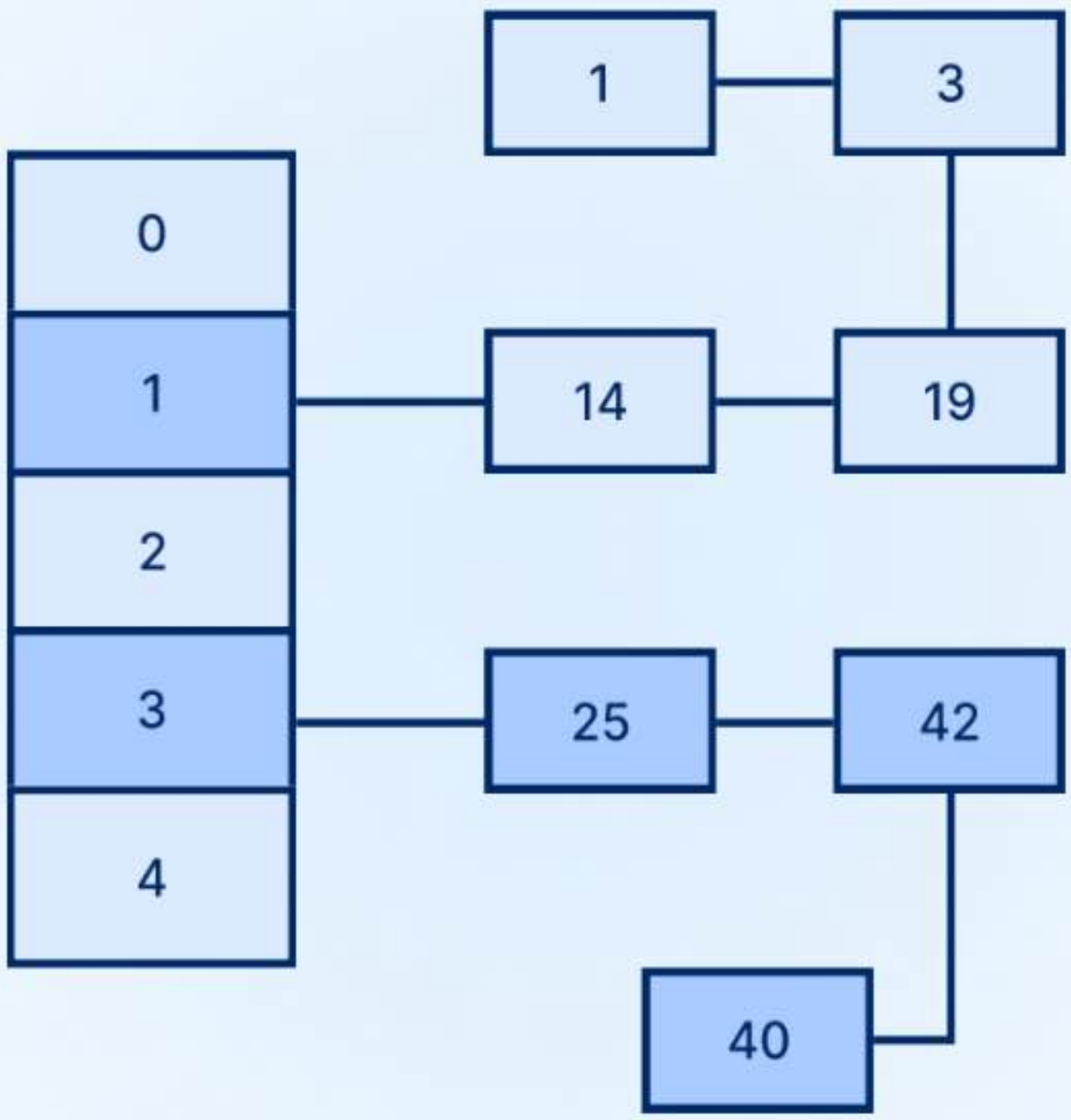
BOSSCODER ACADEMY

TOP 150

DSA Interview Questions



Disclaimer

Interviews test problem-solving, not memory.

Your ability to analyze, reason, and optimize matters more than rote learning.

This doc is a step towards building that mindset.

Array/String

1. Merge Sorted Array (Easy)

You are given two integer arrays nums1 and nums2, sorted in non-decreasing order, and two integers m and n, representing the number of elements in nums1 and nums2 respectively.

Merge nums1 and nums2 into a single array sorted in non-decreasing order.

The final sorted array should not be returned by the function, but instead be stored inside the array nums1. To accommodate this, nums1 has a length of m + n, where the first m elements denote the elements that should be merged, and the last n elements are set to 0 and should be ignored. nums2 has a length of n.

Practice

2. Remove Element (Easy)

Given an integer array nums and an integer val, remove all occurrences of val in nums in-place. The order of the elements may be changed. Then return the number of elements in nums which are not equal to val.

Consider the number of elements in nums which are not equal to val be k, to get accepted, you need to do the following things:

- Change the array nums such that the first k elements of nums contain the - elements which are not equal to val. The remaining elements of nums are not important as well as the size of nums.
- Return k.

Practice

3. Remove Duplicates from Sorted Array (Easy)

Given an integer array nums sorted in non-decreasing order, remove the duplicates in-place such that each unique element appears only once. The relative order of the elements should be kept the same. Then return the number of unique elements in nums.

Consider the number of unique elements of nums to be k, to get accepted, you need to do the following things:

- Change the array nums such that the first k elements of nums contain the - elements which are not equal to val. The remaining elements of nums are not important as well as the size of nums.
- Return k.

