Dynamic Programming

When

Useful for when you are dealing with finding the optimal solution given a constraint, or the number of solutions given a constraint, or if there is a solution given a constraint, in an array or string.

How to use

Let $A[i]$ be the optimal/number-of solution for a prefix of the array/string ending at index i. Or a suffix of the array/string beginning at index i

Think in terms of take or not take. For each $A[i]$ you're finding the maximum of the values of A at indices that represent the optimal/number of solution of the string/array if you had taken, and had not taken a decision at i

Thoughts

DP is a mindset of partitioning/breaking the input up into subproblems that can either build on top of each other (See longest-increasing-subsequence), or mutually exclusive and much easier to solve on their own and compared (See longest-consecutive-sequence)

The hardest part of DP is finding that partition.

When you try DP and find that subproblems are actually more complex than the current problem (given all the partitions that you can think of), then should consider BFS

In context of arrays/strings, when building off of subproblems, you either build off the single subproblem directly before, or consider all subproblems before (and if that were the case sometimes binary search can help if things are sorted nicely).

Detecting Cycles

Undirected

Use either disjoint set data structure or use DFS. For DFS, if an adjacent non-originating vertex is visited then there is a cycle.

Directed

Perform topological sort $O(n+m)$. Topological sort exists iff DAG. We cannot use the same approach as we did in undirected graphs because directed cycles only exists if there are backedges (we are about to visit a node we have traversed from).

Shortest Path

Given s, find shortest $s-v$ path for all v

No Cycles

Directed

Given s, find shortest $s-v$ path for all v

Use topological sort, then use DP.

Use topological sort to find vertex order $1,2,...,n$ so every directed edge (i,j) has i appear before j in list.

Any vertex v that appears before s in list will not have a path from s to v, so we can discard then from list. Now s will be the first element$

Let $A[i]$ be the distance from s to the vertex at index i, n be the length of the list after discarding

```
  A[i] = infty for all i
  A[0] = 0
  for i in 0...n:
    for every edge (vertex at index i, vertex at index j):
      A[i] = min(A[i], A[j] + weight(vertex at index i, vertex at index
j))
```

Undirected

Its a tree, the shortest path is the only path

No negative weights

Dijkstras

Allow negative weights and allow cycles but no negative weight cycles

Bellman-Ford

All shortest paths - no negative weight cycles

Floyd-Warshall

3Sum

Link

https://leetcode.com/problems/3sum/

Where

Leetcode

Difficulty

Medium

Description

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

Notice that the solution set must not contain duplicate triplets.

Solution Main Idea

Use twosum to solve threesum. For each index i in num, let twoSum find the two indices j, k in $num[i+1:]$ such that $num[j] + num[k] = -num[i]$. Sort the array first so that all result tuples follow invariant of solution: a[i] <= a[j] <= a[k]. Store result in tuple so we can hash it. The invariant allows for hashing to remove duplicates

Best Time to Buy and Sell Stock

Link

https://leetcode.com/problems/best-time-to-buy-and-sell-stock/

Where

Leetcode

Difficulty

Easy

Description

You are given an array prices where prices[i] is the price of a given stock on the ith day. You want to maximize your profit by choosing a single day to buy one stock and choosing a different day in the future to sell that stock. Return the maximum profit you can achieve from this transaction. If you cannot achieve any profit, return 0.

Solution Main Idea

Keep a pointer for low and high. If price[low] < price[high] then check if its more than maxProfit. In either case move high pointer to the right. Else if price[low] > price[high] then set low pointer to equal high pointer and move high pointer to the right. Do this while both pointers are in range. Report highest profit

Climbing Stairs

Link

https://leetcode.com/problems/climbing-stairs/

Where

Leetcode

Difficulty

Easy

Description

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

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

Solution Main Idea

Use DP. Let $totalSteps[i]$ denote the num of distinct ways to reach the top when you're at step i. step 0 indicates that you're not on the stairs yet. step n indicates you're at the top.

Fill in $totalSteps$ from n to 0. $totalSteps[i] = totalSteps[i+1] + totalSteps[i+2]$

Clone Graph

Link

https://leetcode.com/problems/clone-graph/

Where

Leetcode

Difficulty

Medium

Description

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

Return a deep copy (clone) of the graph.

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

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

Test case format:

For simplicity, each node's value is the same as the node's index (1-indexed). For example, the first node with val == 1, the second node with val == 2, and so on. The graph is represented in the test case using an adjacency list.

An adjacency list is a collection of unordered lists used to represent a finite graph. Each list describes the set of neighbors of a node in the graph.

The given node will always be the first node with val = 1. You must return the copy of the given node as a reference to the cloned graph.

Solution Main Idea

Use DFS. Have a dictionary where key is the node val and value is the node. For each neighbor, first check that it is in the mapper, if it is then add from the map to the cloned neighbor array. If not then create a new node through DFS. DFS returns the cloned node.

Space is $O(V)$ since each each node is stored in memory only once. All neighbors of a node are pointers to an entry in the mapper in memory.

Coin Change

Link

https://leetcode.com/problems/coin-change/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

DP. Let A[i] denote the least amount of coins needed to fulfill amount i. $A[i] = min(A[i], 1+A[i-coin])$ Let $A[i]$ be the minimum of using that coin or not using that coin.

Combination Sum 4

Link

https://leetcode.com/problems/combination-sum-iv/submissions/

Where

Leetcode

Difficulty

Medium

Description

Given an array of distinct integers nums and a target integer target, return the number of possible combinations that add up to target.

The test cases are generated so that the answer can fit in a 32-bit integer.

Solution Main Idea

Use DP. Let A[i] be the number of permutations to reach i. For each i, go through each num in nums, such that $A[i] = A[i - num] + A[i]$ (We take it or not)

Contains Duplicate

Link

https://leetcode.com/problems/contains-duplicate/

Where

Leetcode

Difficulty

Easy

Description

Given an integer array nums, return true if any value appears at least twice in the array, and return false if every element is distinct.

Solution Main Idea

Keep dictionary of num frequency, find values in dictionary with frequency $\geq$ 2

Decode Ways

Link

https://leetcode.com/problems/decode-ways/

Where

Leetcode

Difficulty

Medium

Description

A message containing letters from A-Z can be encoded into numbers using the following mapping:

'A' -> "1" 'B' -> "2" ... 'Z' -> "26" To decode an encoded message, all the digits must be grouped then mapped back into letters using the reverse of the mapping above (there may be multiple ways). For example, "11106" can be mapped into:

"AAJF" with the grouping (1 1 10 6) "KJF" with the grouping (11 10 6) Note that the grouping (1 11 06) is invalid because "06" cannot be mapped into 'F' since "6" is different from "06".

Given a string s containing only digits, return the number of ways to decode it.

The test cases are generated so that the answer fits in a 32-bit integer.

Solution Main Idea

Use DP. Similar to work-break. Let $A[i]$ represent the number of ways to decode in suffix $s[i:]$. For each $s[i:]$, find all mappings that matches the prefix of $s[i:]$, and add up the number of ways to decode all suffixes that have the prefix removed.

$A[i] += A[i + len(match)]$

Find Minimum in Rotated Sorted Array

Link

https://leetcode.com/problems/find-minimum-in-rotated-sorted-array/

Where

Leetcode

Difficulty

Medium

Description

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

[4,5,6,7,0,1,2] if it was rotated 4 times. [0,1,2,4,5,6,7] if it was rotated 7 times. Notice that rotating an array [a[0], a[1], a[2], ..., a[n-1]] 1 time results in the array [a[n-1], a[0], a[1], a[2], ..., a[n-2]].

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

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

Solution Main Idea

Divide and conquer using binary search. An inflection point is defined to be between i and $i+1$ where $A[i-1] < A[i] > A[i+1]$. $A[i]$ would be the biggest element in nums, and $A[i+1]$ would be the smallest.

All the elements to the left of inflection point > first element of the array. All the elements to the right of inflection point < first element of the array.

