Contents

[**138. Copy List with Random Pointer** 3](#_Toc76192397)

[**200. Number of Islands** 6](#_Toc76192398)

[**973. K Closest Points to Origin** 7](#_Toc76192399)

[**763. Partition Labels** 9](#_Toc76192400)

[**5. Longest Palindromic Substring** 10](#_Toc76192401)

[994. Rotting Oranges 10](#_Toc76192402)

[**1041. Robot Bounded In Circle** 12](#_Toc76192403)

[**692. Top K Frequent Words** 13](#_Toc76192404)

[**253. Meeting Rooms II** 14](#_Toc76192405)

[**1152. Analyze User Website Visit Pattern** 15](#_Toc76192406)

[**56. Merge Intervals** 17](#_Toc76192407)

[**3. Longest Substring Without Repeating Characters** 17](#_Toc76192408)

[957. Prison Cells After N Days 18](#_Toc76192409)

[**48. Rotate Image** 19](#_Toc76192410)

[**103. Binary Tree Zigzag Level Order Traversal** 21](#_Toc76192411)

[**17. Letter Combinations of a Phone Number** 22](#_Toc76192412)

[**79. Word Search** 23](#_Toc76192413)

[**11. Container With Most Water** 25](#_Toc76192414)

[**380. Insert Delete GetRandom O(1)** 27](#_Toc76192415)

[**207. Course Schedule** 28](#_Toc76192416)

[**347. Top K Frequent Elements** 29](#_Toc76192417)

[**54. Spiral Matrix** 30](#_Toc76192418)

[**210. Course Schedule II** 31](#_Toc76192419)

[**12. Integer to Roman** 32](#_Toc76192420)

[**1010. Pairs of Songs With Total Durations Divisible by 60** 34](#_Toc76192421)

[**322. Coin Change** 35](#_Toc76192422)

[**236. Lowest Common Ancestor of a Binary Tree** 36](#_Toc76192423)

[**221. Maximal Square** 37](#_Toc76192424)

[**863. All Nodes Distance K in Binary Tree** 39](#_Toc76192425)

[**22. Generate Parentheses** 40](#_Toc76192426)

[**348. Design Tic-Tac-Toe** 41](#_Toc76192427)

[**227. Basic Calculator II** 43](#_Toc76192428)

[45. Jump Game II 44](#_Toc76192429)

[**8. String to Integer (atoi)** 45](#_Toc76192430)

[**1465. Maximum Area of a Piece of Cake After Horizontal and Vertical Cuts** 47](#_Toc76192431)

[**98. Validate Binary Search Tree** 49](#_Toc76192432)

[**323. Number of Connected Components in an Undirected Graph** 51](#_Toc76192433)

[**560. Subarray Sum Equals K** 52](#_Toc76192434)

[**55. Jump Game** 52](#_Toc76192435)

[**1167. Minimum Cost to Connect Sticks** 53](#_Toc76192436)

[**215. Kth Largest Element in an Array** 54](#_Toc76192437)

[**909. Snakes and Ladders** 55](#_Toc76192438)

[**937. Reorder Data in Log Files** 57](#_Toc76192439)

[**146. LRU Cache** 58](#_Toc76192440)

[**1192. Critical Connections in a Network** 60](#_Toc76192441)

[**819. Most Common Word** 61](#_Toc76192442)

[**42. Trapping Rain Water** 62](#_Toc76192443)

[**127. Word Ladder** 62](#_Toc76192444)

[**23. Merge k Sorted Lists** 63](#_Toc76192445)

[**295. Find Median from Data Stream** 65](#_Toc76192446)

[**103. Binary Tree Zigzag Level Order Traversal** 66](#_Toc76192447)

[**139. Word Break** 67](#_Toc76192448)

[**238. Product of Array Except Self** 68](#_Toc76192449)

[**33. Search in Rotated Sorted Array** 69](#_Toc76192450)

[**49. Group Anagrams** 70](#_Toc76192451)

[**4. Median of Two Sorted Arrays** 71](#_Toc76192452)

[**212. Word Search II** 72](#_Toc76192453)

[**1335. Minimum Difficulty of a Job Schedule** 73](#_Toc76192454)

[**240. Search a 2D Matrix II** 75](#_Toc76192455)

[**126. Word Ladder II** 77](#_Toc76192456)

[**239. Sliding Window Maximum** 78](#_Toc76192457)

[**140. Word Break II** 79](#_Toc76192458)

[**472. Concatenated Words** 80](#_Toc76192459)

[**297. Serialize and Deserialize Binary Tree** 81](#_Toc76192460)

Amazon Leetcode Questions (medium difficulty)

# [**138. Copy List with Random Pointer**](https://leetcode-cn.com/problems/copy-list-with-random-pointer/)

难度中等553收藏分享切换为中文接收动态反馈

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**](https://en.wikipedia.org/wiki/Object_copying#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.

The linked list is represented in the input/output as a list of n nodes. Each node is represented as a pair of [val, random\_index] where:

* val: an integer representing Node.val
* random\_index: the index of the node (range from 0 to n-1) that the random pointer points to, or null if it does not point to any node.

Your code will **only** be given the head of the original linked list.

**Example 1:**



**Input:** head = [[7,null],[13,0],[11,4],[10,2],[1,0]]

**Output:** [[7,null],[13,0],[11,4],[10,2],[1,0]]

**Example 2:**



**Input:** head = [[1,1],[2,1]]

**Output:** [[1,1],[2,1]]

**Example 3:**

****

**Input:** head = [[3,null],[3,0],[3,null]]

**Output:** [[3,null],[3,0],[3,null]]

**Example 4:**

**Input:** head = []

**Output:** []

**Explanation:** The given linked list is empty (null pointer), so return null.

**Constraints:**

* 0 <= n <= 1000
* -10000 <= Node.val <= 10000
* Node.random is null or is pointing to some node in the linked list.

通过次数69,381

提交次数113,875

/\*

// Definition for a Node.

class Node {

    int val;

    Node next;

    Node random;

    public Node(int val) {

        this.val = val;

        this.next = null;

        this.random = null;

    }

}

\*/

class Solution {

    public Node copyRandomList(Node head) {

    }

}

# [**200. Number of Islands**](https://leetcode-cn.com/problems/number-of-islands/)

难度中等1114收藏分享切换为中文接收动态反馈

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.

**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

**Constraints:**

* m == grid.length
* n == grid[i].length
* 1 <= m, n <= 300
* grid[i][j] is '0' or '1'.

通过次数239,144

提交次数447,402

class Solution {

    public int numIslands(char[][] grid) {

    }

}

# [**973. K Closest Points to Origin**](https://leetcode-cn.com/problems/k-closest-points-to-origin/)

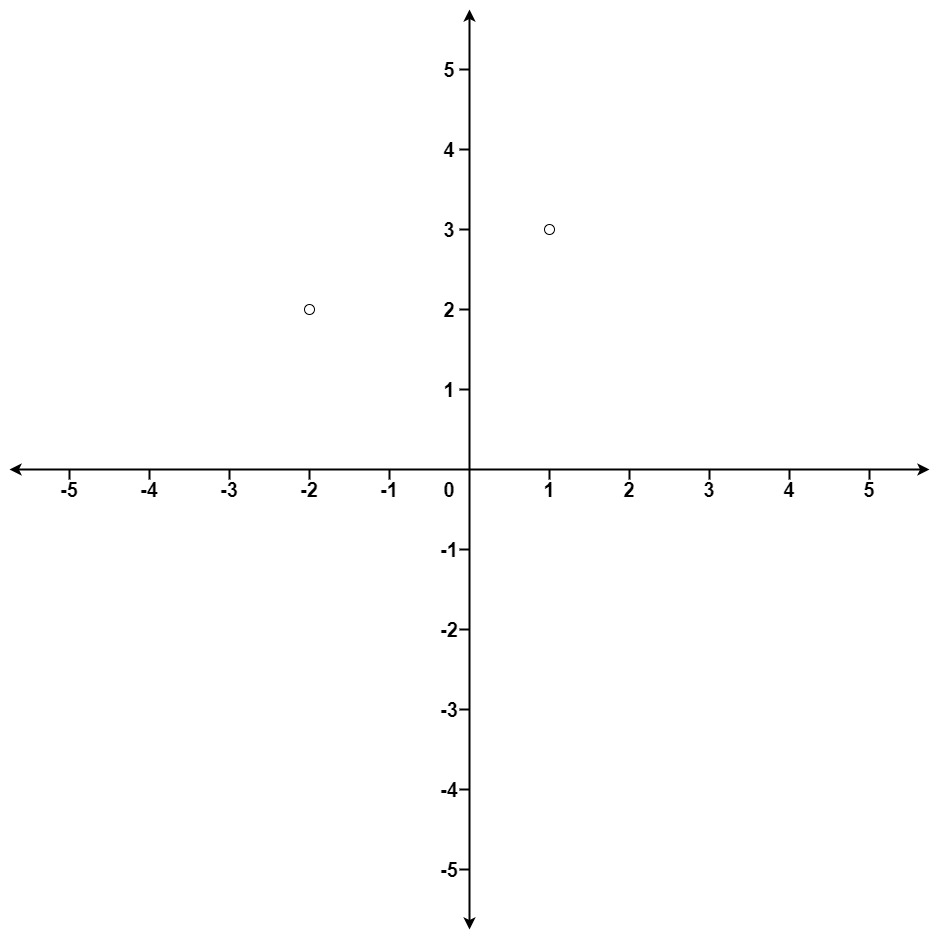
难度中等241收藏分享切换为中文接收动态反馈

Given an array of points where points[i] = [xi, yi] represents a point on the **X-Y** plane and an integer k, return the k closest points to the origin (0, 0).

The distance between two points on the **X-Y** plane is the Euclidean distance (i.e., √(x1 - x2)2 + (y1 - y2)2).

You may return the answer in **any order**. The answer is **guaranteed** to be **unique** (except for the order that it is in).

**Example 1:**



**Input:** points = [[1,3],[-2,2]], k = 1

**Output:** [[-2,2]]

**Explanation:**

The distance between (1, 3) and the origin is sqrt(10).

The distance between (-2, 2) and the origin is sqrt(8).

Since sqrt(8) < sqrt(10), (-2, 2) is closer to the origin.

We only want the closest k = 1 points from the origin, so the answer is just [[-2,2]].

**Example 2:**

**Input:** points = [[3,3],[5,-1],[-2,4]], k = 2

**Output:** [[3,3],[-2,4]]

**Explanation:** The answer [[-2,4],[3,3]] would also be accepted.

**Constraints:**

* 1 <= k <= points.length <= 104
* -104 < xi, yi < 104

通过次数60,114

提交次数94,652

class Solution {

    public int[][] kClosest(int[][] points, int k) {

    }

}

# [**763. Partition Labels**](https://leetcode-cn.com/problems/partition-labels/)

难度中等487收藏分享切换为中文接收动态反馈

A string S of lowercase English letters is given. We want to partition this string into as many parts as possible so that each letter appears in at most one part, and return a list of integers representing the size of these parts.

**Example 1:**

**Input:** S = "ababcbacadefegdehijhklij"

**Output:** [9,7,8]

**Explanation:**

The partition is "ababcbaca", "defegde", "hijhklij".

This is a partition so that each letter appears in at most one part.

A partition like "ababcbacadefegde", "hijhklij" is incorrect, because it splits S into less parts.

**Note:**

* S will have length in range [1, 500].
* S will consist of lowercase English letters ('a' to 'z') only.