Practice

4. Remove Duplicates from Sorted Array II (Medium)

Given an integer array nums sorted in non-decreasing order, remove some duplicates in-place such that each unique element appears at most twice. The relative order of the elements should be kept the same.

Since it is impossible to change the length of the array in some languages, you must instead have the result be placed in the first part of the array nums. More formally, if there are k elements after removing the duplicates, then the first k elements of nums should hold the final result. It does not matter what you leave beyond the first k elements.

Return k after placing the final result in the first k slots of nums.

Do not allocate extra space for another array. You must do this by modifying the input array in-place with O(1) extra memory.

Practice

5. Majority Element (Easy)

Given an array nums of size n, return the majority element.

The majority element is the element that appears more than ⌊n / 2⌋ times. You may assume that the majority element always exists in the array.

Practice

6. Rotate Array (Medium)

Given an integer array nums, rotate the array to the right by k steps, where k is non-negative.

Practice

7. Best Time to Buy and Sell Stock (Easy)

You are given an array prices where prices[i] is the price of a given stock on the ith 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.

Practice

8. Best Time to Buy and Sell Stock II (Medium)

You are given an integer array prices where prices[i] is the price of a given stock on the ith day.

On each day, you may decide to buy and/or sell the stock. You can only hold at most one share of the stock at any time. However, you can buy it then immediately sell it on the same day.

Find and return the maximum profit you can achieve.

Practice

9. Jump Game (Medium)

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.

Practice

10. Jump Game II (Medium)

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.

Practice

11. H-Index (Medium)

Given an array of integers citations where citations[i] is the number of citations a researcher received for their ith paper, return the researcher's h-index.

According to the definition of h-index on Wikipedia: The h-index is defined as the maximum value of h such that the given researcher has published at least h papers that have each been cited at least h times.

Practice

12. Insert Delete GetRandom O(1) (Medium)

Implement the RandomizedSet class:

- RandomizedSet() Initializes the RandomizedSet object.
- bool insert(int val) Inserts an item val into the set if not present. Returns true if the item was not present, false otherwise.
- bool remove(int val) Removes an item val from the set if present. Returns true if the item was present, false otherwise.
- int getRandom() Returns a random element from the current set of elements (it's guaranteed that at least one element exists when this method is called). Each element must have the same probability of being returned.

You must implement the functions of the class such that each function works in average O(1) time complexity.

Practice

13. Product of Array Except Self (Medium)

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.

Practice

14. Gas Station (Medium)

There are n gas stations along a circular route, where the amount of gas at the ith station is gas[i].

You have a car with an unlimited gas tank and it costs cost[i] of gas to travel from the ith station to its next (i + 1)th station. You begin the journey with an empty tank at one of the gas stations.

Given two integer arrays gas and cost, return the starting gas station's index if you can travel around the circuit once in the clockwise direction, otherwise return -1. If there exists a solution, it is guaranteed to be unique.

Practice

15. Candy (Hard)

There are n children standing in a line. Each child is assigned a rating value given in the integer array ratings.

You are giving candies to these children subjected to the following requirements:

- Each child must have at least one candy.
- Children with a higher rating get more candies than their neighbors.

Return the minimum number of candies you need to have to distribute the candies to the children.

Practice

16. Trapping Rain Water (Hard)

Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it can trap after raining.

Practice

17. Roman to Integer (Easy)

Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

(in a table here 👇)

Symbol	Value
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

For example, 2 is written as II in Roman numeral, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used:

- I can be placed before V (5) and X (10) to make 4 and 9.
- X can be placed before L (50) and C (100) to make 40 and 90.
- C can be placed before D (500) and M (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer.

Practice

18. Integer to Roman (Medium)

Like the last question, but reversed this time. Given an integer, convert it to a Roman numeral.

Practice

19. Length of Last Word (Easy)

Given a string s consisting of words and spaces, return the length of the last word in the string.

A word is a maximal substring consisting of non-space characters only.

Practice

20. Longest Common Prefix (Easy)

Write a function to find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string "".

Practice

21. Reverse Words in a String (Medium)

Given an input string s, reverse the order of the words.

A word is defined as a sequence of non-space characters. The words in s will be separated by at least one space.

Return a string of the words in reverse order concatenated by a single space.

Note that s may contain leading or trailing spaces or multiple spaces between two words. The returned string should only have a single space separating the words. Do not include any extra spaces.

Practice

22. Zigzag Conversion (Medium)

Practice

23. Find the Index of the First Occurrence in a String (Easy)

The string "PAYPALISHIRING" is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)

P A H N

A P L S I I G

Y I R

And then read line by line: "PAHNAPLSIIGYIR"

Write the code that will take a string and make this conversion given a number of rows:

string convert(string s, int numRows);

Practice

24. Text Justification (Hard)

Given an array of strings words and a width maxWidth, format the text such that each line has exactly maxWidth characters and is fully (left and right) justified.

You should pack your words in a greedy approach; that is, pack as many words as you can in each line. Pad extra spaces ' ' when necessary so that each line has exactly maxWidth characters.

Extra spaces between words should be distributed as evenly as possible. If the number of spaces on a line does not divide evenly between words, the empty slots on the left will be assigned more spaces than the slots on the right.

For the last line of text, it should be left-justified, and no extra space is inserted between words.

Note:

- A word is defined as a character sequence consisting of non-space characters only.
- Each word's length is guaranteed to be greater than 0 and not exceed maxWidth.
- The input array words contains at least one word.

Practice

Two Pointers

25. Valid Palindrome (Easy)

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.

Practice

26. Is Subsequence (Easy)

Given two strings s and t, return true if s is a subsequence of t, or false otherwise.

A subsequence of a string is a new string that is formed from the original string by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters. (i.e., "ace" is a subsequence of "abcde" while "aec" is not).