4 5 6 7 2 3

1. Find the mid element of the array.
2. If mid element > first element of array this means that we need to look for the inflection point on the right of mid.
3. If mid element < first element of array this that we need to look for the inflection point on the left of mid.

House Robber

Link

https://leetcode.com/problems/house-robber/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

Use DP. Let $A[i]$ be the maximum amount of money you can steal from houses in $nums[0:i]$ (subarray of $nums$ ending at i).

$A[i] = max(nums[i] + A[i-2] \text{(include if i - 2 >=0)}, A[i - 1])$ (Steal from i or not). Return $A[len(nums)-1]$

House Robber 2

Link

https://leetcode.com/problems/house-robber-ii/

Where

Leetcode

Difficulty

Medium

Description

You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed. All houses at this place are arranged in a circle. That means the first house is the neighbor of the last one. Meanwhile, adjacent houses have a security system connected, and it will automatically contact the police if two adjacent houses were broken into on the same night.

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

Solution Main Idea

Just use house-robber twice. Once without the first element, once without the last. Return the max of the two

Jump Game

Link

https://leetcode.com/problems/jump-game/submissions/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

Use DP. Let $A[i]$ represent the furthest one could reach by jumping through a particular subsequence in $nums[0:i]$ that ends at i. If $A[i-1] >= i$, meaning we can jump to i, then $A[i] = max(A[i-1], i + nums[i])$. (If we jump from i, then the furthest we can go from i is $i + nums[i]$, if we don't jump from i, then in order to proceed we must jump from somewhere before i, and furthest we could jump from before i is $A[i-1]$.)

Longest Increasing Subsequence

Link

https://leetcode.com/problems/longest-increasing-subsequence/

Where

Leetcode

Difficulty

Medium

Description

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

A subsequence is a sequence that can be derived from an array by deleting some or no elements without changing the order of the remaining elements. For example, [3,6,2,7] is a subsequence of the array [0,3,1,6,2,2,7].

Solution Main Idea

Let $A[i]$ denote the LIS that ends at index i.

$$A[i] = max(A[j]+1: j<i, N[j]<N[i])$$

max over all $A[i]$'s

Base case: all $A[i]$'s = 1

Maximum Product Subarray

Link

https://leetcode.com/problems/maximum-product-subarray/

Where

Leetcode

Difficulty

Medium

Description

Given an integer array nums, find a contiguous non-empty subarray within the array that has the largest product, and return the product.

The test cases are generated so that the answer will fit in a 32-bit integer.

A subarray is a contiguous subsequence of the array.

Solution Main Idea

Similar to maximum-subarray problem. DP. Let local_max[i] denote the largest product among all subarrays of $nums$ that end at index i. local_max[i] denote the smallest product among all subarrays of $nums$ that end at index i. If $nums[i]$ is negative, then what was local max in the previous i might very well be the smallest in this i. What was local min in the previous i might be the biggest in this i. We also have to consider $num[i]$ on its own as well

Maximum Subarray

Link

https://leetcode.com/problems/maximum-subarray/submissions/

Where

Leetcode

Difficulty

Easy

Description

Given an integer array nums, find the contiguous subarray (containing at least one number) which has the largest sum and return its sum.

A subarray is a contiguous part of an array.

Solution Main Idea

DP. Let local_max[i] be the biggest sum out of all possible subarrays that end at index i.

local_max[i] = max(nums[i], local_max[i-1] + nums[i])

in other words

local_max = max(nums[i], local_max + nums[i])

Get max of all local_max[i]'s to obtain answer

product-of-array-except-self

Link

https://leetcode.com/problems/product-of-array-except-self/

Where

Leetcode

Difficulty

Easy

Description

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.

Solution Main Idea

Keep track of a suffix and prefix array. Prefix[i] = product of the prefix of num bounded by i excluding i. Suffix[i] = product of suffix of num bounded by i excluding i. $answer[i] = prefix[i] = suffix[i]$

Unique Paths

Link

https://leetcode.com/problems/unique-paths/

Where

Leetcode

Difficulty

Medium

Description

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

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

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

Solution Main Idea

Use DP. Let $A[i][j]$ represent the number of ways to reach row i col j. $A[i][j] = A[i-1][j] + A[i][j-1]$. Return $A[n-1][m-1]$

Word Break

Link

https://leetcode.com/problems/word-break/submissions/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

Use DP. let $A[i]$ represent whether or not we can express suffix $s[i:]$ as a combination of wordDict. For each word $matchingPrefix$ that matches the prefix of the suffix $s[i:]$, $A[i] = A[i+len(matchingPrefix)] or A[i]$ (We use it or dont use it). Return $A[0]$

Or

Use BFS. Let s be a vertex such that $vertices[s]$ is a string. Get all possible words in wordDict that is a prefix of $vertices[s]$ and set the suffix of $vertices[s]$ excluding them as adjacent to $vertices[s]$. Let $vertices[""]$ be the vertex we wish to visit. Run BFS from $vertices[s]$ and return true if we visit $vertices[""]$. Else false.

Course Schedule

Link

https://leetcode.com/problems/course-schedule/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

Determine if there is a topological sort. A topological sort exists in a graph iff it is a DAG (Directed Acyclic Graph). Can finish all courses iff it is a DAG.

Pacific Atlantic Water Flow

Link

https://leetcode.com/problems/pacific-atlantic-water-flow/

Where

Leetcode

Difficulty

Medium

Description

There is an m x n rectangular island that borders both the Pacific Ocean and Atlantic Ocean. The Pacific Ocean touches the island's left and top edges, and the Atlantic Ocean touches the island's right and bottom edges.

The island is partitioned into a grid of square cells. You are given an m x n integer matrix heights where heights[r][c] represents the height above sea level of the cell at coordinate (r, c).

The island receives a lot of rain, and the rain water can flow to neighboring cells directly north, south, east, and west if the neighboring cell's height is less than or equal to the current cell's height. Water can flow from any cell adjacent to an ocean into the ocean.

Return a 2D list of grid coordinates result where result[i] = [ri, ci] denotes that rain water can flow from cell (ri, ci) to both the Pacific and Atlantic oceans.

Solution Main Idea

Use floodfill (BFS or DFS). Instead of solving "which cells can flow to the ocean", think about "which cells can the ocean flow to if water can flow upwards". Cells adjacent to ocean are coloured. Colour all cells that are adjacent to colour cells, with condition that cell to traverse to must have height greater or equal to current cell

Longest Consecutive Sequence

Link

https://leetcode.com/problems/longest-consecutive-sequence/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

Use DP. Let maxLength memo the best solution we have so far. Convert $nums$ to a set to ignore duplicates as they do not matter. Find the longest streak by first finding the smallest in a streak and iteratively looking if its +1 exists. Each streak is mutually exclusive so $O(n)$

Insert Interval

Link

https://leetcode.com/problems/insert-interval/

Where

Leetcode

Difficulty

Medium

Description

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

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

Return intervals after the insertion.

Solution Main Idea

Cases: The interval appears entirely before intervals The interval appears entirely after intervals The interval appears inside intervals but not intersecting The interval appears inside intervals but is intersecting.

Deal with fourth case by finding the start and end values of the merged interval using min and max

Reverse Linked List

Link

https://leetcode.com/problems/reverse-linked-list/

Where

Leetcode

Difficulty

Easy

Description

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

Solution Main Idea

Keep track of oldhead and newhead.

Linked List Cycle

Link

https://leetcode.com/problems/linked-list-cycle/

Where

Leetcode

Difficulty

Easy

Description

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

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

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

Solution Main Idea

Use two pointers. One fast and one slow. Fast pointer moves forward twice for every time slow pointer moves forward once. If they land on the same node then there is a cycle

Merge Two Sorted Lists

Link

https://leetcode.com/problems/merge-two-sorted-lists/

Where

Leetcode

Difficulty

Easy

Description

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

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

Return the head of the merged linked list.

Solution Main Idea

Same idea as the merging in merge sort

Merge k Sorted Lists

Link

https://leetcode.com/problems/merge-k-sorted-lists/

Where

Leetcode

Difficulty

Hard

Description

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.

