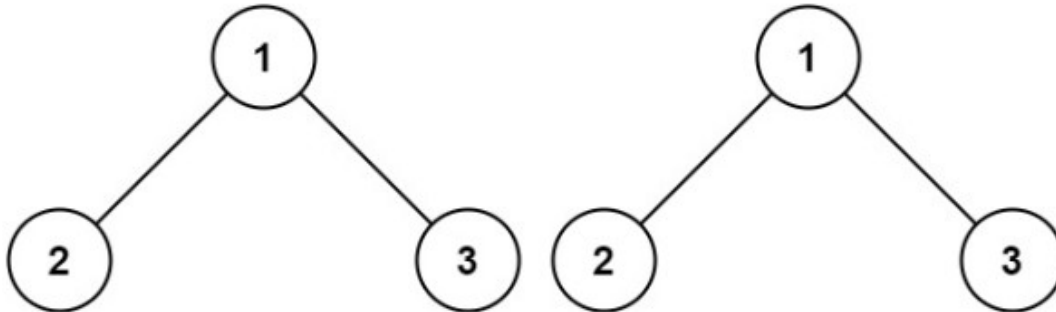
Topics

Companies

Given the roots of two binary trees p and q , write a function to check if they are the same or not.

Two binary trees are considered the same if they are structurally identical, and the nodes have the same value.

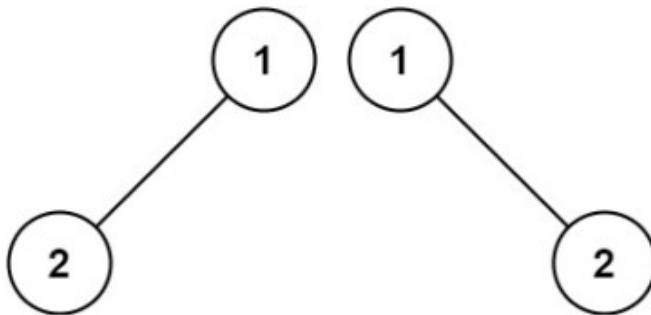
Example 1:



Input: $p = [1,2,3]$, $q = [1,2,3]$

Output: true

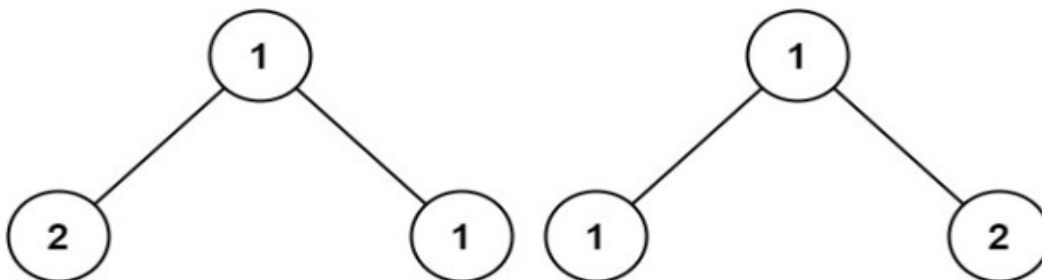
Example 2:



Input: $p = [1,2]$, $q = [1,null,2]$

Output: false

Example 3:



Input: $p = [1,2,1]$, $q = [1,1,2]$

Output: false

572. Subtree of Another Tree

Easy

Topics

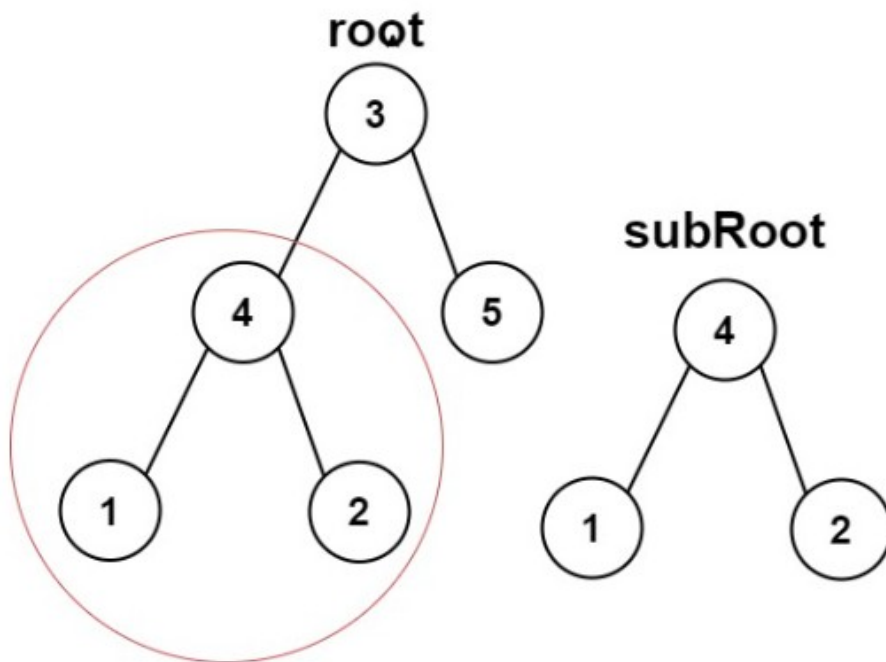
Companies

Hint

Given the roots of two binary trees `root` and `subRoot`, return `true` if there is a subtree of `root` with the same structure and node values of `subRoot` and `false` otherwise.

A subtree of a binary tree `tree` is a tree that consists of a node in `tree` and all of this node's descendants. The tree `tree` could also be considered as a subtree of itself.

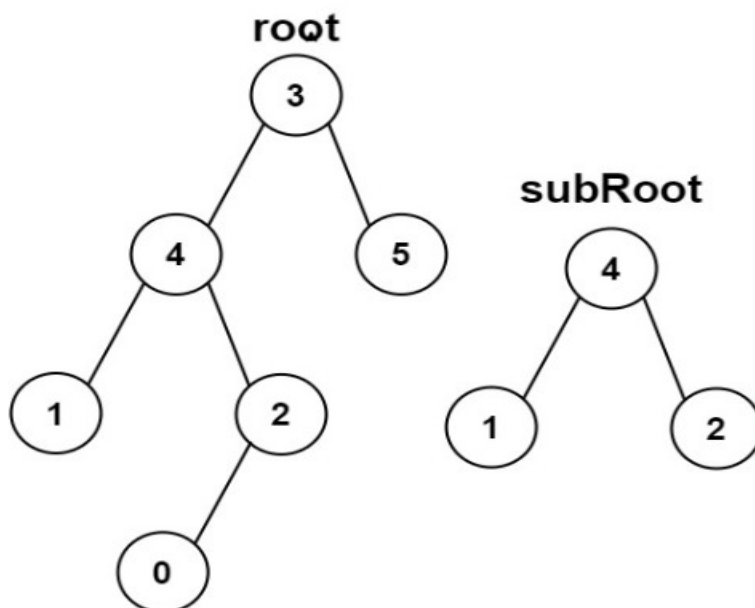
Example 1:



Input: `root = [3,4,5,1,2]`, `subRoot = [4,1,2]`

Output: `true`

Example 2:



Input: `root = [3,4,5,1,2,null,null,null,null,0]`, `subRoot = [4,1,2]`

Output: `false`

78. Subsets

Medium

Topics

Companies

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]]`

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]]`

Example 3:

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

Output: `[]`

703. Kth Largest Element in a Stream

Easy

Topics

Companies

You are part of a university admissions office and need to keep track of the k^{th} highest test score from applicants in real-time. This helps to determine cut-off marks for interviews and admissions dynamically as new applicants submit their scores.

You are tasked to implement a class which, for a given integer k , maintains a stream of test scores and continuously returns the k^{th} highest test score **after** a new score has been submitted. More specifically, we are looking for the k^{th} highest score in the sorted list of all scores.

Implement the `KthLargest` class:

- `KthLargest(int k, int[] nums)` Initializes the object with the integer k and the stream of test scores `nums`.
- `int add(int val)` Adds a new test score `val` to the stream and returns the element representing the k^{th} largest element in the pool of test scores so far.