Practice

27. Two Sum II - Input Array Is Sorted (Medium)

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 <= index1 < index2 <= 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.

Practice

28. Container With Most Water (Medium)

You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the ith line are (i, 0) and (i, height[i]).

Find two lines that together with the x-axis form a container, such that the container contains the most water.

Return the maximum amount of water a container can store.

Notice that you may not slant the container.

Practice

29. 3Sum (Medium)

Given an integer array nums, return all the triplets [nums[i], nums[j], nums[k]] such that i != j, i != k, and j != k, and nums[i] + nums[j] + nums[k] == 0.

Notice that the solution set must not contain duplicate triplets.

Practice

Sliding Window

30. Minimum Size Subarray Sum (Medium)

Given an array of positive integers nums and a positive integer target, return the minimal length of a subarray whose sum is greater than or equal to target. If there is no such subarray, return 0 instead.

Practice

31. Longest Substring Without Repeating Characters (Medium)

Given a string s, find the length of the longest substring without duplicate characters.

Practice

32. Substring with Concatenation of All Words (Hard)

You are given a string s and an array of strings words. All the strings of words are of the same length.

A concatenated string is a string that exactly contains all the strings of any permutation of words concatenated.

For example, if words = ["ab","cd","ef"], then "abcdef", "abefcd", "cdabef", "cdefab", "efabcd", and "efcdab" are all concatenated strings. "acdbef" is not a concatenated string because it is not the concatenation of any permutation of words.

Return an array of the starting indices of all the concatenated substrings in s. You can return the answer in any order.

Practice

33. Minimum Window Substring (Hard)

Given two strings s and t of lengths m and n respectively, return the minimum window substring of s such that every character in t (including duplicates) is included in the window. If there is no such substring, return the empty string "".

The testcases will be generated such that the answer is unique.

Practice

34. Valid Sudoku (Medium)

Determine if a 9 × 9 Sudoku board is valid. Only the filled cells need to be validated according to the following rules:

1. Each row must contain the digits 1-9 without repetition.
2. Each column must contain the digits 1-9 without repetition.
3. Each of the nine 3 × 3 sub-boxes of the grid must contain the digits 1-9 without repetition.

Note:

- A Sudoku board (partially filled) could be valid but is not necessarily solvable.
- Only the filled cells need to be validated according to the mentioned rules.

Practice

35. Spiral Matrix (Medium)

Given an m x n matrix, return all elements of the matrix in spiral order.

Practice

36. Rotate Image (Medium)

You are given an n x n 2D matrix representing an image, rotate the image by 90 degrees (clockwise).

You have to rotate the image in-place, which means you have to modify the input 2D matrix directly. DO NOT allocate another 2D matrix and do the rotation.

Practice

37. Set Matrix Zeroes (Medium)

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.

Practice

38. Game of Life (Medium)

According to Wikipedia's article: "The Game of Life, also known simply as Life, is a cellular automaton devised by the British mathematician John Horton Conway in 1970."

The board is made up of an m x n grid of cells, where each cell has an initial state: live (represented by a 1) or dead (represented by a 0). Each cell interacts with its eight neighbors (horizontal, vertical, diagonal) using the following four rules (taken from the above Wikipedia article):

1. Any live cell with fewer than two live neighbors dies as if caused by under-population.
2. Any live cell with two or three live neighbors lives on to the next generation.
3. Any live cell with more than three live neighbors dies, as if by over-population.
4. Any dead cell with exactly three live neighbors becomes a live cell, as if by reproduction.

The next state of the board is determined by applying the above rules simultaneously to every cell in the current state of the m x n grid board. In this process, births and deaths occur simultaneously.

Given the current state of the board, update the board to reflect its next state.

Note that you do not need to return anything.

Practice

Hashmap

39. Ransom Note (Easy)

Given two strings ransomNote and magazine, return true if ransomNote can be constructed by using the letters from magazine and false otherwise.

Each letter in magazine can only be used once in ransomNote.

Practice

40. Isomorphic Strings (Easy)

Given two strings s and t, determine if they are isomorphic.

Two strings s and t are isomorphic if the characters in s can be replaced to get t.

All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character, but a character may map to itself.

Practice

41. Word Pattern (Easy)

Given a pattern and a string s, find if s follows the same pattern.

Here follow means a full match, such that there is a bijection between a letter in pattern and a non-empty word in s. Specifically:

- Each letter in pattern maps to exactly one unique word in s.
- Each unique word in s maps to exactly one letter in pattern.
- No two letters map to the same word, and no two words map to the same letter.

Practice

42. Valid Anagram (Easy)

Given two strings s and t, return true if t is an anagram of s, and false otherwise.

Practice

43. Group Anagrams (Medium)

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

Practice

44. Two Sum (Easy)

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.

Practice

45. Happy Number (Easy)

Write an algorithm to determine if a number n is happy.

A happy number is a number defined by the following process:

- Starting with any positive integer, replace the number by the sum of the squares of its digits.
- Repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1.
- Those numbers for which this process ends in 1 are happy.

Return true if n is a happy number, and false if not.

Practice

46. Contains Duplicate II (Easy)

Given an integer array nums and an integer k, return true if there are two distinct indices i and j in the array such that nums[i] == nums[j] and abs(i - j) <= k.

Practice

47. Longest Consecutive Sequence (Medium)

Given an unsorted array of integers nums, return the length of the longest consecutive elements sequence.

You must write an algorithm that runs in O(n) time.

Practice

Intervals

48. Summary Ranges (Easy)

You are given a sorted unique integer array nums.

A range [a,b] is the set of all integers from a to b (inclusive).