通过次数63,394

提交次数83,125

class Solution {

    public List<Integer> partitionLabels(String S) {

    }

}

# [**5. Longest Palindromic Substring**](https://leetcode-cn.com/problems/longest-palindromic-substring/)

难度中等3547收藏分享切换为中文关闭提醒反馈

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

**Example 1:**

**Input:** s = "babad"

**Output:** "bab"

**Note:** "aba" is also a valid answer.

**Example 2:**

**Input:** s = "cbbd"

**Output:** "bb"

**Example 3:**

**Input:** s = "a"

**Output:** "a"

**Example 4:**

**Input:** s = "ac"

**Output:** "a"

**Constraints:**

* 1 <= s.length <= 1000
* s consist of only digits and English letters (lower-case and/or upper-case),

通过次数557,653

提交次数1,640,945

class Solution {

    public String longestPalindrome(String s) {

    }

}

# [994. Rotting Oranges](https://leetcode-cn.com/problems/rotting-oranges/)

难度中等351收藏分享切换为中文接收动态反馈

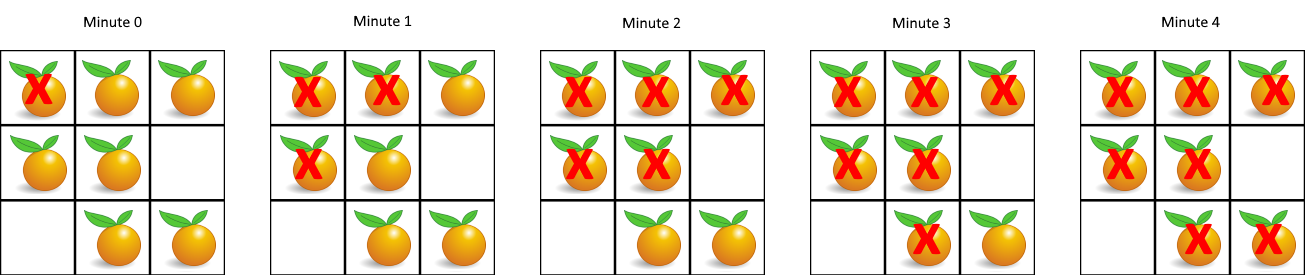
You are given an m x n grid where each cell can have one of three values:

* 0 representing an empty cell,
* 1 representing a fresh orange, or
* 2 representing a rotten orange.

Every minute, any fresh orange that is **4-directionally adjacent** to a rotten orange becomes rotten.

Return the minimum number of minutes that must elapse until no cell has a fresh orange. If this is impossible, return -1.

**Example 1:**



**Input:** grid = [[2,1,1],[1,1,0],[0,1,1]]

**Output:** 4

**Example 2:**

**Input:** grid = [[2,1,1],[0,1,1],[1,0,1]]

**Output:** -1

**Explanation:** The orange in the bottom left corner (row 2, column 0) is never rotten, because rotting only happens 4-directionally.

**Example 3:**

**Input:** grid = [[0,2]]

**Output:** 0

**Explanation:** Since there are already no fresh oranges at minute 0, the answer is just 0.

**Constraints:**

* m == grid.length
* n == grid[i].length
* 1 <= m, n <= 10
* grid[i][j] is 0, 1, or 2.

通过次数40,587

提交次数79,989

class Solution {

    public int orangesRotting(int[][] grid) {

    }

}

# [**1041. Robot Bounded In Circle**](https://leetcode-cn.com/problems/robot-bounded-in-circle/)

难度中等66收藏分享切换为中文接收动态反馈

On an infinite plane, a robot initially stands at (0, 0) and faces north. The robot can receive one of three instructions:

* "G": go straight 1 unit;
* "L": turn 90 degrees to the left;
* "R": turn 90 degrees to the right.

The robot performs the instructions given in order, and repeats them forever.

Return true if and only if there exists a circle in the plane such that the robot never leaves the circle.

**Example 1:**

**Input:** instructions = "GGLLGG"

**Output:** true

**Explanation:** The robot moves from (0,0) to (0,2), turns 180 degrees, and then returns to (0,0).

When repeating these instructions, the robot remains in the circle of radius 2 centered at the origin.

**Example 2:**

**Input:** instructions = "GG"

**Output:** false

**Explanation:** The robot moves north indefinitely.

**Example 3:**

**Input:** instructions = "GL"

**Output:** true

**Explanation:** The robot moves from (0, 0) -> (0, 1) -> (-1, 1) -> (-1, 0) -> (0, 0) -> ...

**Constraints:**

* 1 <= instructions.length <= 100
* instructions[i] is 'G', 'L' or, 'R'.

通过次数5,463

提交次数11,565

class Solution {

    public boolean isRobotBounded(String instructions) {

    }

}

# [**692. Top K Frequent Words**](https://leetcode-cn.com/problems/top-k-frequent-words/)

难度中等238收藏分享切换为中文接收动态反馈

Given a non-empty list of words, return the *k* most frequent elements.

Your answer should be sorted by frequency from highest to lowest. If two words have the same frequency, then the word with the lower alphabetical order comes first.

**Example 1:**

**Input:** ["i", "love", "leetcode", "i", "love", "coding"], k = 2

**Output:** ["i", "love"]

**Explanation:** "i" and "love" are the two most frequent words.

Note that "i" comes before "love" due to a lower alphabetical order.

**Example 2:**

**Input:** ["the", "day", "is", "sunny", "the", "the", "the", "sunny", "is", "is"], k = 4

**Output:** ["the", "is", "sunny", "day"]

**Explanation:** "the", "is", "sunny" and "day" are the four most frequent words,

with the number of occurrence being 4, 3, 2 and 1 respectively.

**Note:**

1. You may assume *k* is always valid, 1 ≤ *k* ≤ number of unique elements.
2. Input words contain only lowercase letters.

**Follow up:**

1. Try to solve it in *O*(*n* log *k*) time and *O*(*n*) extra space.

通过次数27,764

提交次数53,084

class Solution {

    public List<String> topKFrequent(String[] words, int k) {

    }

}

# [**253. Meeting Rooms II**](https://leetcode-cn.com/problems/meeting-rooms-ii/)

难度中等254收藏分享切换为中文接收动态反馈

Given an array of meeting time intervals intervals where intervals[i] = [starti, endi], return the minimum number of conference rooms required.

**Example 1:**

**Input:** intervals = [[0,30],[5,10],[15,20]]

**Output:** 2

**Example 2:**

**Input:** intervals = [[7,10],[2,4]]

**Output:** 1

# 

**Constraints:**

1 <= intervals.length <= 104

0 <= starti < endi <= 106

通过次数22,927

提交次数47,371

class Solution {

    public int minMeetingRooms(int[][] intervals) {

    }

}

#### [139. Word Break](https://leetcode-cn.com/problems/word-break/)

难度中等948收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** s = "leetcode", wordDict = ["leet","code"]

**Output:** true

**Explanation:** Return true because "leetcode" can be segmented as "leet code".

**Example 2:**

**Input:** s = "applepenapple", wordDict = ["apple","pen"]

**Output:** true

**Explanation:** Return true because "applepenapple" can be segmented as "apple pen apple".

Note that you are allowed to reuse a dictionary word.

**Example 3:**

**Input:** s = "catsandog", wordDict = ["cats","dog","sand","and","cat"]

**Output:** false

**Constraints:**

* 1 <= s.length <= 300
* 1 <= wordDict.length <= 1000
* 1 <= wordDict[i].length <= 20
* s and wordDict[i] consist of only lowercase English letters.
* All the strings of wordDict are **unique**.

通过次数137,096

提交次数275,203

class Solution {

    public boolean wordBreak(String s, List<String> wordDict) {

    }

}

# [**1152. Analyze User Website Visit Pattern**](https://leetcode-cn.com/problems/analyze-user-website-visit-pattern/)

难度中等9收藏分享切换为中文接收动态反馈

We are given some website visits: the user with name username[i] visited the website website[i] at time timestamp[i].

A 3-sequence is a list of websites of length 3 sorted in ascending order by the time of their visits.  (The websites in a 3-sequence are not necessarily distinct.)

Find the 3-sequence visited by the largest number of users. If there is more than one solution, return the lexicographically smallest such 3-sequence.

**Example 1:**

**Input:** username = ["joe","joe","joe","james","james","james","james","mary","mary","mary"], timestamp = [1,2,3,4,5,6,7,8,9,10], website = ["home","about","career","home","cart","maps","home","home","about","career"]

**Output:** ["home","about","career"]

**Explanation:**

The tuples in this example are:

["joe", 1, "home"]

["joe", 2, "about"]

["joe", 3, "career"]

["james", 4, "home"]

["james", 5, "cart"]

["james", 6, "maps"]

["james", 7, "home"]

["mary", 8, "home"]

["mary", 9, "about"]

["mary", 10, "career"]

The 3-sequence ("home", "about", "career") was visited at least once by **2** users.

The 3-sequence ("home", "cart", "maps") was visited at least once by 1 user.

The 3-sequence ("home", "cart", "home") was visited at least once by 1 user.

The 3-sequence ("home", "maps", "home") was visited at least once by 1 user.

The 3-sequence ("cart", "maps", "home") was visited at least once by 1 user.

**Note:**

1. 3 <= N = username.length = timestamp.length = website.length <= 50
2. 1 <= username[i].length <= 10
3. 0 <= timestamp[i] <= 10^9
4. 1 <= website[i].length <= 10
5. Both username[i] and website[i] contain only lowercase characters.
6. It is guaranteed that there is at least one user who visited at least 3 websites.
7. No user visits two websites at the same time.

通过次数1,065

提交次数2,846

class Solution {

    public List<String> mostVisitedPattern(String[] username, int[] timestamp, String[] website) {

    }

}

# [**56. Merge Intervals**](https://leetcode-cn.com/problems/merge-intervals/)

难度中等910收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** intervals = [[1,3],[2,6],[8,10],[15,18]]

**Output:** [[1,6],[8,10],[15,18]]

**Explanation:** Since intervals [1,3] and [2,6] overlaps, merge them into [1,6].

**Example 2:**

**Input:** intervals = [[1,4],[4,5]]

**Output:** [[1,5]]

**Explanation:** Intervals [1,4] and [4,5] are considered overlapping.

**Constraints:**

* 1 <= intervals.length <= 104
* intervals[i].length == 2
* 0 <= starti <= endi <= 104

通过次数222,111

提交次数490,947

class Solution {

    public int[][] merge(int[][] intervals) {

    }

}

# [**3. Longest Substring Without Repeating Characters**](https://leetcode-cn.com/problems/longest-substring-without-repeating-characters/)

难度中等5349收藏分享切换为中文接收动态反馈

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 = "bbbbb"

**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.

**Example 4:**

**Input:** s = ""

**Output:** 0

**Constraints:**

* 0 <= s.length <= 5 \* 104
* s consists of English letters, digits, symbols and spaces.

通过次数951,682

提交次数2,574,609

class Solution {

    public int lengthOfLongestSubstring(String s) {

    }

}

# [957. Prison Cells After N Days](https://leetcode-cn.com/problems/prison-cells-after-n-days/)

难度中等90收藏分享切换为中文接收动态反馈

There are 8 prison cells in a row and each cell is either occupied or vacant.

Each day, whether the cell is occupied or vacant changes according to the following rules:

* If a cell has two adjacent neighbors that are both occupied or both vacant, then the cell becomes occupied.
* Otherwise, it becomes vacant.

**Note** that because the prison is a row, the first and the last cells in the row can't have two adjacent neighbors.

You are given an integer array cells where cells[i] == 1 if the ith cell is occupied and cells[i] == 0 if the ith cell is vacant, and you are given an integer n.