Solution Main Idea

Use a min heap PQ that holds the heads of all linked lists. Pop the head from the top of the PQ and append it to the back of the new linked list. Add the next of head back into PQ. Continue until PQ is empty

Remove Nth Node From End of List

Link

https://leetcode.com/problems/remove-nth-node-from-end-of-list/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

Use tail recursion to get a node's location relative to the end of the list. Remove if its location is n

Reorder List

Link

https://leetcode.com/problems/reorder-list/

Where

Leetcode

Difficulty

Medium

Description

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

L0 → L1 → … → Ln - 1 → Ln Reorder the list to be on the following form:

L0 → Ln → L1 → Ln - 1 → L2 → Ln - 2 → … You may not modify the values in the list's nodes. Only nodes themselves may be changed.

Solution Main Idea

Three steps. Find middle node, reverse list at middle node, reorder by alternate setting heads

Longest Substring Without Repeating Characters

Link

https://leetcode.com/problems/longest-substring-without-repeating-characters/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

Sliding Window, head and tail. If we haven't seen $s[head]$ before then $s[tail, head inc]$ is the longest valid substring that ends at $head$. We then inc head. If we have seen $s[head]$ before then we inc $tail$ and remove from set until we reach scenario 1, in which $s[tail, head inc]$ is the longest valid substring that ends at $head$.

Think about how this is a subset of DP. Use a valid $s[tail, head inc]$ to efficiently find the next $s[tail, head inc]$ after inc head by 1.

Unlike longest-repeating-character-replacement, we cannot inc head if its invalid, because we cannot test if $s[tail, head]$ is valid after we have moved head. (Moving head when its invalid is dangerous as we can get false positives for validity from just testing $s[head]$, we don't have enough info from just $s[head]$ alone without knowing if the previous $s[tail,head]$ was valid or not)

Valid Parentheses

Link

https://leetcode.com/problems/valid-parentheses/

Where

Leetcode

Difficulty

Easy

Description

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

An input string is valid if:

Open brackets must be closed by the same type of brackets. Open brackets must be closed in the correct order.

Solution Main Idea

Use a stack that represents opening brackets. A closing bracket must match the opening bracket at top of stack to be valid. If they match then pop from stack. The stack grows when we see an opening bracket. Whole string is valid only if stack is empty in the end.

Longest Repeating Character Replacement

Link

https://leetcode.com/problems/longest-repeating-character-replacement/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

Sliding window. Keep track of freq of all letters inside head and tail inclusive. The number of changes needed to make s[head, tail inc] valid is $head - tail + 1 - freq[mostFreqLetter]$. Valid iff $\le k$.

Inc tail if not valid and remove from freq dict. We can determine if $s[head, tail inc]$ is valid without needing to know if the previous $s[head, tail inc]$ was valid or not, so doesn't matter if its valid or not, always increment head.

Valid Anagram

Link

https://leetcode.com/problems/valid-anagram/submissions/

Where

Leetcode

Difficulty

Easy

Description

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

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.

Solution Main Idea

Build freqDict from s, decrease from freqDict with characters in t. Return True if freqDict is all zeros.

Group Anagrams

Link

https://leetcode.com/problems/group-anagrams/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

Sort the strings and add to dictionary, then convert dictionary into list of lists

Valid Palindrome

Link

https://leetcode.com/problems/valid-palindrome/

Where

Leetcode

Difficulty

Easy

Description

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

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

Solution Main Idea

Left and right pointers

Longest Palindromic Substring

Link

https://leetcode.com/problems/longest-palindromic-substring/submissions/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

At each index i obtain the local longest palindrome rooted at i. Be mindful of palindromes rooted at i and $i+1$ as well. Then achieve a global longest palindrome

Palindromic Substrings

Link

https://leetcode.com/problems/palindromic-substrings/submissions/

Where

Leetcode

Difficulty

Medium

Description

Given a string s, return the number of palindromic substrings in it.

A string is a palindrome when it reads the same backward as forward.

A substring is a contiguous sequence of characters within the string.

Solution Main Idea

For each index i, obtain all palindromes rooted at $s[i]$ and $s[i:i+1]$ (is applicable) and accumulate.

Maximum Depth of Binary Tree

Link

https://leetcode.com/problems/maximum-depth-of-binary-tree/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

Recurse to get maxDepth of left and right subtree, max over both + 1

Same Tree

Link

https://leetcode.com/problems/same-tree/

Where

Leetcode

Difficulty

Easy

Description

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

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

Solution Main Idea

Check if left and right trees are the same

Invert Binary Tree

Link

https://leetcode.com/problems/invert-binary-tree/

Where

Leetcode

Difficulty

Easy

Description

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

Solution Main Idea

Invert left and right subtree

Binary Tree Maximum Path Sum

Link

https://leetcode.com/problems/binary-tree-maximum-path-sum/

Where

Leetcode

Difficulty

Hard

Description

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

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

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

Solution Main Idea

Obtain $pathMax$ from recursing on left and right subtrees, where $pathMax$ represents the maximum path sum of the subtree a that does not include both $a.left$ and $a.right$. Meaning we can construct a longer path from a by adding $root$ to a. However at each node we obtain another sum that is the maximum path that can include both $a.left$ and $a.right$ and have that increment a global max.

Observation: if a path goes from one subtree to another subtree, the parent node cannot build on that path.

Binary Tree Level Order Traversal

Link

https://leetcode.com/problems/binary-tree-level-order-traversal/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

BFS while keep tracking track of the node's distance away from root. Insert to list at index $distance$

Serialize and Deserialize Binary Tree

Link

https://leetcode.com/problems/serialize-and-deserialize-binary-tree/

Where

Leetcode

Difficulty

Hard

Description

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. You do not necessarily need to follow this format, so please be creative and come up with different approaches yourself.

Solution Main Idea

Serialize

Return preorder traversal

Deserialize

Convert preorder to Tree using recursion

Subtree of Another Tree

Link

https://leetcode.com/problems/subtree-of-another-tree/

Where

Leetcode

Difficulty

Easy

Description

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

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

Solution Main Idea

Recurse through tree, find all instances where the root val and subtree val are the same. Recurse on those instances to determine if its the same tree.

Validate Binary Search Tree

Link

https://leetcode.com/problems/validate-binary-search-tree/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

Check if left and right subtrees are BST Check if $root.left.val < root.val && root.val < root.right.val$ Check if the biggest value in left subtree < root.val Check if the smallest value in right subtree > root.val

Kth Smallest Element in a BST

Link

https://leetcode.com/problems/kth-smallest-element-in-a-bst/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

In order traversal traverses the tree in order.

Lowest Common Ancestor of a Binary Search Tree

Link

https://leetcode.com/problems/lowest-common-ancestor-of-a-binary-search-tree/

Where

Leetcode

Difficulty

Easy

Description

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

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

Solution Main Idea

A node is a lowest common ancestor of p and q if:

```
    if p.val < root.val and root.val < q.val

    if q.val < root.val and root.val < p.val

    if p == root and (q.val < root.val or root.val < q.val)

    if q == root and (p.val < root.val or root.val < p.val)
```

We test for it top down to avoid weird edge cases

Top K Frequent Elements

Link

https://leetcode.com/problems/top-k-frequent-elements/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

Get list of freq, heapify into max heap and then heap pop the first k

Find Median from Data Stream

Link

https://leetcode.com/problems/find-median-from-data-stream/

Where

Leetcode

Difficulty

Hard

Description

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.

Solution Main Idea

Have two heaps, one max heap for holding values less than and including median, one minheap for holding values greater than medium. Rebalance to maintain that invariant

Robot Bounded In Circle

Link

https://leetcode.com/problems/robot-bounded-in-circle/

Where

Leetcode

Difficulty

Medium

Description

On an infinite plane, a robot initially stands at (0, 0) and faces north. Note that:

The north direction is the positive direction of the y-axis. The south direction is the negative direction of the y-axis. The east direction is the positive direction of the x-axis. The west direction is the negative direction of the x-axis. The robot can receive one of three instructions:

"G": go straight 1 unit. "L": turn 90 degrees to the left (i.e., anti-clockwise direction). "R": turn 90 degrees to the right (i.e., clockwise direction). 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.