Example 1:

Input:

```
["KthLargest", "add", "add", "add", "add", "add"]  
[[3, [4, 5, 8, 2]], [3], [5], [10], [9], [4]]
```

Output: [null, 4, 5, 5, 8, 8]

Explanation:

```
KthLargest kthLargest = new KthLargest(3, [4, 5, 8, 2]);  
kthLargest.add(3); // return 4  
kthLargest.add(5); // return 5  
kthLargest.add(10); // return 5  
kthLargest.add(9); // return 8  
kthLargest.add(4); // return 8
```

Example 2:

Input:

```
["KthLargest", "add", "add", "add", "add"]  
[[4, [7, 7, 7, 7, 8, 3]], [2], [10], [9], [9]]
```

Output: [null, 7, 7, 7, 8]

Explanation:

```
KthLargest kthLargest = new KthLargest(4, [7, 7, 7, 7, 8, 3]);  
kthLargest.add(2); // return 7  
kthLargest.add(10); // return 7  
kthLargest.add(9); // return 7  
kthLargest.add(9); // return 8
```

1046. Last Stone Weight

Easy

Topics

Companies

Hint

You are given an array of integers `stones` where `stones[i]` is the weight of the i^{th} stone.

We are playing a game with the stones. On each turn, we choose the **heaviest two stones** and smash them together. Suppose the heaviest two stones have weights `x` and `y` with `x <= y`. The result of this smash is:

- If `x == y`, both stones are destroyed, and
- If `x != y`, the stone of weight `x` is destroyed, and the stone of weight `y` has new weight `y - x`.

At the end of the game, there is **at most one** stone left.

Return *the weight of the last remaining stone*. If there are no stones left, return `0`.

Example 1:

Input: `stones = [2,7,4,1,8,1]`

Output: `1`

Explanation:

We combine 7 and 8 to get 1 so the array converts to `[2,4,1,1,1]` then, we combine 2 and 4 to get 2 so the array converts to `[2,1,1,1]` then, we combine 2 and 1 to get 1 so the array converts to `[1,1,1]` then, we combine 1 and 1 to get 0 so the array converts to `[1]` then that's the value of the last stone.

Example 2:

Input: `stones = [1]`

Output: `1`

200. Number of Islands

Medium

Topics

Companies

Given an $m \times n$ 2D binary grid `grid` which represents a map of `'1'` s (land) and `'0'` s (water), return *the number of islands*.

An **island** is surrounded by water and is formed by connecting adjacent lands horizontally or vertically. You may assume all four edges of the grid are all surrounded by water.

Example 1:

```
Input: grid = [
  ["1","1","1","1","0"],
  ["1","1","0","1","0"],
  ["1","1","0","0","0"],
  ["0","0","0","0","0"]
]
```

Output: 1

Example 2:

```
Input: grid = [
  ["1","1","0","0","0"],
  ["1","1","0","0","0"],
  ["0","0","1","0","0"],
  ["0","0","0","1","1"]
]
```

Output: 3

695. Max Area of Island

Medium

Topics


Companies

You are given an $m \times n$ binary matrix `grid`. An island is a group of `1`'s (representing land) connected **4-directionally** (horizontal or vertical.) You may assume all four edges of the grid are surrounded by water.

The **area** of an island is the number of cells with a value `1` in the island.

Return the maximum **area** of an island in `grid`. If there is no island, return `0`.

Example 1:

0	0	1	0	0	0	0	1	0	0	0	0	
0	0	0	0	0	0	0	1	1	1	0	0	0
0	1	1	0	1	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	0	1	0	1	0	0
0	1	0	0	1	1	0	0	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	0	0	0	0

Input: `grid = [[0,0,1,0,0,0,0,1,0,0,0,0,0],[0,0,0,0,0,0,0,1,1,1,0,0,0],[0,1,1,0,1,0,0,0,0,0,0,0,0],[0,1,0,0,1,1,0,0,1,0,1,0,0],[0,1,0,0,1,1,0,0,1,1,1,0,0],[0,0,0,0,0,0,0,0,0,0,1,0,0],[0,0,0,0,0,0,0,1,1,1,0,0,0],[0,0,0,0,0,0,0,1,1,0,0,0,0]]`

Output: 6

Explanation: The answer is not 11, because the island must be connected 4-directionally.

70. Climbing Stairs

Easy

Topics

Companies

Hint

You are climbing a staircase. It takes n steps to reach the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

Example 1:

Input: $n = 2$

Output: 2

Explanation: There are two ways to climb to the top.

1. 1 step + 1 step
2. 2 steps

Example 2:

Input: $n = 3$

Output: 3

Explanation: There are three ways to climb to the top.

1. 1 step + 1 step + 1 step
2. 1 step + 2 steps
3. 2 steps + 1 step

746. Min Cost Climbing Stairs

Easy

Topics

Companies

Hint

You are given an integer array `cost` where `cost[i]` is the cost of i^{th} step on a staircase. Once you pay the cost, you can either climb one or two steps.

You can either start from the step with index `0`, or the step with index `1`.

Return the minimum cost to reach the top of the floor.

Example 1:

Input: `cost = [10,15,20]`

Output: 15

Explanation: You will start at index 1.

– Pay 15 and climb two steps to reach the top.

The total cost is 15.



Example 2:

Input: `cost = [1,100,1,1,1,100,1,1,100,1]`

Output: 6

Explanation: You will start at index 0.

– Pay 1 and climb two steps to reach index 2.

– Pay 1 and climb two steps to reach index 4.

– Pay 1 and climb two steps to reach index 6.

– Pay 1 and climb one step to reach index 7.

– Pay 1 and climb two steps to reach index 9.

– Pay 1 and climb one step to reach the top.

The total cost is 6.



62. Unique Paths

Medium

Topics

Companies

There is a robot on an $m \times n$ grid. The robot is initially located at the **top-left corner** (i.e., `grid[0][0]`). The robot tries to move to the **bottom-right corner** (i.e., `grid[m - 1][n - 1]`). The robot can only move either down or right at any point in time.

Given the two integers m and n , return *the number of possible unique paths that the robot can take to reach the bottom-right corner*.

The test cases are generated so that the answer will be less than or equal to $2 * 10^9$.

Example 1:



Input: $m = 3, n = 7$

Output: 28

Example 2:

Input: $m = 3, n = 2$

Output: 3

Explanation: From the top-left corner, there are a total of 3 ways to reach the bottom-right corner:

1. Right -> Down -> Down
2. Down -> Down -> Right
3. Down -> Right -> Down

1143. Longest Common Subsequence

Medium

Topics

Companies

Hint

Given two strings `text1` and `text2`, return *the length of their longest **common subsequence***. If there is no **common subsequence**, return `0`.

A **subsequence** of a string is a new string generated from the original string with some characters (can be none) deleted without changing the relative order of the remaining characters.

- For example, `"ace"` is a subsequence of `"abcde"`.

A **common subsequence** of two strings is a subsequence that is common to both strings.

Example 1:

Input: `text1 = "abcde", text2 = "ace"`

Output: `3`

Explanation: The longest common subsequence is `"ace"` and its length is 3.

Example 2:

Input: `text1 = "abc", text2 = "abc"`

Output: `3`

Explanation: The longest common subsequence is `"abc"` and its length is 3.

Example 3:

Input: `text1 = "abc", text2 = "def"`

Output: `0`

Explanation: There is no such common subsequence, so the result is 0.

53. Maximum Subarray

Medium

Topics

Companies

Given an integer array `nums`, find the **subarray** with the largest sum, and return *its sum*.

Example 1:

Input: `nums = [-2,1,-3,4,-1,2,1,-5,4]`

Output: 6

Explanation: The subarray `[4,-1,2,1]` has the largest sum 6.

Example 2:

Input: `nums = [1]`

Output: 1

Explanation: The subarray `[1]` has the largest sum 1.

Example 3:

Input: `nums = [5,4,-1,7,8]`

Output: 23

Explanation: The subarray `[5,4,-1,7,8]` has the largest sum 23.

55. Jump Game

Medium

Topics

Companies

You are given an integer array `nums`. You are initially positioned at the array's **first index**, and each element in the array represents your maximum jump length at that position.

Return `true` if you can reach the last index, or `false` otherwise.

Example 1:

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

Output: `true`

Explanation: Jump 1 step from index 0 to 1, then 3 steps to the last index.

Example 2:

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

Output: `false`

Explanation: You will always arrive at index 3 no matter what. Its maximum jump length is 0, which makes it impossible to reach the last index.

45. Jump Game II

Medium

Topics

Companies

You are given a **0-indexed** array of integers `nums` of length `n`. You are initially positioned at `nums[0]`.

Each element `nums[i]` represents the maximum length of a forward jump from index `i`. In other words, if you are at `nums[i]`, you can jump to any `nums[i + j]` where:

- $0 \leq j \leq \text{nums}[i]$ and
- $i + j < n$

Return the *minimum number of jumps to reach* `nums[n - 1]`. The test cases are generated such that you can reach `nums[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