Return the smallest sorted list of ranges that cover all the numbers in the array exactly. That is, each element of nums is covered by exactly one of the ranges, and there is no integer x such that x is in one of the ranges but not in nums.

Each range [a,b] in the list should be output as:

- "a→b" if a != b
- "a" if a == b

Practice

49. Merge Intervals (Medium)

Given an array of intervals where intervals[i] = [starti, endi], merge all overlapping intervals, and return an array of the non-overlapping intervals that cover all the intervals in the input.

Practice

50. Insert Interval (Medium)

You are given an array of non-overlapping intervals intervals where intervals[i] = [starti, endi] represent the start and the end of the ith interval and intervals is sorted in ascending order by starti. You are also given an interval newInterval = [start, end] that represents the start and end of another interval.

Insert newInterval into intervals such that intervals is still sorted in ascending order by starti and intervals still does not have any overlapping intervals (merge overlapping intervals if necessary).

Return intervals after the insertion.

Note that you don't need to modify intervals in-place. You can make a new array and return it.

Practice

51. Minimum Number of Arrows to Burst Balloons (Medium)

There are some spherical balloons taped onto a flat wall that represents the XY-plane. The balloons are represented as a 2D integer array points where points[i] = [xstart, xend] denotes a balloon whose horizontal diameter stretches between xstart and xend. You do not know the exact y-coordinates of the balloons.

Arrows can be shot up directly vertically (in the positive y-direction) from different points along the x-axis. A balloon with xstart and xend is burst by an arrow shot at x if xstart <= x <= xend. There is no limit to the number of arrows that can be shot. A shot arrow keeps traveling up infinitely, bursting any balloons in its path.

Given the array points, return the minimum number of arrows that must be shot to burst all balloons.

Practice

Stack

52. Valid Parentheses (Easy)

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.

Practice

53. Simplify Path (Medium)

You are given an absolute path for a Unix-style file system, which always begins with a slash '/'. Your task is to transform this absolute path into its simplified canonical path.

The rules of a Unix-style file system are as follows:

- A single period '.' represents the current directory.
- A double period '..' represents the previous/parent directory.
- Multiple consecutive slashes such as '//' and '///' are treated as a single slash '/'.
- Any sequence of periods that does not match the rules above should be treated as a valid directory or file name. For example, '...' and '....' are valid directory or file names.

The simplified canonical path should follow these rules:

- The path must start with a single slash '/'.
- Directories within the path must be separated by exactly one slash '/'.

- The path must not end with a slash '/', unless it is the root directory.
- The path must not have any single or double periods ('.' and '..') used to denote current or parent directories.

Return the simplified canonical path.

Practice

54. Min Stack (Medium)

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.

Practice

55. Evaluate Reverse Polish Notation (Medium)

You are given an array of strings tokens that represents an arithmetic expression in a Reverse Polish Notation.

Evaluate the expression. Return an integer that represents the value of the expression.

Note:

- The valid operators are '+', '-', '*', and '/'.
- Each operand may be an integer or another expression.

- The division between two integers always truncates toward zero.
- There will not be any division by zero.
- The input represents a valid arithmetic expression in a reverse polish notation.
- The answer and all the intermediate calculations can be represented in a 32-bit integer.

Practice

56. Basic Calculator (Hard)

Given a string s representing a valid expression, implement a basic calculator to evaluate it, and return the result of the evaluation.

Note:

You are not allowed to use any built-in function which evaluates strings as mathematical expressions, such as eval().

Practice

Linked List

57. Linked List Cycle (Easy)

Given head, the head of a linked list, determine if the linked list has a cycle in it.

There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the next pointer. Internally, pos is used to denote the index of the node that tail's next pointer is connected to. Note that pos is not passed as a parameter.

Return true if there is a cycle in the linked list. Otherwise, return false.

Practice

58. Add Two Numbers (Medium)

You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Practice

59. Merge Two Sorted Lists (Easy)

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.

Practice

60. Copy List with Random Pointer (Medium)

A linked list of length n is given such that each node contains an additional random pointer, which could point to any node in the list, or null.

Construct a deep copy of the list. The deep copy should consist of exactly n brand new nodes, where each new node has its value set to the value of its corresponding original node. Both the next and random pointer of the new nodes should point to new nodes in the copied list such that the pointers in the original list and copied list represent the same list state. None of the pointers in the new list should point to nodes in the original list.

For example, if there are two nodes X and Y in the original list, where X.random → Y, then for the corresponding two nodes x and y in the copied list, x.random → y.

Return the head of the copied linked list.

Practice

61. Reverse Linked List II (Medium)

Given the head of a singly linked list and two integers left and right where left <= right, reverse the nodes of the list from position left to position right, and return the reversed list.

Practice

62. Reverse Nodes in k-Group (Hard)

Given the head of a linked list, reverse the nodes of the list k at a time, and return the modified list.

k is a positive integer and is less than or equal to the length of the linked list. If the number of nodes is not a multiple of k then left-out nodes, in the end, should remain as it is.

You may not alter the values in the list's nodes, only nodes themselves may be changed.

Practice

63. Remove Nth Node From End of List (Medium)

Given the head of a linked list, remove the nth node from the end of the list and return its head.

Practice

64. Remove Duplicates from Sorted List II (Medium)

Given the head of a sorted linked list, delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list. Return the linked list sorted as well.

Practice

65. Rotate List (Medium)

Given the head of a linked list, rotate the list to the right by k places.

Practice

66. Partition List (Medium)

Given the head of a linked list and a value x, partition it such that all nodes less than x come before nodes greater than or equal to x.

You should preserve the original relative order of the nodes in each of the two partitions.

Practice

67. LRU Cache (Medium)

Design a data structure that follows the constraints of a Least Recently Used (LRU) cache.

Implement the LRUCache class:

- LRUCache(int capacity) Initialize the LRU cache with positive size capacity.
- int get(int key) Return the value of the key if the key exists, otherwise return -1.
- void put(int key, int value) Update the value of the key if the key exists. Otherwise, add the key-value pair to the cache. If the number of keys exceeds the capacity from this operation, evict the least recently used key.

The functions get and put must each run in O(1) average time complexity.

Practice

Binary Tree General

68. Maximum Depth of Binary Tree (Easy)

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.

Practice

69. Same Tree (Easy)

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.

Practice

70. Invert Binary Tree (Easy)

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

Practice

71. Symmetric Tree (Easy)

Given the root of a binary tree, check whether it is a mirror of itself (i.e., symmetric around its center).

Practice

72. Construct Binary Tree from Preorder and Inorder Traversal (Medium)

Given two integer arrays preorder and inorder where preorder is the preorder traversal of a binary tree and inorder is the inorder traversal of the same tree, construct and return the binary tree.

Practice

73. Construct Binary Tree from Inorder and Postorder Traversal (Medium)

Given two integer arrays inorder and postorder where inorder is the inorder traversal of a binary tree and postorder is the postorder traversal of the same tree, construct and return the binary tree.

Practice

74. Populating Next Right Pointers in Each Node II (Medium)

Given a binary tree

```
struct Node {
  int val; Node *left;
  Node *right;
  Node *next;
}
```

Populate each next pointer to point to its next right node. If there is no next right node, the next pointer should be set to NULL.

Initially, all next pointers are set to NULL.

Practice

75. Flatten Binary Tree to Linked List (Medium)

Given the root of a binary tree, flatten the tree into a "linked list":

- The "linked list" should use the same TreeNode class where the right child pointer points to the next node in the list and the left child pointer is always null.
- The "linked list" should be in the same order as a pre-order traversal of the binary tree.

Practice

76. Path Sum (Easy)

Given the root of a binary tree and an integer targetSum, return true if the tree has a root-to-leaf path such that adding up all the values along the path equals targetSum.

A leaf is a node with no children.

Practice

77. Sum Root to Leaf Numbers (Medium)

You are given the root of a binary tree containing digits from 0 to 9 only.

Each root-to-leaf path in the tree represents a number.

- For example, the root-to-leaf path 1 → 2 → 3 represents the number 123.

Return the total sum of all root-to-leaf numbers. Test cases are generated so that the answer will fit in a 32-bit integer.

A leaf node is a node with no children.

Practice

78. Binary Tree Maximum Path Sum (Hard)

A path in a binary tree is a sequence of nodes where each pair of adjacent nodes in the sequence has an edge connecting them. A node can only appear in the sequence at most once. Note that the path does not need to pass through the root.

The path sum of a path is the sum of the node's values in the path.

Given the root of a binary tree, return the maximum path sum of any non-empty path.

Practice

79. Binary Search Tree Iterator (Medium)

Implement the BSTIterator class that represents an iterator over the in-order traversal of a binary search tree (BST):

- BSTIterator(TreeNode root) Initializes an object of the BSTIterator class. The root of the BST is given as part of the constructor. The pointer should be initialized to a non-existent number smaller than any element in the BST.
- boolean hasNext() Returns true if there exists a number in the traversal to the right of the pointer, otherwise returns false.
- int next() Moves the pointer to the right, then returns the number at the pointer.

Notice that by initializing the pointer to a non-existent smallest number, the first call to next() will return the smallest element in the BST.

You may assume that next() calls will always be valid. That is, there will be at least a next number in the in-order traversal when next() is called.

Practice

80. Count Complete Tree Nodes (Easy)

Given the root of a complete binary tree, return the number of the nodes in the tree.

According to Wikipedia, every level, except possibly the last, is completely filled in a complete binary tree, and all nodes in the last level are as far left as possible. It can have between 1 and 2h nodes inclusive at the last level h.

Design an algorithm that runs in less than O(n) time complexity.

Practice

81. Lowest Common Ancestor of a Binary Tree (Medium)

Given a binary tree, find the lowest common ancestor (LCA) of two given nodes in the tree.

According to the definition of LCA on Wikipedia: “The lowest common ancestor is defined between two nodes p and q as the lowest node in T that has both p and q as descendants (where we allow a node to be a descendant of itself).”

Practice

Binary Tree BFS

82. Binary Tree Right Side View (Medium)

Given the root of a binary tree, imagine yourself standing on the right side of it, return the values of the nodes you can see ordered from top to bottom.

Practice

83. Average of Levels in Binary Tree (Easy)

Given the root of a binary tree, return the average value of the nodes on each level in the form of an array. Answers within 10-5 of the actual answer will be accepted.

Practice

84. Binary Tree Level Order Traversal (Medium)

Given the root of a binary tree, return the level order traversal of its nodes' values. (i.e., from left to right, level by level).

Practice

85. Binary Tree Zigzag Level Order Traversal (Medium)

Given the root of a binary tree, return the zigzag level order traversal of its nodes' values. (i.e., from left to right, then right to left for the next level and alternate between).

Practice

Binary Tree BFS

86. Minimum Absolute Difference in BST (Easy)

Given the root of a Binary Search Tree (BST), return the minimum absolute difference between the values of any two different nodes in the tree.

Practice

87. Kth Smallest Element in a BST (Medium)

Given the root of a binary search tree, and an integer k, return the kth smallest value (1-indexed) of all the values of the nodes in the tree.

Practice

88. Validate Binary Search Tree (Medium)

Given the root of a binary tree, determine if it is a valid binary search tree (BST).

A valid BST is defined as follows:

- The left subtree of a node contains only nodes with keys strictly less than the node's key.
- The right subtree of a node contains only nodes with keys strictly greater than the node's key.
- Both the left and right subtrees must also be binary search trees.

(A subtree of treeName is a tree consisting of a node in treeName and all of its descendants.)

Practice

Graph General

89. Number of Islands (Medium)

Given an m x 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.

Practice

90. Surrounded Regions (Medium)

You are given an m x n matrix board containing letters 'X' and 'O', capture regions that are surrounded:

- Connect: A cell is connected to adjacent cells horizontally or vertically.
- Region: To form a region connect every 'O' cell.
- Surround: The region is surrounded with 'X' cells if you can connect the region with 'X' cells and none of the region cells are on the edge of the board.

To capture a surrounded region, replace all 'O's with 'X's in-place within the original board. You do not need to return anything.

Practice

91. Clone Graph (Medium)

Given a reference of a node in a connected undirected graph.

Return a deep copy (clone) of the graph.

Each node in the graph contains a value (int) and a list (List[Node]) of its neighbors.

```
class Node {
    public int val;
    public List<Node> neighbors;
}
```

Practice

92. Evaluate Division (Medium)

You are given an array of variable pairs equations and an array of real numbers values, where equations[i] = [Ai, Bi] and values[i] represent the equation Ai / Bi = values[i]. Each Ai or Bi is a string that represents a single variable.

You are also given some queries, where queries[j] = [Cj, Dj] represents the jth query where you must find the answer for Cj / Dj = ?.

Return the answers to all queries. If a single answer cannot be determined, return -1.0.

Note:

The input is always valid. You may assume that evaluating the queries will not result in division by zero and that there is no contradiction.

Note:

The variables that do not occur in the list of equations are undefined, so the answer cannot be determined for them.

Practice

93. Course Schedule (Medium)

There are a total of numCourses courses you have to take, labeled from 0 to numCourses - 1. You are given an array prerequisites where prerequisites[i] = [ai, bi] indicates that you must take course bi first if you want to take course ai.

- For example, the pair [0, 1], indicates that to take course 0 you have to first take course 1.

Return true if you can finish all courses. Otherwise, return false.

Practice

94. Course Schedule II (Medium)

There are a total of numCourses courses you have to take, labeled from 0 to numCourses - 1. You are given an array prerequisites where prerequisites[i] = [ai, bi] indicates that you must take course bi first if you want to take course ai.

- For example, the pair [0, 1], indicates that to take course 0 you have to first take course 1.

Return the ordering of courses you should take to finish all courses. If there are many valid answers, return any of them. If it is impossible to finish all courses, return an empty array.

Practice

Graph BFS

95. Snakes and Ladders (Medium)

You are given an n x n integer matrix board where the cells are labeled from 1 to n2 in a Boustrophedon style starting from the bottom left of the board (i.e. board[n - 1][0]) and alternating direction each row.

Return the least number of dice rolls required to reach the square n2. If it is not possible to reach the square, return -1.

Practice

96. Minimum Genetic Mutation (Medium)

A gene string can be represented by an 8-character long string, with choices from 'A', 'C', 'G', and 'T'.

Suppose we need to investigate a mutation from a gene string startGene to a gene string endGene where one mutation is defined as one single character changed in the gene string.

- For example, "AACCGGTT" → "AACCGGTA" is one mutation.

There is also a gene bank bank that records all the valid gene mutations. A gene must be in bank to make it a valid gene string.

Given the two gene strings startGene and endGene and the gene bank bank, return the minimum number of mutations needed to mutate from startGene to endGene. If there is no such a mutation, return -1.

Note that the starting point is assumed to be valid, so it might not be included in the bank.

Practice

97. Word Ladder (Hard)

A transformation sequence from word beginWord to word endWord using a dictionary wordList is a sequence of words beginWord → s1 → s2 → ... → sk such that:

- Every adjacent pair of words differs by a single letter.
- Every si for 1 <= i <= k is in wordList. Note that beginWord does not need to be in wordList.
- sk == endWord

Given two words, beginWord and endWord, and a dictionary wordList, return the number of words in the shortest transformation sequence from beginWord to endWord, or 0 if no such sequence exists.

Practice

Trie

98. Implement Trie (Prefix Tree) (Medium)

A trie (pronounced as "try") or prefix tree is a tree data structure used to efficiently store and retrieve keys in a dataset of strings. There are various applications of this data structure, such as autocomplete and spellchecker.

Implement the Trie class:

- Trie() Initializes the trie object.
- void insert(String word) Inserts the string word into the trie.
- boolean search(String word) Returns true if the string word is in the trie (i.e., was inserted before), and false otherwise.
- boolean startsWith(String prefix) Returns true if there is a previously inserted string word that has the prefix prefix, and false otherwise.

Practice

99. Design Add and Search Words Data Structure (Medium)

Design a data structure that supports adding new words and finding if a string matches any previously added string.

Implement the WordDictionary class:

- WordDictionary() Initializes the object.
- void addWord(word) Adds word to the data structure, it can be matched later.
- bool search(word) Returns true if there is any string in the data structure that matches word or false otherwise. word may contain dots '.' where dots can be matched with any letter.

Practice

100. Word Search II (Hard)

Given an m x n board of characters and a list of strings words, return all words on the board.

Each word must 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 in a word.

Practice

Backtracking

101. Letter Combinations of a Phone Number (Medium)

Given a string containing digits from 2-9 inclusive, return all possible letter combinations that the number could represent. Return the answer in any order.

A mapping of digits to letters (just like on the telephone buttons) is given below. Note that 1 does not map to any letters.

Practice

102. Combinations (Medium)

Given two integers n and k, return all possible combinations of k numbers chosen from the range [1, n].

You may return the answer in any order.

Practice

103. Permutations (Medium)

Given an array nums of distinct integers, return all the possible permutations. You can return the answer in any order.

Practice

104. Combination Sum (Medium)

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.

Practice

105. N-Queens II (Hard)

The n-queens puzzle is the problem of placing n queens on an n x n chessboard such that no two queens attack each other.

Given an integer n, return the number of distinct solutions to the n-queens puzzle.

Practice

106. Generate Parentheses (Medium)

Given n pairs of parentheses, write a function to generate all combinations of well-formed parentheses.

Practice

107. Word Search (Medium)

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.

Practice

Divide & Conquer

108. Convert Sorted Array to Binary Search Tree (Easy)

Given an integer array nums where the elements are sorted in ascending order, convert it to a height-balanced binary search tree.

(A height-balanced binary tree is a binary tree in which the depth of the two subtrees of every node never differs by more than one.)

Practice

109. Sort List (Medium)

Given the head of a linked list, return the list after sorting it in ascending order.

Practice

110. Construct Quad Tree (Medium)

Given a n * n matrix grid of 0's and 1's only. We want to represent grid with a Quad-Tree.

Return the root of the Quad-Tree representing grid.

Practice

111. Merge k Sorted Lists (Hard)

You are given an array of k linked-lists lists, each linked-list is sorted in ascending order.

Merge all the linked-lists into one sorted linked-list and return it.

Practice

Kadane's Algorithm

112. Maximum Subarray (Medium)

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

Practice

113. Maximum Sum Circular Subarray (Medium)

Given a circular integer array nums of length n, return the maximum possible sum of a non-empty subarray of nums.

A circular array means the end of the array connects to the beginning of the array. Formally, the next element of nums[i] is nums[(i + 1) % n] and the previous element of nums[i] is nums[(i - 1 + n) % n].

A subarray may only include each element of the fixed buffer nums at most once. Formally, for a subarray nums[i], nums[i + 1], ..., nums[j], there does not exist i <= k1, k2 <= j with k1 % n == k2 % n.

Practice

Binary Search

114. Search Insert Position (Easy)

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.

Practice

115. Search a 2D Matrix (Medium)

You are given an m x 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.

Practice

116. Find Peak Element (Medium)

A peak element is an element that is strictly greater than its neighbors.

Given a 0-indexed integer array nums, find a peak element, and return its index. If the array contains multiple peaks, return the index to any of the peaks.

You must write an algorithm that runs in O(log n) time.

Practice

117. Search in Rotated Sorted Array (Medium)

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.

Practice

118. Find First and Last Position of Element in Sorted Array (Medium)

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.

Practice

119. Find Minimum in Rotated Sorted Array (Medium)

Suppose an array of length n sorted in ascending order is rotated between 1 and n times. For example, the array nums = [0,1,2,4,5,6,7] might become:

- [4,5,6,7,0,1,2] if it was rotated 4 times.
- [0,1,2,4,5,6,7] if it was rotated 7 times.

Notice that rotating an array [a[0], a[1], a[2], ..., a[n-1]] 1 time results in the array [a[n-1], a[0], a[1], a[2], ..., a[n-2]].

Given the sorted rotated array nums of unique elements, return the minimum element of this array.

You must write an algorithm that runs in O(log n) time.

Practice

120. Median of Two Sorted Arrays (Hard)

Given two sorted arrays nums1 and nums2 of size m and n respectively, return the median of the two sorted arrays.

The overall run time complexity should be O(log (m+n)).

Practice

Heap

121. Kth Largest Element in an Array (Medium)

Given an integer array nums and an integer k, return the kth largest element in the array.

Note that it is the kth largest element in the sorted order, not the kth distinct element.

Can you solve it without sorting?

Practice

122. IPO (Hard)

Suppose LeetCode will start its IPO soon. In order to sell a good price of its shares to Venture Capital, LeetCode would like to work on some projects to increase its capital before the IPO. Since it has limited resources, it can only finish at most k distinct projects before the IPO. Help LeetCode design the best way to maximize its total capital after finishing at most k distinct projects.

You are given n projects where the ith project has a pure profit profits[i] and a minimum capital of capital[i] is needed to start it.

Initially, you have w capital. When you finish a project, you will obtain its pure profit and the profit will be added to your total capital.

Pick a list of at most k distinct projects from given projects to maximize your final capital, and return the final maximized capital.

The answer is guaranteed to fit in a 32-bit signed integer.

Practice

123. Find K Pairs with Smallest Sums (Medium)

You are given two integer arrays nums1 and nums2 sorted in non-decreasing order and an integer k.

Define a pair (u, v) which consists of one element from the first array and one element from the second array.

Return the k pairs (u1, v1), (u2, v2), ..., (uk, vk) with the smallest sums.

Practice

124. Find Median from Data Stream (Hard)

The median is the middle value in an ordered integer list. If the size of the list is even, there is no middle value, and the median is the mean of the two middle values.

- For example, for arr = [2,3,4], the median is 3.
- For example, for arr = [2,3], the median is (2 + 3) / 2 = 2.5.

Implement the MedianFinder class:

- MedianFinder() initializes the MedianFinder object.
- void addNum(int num) adds the integer num from the data stream to the data structure.
- double findMedian() returns the median of all elements so far. Answers within 10-5 of the actual answer will be accepted.

Practice

Bit Manipulation

125. Add Binary (Easy)

Given two binary strings a and b, return their sum as a binary string.

Practice

126. Reverse Bits (Easy)

Reverse bits of a given 32 bits signed integer.

Practice

127. Number of 1 Bits (Easy)

Reverse bits of a given 32 bits signed integer.

Given a positive integer n, write a function that returns the number of set bits in its binary representation (also known as the Hamming weight).

Practice

128. Single Number (Easy)

Given a non-empty array of integers nums, every element appears twice except for one. Find that single one.

You must implement a solution with a linear runtime complexity and use only constant extra space.

Practice

129. Single Number II (Medium)

Given an integer array nums where every element appears three times except for one, which appears exactly once. Find the single element and return it.

You must implement a solution with a linear runtime complexity and use only constant extra space.

Practice

130. Bitwise AND of Numbers Range (Medium)

Given two integers left and right that represent the range [left, right], return the bitwise AND of all numbers in this range, inclusive.

Practice

Math

131. Palindrome Number (Easy)

Given an integer x, return true if x is a palindrome, and false otherwise.

Practice

132. Plus One (Easy)

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

Practice

133. Factorial Trailing Zeroes (Medium)

Given an integer n, return the number of trailing zeroes in n!.

Note that n! = n * (n - 1) * (n - 2) * ... * 3 * 2 * 1.

Practice

134. Sqrt(x) (Easy)

Given a non-negative integer x, return the square root of x rounded down to the nearest integer. The returned integer should be non-negative as well.

You must not use any built-in exponent function or operator.

- For example, do not use pow(x, 0.5) in c++ or x ** 0.5 in python.

Practice

135. Pow(x, n) (Medium)

Implement pow(x, n), which calculates x raised to the power n (i.e., xn).

Practice

136. Max Points on a Line (Hard)

Given an array of points where points[i] = [xi, yi] represents a point on the X-Y plane, return the maximum number of points that lie on the same straight line.

Practice

1D DP

137. Climbing Stairs (Easy)

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?

Practice

138. House Robber (Medium)

You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have security systems connected and it will automatically contact the police if two adjacent houses were broken into on the same night.

Given an integer array nums representing the amount of money of each house, return the maximum amount of money you can rob tonight without alerting the police.

Practice

139. Word Break (Medium)

Given a string s and a dictionary of strings wordDict, return true if s can be segmented into a space-separated sequence of one or more dictionary words.

Note that the same word in the dictionary may be reused multiple times in the segmentation.

Practice

140. Coin Change (Medium)

You are given an integer array coins representing coins of different denominations and an integer amount representing a total amount of money.

Return the fewest number of coins that you need to make up that amount. If that amount of money cannot be made up by any combination of the coins, return -1.

You may assume that you have an infinite number of each kind of coin.

Practice

141. Longest Increasing Subsequence (Medium)

Given an integer array nums, return the length of the longest strictly increasing subsequence.

Practice

Multidimensional DP

142. Triangle (Medium)

Given a triangle array, return the minimum path sum from top to bottom.

For each step, you may move to an adjacent number of the row below. More formally, if you are on index i on the current row, you may move to either index i or index i + 1 on the next row.

Practice

143. Minimum Path Sum (Medium)

Given a m x n grid filled with non-negative numbers, find a path from top left to bottom right, which minimizes the sum of all numbers along its path.

Note: You can only move either down or right at any point in time.

Practice

144. Unique Paths II (Medium)

You are given an m x n integer array grid. There is a robot 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.

An obstacle and space are marked as 1 or 0 respectively in grid. A path that the robot takes cannot include any square that is an obstacle.

Return the number of possible unique paths that the robot can take to reach the bottom-right corner.

The testcases are generated so that the answer will be less than or equal to 2 * 109.

Practice

145. Longest Palindromic Substring (Medium)

Given a string s, return the longest palindromic substring in s.

Practice

146. Interleaving String (Medium)

Given strings s1, s2, and s3, find whether s3 is formed by an interleaving of s1 and s2.

An interleaving of two strings s and t is a configuration where s and t are divided into n and m substrings respectively, such that:

- s = s1 + s2 + ... + sn
- t = t1 + t2 + ... + tm
- |n - m| <= 1
- The interleaving is s1 + t1 + s2 + t2 + s3 + t3 + ... or t1 + s1 + t2 + s2 + t3 + s3 + ...

Note:

a + b is the concatenation of strings a and b.

Practice

147. Edit Distance (Medium)

Given two strings word1 and word2, return the minimum number of operations required to convert word1 to word2.

You have the following three operations permitted on a word:

- Insert a character
- Delete a character
- Replace a character

Practice

148. Best Time to Buy and Sell Stock III (Hard)

You are given an array prices where prices[i] is the price of a given stock on the ith day.

Find the maximum profit you can achieve. You may complete at most two transactions.

Note: You may not engage in multiple transactions simultaneously (i.e., you must sell the stock before you buy again).

Practice

149. Best Time to Buy and Sell Stock IV (Hard)

You are given an integer array prices where prices[i] is the price of a given stock on the ith day, and an integer k.

Find the maximum profit you can achieve. You may complete at most k transactions: i.e. you may buy at most k times and sell at most k times.

Note:

You may not engage in multiple transactions simultaneously (i.e., you must sell the stock before you buy again).

Practice

150. Maximal Square (Medium)

Given an m x n binary matrix filled with 0's and 1's, find the largest square containing only 1's and return its area.

Practice

BOSSCODER ACADEMY

WHY BOSSCODER?

2200+ Alumni placed at Top Product-based companies.

More than **120% hike** for every 2 out of 3 Working Professional.

Average Package of **24LPA.**

The syllabus is most **up-to-date** and the list of problems provided covers **all important topics.**

Lavanya
Meta

Course is very well **structured** and **streamlined** to crack any **MAANG** company .

Rahul
Google

EXPLORE MORE