Solution Main Idea

Determine if the total displacement is 0 after four cycles of instructions.

Number of Islands

Link

https://leetcode.com/problems/number-of-islands/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

Component finding. Floodfill (DFS) on unvisited cell that is land and inc island count

Bus Routes

Link

https://leetcode.com/problems/bus-routes/

Where

Leetcode

Difficulty

Hard

Description

You are given an array routes representing bus routes where routes[i] is a bus route that the ith bus repeats forever.

For example, if routes[0] = [1, 5, 7], this means that the 0th bus travels in the sequence 1 -> 5 -> 7 -> 1 -> 5 -> 7 -> 1 -> ... forever. You will start at the bus stop source (You are not on any bus initially), and you want to go to the bus stop target. You can travel between bus stops by buses only.

Return the least number of buses you must take to travel from source to target. Return -1 if it is not possible.

Solution Main Idea

Use BFS.

To fulfill time constraint:

1. We do not want to go on the same bus more than once
2. We do not want to go to the same station more than once (dealt with by visited set)

To fulfill 1, our adjacency list should maintain info about each bus separately so that we can prune it from routes after we've used that bus route once

Permutations II

Link

https://leetcode.com/problems/permutations-ii/

Where

Leetcode

Difficulty

Medium

Description

Given a collection of numbers, nums, that might contain duplicates, return all possible unique permutations in any order.

Example 1:

Input: nums = [1,1,2] Output: [[1,1,2], [1,2,1], [2,1,1]] Example 2:

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

Solution Main Idea

Use backtracking. Keep track of each num's frequency (amount of each num that is available). For each num that is available and has not yet been chosen already in this stage of recursion, choose it (prune others by not recursing into it - backtracking). Recurse and treat result as list of prefix of the array. Append num to the end of the array. Add array to the result array and return that.

(We do this to use .append which takes O(1))

{1,1,2,3}, choose 1: {1,2,3}[1], choose 2: {1,3}[2,1],

choose 3: {1}[3,2,1], choose 1: {}[1,3,2,1]

, choose 1: {3}[1,2,1], choose 3: {}[3,1,2,1]

, choose 1: {2,3}[1,1], choose 2: {3}[2,1,1], choose 3: {}[3,2,1,1]

, choose 3: {2}[3,1,1], choose 2: {}[2,3,1,1]

...

, #DONT CHOOSE 1 AGAIN

, choose 2: {1,1,3}[2]...

....

Find a Corresponding Node of a Binary Tree in a Clone of That Tree

Link

https://leetcode.com/problems/find-a-corresponding-node-of-a-binary-tree-in-a-clone-of-that-tree/

Where

Leetcode

Difficulty

Medium

Description

Given two binary trees original and cloned and given a reference to a node target in the original tree.

The cloned tree is a copy of the original tree.

Return a reference to the same node in the cloned tree.

Note that you are not allowed to change any of the two trees or the target node and the answer must be a reference to a node in the cloned tree.

Solution Main Idea

Find the potential target in original, and then check if original and cloned are the same tree from that root

Balanced Binary Tree

Link

https://leetcode.com/problems/balanced-binary-tree/

Where

Leetcode

Difficulty

Medium

Description

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

For this problem, a height-balanced binary tree is defined as:

a binary tree in which the left and right subtrees of every node differ in height by no more than 1.

Solution Main Idea

Recurse on left and right tree, returning both the height of the subtree as well as if the subtree is a balanced tree. If subtree isnt a balanced tree then return not balanced. Otherwise, the current tree is balanced if abs(left[1] - right[1]) <=1.>

Diameter of Binary Tree

Link

https://leetcode.com/problems/diameter-of-binary-tree/

Where

Leetcode

Difficulty

Easy

Description

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

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

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

Solution Main Idea

Recurse to get the max len from left subtree to a leaf in the left subtree, likewise for right subtree. The max len that includes the cur node is the summation of the recursion results. Update max len if that max cur that includes the cur node is more than max len globally.

Middle of the Linked List

Link

https://leetcode.com/problems/middle-of-the-linked-list/

Where

Leetcode

Difficulty

Easy

Description

Given the head of a singly linked list, return the middle node of the linked list.

If there are two middle nodes, return the second middle node.

Solution Main Idea

Traverse once to get length, traverse again until you get to the middle

Maximum Profit in Job Scheduling

Link

https://leetcode.com/problems/maximum-profit-in-job-scheduling/

Where

Leetcode

Difficulty

Hard

Description

We have n jobs, where every job is scheduled to be done from startTime[i] to endTime[i], obtaining a profit of profit[i].

You're given the startTime, endTime and profit arrays, return the maximum profit you can take such that there are no two jobs in the subset with overlapping time range.

If you choose a job that ends at time X you will be able to start another job that starts at time X.

Solution Main Idea

Sort by endTime. For each job you take or dont take. If you take, the best profit in that scenario would be the profit of the job + the most profit you can obtain right before that jobs start time. If you dont take, the best profit is the profit of the job before this job. Max over both scenarios.

`A[i] = max(A[i-1],p+A[justBefore])`, where `justBefore` is found via binary search on endTime. If `justBefore` is not found then `A[i] = max(A[i-1],p)`

`A[i]` denotes the most profit you can obtain from the prefix of the jobs array up to including index `i`

Perfect Squares

Link

https://leetcode.com/problems/perfect-squares/description/

Where

Leetcode

Difficulty

Medium

Description

Given an integer n, return the least number of perfect square numbers that sum to n.

A perfect square is an integer that is the square of an integer; in other words, it is the product of some integer with itself. For example, 1, 4, 9, and 16 are perfect squares while 3 and 11 are not.

Solution Main Idea

Similar to coin change problem, expect you have to make your own coins. Generate all squares up to n and then run coin change algo.

First Bad Version

Link

https://leetcode.com/problems/first-bad-version/

Where

Leetcode

Difficulty

Easy

Description

You are a product manager and currently leading a team to develop a new product. Unfortunately, the latest version of your product fails the quality check. Since each version is developed based on the previous version, all the versions after a bad version are also bad.

Suppose you have n versions [1, 2, ..., n] and you want to find out the first bad one, which causes all the following ones to be bad.

You are given an API bool isBadVersion(version) which returns whether version is bad. Implement a function to find the first bad version. You should minimize the number of calls to the API.

Solution Main Idea

Binary search. If it is a bad version we set tail to be middle to not lose the first bad version

Ransom Note

Link

https://leetcode.com/problems/ransom-note/

Where

Leetcode

Difficulty

Easy

Description

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

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

Solution Main Idea

Use word freq counter

Longest Palindrome

Link

https://leetcode.com/problems/longest-palindrome/

Where

Leetcode

Difficulty

Easy

Description

Given a string s which consists of lowercase or uppercase letters, return the length of the longest palindrome that can be built with those letters.

Letters are case sensitive, for example, "Aa" is not considered a palindrome here.

Solution Main Idea

Use counter and add multiples of 2. If there are any word freqs left then can add 1 as the middle

Majority Element

Link

https://leetcode.com/problems/majority-element/

Where

Leetcode

Difficulty

Easy

Description

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

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

Solution Main Idea

Freq dict

Add Binary

Link

https://leetcode.com/problems/add-binary/

Where

Leetcode

Difficulty

Easy

Description

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

Solution Main Idea

Implement binary adding or use python built-in

01 Matrix

Link

https://leetcode.com/problems/01-matrix/description/

Where

Leetcode

Difficulty

Medium

Description

Given an m x n binary matrix mat, return the distance of the nearest 0 for each cell. The distance between two adjacent cells is 1.