Return the state of the prison after n days (i.e., n such changes described above).

**Example 1:**

**Input:** cells = [0,1,0,1,1,0,0,1], n = 7

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

**Explanation:** The following table summarizes the state of the prison on each day:

Day 0: [0, 1, 0, 1, 1, 0, 0, 1]

Day 1: [0, 1, 1, 0, 0, 0, 0, 0]

Day 2: [0, 0, 0, 0, 1, 1, 1, 0]

Day 3: [0, 1, 1, 0, 0, 1, 0, 0]

Day 4: [0, 0, 0, 0, 0, 1, 0, 0]

Day 5: [0, 1, 1, 1, 0, 1, 0, 0]

Day 6: [0, 0, 1, 0, 1, 1, 0, 0]

Day 7: [0, 0, 1, 1, 0, 0, 0, 0]

**Example 2:**

**Input:** cells = [1,0,0,1,0,0,1,0], n = 1000000000

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

**Constraints:**

* cells.length == 8
* cells[i] is either 0 or 1.
* 1 <= n <= 109

通过次数12,444

提交次数35,067

class Solution {

    public int[] prisonAfterNDays(int[] cells, int n) {

    }

}

# [**48. Rotate Image**](https://leetcode-cn.com/problems/rotate-image/)

难度中等860收藏分享切换为中文接收动态反馈

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**](https://en.wikipedia.org/wiki/In-place_algorithm), which means you have to modify the input 2D matrix directly. **DO NOT** allocate another 2D matrix and do the rotation.

**Example 1:**



**Input:** matrix = [[1,2,3],[4,5,6],[7,8,9]]

**Output:** [[7,4,1],[8,5,2],[9,6,3]]

**Example 2:**



**Input:** matrix = [[5,1,9,11],[2,4,8,10],[13,3,6,7],[15,14,12,16]]

**Output:** [[15,13,2,5],[14,3,4,1],[12,6,8,9],[16,7,10,11]]

**Example 3:**

**Input:** matrix = [[1]]

**Output:** [[1]]

**Example 4:**

**Input:** matrix = [[1,2],[3,4]]

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

**Constraints:**

* matrix.length == n
* matrix[i].length == n
* 1 <= n <= 20
* -1000 <= matrix[i][j] <= 1000

通过次数164,382

提交次数226,252

class Solution {

    public void rotate(int[][] matrix) {

    }

}

# [**103. Binary Tree Zigzag Level Order Traversal**](https://leetcode-cn.com/problems/binary-tree-zigzag-level-order-traversal/)

难度中等436收藏分享切换为中文接收动态反馈

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).

**Example 1:**



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

**Output:** [[3],[20,9],[15,7]]

**Example 2:**

**Input:** root = [1]

**Output:** [[1]]

**Example 3:**

**Input:** root = []

**Output:** []

**Constraints:**

* The number of nodes in the tree is in the range [0, 2000].
* -100 <= Node.val <= 100

通过次数132,481

提交次数231,961

/\*\*

 \* Definition for a binary tree node.

 \* public class TreeNode {

 \*     int val;

 \*     TreeNode left;

 \*     TreeNode right;

 \*     TreeNode() {}

 \*     TreeNode(int val) { this.val = val; }

 \*     TreeNode(int val, TreeNode left, TreeNode right) {

 \*         this.val = val;

 \*         this.left = left;

 \*         this.right = right;

 \*     }

 \* }

 \*/

class Solution {

    public List<List<Integer>> zigzagLevelOrder(TreeNode root) {

    }

}

# [**17. Letter Combinations of a Phone Number**](https://leetcode-cn.com/problems/letter-combinations-of-a-phone-number/)

难度中等1263收藏分享切换为中文接收动态反馈

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



**Example 1:**

**Input:** digits = "23"

**Output:** ["ad","ae","af","bd","be","bf","cd","ce","cf"]

**Example 2:**

**Input:** digits = ""

**Output:** []

**Example 3:**

**Input:** digits = "2"

**Output:** ["a","b","c"]

**Constraints:**

* 0 <= digits.length <= 4
* digits[i] is a digit in the range ['2', '9'].

class Solution {

    public List<String> letterCombinations(String digits) {

    }

}

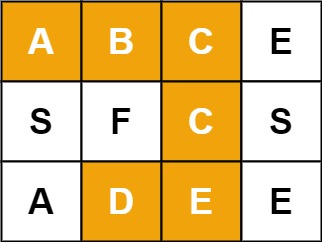
# [**79. Word Search**](https://leetcode-cn.com/problems/word-search/)

难度中等871收藏分享切换为中文接收动态反馈

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

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

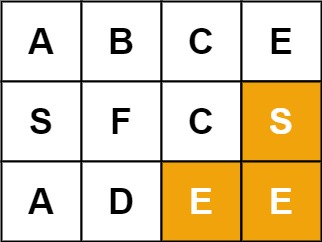
**Example 1:**



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

**Output:** true

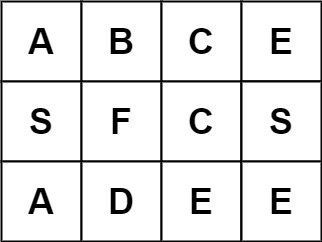
**Example 2:**



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

**Output:** true

**Example 3:**



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

**Output:** false

**Constraints:**

* m == board.length
* n = board[i].length
* 1 <= m, n <= 6
* 1 <= word.length <= 15
* board and word consists of only lowercase and uppercase English letters.

**Follow up:** Could you use search pruning to make your solution faster with a larger board?

通过次数163,879

提交次数366,701

class Solution {

    public boolean exist(char[][] board, String word) {

    }

}

# [**11. Container With Most Water**](https://leetcode-cn.com/problems/container-with-most-water/)

难度中等2389收藏分享切换为中文接收动态反馈

Given n non-negative integers a1, a2, ..., an, where each represents a point at coordinate (i, ai). n vertical lines are drawn such that the two endpoints of the line i is at (i, ai) and (i, 0). Find two lines, which, together with the x-axis forms a container, such that the container contains the most water.

**Notice** that you may not slant the container.

**Example 1:**



**Input:** height = [1,8,6,2,5,4,8,3,7]

**Output:** 49

**Explanation:** The above vertical lines are represented by array [1,8,6,2,5,4,8,3,7]. In this case, the max area of water (blue section) the container can contain is 49.

**Example 2:**

**Input:** height = [1,1]

**Output:** 1

**Example 3:**

**Input:** height = [4,3,2,1,4]

**Output:** 16

**Example 4:**

**Input:** height = [1,2,1]

**Output:** 2

**Constraints:**

* n == height.length
* 2 <= n <= 105
* 0 <= height[i] <= 104

通过次数420,807

提交次数654,861

class Solution {

    public int maxArea(int[] height) {

    }

}

# [**380. Insert Delete GetRandom O(1)**](https://leetcode-cn.com/problems/insert-delete-getrandom-o1/)

难度中等305收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input**

["RandomizedSet", "insert", "remove", "insert", "getRandom", "remove", "insert", "getRandom"]

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

**Output**

[null, true, false, true, 2, true, false, 2]

**Explanation**

RandomizedSet randomizedSet = new RandomizedSet();

randomizedSet.insert(1); // Inserts 1 to the set. Returns true as 1 was inserted successfully.

randomizedSet.remove(2); // Returns false as 2 does not exist in the set.

randomizedSet.insert(2); // Inserts 2 to the set, returns true. Set now contains [1,2].

randomizedSet.getRandom(); // getRandom() should return either 1 or 2 randomly.

randomizedSet.remove(1); // Removes 1 from the set, returns true. Set now contains [2].

randomizedSet.insert(2); // 2 was already in the set, so return false.

randomizedSet.getRandom(); // Since 2 is the only number in the set, getRandom() will always return 2.

**Constraints:**

* -231 <= val <= 231 - 1
* At most 105 calls will be made to insert, remove, and getRandom.
* There will be **at least one** element in the data structure when getRandom is called.

**Follow up:** Could you implement the functions of the class with each function works in **average** O(1) time?

通过次数25,657

提交次数51,627

class RandomizedSet {

    /\*\* Initialize your data structure here. \*/

    public RandomizedSet() {

    }

    /\*\* Inserts a value to the set. Returns true if the set did not already contain the specified element. \*/

    public boolean insert(int val) {

    }

    /\*\* Removes a value from the set. Returns true if the set contained the specified element. \*/

    public boolean remove(int val) {

    }

    /\*\* Get a random element from the set. \*/

    public int getRandom() {

    }

}

/\*\*

 \* Your RandomizedSet object will be instantiated and called as such:

 \* RandomizedSet obj = new RandomizedSet();

 \* boolean param\_1 = obj.insert(val);

 \* boolean param\_2 = obj.remove(val);

 \* int param\_3 = obj.getRandom();

 \*/

# [**207. Course Schedule**](https://leetcode-cn.com/problems/course-schedule/)

难度中等784收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** numCourses = 2, prerequisites = [[1,0]]

**Output:** true

**Explanation:** There are a total of 2 courses to take.

To take course 1 you should have finished course 0. So it is possible.

**Example 2:**

**Input:** numCourses = 2, prerequisites = [[1,0],[0,1]]

**Output:** false

**Explanation:** There are a total of 2 courses to take.

To take course 1 you should have finished course 0, and to take course 0 you should also have finished course 1. So it is impossible.

**Constraints:**

* 1 <= numCourses <= 105
* 0 <= prerequisites.length <= 5000
* prerequisites[i].length == 2
* 0 <= ai, bi < numCourses
* All the pairs prerequisites[i] are **unique**.

通过次数106,404

提交次数194,493

class Solution {

    public boolean canFinish(int numCourses, int[][] prerequisites) {

    }

}

# [**347. Top K Frequent Elements**](https://leetcode-cn.com/problems/top-k-frequent-elements/)

难度中等729收藏分享切换为中文接收动态反馈

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
* k is in the range [1, the number of unique elements in the array].
* It is **guaranteed** that the answer is **unique**.

**Follow up:** Your algorithm's time complexity must be better than O(n log n), where n is the array's size.

通过次数153,311

提交次数247,446

class Solution {

    public int[] topKFrequent(int[] nums, int k) {

    }

}

# [**54. Spiral Matrix**](https://leetcode-cn.com/problems/spiral-matrix/)

难度中等758收藏分享切换为中文接收动态反馈

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

**Example 1:**



**Input:** matrix = [[1,2,3],[4,5,6],[7,8,9]]

**Output:** [1,2,3,6,9,8,7,4,5]

**Example 2:**



**Input:** matrix = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]

**Output:** [1,2,3,4,8,12,11,10,9,5,6,7]

**Constraints:**

* m == matrix.length
* n == matrix[i].length
* 1 <= m, n <= 10
* -100 <= matrix[i][j] <= 100

通过次数147,977

提交次数317,288

class Solution {

    public List<Integer> spiralOrder(int[][] matrix) {

    }

}

# [**210. Course Schedule II**](https://leetcode-cn.com/problems/course-schedule-ii/)

难度中等390收藏分享切换为中文接收动态反馈

There are a total of n courses you have to take labelled from 0 to n - 1.

Some courses may have prerequisites, for example, if prerequisites[i] = [ai, bi] this means you must take the course bi before the course ai.

Given the total number of courses numCourses and a list of the prerequisite pairs, 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**.

**Example 1:**

**Input:** numCourses = 2, prerequisites = [[1,0]]

**Output:** [0,1]

**Explanation:** There are a total of 2 courses to take. To take course 1 you should have finished course 0. So the correct course order is [0,1].