Solution Main Idea

Use BFS. Add all zeros cells to the queue first. Add neighbours to the queue. This ensures that all neighbours popped from front of queue have distance <= distance of neighbours popped later on. Therefore we only need a simple seen set. Add distance to a result matrix for each popped neighbour. We can think of the algo as dropping n many beads of water into bed of water simultaneously. Where the ripples meet are where its equal distance from the source.

Fastest Mode of Transportation

Link

None

Where

Databricks onsite

Difficulty

Medium

Description

Given an m x n matrix, and three modes of transportation: bike, car, and walk, and an array of size 3 representing the cost of each mode. Given a source and a destination. Determine the best mode of transport. You cannot switch modes of transport.

Solution Main Idea

K Closest Points to Origin

Link

https://leetcode.com/problems/k-closest-points-to-origin/description/

Where

Leetcode

Difficulty

Medium

Description

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., $\sqrt{(x1 - x2)2 + (y1 - y2)2}$).

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

Solution Main Idea

Use quick select to find index of k smallest

Evaluate Reverse Polish Notation

Link

https://leetcode.com/problems/evaluate-reverse-polish-notation/description/

Where

Leetcode

Difficulty

Medium

Description

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

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

Note that:

The valid operators are '+', '-', '*', and '/'. Each operand may be an integer or another expression. The division between two integers always truncates toward zero. There will not be any division by zero. The input represents a valid arithmetic expression in a reverse polish notation. The answer and all the intermediate calculations can be represented in a 32-bit integer.

Solution Main Idea

Write a postfix evaluator. Pop from back. When all left is popped away, right is what is left. Or you can write an AST and traverse it

Implement Trie (Prefix Tree)

Link

https://leetcode.com/problems/implement-trie-prefix-tree/

Where

Leetcode

Difficulty

Medium

Description

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

Implement the Trie class:

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

Solution Main Idea

Make trie lol

Min Stack

Link

https://leetcode.com/problems/min-stack/

Where

Leetcode

Difficulty

Medium

Description

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

Implement the MinStack class:

MinStack() initializes the stack object. void push(int val) pushes the element val onto the stack. void pop() removes the element on the top of the stack. int top() gets the top element of the stack. int getMin() retrieves the minimum element in the stack. You must implement a solution with O(1) time complexity for each function.

Solution Main Idea

Have a container dict to keep track of which elements are actually in the data structure. When we pop min or just pop, pop until all shadow elements are gone.

Rotting Oranges

Link

https://leetcode.com/problems/rotting-oranges/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

Use BFS with all rotten oranges as start nodes. Obtain distance matrix of all nodes from rotten oranges. Return max distance among all oranges. But return -1 if there are oranges that cant be rotted

Reconstruct Itinerary

Link

https://leetcode.com/problems/reconstruct-itinerary/description/

Where

Leetcode

Difficulty

Hard

Description

You are given a list of airline tickets where tickets[i] = [fromi, toi] represent the departure and the arrival airports of one flight. Reconstruct the itinerary in order and return it.

All of the tickets belong to a man who departs from "JFK", thus, the itinerary must begin with "JFK". If there are multiple valid itineraries, you should return the itinerary that has the smallest lexical order when read as a single string.

For example, the itinerary ["JFK", "LGA"] has a smaller lexical order than ["JFK", "LGB"]. You may assume all tickets form at least one valid itinerary. You must use all the tickets once and only once.

Solution Main Idea

Create adj list, modify list as you DFS to ensure you only traverse an edge once. Restore that edge once youre done DFS'ing. Sort the neighbours lexicalgraphically. Return first result that uses all edges.

Insert Delete GetRandom O(1)

Link

https://leetcode.com/problems/insert-delete-getrandom-o1/description/

Where

Leetcode

Difficulty

Medium

Description

Implement the RandomizedSet class:

RandomizedSet() Initializes the RandomizedSet object. bool insert(int val) Inserts an item val into the set if not present. Returns true if the item was not present, false otherwise. bool remove(int val) Removes an item val from the set if present. Returns true if the item was present, false otherwise. int getRandom() Returns a random element from the current set of elements (it's guaranteed that at least one element exists when this method is called). Each element must have the same probability of being returned. You must implement the functions of the class such that each function works in average O(1) time complexity.

Solution Main Idea

Have indices dict Have unusedIndices set Have values arr

remove is avg O(1) cuz we keep trying for indices that are not in unusedIndices

Insert Delete GetRandom O(1) - Duplicates allowed

Link

https://leetcode.com/problems/insert-delete-getrandom-o1-duplicates-allowed/description/

Where

Leetcode

Difficulty

Hard

Description

RandomizedCollection is a data structure that contains a collection of numbers, possibly duplicates (i.e., a multiset). It should support inserting and removing specific elements and also reporting a random element.

Implement the RandomizedCollection class:

RandomizedCollection() Initializes the empty RandomizedCollection object. bool insert(int val) Inserts an item val into the multiset, even if the item is already present. Returns true if the item is not present, false otherwise. bool remove(int val) Removes an item val from the multiset if present. Returns true if the item is present, false otherwise. Note that if val has multiple occurrences in the multiset, we only remove one of them. int getRandom() Returns a random element from the current multiset of elements. The probability of each element being returned is linearly related to the number of the same values the multiset contains. You must implement the functions of the class such that each function works on average O(1) time complexity.

Note: The test cases are generated such that getRandom will only be called if there is at least one item in the RandomizedCollection.

Solution Main Idea

Same as insert-delete-getrandom-o1 but use dict of array for indices

Design Underground System

Link

https://leetcode.com/problems/design-underground-system/description/

Where

Leetcode

Difficulty

Medium

Description

An underground railway system is keeping track of customer travel times between different stations. They are using this data to calculate the average time it takes to travel from one station to another.

Implement the UndergroundSystem class:

void checkIn(int id, string stationName, int t) A customer with a card ID equal to id, checks in at the station stationName at time t. A customer can only be checked into one place at a time. void checkOut(int id, string stationName, int t) A customer with a card ID equal to id, checks out from the station stationName at time t. double getAverageTime(string startStation, string endStation) Returns the average time it takes to travel from startStation to endStation. The average time is computed from all the previous traveling times from startStation to endStation that happened directly, meaning a check in at startStation followed by a check out from endStation. The time it takes to travel from startStation to endStation may be different from the time it takes to travel from endStation to startStation. There will be at least one customer that has traveled from startStation to endStation before getAverageTime is called. You may assume all calls to the checkIn and checkOut methods are consistent. If a customer checks in at time t1 then checks out at time t2, then t1 < t2. All events happen in chronological order.

Solution Main Idea

Have a dict that uses (startStation,endStation) as key, keep track of cumulative time and n for each. Have another dict that keeps track of the current station that each id is located at. Use second dict to store time of getting on

Pow(x, n)

Link

https://leetcode.com/problems/powx-n/description/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

Divide and conquer with memoization

String to Integer (atoi)

Link

https://leetcode.com/problems/string-to-integer-atoi/description/

Where

Leetcode

Difficulty

Medium

Description

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:

Read in and ignore any leading whitespace. 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. Read in next the characters until the next non-digit character or the end of the input is reached. The rest of the string is ignored. 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). 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. 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.

Solution Main Idea

Regex [^1]*[-+]?[0-9]+

Decode String

Link

https://leetcode.com/problems/decode-string/description/

Where

Leetcode

Difficulty

Medium

Description

Given an encoded string, return its decoded string.

The encoding rule is: k[encoded_string], where the encoded_string inside the square brackets is being repeated exactly k times. Note that k is guaranteed to be a positive integer.

You may assume that the input string is always valid; there are no extra white spaces, square brackets are well-formed, etc. Furthermore, you may assume that the original data does not contain any digits and that digits are only for those repeat numbers, k. For example, there will not be input like 3a or 2[4].

The test cases are generated so that the length of the output will never exceed 105.

Solution Main Idea

Use a stack. Invariant: by the time you reach a closing bracket, the top of the stack contains only terminal strings. So just multiply by freq and add to next level in stack.

First Missing Positive

Link

https://leetcode.com/problems/first-missing-positive/

Where

Leetcode

Difficulty

Hard

Description

Given an unsorted integer array nums, return the smallest missing positive integer.

You must implement an algorithm that runs in O(n) time and uses constant extra space.

Solution Main Idea

missing pos integer must be in range [1,n] look at code

Two City Scheduling

Link

https://leetcode.com/problems/two-city-scheduling/description/

Where

Leetcode

Difficulty

Medium

Description

A company is planning to interview 2n people. Given the array costs where costs[i] = [aCosti, bCosti], the cost of flying the ith person to city a is aCosti, and the cost of flying the ith person to city b is bCosti.

Return the minimum cost to fly every person to a city such that exactly n people arrive in each city.

Solution Main Idea

Sort by abs diff between two methods, and greedily take the lowest cost of each.

Remove All Adjacent Duplicates in String II

Link

https://leetcode.com/problems/remove-all-adjacent-duplicates-in-string-ii/description/

Where

Leetcode

Difficulty

Medium

Description

You are given a string s and an integer k, a k duplicate removal consists of choosing k adjacent and equal letters from s and removing them, causing the left and the right side of the deleted substring to concatenate together.

We repeatedly make k duplicate removals on s until we no longer can.

Return the final string after all such duplicate removals have been made. It is guaranteed that the answer is unique.

Solution Main Idea

Stack, each element holds char and its frequency. If frequency == k then pop off stack. Afterwards just return char * frequency

Merge Intervals

Link

https://leetcode.com/problems/merge-intervals/description/

Where

Leetcode

Difficulty

Medium

Description

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.

Solution Main Idea

Sort by start, have result list, check if last element in result list overlaps with cur interval, merge and append.

Flatten a Multilevel Doubly Linked List

Link

https://leetcode.com/problems/flatten-a-multilevel-doubly-linked-list/description/

Where

Leetcode

Difficulty

Medium

Description

You are given a doubly linked list, which contains nodes that have a next pointer, a previous pointer, and an additional child pointer. This child pointer may or may not point to a separate doubly linked list, also containing these special nodes. These child lists may have one or more children of their own, and so on, to produce a multilevel data structure as shown in the example below.

Given the head of the first level of the list, flatten the list so that all the nodes appear in a single-level, doubly linked list. Let curr be a node with a child list. The nodes in the child list should appear after curr and before curr.next in the flattened list.

Return the head of the flattened list. The nodes in the list must have all of their child pointers set to null.

Solution Main Idea

Recurse on child nodes. Get recursion result and splice it into current linked list. Return last node of current linked list for splicing.

All Paths From Source to Target

Link

https://leetcode.com/problems/all-paths-from-source-to-target/description/

Where

Leetcode

Difficulty

Medium

Description

Given a directed acyclic graph (DAG) of n nodes labeled from 0 to n - 1, find all possible paths from node 0 to node n - 1 and return them in any order.

The graph is given as follows: graph[i] is a list of all nodes you can visit from node i (i.e., there is a directed edge from node i to node graph[i][j]).

Solution Main Idea

DFS while holding onto buffer. Flush buffer into result when reach end node. Can reuse same buffer by popping before returning

LRU Cache

Link

https://leetcode.com/problems/lru-cache/description/

Where

Leetcode

Difficulty

Medium

Description

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

Implement the LRUCache class:

LRUCache(int capacity) Initialize the LRU cache with positive size capacity. int get(int key) Return the value of the key if the key exists, otherwise return -1. void put(int key, int value) Update the value of the key if the key exists. Otherwise, add the key-value pair to the cache. If the number of keys exceeds the capacity from this operation, evict the least recently used key. The functions get and put must each run in O(1) average time complexity.

Solution Main Idea

Use deque (doubly linked list) to keep track of order. Dict to point to each node via key An actual dict to store key vals

Design A Leaderboard

Link

https://leetcode.com/problems/design-a-leaderboard/description/

Where

Leetcode

Difficulty

Medium

Description

Design a Leaderboard class, which has 3 functions:

addScore(playerId, score): Update the leaderboard by adding score to the given player's score. If there is no player with such id in the leaderboard, add him to the leaderboard with the given score. top(K): Return the score sum of the top K players. reset(playerId): Reset the score of the player with the given id to 0 (in other words erase it from the leaderboard). It is guaranteed that the player was added to the leaderboard before calling this function. Initially, the leaderboard is empty.

Solution Main Idea

Use sorteddict or heap

Meeting Rooms II

Link

https://leetcode.com/problems/meeting-rooms-ii/description/

Where

Leetcode

Difficulty

Medium

Description

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

Solution Main Idea

Sort intervals by starttime. Have a minheap on endtime, where it represents the list of rooms. Top of heap is therefore most likely to be free. Add to heap a new room if no room free

Number of Ships in a Rectangle

Link

https://leetcode.com/problems/number-of-ships-in-a-rectangle/description/

Where

Leetcode

Difficulty

Hard

Description

(This problem is an interactive problem.)

Each ship is located at an integer point on the sea represented by a cartesian plane, and each integer point may contain at most 1 ship.

You have a function Sea.hasShips(topRight, bottomLeft) which takes two points as arguments and returns true If there is at least one ship in the rectangle represented by the two points, including on the boundary.

Given two points: the top right and bottom left corners of a rectangle, return the number of ships present in that rectangle. It is guaranteed that there are at most 10 ships in that rectangle.

Submissions making more than 400 calls to hasShips will be judged Wrong Answer. Also, any solutions that attempt to circumvent the judge will result in disqualification.

Solution Main Idea

Split into 4 quadrants. Divide and conquer on each. We can satisfy constraint because at each level there can only be at most 10 subrectangles with ships. And for each of them we can split at most 4 times. There

are log10 1000000 = 10 levels. So at most 10 x 10 x 4 = 400 calls to API

Search in Rotated Sorted Array

Link

https://leetcode.com/problems/search-in-rotated-sorted-array/description/

Where

Leetcode

Difficulty

Medium

Description

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

Prior to being passed to your function, nums is possibly rotated at an unknown pivot index k (1 <= k < nums.length) such that the resulting array is [nums[k], nums[k+1], ..., nums[n-1], nums[0], nums[1], ..., nums[k-1]] (0-indexed). For example, [0,1,2,4,5,6,7] might be rotated at pivot index 3 and become [4,5,6,7,0,1,2].

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

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

Solution Main Idea

Binary search for min Use min index as offset to map

Lowest Common Ancestor of a Binary Tree

Link

https://leetcode.com/problems/lowest-common-ancestor-of-a-binary-tree/description/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

Recurse on left and right, keep track of which nodes are found in which subtree. Propagate back up

Move Zeroes

Link

https://leetcode.com/problems/move-zeroes/description/

Where

Leetcode

Difficulty

Easy

Description

Given an integer array nums, move all 0's to the end of it while maintaining the relative order of the non-zero elements.

Note that you must do this in-place without making a copy of the array.

Solution Main Idea

Iterate from left to right, keep track of index of first 0 in list. When we see a non-zero, swap with first 0.

Lowest Common Ancestor of a Binary Tree III

Link

https://leetcode.com/problems/lowest-common-ancestor-of-a-binary-tree-iii/description/

Where

Leetcode

Difficulty

Medium

Description

Given two nodes of a binary tree p and q, return their lowest common ancestor (LCA).

Each node will have a reference to its parent node. The definition for Node is below:

`class Node { public int val; public Node left; public Node right; public Node parent; } According to the definition of LCA on Wikipedia: "The lowest common ancestor of two nodes p and q in a tree T is the lowest node that has both p and q as descendants (where we allow a node to be a descendant of itself)."

Solution Main Idea

Traverse to parent. Single path to root node. Return point of intersection for both nodes. ````

Word Break II

Link

https://leetcode.com/problems/word-break-ii/description/

Where

Leetcode

Difficulty

Hard

Description

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.

Solution Main Idea

Split the string at all indexes. Then basically there is always two parts - the first part (word) and rest of the string. If word is in the wordDict, take the second part, and do the same thing until there is no string left. At that point, simply join the path with space and append to the output.