**Example 2:**

**Input:** numCourses = 4, prerequisites = [[1,0],[2,0],[3,1],[3,2]]

**Output:** [0,2,1,3]

**Explanation:** There are a total of 4 courses to take. To take course 3 you should have finished both courses 1 and 2. Both courses 1 and 2 should be taken after you finished course 0.

So one correct course order is [0,1,2,3]. Another correct ordering is [0,2,1,3].

**Example 3:**

**Input:** numCourses = 1, prerequisites = []

**Output:** [0]

**Constraints:**

* 1 <= numCourses <= 2000
* 0 <= prerequisites.length <= numCourses \* (numCourses - 1)
* prerequisites[i].length == 2
* 0 <= ai, bi < numCourses
* ai != bi
* All the pairs [ai, bi] are **distinct**.

通过次数67,393

提交次数126,813

class Solution {

    public int[] findOrder(int numCourses, int[][] prerequisites) {

    }

}

# [**12. Integer to Roman**](https://leetcode-cn.com/problems/integer-to-roman/)

难度中等543收藏分享切换为中文接收动态反馈

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

**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 one's 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 an integer, convert it to a roman numeral.

**Example 1:**

**Input:** num = 3

**Output:** "III"

**Example 2:**

**Input:** num = 4

**Output:** "IV"

**Example 3:**

**Input:** num = 9

**Output:** "IX"

**Example 4:**

**Input:** num = 58

**Output:** "LVIII"

**Explanation:** L = 50, V = 5, III = 3.

**Example 5:**

**Input:** num = 1994

**Output:** "MCMXCIV"

**Explanation:** M = 1000, CM = 900, XC = 90 and IV = 4.

**Constraints:**

* 1 <= num <= 3999

通过次数159,139

提交次数245,054

class Solution {

    public String intToRoman(int num) {

    }

}

# [**1010. Pairs of Songs With Total Durations Divisible by 60**](https://leetcode-cn.com/problems/pairs-of-songs-with-total-durations-divisible-by-60/)

难度中等143收藏分享切换为中文接收动态反馈

You are given a list of songs where the ith song has a duration of time[i] seconds.

Return the number of pairs of songs for which their total duration in seconds is divisible by 60. Formally, we want the number of indices i, j such that i < j with (time[i] + time[j]) % 60 == 0.

**Example 1:**

**Input:** time = [30,20,150,100,40]

**Output:** 3

**Explanation:** Three pairs have a total duration divisible by 60:

(time[0] = 30, time[2] = 150): total duration 180

(time[1] = 20, time[3] = 100): total duration 120

(time[1] = 20, time[4] = 40): total duration 60

**Example 2:**

**Input:** time = [60,60,60]

**Output:** 3

**Explanation:** All three pairs have a total duration of 120, which is divisible by 60.

**Constraints:**

* 1 <= time.length <= 6 \* 104
* 1 <= time[i] <= 500

通过次数17,015

提交次数38,320

class Solution {

    public int numPairsDivisibleBy60(int[] time) {

    }

}

# [**322. Coin Change**](https://leetcode-cn.com/problems/coin-change/)

难度中等1216收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** coins = [1,2,5], amount = 11

**Output:** 3

**Explanation:** 11 = 5 + 5 + 1

**Example 2:**

**Input:** coins = [2], amount = 3

**Output:** -1

**Example 3:**

**Input:** coins = [1], amount = 0

**Output:** 0

**Example 4:**

**Input:** coins = [1], amount = 1

**Output:** 1

**Example 5:**

**Input:** coins = [1], amount = 2

**Output:** 2

**Constraints:**

* 1 <= coins.length <= 12
* 1 <= coins[i] <= 231 - 1
* 0 <= amount <= 104

通过次数215,884

提交次数500,111

class Solution {

    public int coinChange(int[] coins, int amount) {

    }

}

# [**236. Lowest Common Ancestor of a Binary Tree**](https://leetcode-cn.com/problems/lowest-common-ancestor-of-a-binary-tree/)

难度中等1087收藏分享切换为中文接收动态反馈

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](https://en.wikipedia.org/wiki/Lowest_common_ancestor): “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**).”

**Example 1:**



**Input:** root = [3,5,1,6,2,0,8,null,null,7,4], p = 5, q = 1

**Output:** 3

**Explanation:** The LCA of nodes 5 and 1 is 3.

**Example 2:**



**Input:** root = [3,5,1,6,2,0,8,null,null,7,4], p = 5, q = 4

**Output:** 5

**Explanation:** The LCA of nodes 5 and 4 is 5, since a node can be a descendant of itself according to the LCA definition.

**Example 3:**

**Input:** root = [1,2], p = 1, q = 2

**Output:** 1

**Constraints:**

* The number of nodes in the tree is in the range [2, 105].
* -109 <= Node.val <= 109
* All Node.val are **unique**.
* p != q
* p and q will exist in the tree.

通过次数188,612

提交次数281,557

/\*\*

 \* Definition for a binary tree node.

 \* public class TreeNode {

 \*     int val;

 \*     TreeNode left;

 \*     TreeNode right;

 \*     TreeNode(int x) { val = x; }

 \* }

 \*/

class Solution {

    public TreeNode lowestCommonAncestor(TreeNode root, TreeNode p, TreeNode q) {

    }

}

# [**221. Maximal Square**](https://leetcode-cn.com/problems/maximal-square/)

难度中等743收藏分享切换为中文接收动态反馈

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.

**Example 1:**



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

**Output:** 4

**Example 2:**



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

**Output:** 1

**Example 3:**

**Input:** matrix = [["0"]]

**Output:** 0

**Constraints:**

* m == matrix.length
* n == matrix[i].length
* 1 <= m, n <= 300
* matrix[i][j] is '0' or '1'.

通过次数102,859

提交次数226,892

class Solution {

    public int maximalSquare(char[][] matrix) {

    }

}

# [**863. All Nodes Distance K in Binary Tree**](https://leetcode-cn.com/problems/all-nodes-distance-k-in-binary-tree/)

难度中等269收藏分享切换为中文接收动态反馈

We are given a binary tree (with root node root), a target node, and an integer value K.

Return a list of the values of all nodes that have a distance K from the target node.  The answer can be returned in any order.

**Example 1:**

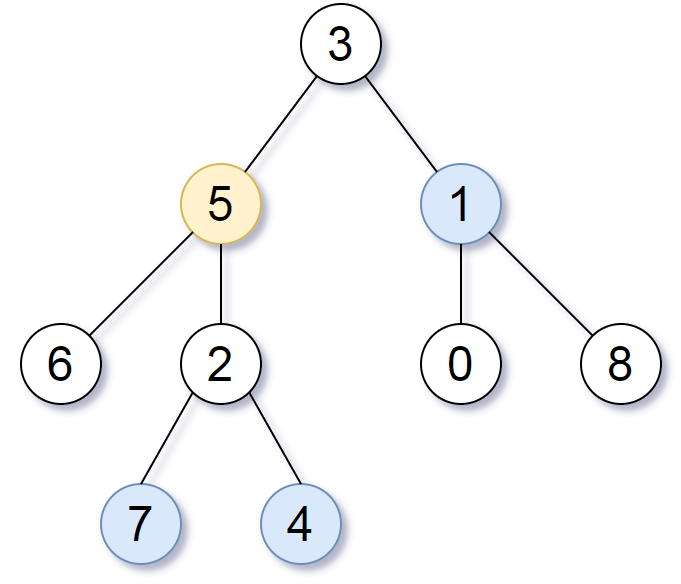
**Input:** root = [3,5,1,6,2,0,8,null,null,7,4], target = 5, K = 2

**Output:** [7,4,1]

**Explanation:**

The nodes that are a distance 2 from the target node (with value 5)

have values 7, 4, and 1.



Note that the inputs "root" and "target" are actually TreeNodes.

The descriptions of the inputs above are just serializations of these objects.

**Note:**

1. The given tree is non-empty.
2. Each node in the tree has unique values 0 <= node.val <= 500.
3. The target node is a node in the tree.
4. 0 <= K <= 1000.

通过次数12,298

提交次数22,566

/\*\*

 \* Definition for a binary tree node.

 \* public class TreeNode {

 \*     int val;

 \*     TreeNode left;

 \*     TreeNode right;

 \*     TreeNode(int x) { val = x; }

 \* }

 \*/

class Solution {

    public List<Integer> distanceK(TreeNode root, TreeNode target, int K) {

    }

}

# [**22. Generate Parentheses**](https://leetcode-cn.com/problems/generate-parentheses/)

难度中等1737收藏分享切换为中文接收动态反馈

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

**Example 1:**

**Input:** n = 3

**Output:** ["((()))","(()())","(())()","()(())","()()()"]

**Example 2:**

**Input:** n = 1

**Output:** ["()"]

**Constraints:**

* 1 <= n <= 8

通过次数262,832

提交次数341,012

class Solution {

    public List<String> generateParenthesis(int n) {

    }

}

# [**348. Design Tic-Tac-Toe**](https://leetcode-cn.com/problems/design-tic-tac-toe/)

难度中等68收藏分享切换为中文接收动态反馈

Assume the following rules are for the tic-tac-toe game on an n x n board between two players:

1. A move is guaranteed to be valid and is placed on an empty block.
2. Once a winning condition is reached, no more moves are allowed.
3. A player who succeeds in placing n of their marks in a horizontal, vertical, or diagonal row wins the game.

Implement the TicTacToe class:

* TicTacToe(int n) Initializes the object the size of the board n.
* int move(int row, int col, int player) Indicates that player with id player plays at the cell (row, col) of the board. The move is guaranteed to be a valid move.

**Follow up:**  
Could you do better than O(*n*2) per move() operation?

**Example 1:**

**Input**

["TicTacToe", "move", "move", "move", "move", "move", "move", "move"]

[[3], [0, 0, 1], [0, 2, 2], [2, 2, 1], [1, 1, 2], [2, 0, 1], [1, 0, 2], [2, 1, 1]]

**Output**

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

**Explanation**

TicTacToe ticTacToe = new TicTacToe(3);

Assume that player 1 is "X" and player 2 is "O" in the board.

ticTacToe.move(0, 0, 1); // return 0 (no one wins)

|X| | |

| | | | // Player 1 makes a move at (0, 0).

| | | |

ticTacToe.move(0, 2, 2); // return 0 (no one wins)

|X| |O|

| | | | // Player 2 makes a move at (0, 2).

| | | |

ticTacToe.move(2, 2, 1); // return 0 (no one wins)

|X| |O|

| | | | // Player 1 makes a move at (2, 2).

| | |X|

ticTacToe.move(1, 1, 2); // return 0 (no one wins)

|X| |O|

| |O| | // Player 2 makes a move at (1, 1).

| | |X|

ticTacToe.move(2, 0, 1); // return 0 (no one wins)

|X| |O|

| |O| | // Player 1 makes a move at (2, 0).

|X| |X|

ticTacToe.move(1, 0, 2); // return 0 (no one wins)

|X| |O|

|O|O| | // Player 2 makes a move at (1, 0).

|X| |X|

ticTacToe.move(2, 1, 1); // return 1 (player 1 wins)

|X| |O|

|O|O| | // Player 1 makes a move at (2, 1).

|X|X|X|

**Constraints:**

* 2 <= n <= 100
* player is 1 or 2.
* 1 <= row, col <= n
* (row, col) are **unique** for each different call to move.
* At most n2 calls will be made to move.

通过次数3,918

提交次数6,591

class TicTacToe {

    /\*\* Initialize your data structure here. \*/

    public TicTacToe(int n) {

    }

    /\*\* Player {player} makes a move at ({row}, {col}).

        @param row The row of the board.

        @param col The column of the board.

        @param player The player, can be either 1 or 2.

        @return The current winning condition, can be either:

                0: No one wins.

                1: Player 1 wins.

                2: Player 2 wins. \*/

    public int move(int row, int col, int player) {

    }

}

/\*\*

 \* Your TicTacToe object will be instantiated and called as such:

 \* TicTacToe obj = new TicTacToe(n);

 \* int param\_1 = obj.move(row,col,player);

 \*/

# [**227. Basic Calculator II**](https://leetcode-cn.com/problems/basic-calculator-ii/)

难度中等380收藏分享切换为中文接收动态反馈

Given a string s which represents an expression, evaluate this expression and return its value.

The integer division should truncate toward zero.

**Example 1:**

**Input:** s = "3+2\*2"

**Output:** 7

**Example 2:**

**Input:** s = " 3/2 "

**Output:** 1

**Example 3:**

**Input:** s = " 3+5 / 2 "

**Output:** 5

**Constraints:**

* 1 <= s.length <= 3 \* 105
* s consists of integers and operators ('+', '-', '\*', '/') separated by some number of spaces.
* s represents **a valid expression**.
* All the integers in the expression are non-negative integers in the range [0, 231 - 1].
* The answer is **guaranteed** to fit in a **32-bit integer**.

通过次数67,349

提交次数155,540

class Solution {

    public int calculate(String s) {

    }

}

# [45. Jump Game II](https://leetcode-cn.com/problems/jump-game-ii/)

难度中等928收藏分享切换为中文接收动态反馈

Given an array of non-negative integers nums, you are initially positioned at the first index of the array.

Each element in the array represents your maximum jump length at that position.

Your goal is to reach the last index in the minimum number of jumps.

You can assume that you can always reach the last index.

**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

**Constraints:**

* 1 <= nums.length <= 1000
* 0 <= nums[i] <= 105

通过次数122,124

提交次数310,306

class Solution {

    public int jump(int[] nums) {

    }

}

# [**8. String to Integer (atoi)**](https://leetcode-cn.com/problems/string-to-integer-atoi/)

难度中等1092收藏分享切换为中文关闭提醒反馈

Implement the myAtoi(string s) function, which converts a string to a 32-bit signed integer (similar to C/C++'s atoi function).

The algorithm for myAtoi(string s) is as follows:

1. Read in and ignore any leading whitespace.
2. Check if the next character (if not already at the end of the string) is '-' or '+'. Read this character in if it is either. This determines if the final result is negative or positive respectively. Assume the result is positive if neither is present.
3. Read in next the characters until the next non-digit charcter or the end of the input is reached. The rest of the string is ignored.
4. Convert these digits into an integer (i.e. "123" -> 123, "0032" -> 32). If no digits were read, then the integer is 0. Change the sign as necessary (from step 2).
5. If the integer is out of the 32-bit signed integer range [-231, 231 - 1], then clamp the integer so that it remains in the range. Specifically, integers less than -231 should be clamped to -231, and integers greater than 231 - 1 should be clamped to 231 - 1.
6. Return the integer as the final result.

**Note:**

* Only the space character ' ' is considered a whitespace character.
* **Do not ignore** any characters other than the leading whitespace or the rest of the string after the digits.

**Example 1:**

**Input:** s = "42"

**Output:** 42

**Explanation:** The underlined characters are what is read in, the caret is the current reader position.

Step 1: "42" (no characters read because there is no leading whitespace)

^

Step 2: "42" (no characters read because there is neither a '-' nor '+')

^

Step 3: "42" ("42" is read in)

^

The parsed integer is 42.

Since 42 is in the range [-231, 231 - 1], the final result is 42.

**Example 2:**

**Input:** s = " -42"

**Output:** -42

**Explanation:**

Step 1: " -42" (leading whitespace is read and ignored)

^

Step 2: " -42" ('-' is read, so the result should be negative)

^

Step 3: " -42" ("42" is read in)

^

The parsed integer is -42.

Since -42 is in the range [-231, 231 - 1], the final result is -42.

**Example 3:**

**Input:** s = "4193 with words"

**Output:** 4193

**Explanation:**

Step 1: "4193 with words" (no characters read because there is no leading whitespace)

^

Step 2: "4193 with words" (no characters read because there is neither a '-' nor '+')

^

Step 3: "4193 with words" ("4193" is read in; reading stops because the next character is a non-digit)

^

The parsed integer is 4193.

Since 4193 is in the range [-231, 231 - 1], the final result is 4193.

**Example 4:**

**Input:** s = "words and 987"

**Output:** 0

**Explanation:**

Step 1: "words and 987" (no characters read because there is no leading whitespace)

^

Step 2: "words and 987" (no characters read because there is neither a '-' nor '+')

^

Step 3: "words and 987" (reading stops immediately because there is a non-digit 'w')

^

The parsed integer is 0 because no digits were read.

Since 0 is in the range [-231, 231 - 1], the final result is 0.

**Example 5:**

**Input:** s = "-91283472332"

**Output:** -2147483648

**Explanation:**

Step 1: "-91283472332" (no characters read because there is no leading whitespace)

^

Step 2: "-91283472332" ('-' is read, so the result should be negative)

^

Step 3: "-91283472332" ("91283472332" is read in)

^

The parsed integer is -91283472332.

Since -91283472332 is less than the lower bound of the range [-231, 231 - 1], the final result is clamped to -231 = -2147483648.

**Constraints:**

* 0 <= s.length <= 200
* s consists of English letters (lower-case and upper-case), digits (0-9), ' ', '+', '-', and '.'.

通过次数280,136

提交次数1,307,268

class Solution {

    public int myAtoi(String s) {

    }

}

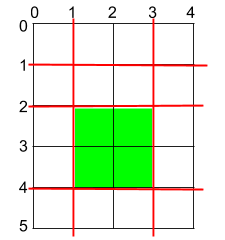
# [**1465. Maximum Area of a Piece of Cake After Horizontal and Vertical Cuts**](https://leetcode-cn.com/problems/maximum-area-of-a-piece-of-cake-after-horizontal-and-vertical-cuts/)

难度中等11收藏分享切换为中文接收动态反馈

Given a rectangular cake with height h and width w, and two arrays of integers horizontalCuts and verticalCuts where horizontalCuts[i] is the distance from the top of the rectangular cake to the ith horizontal cut and similarly, verticalCuts[j] is the distance from the left of the rectangular cake to the jth vertical cut.

Return the maximum area of a piece of cake after you cut at each horizontal and vertical position provided in the arrays *horizontalCuts* and *verticalCuts*. Since the answer can be a huge number, return this modulo 10^9 + 7.

**Example 1:**

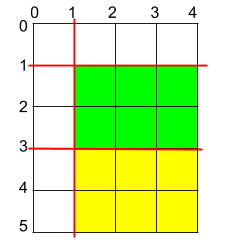


**Input:** h = 5, w = 4, horizontalCuts = [1,2,4], verticalCuts = [1,3]

**Output:** 4

**Explanation:** The figure above represents the given rectangular cake. Red lines are the horizontal and vertical cuts. After you cut the cake, the green piece of cake has the maximum area.

**Example 2:**

****

**Input:** h = 5, w = 4, horizontalCuts = [3,1], verticalCuts = [1]

**Output:** 6

**Explanation:** The figure above represents the given rectangular cake. Red lines are the horizontal and vertical cuts. After you cut the cake, the green and yellow pieces of cake have the maximum area.

**Example 3:**

**Input:** h = 5, w = 4, horizontalCuts = [3], verticalCuts = [3]

**Output:** 9

**Constraints:**

* 2 <= h, w <= 10^9
* 1 <= horizontalCuts.length < min(h, 10^5)
* 1 <= verticalCuts.length < min(w, 10^5)
* 1 <= horizontalCuts[i] < h
* 1 <= verticalCuts[i] < w
* It is guaranteed that all elements in horizontalCuts are distinct.
* It is guaranteed that all elements in verticalCuts are distinct.

通过次数5,485

提交次数18,296

class Solution {

    public int maxArea(int h, int w, int[] horizontalCuts, int[] verticalCuts) {

    }

}

# [**98. Validate Binary Search Tree**](https://leetcode-cn.com/problems/validate-binary-search-tree/)

难度中等1027收藏分享切换为中文接收动态反馈

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

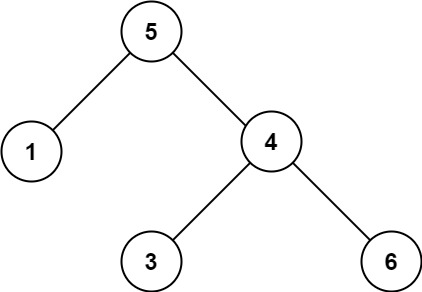
**Example 1:**



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

**Output:** true

**Example 2:**



**Input:** root = [5,1,4,null,null,3,6]

**Output:** false

**Explanation:** The root node's value is 5 but its right child's value is 4.

**Constraints:**

* The number of nodes in the tree is in the range [1, 104].
* -231 <= Node.val <= 231 - 1

通过次数257,592

提交次数756,101

/\*\*

 \* Definition for a binary tree node.

 \* public class TreeNode {

 \*     int val;

 \*     TreeNode left;

 \*     TreeNode right;

 \*     TreeNode() {}

 \*     TreeNode(int val) { this.val = val; }

 \*     TreeNode(int val, TreeNode left, TreeNode right) {

 \*         this.val = val;

 \*         this.left = left;

 \*         this.right = right;

 \*     }

 \* }

 \*/

class Solution {

    public boolean isValidBST(TreeNode root) {

    }

}

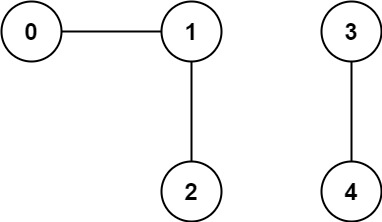
# [**323. Number of Connected Components in an Undirected Graph**](https://leetcode-cn.com/problems/number-of-connected-components-in-an-undirected-graph/)

难度中等79收藏分享切换为中文接收动态反馈

You have a graph of n nodes. You are given an integer n and an array edges where edges[i] = [ai, bi] indicates that there is an edge between ai and bi in the graph.

Return the number of connected components in the graph.

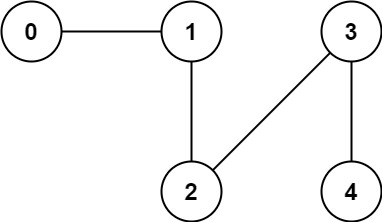
**Example 1:**



**Input:** n = 5, edges = [[0,1],[1,2],[3,4]]

**Output:** 2

**Example 2:**



**Input:** n = 5, edges = [[0,1],[1,2],[2,3],[3,4]]

**Output:** 1

**Constraints:**

* 1 <= n <= 2000
* 1 <= edges.length <= 5000
* edges[i].length == 2
* 0 <= ai <= bi < n
* ai != bi
* There are no repeated edges.

通过次数6,993

提交次数11,154

class Solution {

    public int countComponents(int n, int[][] edges) {

    }

}

# [**560. Subarray Sum Equals K**](https://leetcode-cn.com/problems/subarray-sum-equals-k/)

难度中等862收藏分享切换为中文接收动态反馈

Given an array of integers nums and an integer k, return the total number of continuous subarrays whose sum equals to *k*.

**Example 1:**

**Input:** nums = [1,1,1], k = 2

**Output:** 2

**Example 2:**

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

**Output:** 2

**Constraints:**

* 1 <= nums.length <= 2 \* 104
* -1000 <= nums[i] <= 1000
* -107 <= k <= 107

通过次数103,982

提交次数233,538

class Solution {

    public int subarraySum(int[] nums, int k) {

    }

}

# [**55. Jump Game**](https://leetcode-cn.com/problems/jump-game/)

难度中等1155收藏分享切换为中文接收动态反馈

Given an array of non-negative integers nums, you are initially positioned at the **first index** of the array.

Each element in the array represents your maximum jump length at that position.

Determine if you are able to reach the last index.

**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.

**Constraints:**

* 1 <= nums.length <= 3 \* 104
* 0 <= nums[i] <= 105

通过次数224,442

提交次数534,571

class Solution {

    public boolean canJump(int[] nums) {

    }

}

# [**1167. Minimum Cost to Connect Sticks**](https://leetcode-cn.com/problems/minimum-cost-to-connect-sticks/)

难度中等40收藏分享切换为中文接收动态反馈

You have some number of sticks with positive integer lengths. These lengths are given as an array sticks, where sticks[i] is the length of the ith stick.

You can connect any two sticks of lengths x and y into one stick by paying a cost of x + y. You must connect all the sticks until there is only one stick remaining.

Return the minimum cost of connecting all the given sticks into one stick in this way.

**Example 1:**

**Input:** sticks = [2,4,3]

**Output:** 14

**Explanation:** You start with sticks = [2,4,3].

1. Combine sticks 2 and 3 for a cost of 2 + 3 = 5. Now you have sticks = [5,4].

2. Combine sticks 5 and 4 for a cost of 5 + 4 = 9. Now you have sticks = [9].

There is only one stick left, so you are done. The total cost is 5 + 9 = 14.

**Example 2:**

**Input:** sticks = [1,8,3,5]

**Output:** 30

**Explanation:** You start with sticks = [1,8,3,5].

1. Combine sticks 1 and 3 for a cost of 1 + 3 = 4. Now you have sticks = [4,8,5].

2. Combine sticks 4 and 5 for a cost of 4 + 5 = 9. Now you have sticks = [9,8].

3. Combine sticks 9 and 8 for a cost of 9 + 8 = 17. Now you have sticks = [17].

There is only one stick left, so you are done. The total cost is 4 + 9 + 17 = 30.

**Example 3:**

**Input:** sticks = [5]

**Output:** 0

**Explanation:** There is only one stick, so you don't need to do anything. The total cost is 0.

**Constraints:**

* 1 <= sticks.length <= 104
* 1 <= sticks[i] <= 104

通过次数1,905

提交次数4,751

class Solution {

    public int connectSticks(int[] sticks) {

    }

}

# [**215. Kth Largest Element in an Array**](https://leetcode-cn.com/problems/kth-largest-element-in-an-array/)

难度中等1042收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** nums = [3,2,1,5,6,4], k = 2

**Output:** 5

**Example 2:**

**Input:** nums = [3,2,3,1,2,4,5,5,6], k = 4

**Output:** 4

**Constraints:**

* 1 <= k <= nums.length <= 104
* -104 <= nums[i] <= 104

通过次数314,987

提交次数487,554

class Solution {

    public int findKthLargest(int[] nums, int k) {

    }

}

# [**909. Snakes and Ladders**](https://leetcode-cn.com/problems/snakes-and-ladders/)

难度中等39收藏分享切换为中文接收动态反馈

On an N x N board, the numbers from 1 to N\*N are written boustrophedonically **starting from the bottom left of the board**, and alternating direction each row.  For example, for a 6 x 6 board, the numbers are written as follows:



You start on square 1 of the board (which is always in the last row and first column).  Each move, starting from square x, consists of the following:

* You choose a destination square S with number x+1, x+2, x+3, x+4, x+5, or x+6, provided this number is <= N\*N.
  + (This choice simulates the result of a standard 6-sided die roll: ie., there are always **at most 6 destinations, regardless of the size of the board**.)
* If S has a snake or ladder, you move to the destination of that snake or ladder.  Otherwise, you move to S.

A board square on row r and column c has a "snake or ladder" if board[r][c] != -1.  The destination of that snake or ladder is board[r][c].

Note that you only take a snake or ladder at most once per move: if the destination to a snake or ladder is the start of another snake or ladder, you do **not** continue moving.  (For example, if the board is `[[4,-1],[-1,3]]`, and on the first move your destination square is `2`, then you finish your first move at `3`, because you do **not** continue moving to `4`.)

Return the least number of moves required to reach square N\*N.  If it is not possible, return -1.

**Example 1:**

**Input:** [

[-1,-1,-1,-1,-1,-1],

[-1,-1,-1,-1,-1,-1],

[-1,-1,-1,-1,-1,-1],

[-1,35,-1,-1,13,-1],

[-1,-1,-1,-1,-1,-1],

[-1,15,-1,-1,-1,-1]]

**Output:** 4

**Explanation:**

At the beginning, you start at square 1 [at row 5, column 0].

You decide to move to square 2, and must take the ladder to square 15.

You then decide to move to square 17 (row 3, column 5), and must take the snake to square 13.

You then decide to move to square 14, and must take the ladder to square 35.

You then decide to move to square 36, ending the game.

It can be shown that you need at least 4 moves to reach the N\*N-th square, so the answer is 4.

**Note:**

1. 2 <= board.length = board[0].length <= 20
2. board[i][j] is between 1 and N\*N or is equal to -1.
3. The board square with number 1 has no snake or ladder.
4. The board square with number N\*N has no snake or ladder.

通过次数3,078

提交次数9,123

class Solution {

    public int snakesAndLadders(int[][] board) {

    }

}

# [**937. Reorder Data in Log Files**](https://leetcode-cn.com/problems/reorder-data-in-log-files/)

难度简单73收藏分享切换为中文接收动态反馈

You are given an array of logs. Each log is a space-delimited string of words, where the first word is the **identifier**.

There are two types of logs:

* **Letter-logs**: All words (except the identifier) consist of lowercase English letters.
* **Digit-logs**: All words (except the identifier) consist of digits.

Reorder these logs so that:

1. The **letter-logs** come before all **digit-logs**.
2. The **letter-logs** are sorted lexicographically by their contents. If their contents are the same, then sort them lexicographically by their identifiers.
3. The **digit-logs** maintain their relative ordering.

Return the final order of the logs.

**Example 1:**

**Input:** logs = ["dig1 8 1 5 1","let1 art can","dig2 3 6","let2 own kit dig","let3 art zero"]

**Output:** ["let1 art can","let3 art zero","let2 own kit dig","dig1 8 1 5 1","dig2 3 6"]

**Explanation:**

The letter-log contents are all different, so their ordering is "art can", "art zero", "own kit dig".

The digit-logs have a relative order of "dig1 8 1 5 1", "dig2 3 6".

**Example 2:**

**Input:** logs = ["a1 9 2 3 1","g1 act car","zo4 4 7","ab1 off key dog","a8 act zoo"]

**Output:** ["g1 act car","a8 act zoo","ab1 off key dog","a1 9 2 3 1","zo4 4 7"]

**Constraints:**

* 1 <= logs.length <= 100
* 3 <= logs[i].length <= 100
* All the tokens of logs[i] are separated by a **single** space.
* logs[i] is guaranteed to have an identifier and at least one word after the identifier.

class Solution {

    public String[] reorderLogFiles(String[] logs) {

    }

}

# [**146. LRU Cache**](https://leetcode-cn.com/problems/lru-cache/)

难度中等1473收藏分享切换为中文接收动态反馈

Design a data structure that follows the constraints of a [**Least Recently Used (LRU) cache**](https://en.wikipedia.org/wiki/Cache_replacement_policies#LRU).

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.

**Example 1:**

**Input**

["LRUCache", "put", "put", "get", "put", "get", "put", "get", "get", "get"]

[[2], [1, 1], [2, 2], [1], [3, 3], [2], [4, 4], [1], [3], [4]]

**Output**

[null, null, null, 1, null, -1, null, -1, 3, 4]

**Explanation**

LRUCache lRUCache = new LRUCache(2);

lRUCache.put(1, 1); // cache is {1=1}

lRUCache.put(2, 2); // cache is {1=1, 2=2}

lRUCache.get(1); // return 1

lRUCache.put(3, 3); // LRU key was 2, evicts key 2, cache is {1=1, 3=3}

lRUCache.get(2); // returns -1 (not found)

lRUCache.put(4, 4); // LRU key was 1, evicts key 1, cache is {4=4, 3=3}

lRUCache.get(1); // return -1 (not found)

lRUCache.get(3); // return 3

lRUCache.get(4); // return 4

**Constraints:**

* 1 <= capacity <= 3000
* 0 <= key <= 104
* 0 <= value <= 105
* At most 2 \* 105 calls will be made to get and put.

class LRUCache {

public LRUCache(int capacity) {

}

public int get(int key) {

}

public void put(int key, int value) {

}

}

/\*\*

\* Your LRUCache object will be instantiated and called as such:

\* LRUCache obj = new LRUCache(capacity);

\* int param\_1 = obj.get(key);

\* obj.put(key,value);

\*/

# [**1192. Critical Connections in a Network**](https://leetcode-cn.com/problems/critical-connections-in-a-network/)

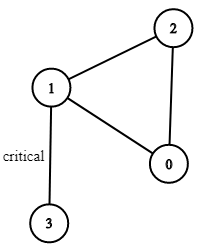
难度困难143收藏分享切换为中文接收动态反馈

There are n servers numbered from 0 to n - 1 connected by undirected server-to-server connections forming a network where connections[i] = [ai, bi] represents a connection between servers ai and bi. Any server can reach other servers directly or indirectly through the network.

A critical connection is a connection that, if removed, will make some servers unable to reach some other server.

Return all critical connections in the network in any order.

**Example 1:**

****

**Input:** n = 4, connections = [[0,1],[1,2],[2,0],[1,3]]

**Output:** [[1,3]]

**Explanation:** [[3,1]] is also accepted.

**Constraints:**

* 2 <= n <= 105
* n - 1 <= connections.length <= 105
* 0 <= ai, bi <= n - 1
* ai != bi
* There are no repeated connections.

class Solution {

    public List<List<Integer>> criticalConnections(int n, List<List<Integer>> connections) {

    }

}

# [**819. Most Common Word**](https://leetcode-cn.com/problems/most-common-word/)

难度简单98收藏分享切换为中文接收动态反馈

Given a string paragraph and a string array of the banned words banned, return the most frequent word that is not banned. It is **guaranteed** there is **at least one word** that is not banned, and that the answer is **unique**.

The words in paragraph are **case-insensitive** and the answer should be returned in **lowercase**.

**Example 1:**

**Input:** paragraph = "Bob hit a ball, the hit BALL flew far after it was hit.", banned = ["hit"]

**Output:** "ball"

**Explanation:**

"hit" occurs 3 times, but it is a banned word.

"ball" occurs twice (and no other word does), so it is the most frequent non-banned word in the paragraph.

Note that words in the paragraph are not case sensitive,

that punctuation is ignored (even if adjacent to words, such as "ball,"),

and that "hit" isn't the answer even though it occurs more because it is banned.

**Example 2:**

**Input:** paragraph = "a.", banned = []

**Output:** "a"

**Constraints:**

* 1 <= paragraph.length <= 1000
* paragraph consists of English letters, space ' ', or one of the symbols: "!?',;.".
* 0 <= banned.length <= 100
* 1 <= banned[i].length <= 10
* banned[i] consists of only lowercase English letters.

class Solution {

    public String mostCommonWord(String paragraph, String[] banned) {

    }

}

# [**42. Trapping Rain Water**](https://leetcode-cn.com/problems/trapping-rain-water/)

难度困难2449收藏分享切换为中文接收动态反馈

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.

**Example 1:**



**Input:** height = [0,1,0,2,1,0,1,3,2,1,2,1]

**Output:** 6

**Explanation:** The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.

**Example 2:**

**Input:** height = [4,2,0,3,2,5]

**Output:** 9

**Constraints:**

* n == height.length
* 0 <= n <= 3 \* 104
* 0 <= height[i] <= 105

class Solution {

    public int trap(int[] height) {

    }

}

# [**127. Word Ladder**](https://leetcode-cn.com/problems/word-ladder/)

难度困难791收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** beginWord = "hit", endWord = "cog", wordList = ["hot","dot","dog","lot","log","cog"]

**Output:** 5

**Explanation:** One shortest transformation sequence is "hit" -> "hot" -> "dot" -> "dog" -> cog", which is 5 words long.

**Example 2:**

**Input:** beginWord = "hit", endWord = "cog", wordList = ["hot","dot","dog","lot","log"]

**Output:** 0

**Explanation:** The endWord "cog" is not in wordList, therefore there is no valid transformation sequence.

**Constraints:**

* 1 <= beginWord.length <= 10
* endWord.length == beginWord.length
* 1 <= wordList.length <= 5000
* wordList[i].length == beginWord.length
* beginWord, endWord, and wordList[i] consist of lowercase English letters.
* beginWord != endWord
* All the words in wordList are **unique**.

class Solution {

    public int ladderLength(String beginWord, String endWord, List<String> wordList) {

    }

}

# [**23. Merge k Sorted Lists**](https://leetcode-cn.com/problems/merge-k-sorted-lists/)

难度困难1364收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** lists = [[1,4,5],[1,3,4],[2,6]]

**Output:** [1,1,2,3,4,4,5,6]

**Explanation:** The linked-lists are:

[

1->4->5,

1->3->4,

2->6

]

merging them into one sorted list:

1->1->2->3->4->4->5->6

**Example 2:**

**Input:** lists = []

**Output:** []

**Example 3:**

**Input:** lists = [[]]

**Output:** []

**Constraints:**

* k == lists.length
* 0 <= k <= 10^4
* 0 <= lists[i].length <= 500
* -10^4 <= lists[i][j] <= 10^4
* lists[i] is sorted in **ascending order**.
* The sum of lists[i].length won't exceed 10^4.

/\*\*

 \* Definition for singly-linked list.

 \* public class ListNode {

 \*     int val;

 \*     ListNode next;

 \*     ListNode() {}

 \*     ListNode(int val) { this.val = val; }

 \*     ListNode(int val, ListNode next) { this.val = val; this.next = next; }

 \* }

 \*/

class Solution {

    public ListNode mergeKLists(ListNode[] lists) {

    }

}

# [**295. Find Median from Data Stream**](https://leetcode-cn.com/problems/find-median-from-data-stream/)

难度困难434收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input**

["MedianFinder", "addNum", "addNum", "findMedian", "addNum", "findMedian"]

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

**Output**

[null, null, null, 1.5, null, 2.0]

**Explanation**

MedianFinder medianFinder = new MedianFinder();

medianFinder.addNum(1); // arr = [1]

medianFinder.addNum(2); // arr = [1, 2]

medianFinder.findMedian(); // return 1.5 (i.e., (1 + 2) / 2)

medianFinder.addNum(3); // arr[1, 2, 3]

medianFinder.findMedian(); // return 2.0

**Constraints:**

* -105 <= num <= 105
* There will be at least one element in the data structure before calling findMedian.
* At most 5 \* 104 calls will be made to addNum and findMedian.

**Follow up:**

* If all integer numbers from the stream are in the range [0, 100], how would you optimize your solution?
* If 99% of all integer numbers from the stream are in the range [0, 100], how would you optimize your solution?

class MedianFinder {

    /\*\* initialize your data structure here. \*/

    public MedianFinder() {

    }

    public void addNum(int num) {

    }

    public double findMedian() {

    }

}

/\*\*

 \* Your MedianFinder object will be instantiated and called as such:

 \* MedianFinder obj = new MedianFinder();

 \* obj.addNum(num);

 \* double param\_2 = obj.findMedian();

 \*/

# [**103. Binary Tree Zigzag Level Order Traversal**](https://leetcode-cn.com/problems/binary-tree-zigzag-level-order-traversal/)

难度中等462收藏分享切换为中文接收动态反馈

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).

**Example 1:**



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

**Output:** [[3],[20,9],[15,7]]

**Example 2:**

**Input:** root = [1]

**Output:** [[1]]

**Example 3:**

**Input:** root = []

**Output:** []

**Constraints:**

* The number of nodes in the tree is in the range [0, 2000].
* -100 <= Node.val <= 100

/\*\*

 \* Definition for a binary tree node.

 \* public class TreeNode {

 \*     int val;

 \*     TreeNode left;

 \*     TreeNode right;

 \*     TreeNode() {}

 \*     TreeNode(int val) { this.val = val; }

 \*     TreeNode(int val, TreeNode left, TreeNode right) {

 \*         this.val = val;

 \*         this.left = left;

 \*         this.right = right;

 \*     }

 \* }

 \*/

class Solution {

    public List<List<Integer>> zigzagLevelOrder(TreeNode root) {

    }

}

# [**139. Word Break**](https://leetcode-cn.com/problems/word-break/)

难度中等1044收藏分享切换为中文接收动态反馈

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.

**Example 1:**

**Input:** s = "leetcode", wordDict = ["leet","code"]

**Output:** true

**Explanation:** Return true because "leetcode" can be segmented as "leet code".

**Example 2:**

**Input:** s = "applepenapple", wordDict = ["apple","pen"]

**Output:** true

**Explanation:** Return true because "applepenapple" can be segmented as "apple pen apple".

Note that you are allowed to reuse a dictionary word.

**Example 3:**

**Input:** s = "catsandog", wordDict = ["cats","dog","sand","and","cat"]

**Output:** false

**Constraints:**

* 1 <= s.length <= 300
* 1 <= wordDict.length <= 1000
* 1 <= wordDict[i].length <= 20
* s and wordDict[i] consist of only lowercase English letters.
* All the strings of wordDict are **unique**.

class Solution {

    public boolean wordBreak(String s, List<String> wordDict) {

    }

}

# [**238. Product of Array Except Self**](https://leetcode-cn.com/problems/product-of-array-except-self/)

难度中等856收藏分享切换为中文接收动态反馈

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 <= nums.length <= 105
* -30 <= nums[i] <= 30
* The product of any prefix or suffix of nums is **guaranteed** to fit in a **32-bit** integer.

**Follow up:** Can you solve the problem in O(1) extra space complexity? (The output array **does not** count as extra space for space complexity analysis.)

通过次数120,434

提交次数167,304

class Solution {

    public int[] productExceptSelf(int[] nums) {

    }

}

# [**33. Search in Rotated Sorted Array**](https://leetcode-cn.com/problems/search-in-rotated-sorted-array/)

难度中等1425收藏分享切换为中文接收动态反馈

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

Prior to being passed to your function, nums is **rotated** at an unknown pivot index k (0 <= 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 rotated at pivot index 3 and become [4,5,6,7,0,1,2].

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

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

**Example 1:**

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

**Output:** 4

**Example 2:**

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

**Output:** -1

**Example 3:**

**Input:** nums = [1], target = 0

**Output:** -1

**Constraints:**

* 1 <= nums.length <= 5000
* -104 <= nums[i] <= 104
* All values of nums are **unique**.
* nums is guaranteed to be rotated at some pivot.
* -104 <= target <= 104

class Solution {

    public int search(int[] nums, int target) {

    }

}

# [**49. Group Anagrams**](https://leetcode-cn.com/problems/group-anagrams/)

难度中等773收藏分享切换为中文接收动态反馈

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"]]

**Constraints:**

* 1 <= strs.length <= 104
* 0 <= strs[i].length <= 100
* strs[i] consists of lower-case English letters.

class Solution {

    public List<List<String>> groupAnagrams(String[] strs) {

    }

}

# [**4. Median of Two Sorted Arrays**](https://leetcode-cn.com/problems/median-of-two-sorted-arrays/)

难度困难4220收藏分享切换为中文接收动态反馈

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)).

**Example 1:**

**Input:** nums1 = [1,3], nums2 = [2]

**Output:** 2.00000

**Explanation:** merged array = [1,2,3] and median is 2.

**Example 2:**

**Input:** nums1 = [1,2], nums2 = [3,4]

**Output:** 2.50000

**Explanation:** merged array = [1,2,3,4] and median is (2 + 3) / 2 = 2.5.

**Example 3:**

**Input:** nums1 = [0,0], nums2 = [0,0]

**Output:** 0.00000

**Example 4:**

**Input:** nums1 = [], nums2 = [1]

**Output:** 1.00000

**Example 5:**

**Input:** nums1 = [2], nums2 = []

**Output:** 2.00000

**Constraints:**

* nums1.length == m
* nums2.length == n
* 0 <= m <= 1000
* 0 <= n <= 1000
* 1 <= m + n <= 2000
* -106 <= nums1[i], nums2[i] <= 106

class Solution {

    public double findMedianSortedArrays(int[] nums1, int[] nums2) {

    }

}

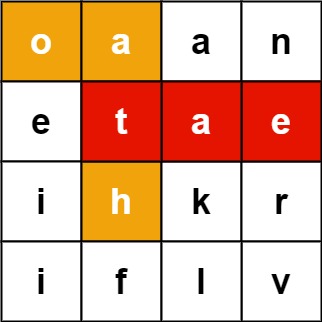
# [**212. Word Search II**](https://leetcode-cn.com/problems/word-search-ii/)

难度困难405收藏分享切换为中文接收动态反馈

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.

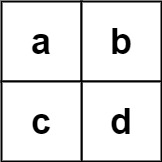
**Example 1:**



**Input:** board = [["o","a","a","n"],["e","t","a","e"],["i","h","k","r"],["i","f","l","v"]], words = ["oath","pea","eat","rain"]

**Output:** ["eat","oath"]

**Example 2:**



**Input:** board = [["a","b"],["c","d"]], words = ["abcb"]

**Output:** []

**Constraints:**

* m == board.length
* n == board[i].length
* 1 <= m, n <= 12
* board[i][j] is a lowercase English letter.
* 1 <= words.length <= 3 \* 104
* 1 <= words[i].length <= 10
* words[i] consists of lowercase English letters.
* All the strings of words are unique.

通过次数37,380

提交次数82,404

class Solution {

    public List<String> findWords(char[][] board, String[] words) {

    }

}

# [**1335. Minimum Difficulty of a Job Schedule**](https://leetcode-cn.com/problems/minimum-difficulty-of-a-job-schedule/)

难度困难51收藏分享切换为中文接收动态反馈

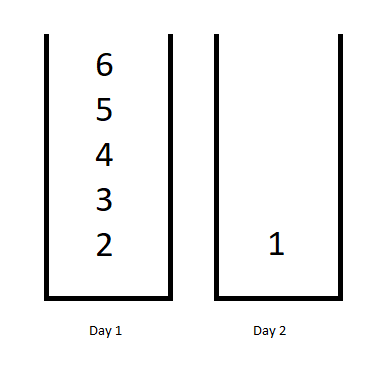
You want to schedule a list of jobs in d days. Jobs are dependent (i.e To work on the i-th job, you have to finish all the jobs j where 0 <= j < i).

You have to finish **at least** one task every day. The difficulty of a job schedule is the sum of difficulties of each day of the d days. The difficulty of a day is the maximum difficulty of a job done in that day.

Given an array of integers jobDifficulty and an integer d. The difficulty of the i-th job is jobDifficulty[i].

Return the minimum difficulty of a job schedule. If you cannot find a schedule for the jobs return **-1**.

**Example 1:**



**Input:** jobDifficulty = [6,5,4,3,2,1], d = 2

**Output:** 7

**Explanation:** First day you can finish the first 5 jobs, total difficulty = 6.

Second day you can finish the last job, total difficulty = 1.

The difficulty of the schedule = 6 + 1 = 7

**Example 2:**

**Input:** jobDifficulty = [9,9,9], d = 4

**Output:** -1

**Explanation:** If you finish a job per day you will still have a free day. you cannot find a schedule for the given jobs.

**Example 3:**

**Input:** jobDifficulty = [1,1,1], d = 3

**Output:** 3

**Explanation:** The schedule is one job per day. total difficulty will be 3.

**Example 4:**

**Input:** jobDifficulty = [7,1,7,1,7,1], d = 3

**Output:** 15

**Example 5:**

**Input:** jobDifficulty = [11,111,22,222,33,333,44,444], d = 6

**Output:** 843

**Constraints:**

* 1 <= jobDifficulty.length <= 300
* 0 <= jobDifficulty[i] <= 1000
* 1 <= d <= 10

通过次数3,075

提交次数5,299

class Solution {

    public int minDifficulty(int[] jobDifficulty, int d) {

    }

}

# [**240. Search a 2D Matrix II**](https://leetcode-cn.com/problems/search-a-2d-matrix-ii/)

难度中等661收藏分享切换为中文接收动态反馈

Write an efficient algorithm that searches for a target value in an m x n integer matrix. The matrix has the following properties:

* Integers in each row are sorted in ascending from left to right.
* Integers in each column are sorted in ascending from top to bottom.

**Example 1:**



**Input:** matrix = [[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]], target = 5

**Output:** true

**Example 2:**



**Input:** matrix = [[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]], target = 20

**Output:** false

**Constraints:**

* m == matrix.length
* n == matrix[i].length
* 1 <= n, m <= 300
* -109 <= matix[i][j] <= 109
* All the integers in each row are **sorted** in ascending order.
* All the integers in each column are **sorted** in ascending order.
* -109 <= target <= 109

通过次数141,015

提交次数303,179

class Solution {

    public boolean searchMatrix(int[][] matrix, int target) {

    }

}

# [**126. Word Ladder II**](https://leetcode-cn.com/problems/word-ladder-ii/)

难度困难445收藏分享切换为中文接收动态反馈

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 all the ***shortest transformation sequences*** from beginWord to endWord, or an empty list if no such sequence exists. Each sequence should be returned as a list of the words [beginWord, s1, s2, ..., sk].

**Example 1:**

**Input:** beginWord = "hit", endWord = "cog", wordList = ["hot","dot","dog","lot","log","cog"]

**Output:** [["hit","hot","dot","dog","cog"],["hit","hot","lot","log","cog"]]

**Explanation:** There are 2 shortest transformation sequences:

"hit" -> "hot" -> "dot" -> "dog" -> "cog"

"hit" -> "hot" -> "lot" -> "log" -> "cog"

**Example 2:**

**Input:** beginWord = "hit", endWord = "cog", wordList = ["hot","dot","dog","lot","log"]

**Output:** []

**Explanation:** The endWord "cog" is not in wordList, therefore there is no valid transformation sequence.

**Constraints:**

* 1 <= beginWord.length <= 5
* endWord.length == beginWord.length
* 1 <= wordList.length <= 1000
* wordList[i].length == beginWord.length
* beginWord, endWord, and wordList[i] consist of lowercase English letters.
* beginWord != endWord
* All the words in wordList are **unique**.

class Solution {

    public List<List<String>> findLadders(String beginWord, String endWord, List<String> wordList) {

    }

}

# [**239. Sliding Window Maximum**](https://leetcode-cn.com/problems/sliding-window-maximum/)

难度困难1054收藏分享切换为中文接收动态反馈

You are given an array of integers nums, there is a sliding window of size k which is moving from the very left of the array to the very right. You can only see the k numbers in the window. Each time the sliding window moves right by one position.

Return the max sliding window.

**Example 1:**

**Input:** nums = [1,3,-1,-3,5,3,6,7], k = 3

**Output:** [3,3,5,5,6,7]

**Explanation:**

Window position Max

--------------- -----

[1 3 -1] -3 5 3 6 7 **3**

1 [3 -1 -3] 5 3 6 7 **3**

1 3 [-1 -3 5] 3 6 7  **5**

1 3 -1 [-3 5 3] 6 7 **5**

1 3 -1 -3 [5 3 6] 7 **6**

1 3 -1 -3 5 [3 6 7] **7**

**Example 2:**

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

**Output:** [1]

**Example 3:**

**Input:** nums = [1,-1], k = 1

**Output:** [1,-1]

**Example 4:**

**Input:** nums = [9,11], k = 2

**Output:** [11]

**Example 5:**

**Input:** nums = [4,-2], k = 2

**Output:** [4]

**Constraints:**

* 1 <= nums.length <= 105
* -104 <= nums[i] <= 104
* 1 <= k <= nums.length

通过次数163,500

提交次数330,113

class Solution {

    public int[] maxSlidingWindow(int[] nums, int k) {

    }

}

# [**140. Word Break II**](https://leetcode-cn.com/problems/word-break-ii/)

难度困难476收藏分享切换为中文接收动态反馈

Given a string s and a dictionary of strings wordDict, add spaces in s to construct a sentence where each word is a valid dictionary word. Return all such possible sentences in **any order**.

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

**Example 1:**

**Input:** s = "catsanddog", wordDict = ["cat","cats","and","sand","dog"]

**Output:** ["cats and dog","cat sand dog"]

**Example 2:**

**Input:** s = "pineapplepenapple", wordDict = ["apple","pen","applepen","pine","pineapple"]

**Output:** ["pine apple pen apple","pineapple pen apple","pine applepen apple"]

**Explanation:** Note that you are allowed to reuse a dictionary word.

**Example 3:**

**Input:** s = "catsandog", wordDict = ["cats","dog","sand","and","cat"]

**Output:** []

**Constraints:**

* 1 <= s.length <= 20
* 1 <= wordDict.length <= 1000
* 1 <= wordDict[i].length <= 10
* s and wordDict[i] consist of only lowercase English letters.
* All the strings of wordDict are **unique**.

通过次数47,729

提交次数101,597

class Solution {

    public List<String> wordBreak(String s, List<String> wordDict) {

    }

}

# [**472. Concatenated Words**](https://leetcode-cn.com/problems/concatenated-words/)

难度困难104收藏分享切换为中文接收动态反馈

Given an array of strings words (**without duplicates**), return all the ***concatenated words*** in the given list of words.

A **concatenated word** is defined as a string that is comprised entirely of at least two shorter words in the given array.

**Example 1:**

**Input:** words = ["cat","cats","catsdogcats","dog","dogcatsdog","hippopotamuses","rat","ratcatdogcat"]

**Output:** ["catsdogcats","dogcatsdog","ratcatdogcat"]

**Explanation:** "catsdogcats" can be concatenated by "cats", "dog" and "cats";

"dogcatsdog" can be concatenated by "dog", "cats" and "dog";

"ratcatdogcat" can be concatenated by "rat", "cat", "dog" and "cat".

**Example 2:**

**Input:** words = ["cat","dog","catdog"]

**Output:** ["catdog"]

**Constraints:**

* 1 <= words.length <= 104
* 0 <= words[i].length <= 1000
* words[i] consists of only lowercase English letters.
* 0 <= sum(words[i].length) <= 105

通过次数5,861

提交次数14,218

class Solution {

    public List<String> findAllConcatenatedWordsInADict(String[] words) {

    }

}

# [**297. Serialize and Deserialize Binary Tree**](https://leetcode-cn.com/problems/serialize-and-deserialize-binary-tree/)

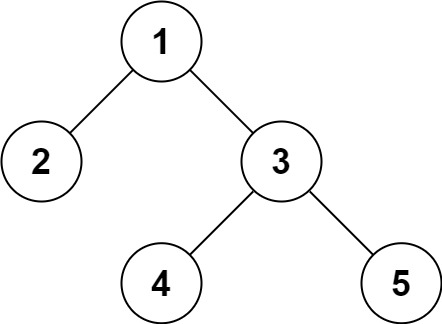
难度困难592收藏分享切换为中文接收动态反馈

Serialization is the process of converting a data structure or object into a sequence of bits so that it can be stored in a file or memory buffer, or transmitted across a network connection link to be reconstructed later in the same or another computer environment.

Design an algorithm to serialize and deserialize a binary tree. There is no restriction on how your serialization/deserialization algorithm should work. You just need to ensure that a binary tree can be serialized to a string and this string can be deserialized to the original tree structure.

**Clarification:** The input/output format is the same as [how LeetCode serializes a binary tree](https://leetcode-cn.com/faq/#binary-tree). You do not necessarily need to follow this format, so please be creative and come up with different approaches yourself.

**Example 1:**



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

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

**Example 2:**

**Input:** root = []

**Output:** []

**Example 3:**

**Input:** root = [1]

**Output:** [1]

**Example 4:**

**Input:** root = [1,2]

**Output:** [1,2]

**Constraints:**

* The number of nodes in the tree is in the range [0, 104].
* -1000 <= Node.val <= 1000

通过次数89,829

提交次数162,723

/\*\*

 \* Definition for a binary tree node.

 \* public class TreeNode {

 \*     int val;

 \*     TreeNode left;

 \*     TreeNode right;

 \*     TreeNode(int x) { val = x; }

 \* }

 \*/

public class Codec {

    // Encodes a tree to a single string.

    public String serialize(TreeNode root) {

    }

    // Decodes your encoded data to tree.

    public TreeNode deserialize(String data) {

    }

}

// Your Codec object will be instantiated and called as such:

// Codec ser = new Codec();

// Codec deser = new Codec();

// TreeNode ans = deser.deserialize(ser.serialize(root));

# [**1833. Maximum Ice Cream Bars**](https://leetcode-cn.com/problems/maximum-ice-cream-bars/)

难度中等73收藏分享切换为中文接收动态反馈

It is a sweltering summer day, and a boy wants to buy some ice cream bars.

At the store, there are n ice cream bars. You are given an array costs of length n, where costs[i] is the price of the ith ice cream bar in coins. The boy initially has coins coins to spend, and he wants to buy as many ice cream bars as possible.

Return the ***maximum*** number of ice cream bars the boy can buy with coins coins.

**Note:** The boy can buy the ice cream bars in any order.

**Example 1:**

**Input:** costs = [1,3,2,4,1], coins = 7

**Output:** 4

**Explanation:** The boy can buy ice cream bars at indices 0,1,2,4 for a total price of 1 + 3 + 2 + 1 = 7.

**Example 2:**

**Input:** costs = [10,6,8,7,7,8], coins = 5

**Output:** 0

**Explanation:** The boy cannot afford any of the ice cream bars.

**Example 3:**

**Input:** costs = [1,6,3,1,2,5], coins = 20

**Output:** 6

**Explanation:** The boy can buy all the ice cream bars for a total price of 1 + 6 + 3 + 1 + 2 + 5 = 18.

**Constraints:**

* costs.length == n
* 1 <= n <= 105
* 1 <= costs[i] <= 105
* 1 <= coins <= 108

通过次数39,227

提交次数56,787

class Solution {

    public int maxIceCream(int[] costs, int coins) {

    }

}