Convert wordDict to hashset for O(1) lookup time.

Minimum Number of Steps to Make Two Strings Anagram

Link

https://leetcode.com/problems/minimum-number-of-steps-to-make-two-strings-anagram/description/

Where

Leetcode

Difficulty

Medium

Description

You are given two strings of the same length s and t. In one step you can choose any character of t and replace it with another character.

Return the minimum number of steps to make t an anagram of s.

An Anagram of a string is a string that contains the same characters with a different (or the same) ordering.

Solution Main Idea

Count delta of char freq. Divide by 2. Delta is always divisible by 2 because if a char is missing in one string at location i. Then a different char must be in that string somewhere that does not exist as frequently in the other string to take its place.

Count Univalue Subtrees

Link

https://leetcode.com/problems/count-univalue-subtrees/description/

Where

Leetcode

Difficulty

Medium

Description

Given the root of a binary tree, return the number of uni-value subtrees.

A uni-value subtree means all nodes of the subtree have the same value.

Solution Main Idea

Recurse and return tuple of if child nodes are unival and their values

Integer to English Words

Link

https://leetcode.com/problems/integer-to-english-words/

Where

Leetcode

Difficulty

Hard

Description

Convert a non-negative integer num to its English words representation.

Solution Main Idea

Mapper and recursion

Sequential Digits

Link

https://leetcode.com/problems/sequential-digits/description/

Where

Leetcode

Difficulty

Medium

Description

An integer has sequential digits if and only if each digit in the number is one more than the previous digit.

Return a sorted list of all the integers in the range [low, high] inclusive that have sequential digits.

Solution Main Idea

BFS on digits.

Kth Smallest Element in a Sorted Matrix

Link

https://leetcode.com/problems/kth-smallest-element-in-a-sorted-matrix/description/

Where

Leetcode

Difficulty

Medium

Description

Given an n x n matrix where each of the rows and columns is sorted in ascending order, return the kth smallest element in the matrix.

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

You must find a solution with a memory complexity better than O(n2).

Solution Main Idea

Similar idea to when we merge n sorted lists. Use minheap k times.

Maximum Level Sum of a Binary Tree

Link

https://leetcode.com/problems/maximum-level-sum-of-a-binary-tree/description/

Where

Leetcode

Difficulty

Medium

Description

Given the root of a binary tree, the level of its root is 1, the level of its children is 2, and so on.

Return the smallest level x such that the sum of all the values of nodes at level x is maximal.

Solution Main Idea

BFS from root.

Number of Times Binary String Is Prefix-Aligned

Link

https://leetcode.com/problems/number-of-times-binary-string-is-prefix-aligned/description/

Where

Leetcode

Difficulty

Medium

Description

You have a 1-indexed binary string of length n where all the bits are 0 initially. We will flip all the bits of this binary string (i.e., change them from 0 to 1) one by one. You are given a 1-indexed integer array flips where flips[i] indicates that the bit at index i will be flipped in the ith step.

A binary string is prefix-aligned if, after the ith step, all the bits in the inclusive range [1, i] are ones and all the other bits are zeros.

Return the number of times the binary string is prefix-aligned during the flipping process.

Solution Main Idea

right is the number of the right most lighted bulb.

Iterate the input light A, update right = max(right, A[i]).

Now we have lighted up i + 1 bulbs, if right == i + 1, it means that all the previous bulbs (to the left) are turned on too. Then we increment res

Reorder Routes to Make All Paths Lead to the City Zero

Link

https://leetcode.com/problems/reorder-routes-to-make-all-paths-lead-to-the-city-zero/

Where

Leetcode

Difficulty

Medium

Description

There are n cities numbered from 0 to n - 1 and n - 1 roads such that there is only one way to travel between two different cities (this network form a tree). Last year, The ministry of transport decided to orient the roads in one direction because they are too narrow.

Roads are represented by connections where connections[i] = [ai, bi] represents a road from city ai to city bi.

This year, there will be a big event in the capital (city 0), and many people want to travel to this city.

Your task consists of reorienting some roads such that each city can visit the city 0. Return the minimum number of edges changed.

It's guaranteed that each city can reach city 0 after reorder.

Solution Main Idea

Have two adj list, one in one direction and another going opposite direction. BFS and count number of opposite edges taken.

Subarray Sum Equals K

Link

https://leetcode.com/problems/subarray-sum-equals-k/

Where

Leetcode

Difficulty

Medium

Description

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

A subarray is a contiguous non-empty sequence of elements within an array.

Solution Main Idea

The idea behind this approach is as follows: If the cumulative sum up to two indices is the same, the sum of the elements lying in between those indices is zero. Extending the same thought further, if the cumulative sum up to two indices, say sum[i] and sum[j] is at a difference of k, the sum of elements lying between indices i and j is k. Store this in hashmap alongside num of occurrences. Key is cum sum. Value is num occurrences. x

Maximum Length of a Concatenated String with Unique Characters

Link

https://leetcode.com/problems/maximum-length-of-a-concatenated-string-with-unique-characters/description/

Where

Leetcode

Difficulty

Medium

Description

You are given an array of strings arr. A string s is formed by the concatenation of a subsequence of arr that has unique characters.

Return the maximum possible length of s.

A subsequence is an array that can be derived from another array by deleting some or no elements without changing the order of the remaining elements.

Solution Main Idea

O(2^n) solution since we explore all possible subsequences of arr of len N

Solution 1: Recursion For each index, explore adding each index after it. Check if there are duplicates

Solution 2: Build on top of solution 1 to use bitmaps to store result. Greatly reduces space complexity O(N) space

Minimum Time to Make Rope Colorful

Link

https://leetcode.com/problems/minimum-time-to-make-rope-colorful/description/

Where

Leetcode

Difficulty

Medium

Description

Alice has n balloons arranged on a rope. You are given a 0-indexed string colors where colors[i] is the color of the ith balloon.

Alice wants the rope to be colorful. She does not want two consecutive balloons to be of the same color, so she asks Bob for help. Bob can remove some balloons from the rope to make it colorful. You are given a 0-

indexed integer array neededTime where neededTime[i] is the time (in seconds) that Bob needs to remove the ith balloon from the rope.

Return the minimum time Bob needs to make the rope colorful.

Solution Main Idea

The minimize time depends on whether if we are cutting the left string or the right string when we encounter two consecutive balloons of the same colour. Just pick the one that takes the least amount of time, and set the colour and the index of that last colour that we have seen to be the index of the string we did not cut. Effectively ignoring the cut string from future considerations

Maximum Number of Balloons

Link

https://leetcode.com/problems/maximum-number-of-balloons/description/

Where

Leetcode

Difficulty

Easy

Description

Given a string text, you want to use the characters of text to form as many instances of the word "balloon" as possible.

You can use each character in text at most once. Return the maximum number of instances that can be formed.

Solution Main Idea

Create a counter of letters. Keep trying to more freq of "balloon" until you cant.

Maximum Value after Insertion

Link

https://leetcode.com/problems/maximum-value-after-insertion/description/

Where

Leetcode

Difficulty

Medium

Description

You are given a very large integer n, represented as a string, and an integer digit x. The digits in n and the digit x are in the inclusive range [1, 9], and n may represent a negative number.

You want to maximize n's numerical value by inserting x anywhere in the decimal representation of n. You cannot insert x to the left of the negative sign.

For example, if n = 73 and x = 6, it would be best to insert it between 7 and 3, making n = 763. If n = -55 and x = 2, it would be best to insert it before the first 5, making n = -255. Return a string representing the maximum value of n after the insertion.

Solution Main Idea

Digits further to the left have more impact on the value of the number. We wish to add the digit at the location with most impact while not overriding digits that would have more impact than the digit we are inserting. Therefore, we greedily go from left to right, finding a location where the value of digits is less than the digit we are adding. Opposite is true for negative

Spiral Matrix III

Link

https://leetcode.com/problems/spiral-matrix-iii/description/

Where

Leetcode

Difficulty

Medium

Description

You start at the cell (rStart, cStart) of an rows x cols grid facing east. The northwest corner is at the first row and column in the grid, and the southeast corner is at the last row and column.

You will walk in a clockwise spiral shape to visit every position in this grid. Whenever you move outside the grid's boundary, we continue our walk outside the grid (but may return to the grid boundary later.). Eventually, we reach all rows * cols spaces of the grid.

Return an array of coordinates representing the positions of the grid in the order you visited them.

Solution Main Idea

We increase the stepVal every time after we have moved two directions. Set upper bound to be 2 * max(rows,cols) increases

Find Winner on a Tic Tac Toe Game

Link

https://leetcode.com/problems/find-winner-on-a-tic-tac-toe-game/description/

Where

Leetcode

Difficulty

Easy

Description

Tic-tac-toe is played by two players A and B on a 3 x 3 grid. The rules of Tic-Tac-Toe are:

Players take turns placing characters into empty squares ' '. The first player A always places 'X' characters, while the second player B always places 'O' characters. 'X' and 'O' characters are always placed into empty squares, never on filled ones. The game ends when there are three of the same (non-empty) character filling any row, column, or diagonal. The game also ends if all squares are non-empty. No more moves can be played if the game is over. Given a 2D integer array moves where moves[i] = [rowi, coli] indicates that the ith move will be played on grid[rowi][coli]. return the winner of the game if it exists (A or B). In case the game ends in a draw return "Draw". If there are still movements to play return "Pending".

You can assume that moves is valid (i.e., it follows the rules of Tic-Tac-Toe), the grid is initially empty, and A will play first.

Solution Main Idea

Solution 1 is to brute force check all cells after each move Solution 2 is to treat "A" as adding 1, "B" as removing 1 to each row,col it affects. -3 means all "B", 3 means all "A"

Find N Unique Integers Sum up to Zero

Link

https://leetcode.com/problems/find-n-unique-integers-sum-up-to-zero/description/

Where

Leetcode

Difficulty

Easy

Description

Given an integer n, return any array containing n unique integers such that they add up to 0.

Solution Main Idea

Palindrome of negative and positive

Minimum Changes To Make Alternating Binary String

Link

https://leetcode.com/problems/minimum-changes-to-make-alternating-binary-string/

Where

Leetcode

Difficulty

Easy

Description

You are given a string s consisting only of the characters '0' and '1'. In one operation, you can change any '0' to '1' or vice versa.

The string is called alternating if no two adjacent characters are equal. For example, the string "010" is alternating, while the string "0100" is not.

Return the minimum number of operations needed to make s alternating.

Solution Main Idea

Small observation that the sequence of index is [0,1,2,3..], if we get its module by 2, we get [0,1,0,1,0..], which is the alternating binary sequence we want.

So we iterate the string, check if the characters[i] is same as the i % 2. Note that s[i] is a character, and s[i] - '0' making it to integer.

We accumulate the number of difference, which is the number of operation to make it into 01 string.

We can do the same to find out res, the number of operation for 10 string. But we don't have to, becaue this equals to s.length - res.

Maximal Network Rank

Link

https://leetcode.com/problems/maximal-network-rank/description/

Where

Leetcode

Difficulty

Medium

Description

There is an infrastructure of n cities with some number of roads connecting these cities. Each roads[i] = [ai, bi] indicates that there is a bidirectional road between cities ai and bi.

The network rank of two different cities is defined as the total number of directly connected roads to either city. If a road is directly connected to both cities, it is only counted once.

The maximal network rank of the infrastructure is the maximum network rank of all pairs of different cities.

Given the integer n and the array roads, return the maximal network rank of the entire infrastructure.

Solution Main Idea

Get adj, obtain deg of all cities. Iterate through all pairs and return largest sum. O(n^2)

Dot Product of Two Sparse Vectors

Link

https://leetcode.com/problems/dot-product-of-two-sparse-vectors/description/

Where

Leetcode

Difficulty

Medium

Description

Given two sparse vectors, compute their dot product.

Implement class SparseVector:

SparseVector(nums) Initializes the object with the vector nums dotProduct(vec) Compute the dot product between the instance of SparseVector and vec A sparse vector is a vector that has mostly zero values, you should store the sparse vector efficiently and compute the dot product between two SparseVector.

Follow up: What if only one of the vectors is sparse?

Solution Main Idea

Keep track of only non zero elements with dict

Minimum Deletions to Make Character Frequencies Unique

Link

https://leetcode.com/problems/minimum-deletions-to-make-character-frequencies-unique/description/

Where

Leetcode

Difficulty

Medium

Description

A string s is called good if there are no two different characters in s that have the same frequency.

Given a string s, return the minimum number of characters you need to delete to make s good.

The frequency of a character in a string is the number of times it appears in the string. For example, in the string "aab", the frequency of 'a' is 2, while the frequency of 'b' is 1.

Solution Main Idea

Get freq, only care about freq. Make sure freq are unique by using set and counting number of times we need to dec freq to be unique

Largest Perimeter Triangle

Link

https://leetcode.com/problems/largest-perimeter-triangle/description/

Where

Leetcode

Difficulty

Easy

Description

Given an integer array nums, return the largest perimeter of a triangle with a non-zero area, formed from three of these lengths. If it is impossible to form any triangle of a non-zero area, return 0.

Solution Main Idea

Sort the list ascending. At any index, the best bet that it will be a valid triangle are the two indices before it (they're the next biggest). It also happens that if they are valid, then those two will be the biggest triangle for that root index.

Minimum Path Sum

Link

https://leetcode.com/problems/minimum-path-sum/description/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

DP, min of top and left

Letter Combinations of a Phone Number

Link

https://leetcode.com/problems/letter-combinations-of-a-phone-number/description/

Where

Leetcode

Difficulty

Medium

Description

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

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

Solution Main Idea

Create adj based on mapping Perform DFS

Minimum Cost For Tickets

Link

https://leetcode.com/problems/minimum-cost-for-tickets/description/

Where

Leetcode

Difficulty

Medium

Description

You have planned some train traveling one year in advance. The days of the year in which you will travel are given as an integer array days. Each day is an integer from 1 to 365.

Train tickets are sold in three different ways:

a 1-day pass is sold for costs[0] dollars, a 7-day pass is sold for costs[1] dollars, and a 30-day pass is sold for costs[2] dollars. The passes allow that many days of consecutive travel.

For example, if we get a 7-day pass on day 2, then we can travel for 7 days: 2, 3, 4, 5, 6, 7, and 8. Return the minimum number of dollars you need to travel every day in the given list of days.

Solution Main Idea

DP On days you are not travelling, the cost is the same as the day before. Else, it is the min of the three possible ways to get to this day.

Reducing Dishes

Link

https://leetcode.com/problems/reducing-dishes/description/

Where

Leetcode

Difficulty

Hard

Description

A chef has collected data on the satisfaction level of his n dishes. Chef can cook any dish in 1 unit of time.

Like-time coefficient of a dish is defined as the time taken to cook that dish including previous dishes multiplied by its satisfaction level i.e. time[i] * satisfaction[i].

Return the maximum sum of like-time coefficient that the chef can obtain after dishes preparation.

Dishes can be prepared in any order and the chef can discard some dishes to get this maximum value.

Solution Main Idea

Sort the array and reverse it. Notice that by adding an element we can increase the previous result as we are "shifting" it. The array being sorted and reversed gives the best chance as we can greedily go through it.

Use DP as well to store prev results. Recurrence relation: e.g going from -1,0,5 -> -8,-1,0,5 # accum1 = -1*1 + 0*2 + 5*3, accum2 = -1+0+5 # newaccum1 = -8*1 + -1*2 + 0*3 + 5*4 # = -8*1 + -1 + -1*1 + 0 + 0*2 + 5 + 5*3 # = -8*1 + -1 + 0 + 5 + -1*1 + 0*2 + 5*3 # = -8*1 + accum2 + accum1 # newaccum2 = -8 + accum2

[^1]: