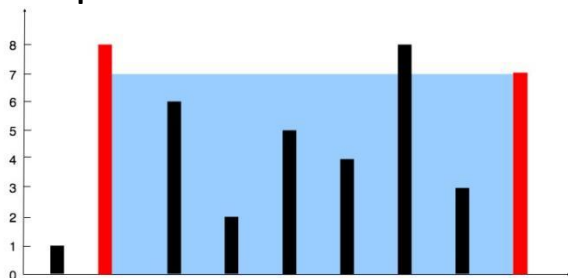


Python Coding Questions

S.NO	Description
101.	<p>The string "PAYPALISHIRING" is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)</p> <pre>P A H N A P L S I I G Y I R</pre> <p>And then read line by line: "PAHNAPLSIIGYIR"</p> <p>Write the code that will take a string and make this conversion given a number of rows:</p> <pre>string convert(string s, int numRows);</pre> <p>Example 1: Input: s = "PAYPALISHIRING", numRows = 3 Output: "PAHNAPLSIIGYIR"</p> <p>Example 2: Input: s = "PAYPALISHIRING", numRows = 4 Output: "PINALSIGYAHRPI"</p> <p>Explanation:</p> <pre>P I N A L S I G Y A H R P I</pre> <p>Constraints: $1 \leq s.length \leq 1000$ s consists of English letters (lower-case and upper-case), ',' and '.'. $1 \leq numRows \leq 1000$</p>
102.	<p>You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the ith line are (i, 0) and (i, height[i]). Find two lines that together with the x-axis form a container, such that the container contains the most water.</p> <p>Return the maximum amount of water a container can store.</p> <p>Notice that you may not slant the container.</p> <p>Example 1:</p>  <p>Input: height = [1,8,6,2,5,4,8,3,7]</p>

	<p>Output: 49</p> <p>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.</p> <p>Constraints: $n == \text{height.length}$ $2 \leq n \leq 10^5$ $0 \leq \text{height}[i] \leq 10^4$</p>
103.	<p>Given an integer array nums, return all the triplets [nums[i], nums[j], nums[k]] such that $i \neq j$, $i \neq k$, and $j \neq k$, and $\text{nums}[i] + \text{nums}[j] + \text{nums}[k] == 0$.</p> <p>Notice that the solution set must not contain duplicate triplets.</p> <p>Example 1: Input: nums = [-1,0,1,2,-1,-4] Output: [[-1,-1,2],[-1,0,1]] Explanation: $\text{nums}[0] + \text{nums}[1] + \text{nums}[2] = (-1) + 0 + 1 = 0$. $\text{nums}[1] + \text{nums}[2] + \text{nums}[4] = 0 + 1 + (-1) = 0$. $\text{nums}[0] + \text{nums}[3] + \text{nums}[4] = (-1) + 2 + (-1) = 0$. The distinct triplets are [-1,0,1] and [-1,-1,2]. Notice that the order of the output and the order of the triplets does not matter.</p> <p>Constraints: $3 \leq \text{nums.length} \leq 3000$ $-10^5 \leq \text{nums}[i] \leq 10^5$</p>
104.	<p>Given an integer array nums of length n and an integer target, find three integers in nums such that the sum is closest to target.</p> <p>Return the sum of the three integers.</p> <p>You may assume that each input would have exactly one solution.</p> <p>Example 1: Input: nums = [-1,2,1,-4], target = 1 Output: 2 Explanation: The sum that is closest to the target is 2. $(-1 + 2 + 1 = 2)$.</p> <p>Constraints: $3 \leq \text{nums.length} \leq 500$ $-1000 \leq \text{nums}[i] \leq 1000$ $-10^4 \leq \text{target} \leq 10^4$</p>
105.	<p>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.</p> <p>A mapping of digits to letters (just like on the telephone buttons) is given below. Note that 1 does not map to any letters.</p>



Example 1:

Input: digits = "23"

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

Constraints:

0 <= digits.length <= 4

digits[i] is a digit in the range ['2', '9'].

106. Given an array `nums` of `n` integers, return an array of all the unique quadruplets `[nums[a], nums[b], nums[c], nums[d]]` such that:
0 <= a, b, c, d < n
a, b, c, and d are distinct.
`nums[a] + nums[b] + nums[c] + nums[d] == target`
You may return the answer in any order.

Example 1:

Input: `nums = [1,0,-1,0,-2,2]`, `target = 0`

Output: `[[-2,-1,1,2],[-2,0,0,2],[-1,0,0,1]]`

Constraints:

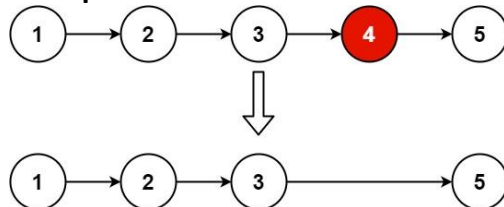
1 <= `nums.length` <= 200

-10⁹ <= `nums[i]` <= 10⁹

-10⁹ <= `target` <= 10⁹

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

Example 1:



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

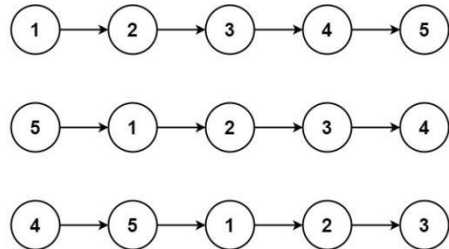

Output: `[1,2,3,5]`

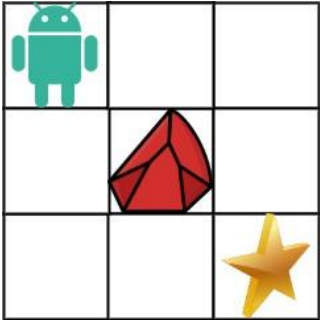
	<p>Constraints:</p> <p>The number of nodes in the list is sz.</p> <p>$1 \leq sz \leq 30$</p> <p>$0 \leq \text{Node.val} \leq 100$</p> <p>$1 \leq n \leq sz$</p>
108.	<p>Given n pairs of parentheses, write a function to generate all combinations of well-formed parentheses.</p> <p>Example 1: Input: n = 3 Output: ["((()))", "(()())", "(())()", "()(())", "()()()"]</p> <p>Example 2: Input: n = 1 Output: ["()"]</p> <p>Constraints: $1 \leq n \leq 8$</p>
109.	<p>There is an integer array nums sorted in ascending order (with distinct values).</p> <p>Prior to being passed to your function, nums is possibly rotated at an unknown pivot index k ($1 \leq k < \text{nums.length}$) 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].</p> <p>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.</p> <p>You must write an algorithm with $O(\log n)$ runtime complexity.</p> <p>Example 1: Input: nums = [4,5,6,7,0,1,2], target = 0 Output: 4</p> <p>Example 2: Input: nums = [4,5,6,7,0,1,2], target = 3 Output: -1</p> <p>Constraints: $1 \leq \text{nums.length} \leq 5000$ $-10^4 \leq \text{nums}[i] \leq 10^4$ All values of nums are unique. nums is an ascending array that is possibly rotated. $-10^4 \leq \text{target} \leq 10^4$</p>

110.	<p>Given an array of integers <code>nums</code> sorted in non-decreasing order, find the starting and ending position of a given target value.</p> <p>If target is not found in the array, return <code>[-1, -1]</code>.</p> <p>You must write an algorithm with $O(\log n)$ runtime complexity.</p> <p>Example 1: Input: <code>nums = [5,7,7,8,8,10]</code>, <code>target = 8</code> Output: <code>[3,4]</code></p> <p>Example 2: Input: <code>nums = [5,7,7,8,8,10]</code>, <code>target = 6</code> Output: <code>[-1,-1]</code></p> <p>Constraints: $0 \leq \text{nums.length} \leq 10^5$ $-10^9 \leq \text{nums}[i] \leq 10^9$ <code>nums</code> is a non-decreasing array. $-10^9 \leq \text{target} \leq 10^9$</p>
111.	<p>The count-and-say sequence is a sequence of digit strings defined by the recursive formula:</p> <p><code>countAndSay(1) = "1"</code></p> <p><code>countAndSay(n)</code> is the way you would "say" the digit string from <code>countAndSay(n-1)</code>, which is then converted into a different digit string.</p> <p>To determine how you "say" a digit string, split it into the minimal number of substrings such that each substring contains exactly one unique digit. Then for each substring, say the number of digits, then say the digit. Finally, concatenate every said digit.</p> <p>For example, the saying and conversion for digit string "3322251":</p> <pre> "3322251" two 3's, three 2's, one 5, and one 1 2 3 + 3 2 + 1 5 + 1 1 "23321511" </pre> <p>Given a positive integer <code>n</code>, return the <code>n</code>th term of the count-and-say sequence.</p> <p>Example 1: Input: <code>n = 1</code> Output: <code>"1"</code> Explanation: This is the base case.</p> <p>Example 2: Input: <code>n = 4</code> Output: <code>"1211"</code></p> <p>Constraints:</p>

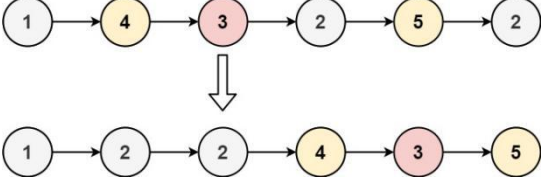
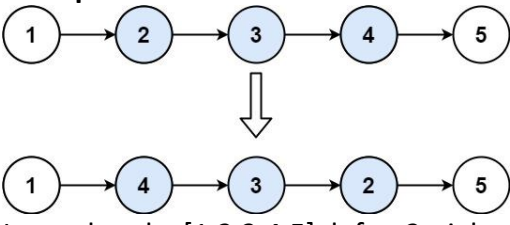
	$1 \leq n \leq 30$
112.	<p>Given an array of distinct integers candidates and a target integer target, return a list of all unique combinations of candidates where the chosen numbers sum to target. You may return the combinations in any order. The same number may be chosen from candidates an unlimited number of times. Two combinations are unique if the frequency of at least one of the chosen numbers is different. The test cases are generated such that the number of unique combinations that sum up to target is less than 150 combinations for the given input.</p> <p>Example 1: Input: candidates = [2,3,6,7], target = 7 Output: [[2,2,3],[7]]</p> <p>Explanation: 2 and 3 are candidates, and $2 + 2 + 3 = 7$. Note that 2 can be used multiple times. 7 is a candidate, and $7 = 7$. These are the only two combinations.</p> <p>Constraints: $1 \leq \text{candidates.length} \leq 30$ $1 \leq \text{candidates}[i] \leq 40$ All elements of candidates are distinct. $1 \leq \text{target} \leq 40$</p>
113.	<p>Given a collection of candidate numbers (candidates) and a target number (target), find all unique combinations in candidates where the candidate numbers sum to target. Each number in candidates may only be used once in the combination. Note: The solution set must not contain duplicate combinations.</p> <p>Example 1: Input: candidates = [10,1,2,7,6,1,5], target = 8 Output: [[1,1,6], [1,2,5], [1,7], [2,6]]</p> <p>Constraints: $1 \leq \text{candidates.length} \leq 100$ $1 \leq \text{candidates}[i] \leq 50$ $1 \leq \text{target} \leq 30$</p>

114.	<p>Given an m x n matrix, return all elements of the matrix in spiral order.</p> <p>Example 1:</p> <table border="1"><tr><td>1 →</td><td>2 →</td><td>3 ↓</td></tr><tr><td>4 →</td><td>5</td><td>6 ↓</td></tr><tr><td>↑ 7</td><td>← 8</td><td>← 9</td></tr></table> <p>Input: matrix = [[1,2,3],[4,5,6],[7,8,9]] Output: [1,2,3,6,9,8,7,4,5]</p> <p>Constraints: m == matrix.length n == matrix[i].length 1 <= m, n <= 10 -100 <= matrix[i][j] <= 100</p>	1 →	2 →	3 ↓	4 →	5	6 ↓	↑ 7	← 8	← 9
1 →	2 →	3 ↓								
4 →	5	6 ↓								
↑ 7	← 8	← 9								
115.	<p>You are given a 0-indexed array of integers nums of length n. You are initially positioned at nums[0].</p> <p>Each element nums[i] represents the maximum length of a forward jump from index i. In other words, if you are at nums[i], you can jump to any nums[i + j] where:</p> <p>0 <= j <= nums[i] and i + j < n</p> <p>Return the minimum number of jumps to reach nums[n - 1]. The test cases are generated such that you can reach nums[n - 1].</p> <p>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.</p> <p>Constraints: 1 <= nums.length <= 10⁴ 0 <= nums[i] <= 1000 It's guaranteed that you can reach nums[n - 1].</p>									
116.	<p>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.</p> <p>Example 1:</p>									

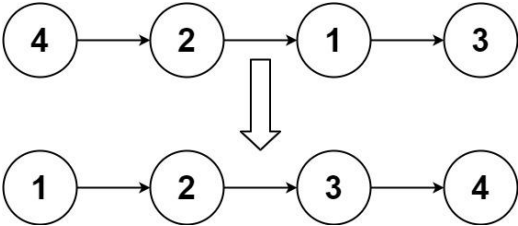
	<p>Input: intervals = [[1,3],[2,6],[8,10],[15,18]]</p> <p>Output: [[1,6],[8,10],[15,18]]</p> <p>Explanation: Since intervals [1,3] and [2,6] overlap, merge them into [1,6].</p> <p>Example 2:</p> <p>Input: intervals = [[1,4],[4,5]]</p> <p>Output: [[1,5]]</p> <p>Explanation: Intervals [1,4] and [4,5] are considered overlapping.</p> <p>Constraints:</p> <p>1 <= intervals.length <= 10⁴</p> <p>intervals[i].length == 2</p> <p>0 <= starti <= endi <= 10⁴</p>
117.	<p>Given the head of a linked list, rotate the list to the right by k places.</p> <p>Example 1:</p>  <p>rotate 1</p> <p>rotate 2</p> <p>Input: head = [1,2,3,4,5], k = 2</p> <p>Output: [4,5,1,2,3]</p> <p>Constraints:</p> <p>The number of nodes in the list is in the range [0, 500].</p> <p>-100 <= Node.val <= 100</p> <p>0 <= k <= 2 * 10⁹</p>
118.	<p>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.</p> <p>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.</p> <p>The test cases are generated so that the answer will be less than or equal to 2 * 10⁹.</p> <p>Example 1:</p>  <p>Input: m = 3, n = 7</p>

	<p>Output: 28</p> <p>Example 2:</p> <p>Input: m = 3, n = 2</p> <p>Output: 3</p> <p>Explanation: From the top-left corner, there are a total of 3 ways to reach the bottom-right corner:</p> <ol style="list-style-type: none"> 1. Right -> Down -> Down 2. Down -> Down -> Right 3. Down -> Right -> Down <p>Constraints:</p> <p>1 <= m, n <= 100</p>
119.	<p>You are given an m x n integer array grid. There is a robot initially located at the top-left corner (i.e., grid[0][0]). The robot tries to move to the bottom-right corner (i.e., grid[m - 1][n - 1]). The robot can only move either down or right at any point in time.</p> <p>An obstacle and space are marked as 1 or 0 respectively in grid. A path that the robot takes cannot include any square that is an obstacle.</p> <p>Return the number of possible unique paths that the robot can take to reach the bottom-right corner.</p> <p>The testcases are generated so that the answer will be less than or equal to $2 * 10^9$.</p> <p>Example 1:</p>  <p>Input: obstacleGrid = [[0,0,0],[0,1,0],[0,0,0]]</p> <p>Output: 2</p> <p>Explanation: There is one obstacle in the middle of the 3x3 grid above. There are two ways to reach the bottom-right corner:</p> <ol style="list-style-type: none"> 1. Right -> Right -> Down -> Down 2. Down -> Down -> Right -> Right <p>Constraints:</p> <p>m == obstacleGrid.length</p> <p>n == obstacleGrid[i].length</p> <p>1 <= m, n <= 100</p> <p>obstacleGrid[i][j] is 0 or 1.</p>
120.	<p>Given an array nums with n objects colored red, white, or blue, sort them in-</p>

	<p>place so that objects of the same color are adjacent, with the colors in the order red, white, and blue.</p> <p>We will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively.</p> <p>You must solve this problem without using the library's sort function.</p> <p>Example 1: Input: nums = [2,0,2,1,1,0]Output: [0,0,1,1,2,2]</p> <p>Example 2: Input: nums = [2,0,1]Output: [0,1,2]</p> <p>Constraints: n == nums.length 1 <= n <= 300 nums[i] is either 0, 1, or 2.</p>												
121.	<p>You are given an m x n integer matrix matrix with the following two properties:</p> <p>Each row is sorted in non-decreasing order.</p> <p>The first integer of each row is greater than the last integer of the previous row.</p> <p>Given an integer target, return true if target is in matrix or false otherwise.</p> <p>You must write a solution in O(log(m * n)) time complexity.</p> <p>Example 1:</p> <table border="1"><tr><td>1</td><td>3</td><td>5</td><td>7</td></tr><tr><td>10</td><td>11</td><td>16</td><td>20</td></tr><tr><td>23</td><td>30</td><td>34</td><td>60</td></tr></table> <p>Input: matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 3 Output: true</p> <p>Constraints: m == matrix.length n == matrix[i].length 1 <= m, n <= 100 -10^4 <= matrix[i][j], target <= 10^4</p>	1	3	5	7	10	11	16	20	23	30	34	60
1	3	5	7										
10	11	16	20										
23	30	34	60										
122.	<p>Given the head of a linked list and a value x, partition it such that all nodes less than x come before nodes greater than or equal to x.</p> <p>You should preserve the original relative order of the nodes in each of the two partitions.</p> <p>Example 1:</p>												

	 <p>Input: head = [1,4,3,2,5,2], x = 3 Output: [1,2,2,4,3,5]</p> <p>Constraints: The number of nodes in the list is in the range [0, 200]. -100 ≤ Node.val ≤ 100 -200 ≤ x ≤ 200</p>
123.	<p>Given an integer array nums that may contain duplicates, return all possible subsets (the power set). The solution set must not contain duplicate subsets. Return the solution in any order.</p> <p>Example 1: Input: nums = [1,2,2] Output: [[],[1],[1,2],[1,2,2],[2],[2,2]]</p> <p>Example 2: Input: nums = [0] Output: [[],[0]]</p> <p>Constraints: 1 ≤ nums.length ≤ 10 -10 ≤ nums[i] ≤ 10</p>
124.	<p>Given the head of a singly linked list and two integers left and right where left ≤ right, reverse the nodes of the list from position left to position right, and return the reversed list.</p> <p>Example 1:</p>  <p>Input: head = [1,2,3,4,5], left = 2, right = 4 Output: [1,4,3,2,5]</p> <p>Example 2: Input: head = [5], left = 1, right = 1 Output: [5]</p> <p>Constraints: The number of nodes in the list is n.</p>

	$1 \leq n \leq 500$ $-500 \leq \text{Node.val} \leq 500$ $1 \leq \text{left} \leq \text{right} \leq n$
125.	<p>Given a triangle array, return the minimum path sum from top to bottom. For each step, you may move to an adjacent number of the row below. More formally, if you are on index i on the current row, you may move to either index i or index $i + 1$ on the next row.</p> <p>Example 1: Input: triangle = [[2],[3,4],[6,5,7],[4,1,8,3]] Output: 11 Explanation: The triangle looks like:</p> <pre> 2 3 4 6 5 7 4 1 8 3 </pre> <p>The minimum path sum from top to bottom is $2 + 3 + 5 + 1 = 11$</p> <p>Constraints: $1 \leq \text{triangle.length} \leq 200$ $\text{triangle}[0].\text{length} == 1$ $\text{triangle}[i].\text{length} == \text{triangle}[i - 1].\text{length} + 1$ $-10^4 \leq \text{triangle}[i][j] \leq 10^4$</p>
126.	<p>Given an integer array nums where every element appears three times except for one, which appears exactly once. Find the single element and return it.</p> <p>You must implement a solution with a linear runtime complexity and use only constant extra space.</p> <p>Example 1: Input: nums = [2,2,3,2] Output: 3</p> <p>Example 2: Input: nums = [0,1,0,1,0,1,99] Output: 99</p> <p>Constraints: $1 \leq \text{nums.length} \leq 3 * 10^4$ $-231 \leq \text{nums}[i] \leq 231 - 1$ Each element in nums appears exactly three times except for one element which appears once.</p>
127.	<p>Given the head of a linked list, return the list after sorting it in ascending order.</p>

	<p>Example 1:</p>  <p>Input: head = [4,2,1,3] Output: [1,2,3,4]</p> <p>Constraints:</p> <p>The number of nodes in the list is in the range [0, 5 * 10⁴]. -10⁵ ≤ Node.val ≤ 10⁵</p>
128.	<p>Given a string <i>s</i> and a dictionary of strings <i>wordDict</i>, return true if <i>s</i> can be segmented into a space-separated sequence of one or more dictionary words.</p> <p>Note that the same word in the dictionary may be reused multiple times in the segmentation.</p> <p>Example 1: Input: <i>s</i> = "applepenapple", <i>wordDict</i> = ["apple", "pen"] Output: true Explanation: Return true because "applepenapple" can be segmented as "apple pen apple".</p> <p>Example 2: Input: <i>s</i> = "catsanddog", <i>wordDict</i> = ["cats", "dog", "sand", "and", "cat"] Output: false</p> <p>Constraints: 1 ≤ <i>s</i>.length ≤ 300 1 ≤ <i>wordDict</i>.length ≤ 1000 1 ≤ <i>wordDict</i>[<i>i</i>].length ≤ 20 <i>s</i> and <i>wordDict</i>[<i>i</i>] consist of only lowercase English letters. All the strings of <i>wordDict</i> are unique.</p>
129.	<p>Given a string <i>s</i>, partition <i>s</i> such that every substring of the partition is a palindrome. Return all possible palindrome partitioning of <i>s</i>.</p> <p>Example 1: Input: <i>s</i> = "aab"</p>

	<p>Output: <code>[["a","a","b"],["aa","b"]]</code></p> <p>Example 2: Input: <code>s = "a"</code> Output: <code>[["a"]]</code></p> <p>Constraints: $1 \leq s.length \leq 16$ <code>s</code> contains only lowercase English letters.</p>
130.	<p>Given an input string <code>s</code>, reverse the order of the words. A word is defined as a sequence of non-space characters. The words in <code>s</code> will be separated by at least one space. Return a string of the words in reverse order concatenated by a single space. Note that <code>s</code> may contain leading or trailing spaces or multiple spaces between two words. The returned string should only have a single space separating the words. Do not include any extra spaces.</p> <p>Example 1: Input: <code>s = "the sky is blue"</code> Output: <code>"blue is sky the"</code></p> <p>Example 2: Input: <code>s = " hello world "</code> Output: <code>"world hello"</code></p> <p>Constraints: $1 \leq s.length \leq 104$ <code>s</code> contains English letters (upper-case and lower-case), digits, and spaces ' '. There is at least one word in <code>s</code>.</p>
131.	<p>Given a 1-indexed array of integers numbers that is already sorted in non-decreasing order, find two numbers such that they add up to a specific target number. Let these two numbers be <code>numbers[index1]</code> and <code>numbers[index2]</code> where $1 \leq index1 < index2 < numbers.length$. Return the indices of the two numbers, <code>index1</code> and <code>index2</code>, added by one as an integer array <code>[index1, index2]</code> of length 2. The tests are generated such that there is exactly one solution. You may not use the same element twice. Your solution must use only constant extra space.</p> <p>Example 1: Input: <code>numbers = [2,7,11,15]</code>, <code>target = 9</code> Output: <code>[1,2]</code> Explanation: The sum of 2 and 7 is 9. Therefore, <code>index1 = 1</code>, <code>index2 = 2</code>. We return <code>[1, 2]</code>.</p>

	<p>Constraints:</p> <p>$2 \leq \text{numbers.length} \leq 3 * 10^4$</p> <p>$-1000 \leq \text{numbers}[i] \leq 1000$</p> <p>numbers is sorted in non-decreasing order.</p> <p>$-1000 \leq \text{target} \leq 1000$</p> <p>The tests are generated such that there is exactly one solution.</p>
132.	<p>Given an integer array nums, rotate the array to the right by k steps, where k is non-negative.</p> <p>Example 1: Input: nums = [1,2,3,4,5,6,7], k = 3 Output: [5,6,7,1,2,3,4] Explanation: rotate 1 steps to the right: [7,1,2,3,4,5,6] rotate 2 steps to the right: [6,7,1,2,3,4,5] rotate 3 steps to the right: [5,6,7,1,2,3,4]</p> <p>Constraints:</p> <p>$1 \leq \text{nums.length} \leq 10^5$</p> <p>$-231 \leq \text{nums}[i] \leq 231 - 1$</p> <p>$0 \leq k \leq 10^5$</p>
133.	<p>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.</p> <p>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.</p> <p>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</p> <p>Constraints:</p> <p>$m == \text{grid.length}$</p> <p>$n == \text{grid}[i].\text{length}$</p> <p>$1 \leq m, n \leq 300$</p> <p>grid[i][j] is '0' or '1'.</p>
134.	<p>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</p>

	<p>systems connected and it will automatically contact the police if two adjacent houses were broken into on the same night.</p> <p>Given an integer array <code>nums</code> representing the amount of money of each house, return the maximum amount of money you can rob tonight without alerting the police.</p> <p>Example 1: Input: <code>nums = [1,2,3,1]</code> Output: 4 Explanation: Rob house 1 (money = 1) and then rob house 3 (money = 3). Total amount you can rob = 1 + 3 = 4.</p> <p>Example 2: Input: <code>nums = [2,7,9,3,1]</code> Output: 12 Explanation: Rob house 1 (money = 2), rob house 3 (money = 9) and rob house 5 (money = 1). Total amount you can rob = 2 + 9 + 1 = 12.</p> <p>Constraints: $1 \leq \text{nums.length} \leq 100$ $0 \leq \text{nums}[i] \leq 400$</p>
135.	<p>Given an integer <code>n</code>, return the number of prime numbers that are strictly less than <code>n</code>.</p> <p>Example 1: Input: <code>n = 10</code> Output: 4 Explanation: There are 4 prime numbers less than 10, they are 2, 3, 5, 7.</p> <p>Constraints: $0 \leq n \leq 5 * 10^6$</p>
136.	<p>There are a total of <code>numCourses</code> courses you have to take, labeled from 0 to <code>numCourses - 1</code>. You are given an array <code>prerequisites</code> where <code>prerequisites[i] = [ai, bi]</code> indicates that you must take course <code>bi</code> first if you want to take course <code>ai</code>.</p> <p>For example, the pair <code>[0, 1]</code>, indicates that to take course 0 you have to first take course 1.</p> <p>Return true if you can finish all courses. Otherwise, return false.</p> <p>Example 1: Input: <code>numCourses = 2, prerequisites = [[1,0]]</code> 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.</p> <p>Example 2: Input: <code>numCourses = 2, prerequisites = [[1,0],[0,1]]</code> Output: false</p>

	<p>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.</p> <p>Constraints: $1 \leq \text{numCourses} \leq 2000$ $0 \leq \text{prerequisites.length} \leq 5000$ $\text{prerequisites}[i].\text{length} == 2$ $0 \leq a_i, b_i < \text{numCourses}$ All the pairs $\text{prerequisites}[i]$ are unique.</p>
137.	<p>There are a total of numCourses courses you have to take, labeled from 0 to $\text{numCourses} - 1$. You are given an array prerequisites where $\text{prerequisites}[i] = [a_i, b_i]$ indicates that you must take course b_i first if you want to take course a_i. For example, the pair $[0, 1]$, indicates that to take course 0 you have to first take course 1.</p> <p>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.</p> <p>Example 1: Input: $\text{numCourses} = 2$, $\text{prerequisites} = [[1,0]]$ 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 $[0,1]$.</p> <p>Example 2: Input: $\text{numCourses} = 4$, $\text{prerequisites} = [[1,0],[2,0],[3,1],[3,2]]$ 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 $[0,2,1,3]$.</p> <p>Constraints: $1 \leq \text{numCourses} \leq 2000$ $0 \leq \text{prerequisites.length} \leq \text{numCourses} * (\text{numCourses} - 1)$ $\text{prerequisites}[i].\text{length} == 2$ $0 \leq a_i, b_i < \text{numCourses}$ $a_i \neq b_i$ All the pairs $[a_i, b_i]$ are distinct.</p>
138.	<p>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</p>

	<p>on the same night.</p> <p>Given an integer array <code>nums</code> representing the amount of money of each house, return the maximum amount of money you can rob tonight without alerting the police.</p> <p>Example 1: Input: <code>nums = [2,3,2]</code> Output: 3 Explanation: You cannot rob house 1 (money = 2) and then rob house 3 (money = 2), because they are adjacent houses.</p> <p>Example 2: Input: <code>nums = [1,2,3,1]</code> Output: 4 Explanation: Rob house 1 (money = 1) and then rob house 3 (money = 3). Total amount you can rob = 1 + 3 = 4.</p> <p>Constraints: $1 \leq \text{nums.length} \leq 100$ $0 \leq \text{nums}[i] \leq 1000$</p>
139.	<p>Find all valid combinations of <code>k</code> numbers that sum up to <code>n</code> such that the following conditions are true:</p> <ul style="list-style-type: none"> Only numbers 1 through 9 are used. Each number is used at most once. <p>Return a list of all possible valid combinations. The list must not contain the same combination twice, and the combinations may be returned in any order.</p> <p>Example 1: Input: <code>k = 3, n = 7</code> Output: <code>[[1,2,4]]</code> Explanation: $1 + 2 + 4 = 7$ There are no other valid combinations.</p> <p>Example 2: Input: <code>k = 3, n = 9</code> Output: <code>[[1,2,6],[1,3,5],[2,3,4]]</code> Explanation: $1 + 2 + 6 = 9$ $1 + 3 + 5 = 9$ $2 + 3 + 4 = 9$ There are no other valid combinations.</p> <p>Constraints: $2 \leq k \leq 9$ $1 \leq n \leq 60$</p>

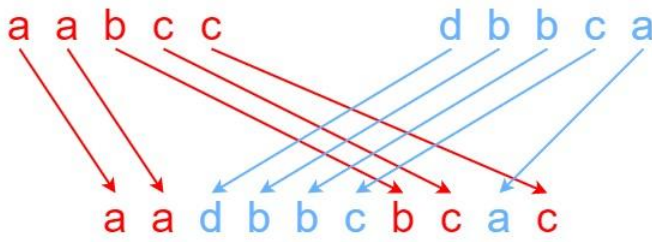
140.	<p>Given an m x n binary matrix filled with 0's and 1's, find the largest square containing only 1's and return its area.</p> <p>Example 1:</p> <table><tr><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr></table> <p>Input: matrix = [[["1","0","1","0","0"],["1","0","1","1","1"],["1","1","1","1","1"],["1","0","0","1","0"]]]</p> <p>Output: 4</p> <p>Example 2:</p> <table><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table> <p>Input: matrix = [[["0","1"],["1","0"]]]</p> <p>Output: 1</p> <p>Constraints:</p> <p>m == matrix.length n == matrix[i].length 1 <= m, n <= 300 matrix[i][j] is '0' or '1'.</p>	1	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	0	0	1	0	0	1	1	0
1	0	1	0	0																					
1	0	1	1	1																					
1	1	1	1	1																					
1	0	0	1	0																					
0	1																								
1	0																								
141.	<p>An ugly number is a positive integer whose prime factors are limited to 2, 3, and 5.</p> <p>Given an integer n, return the nth ugly number.</p> <p>Example 1:</p> <p>Input: n = 10</p> <p>Output: 12</p> <p>Explanation: [1, 2, 3, 4, 5, 6, 8, 9, 10, 12] is the sequence of the first 10 ugly numbers.</p> <p>Constraints:</p> <p>1 <= n <= 1690</p>																								
142.	<p>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].</p> <p>The product of any prefix or suffix of nums is guaranteed to fit in a 32-bit integer.</p>																								

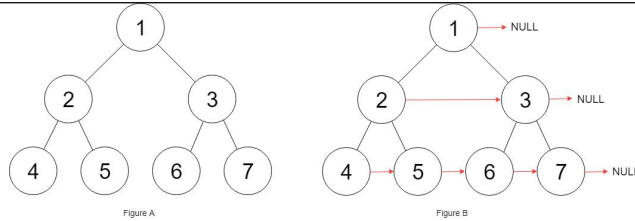
	<p>You must write an algorithm that runs in $O(n)$ time and without using the division operation.</p> <p>Example 1: Input: nums = [1,2,3,4] Output: [24,12,8,6]</p> <p>Example 2: Input: nums = [-1,1,0,-3,3] Output: [0,0,9,0,0]</p> <p>Constraints: $2 \leq \text{nums.length} \leq 10^5$ $-30 \leq \text{nums}[i] \leq 30$ The product of any prefix or suffix of nums is guaranteed to fit in a 32-bit integer.</p>
143.	<p>Given an integer array of size n, find all elements that appear more than $n/3$ times.</p> <p>Example 1: Input: nums = [3,2,3] Output: [3]</p> <p>Example 2: Input: nums = [1] Output: [1]</p> <p>Example 3: Input: nums = [1,2] Output: [1,2]</p> <p>Constraints: $1 \leq \text{nums.length} \leq 5 * 10^4$ $-10^9 \leq \text{nums}[i] \leq 10^9$</p>
144.	<p>Given a string s which represents an expression, evaluate this expression and return its value. The integer division should truncate toward zero. You may assume that the given expression is always valid. All intermediate results will be in the range of $[-231, 231 - 1]$. Note: You are not allowed to use any built-in function which evaluates strings as mathematical expressions, such as eval().</p> <p>Example 1: Input: s = "3+2*2" Output: 7</p> <p>Example 2: Input: s = " 3/2 " Output: 1</p>

	<p>Example 3: Input: s = " 3+5 / 2 " Output: 5 Constraints: $1 \leq s.length \leq 3 * 10^5$ 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, $2^{31} - 1$]. The answer is guaranteed to fit in a 32-bit integer.</p>
145.	<p>Given an integer array nums, find a subarray 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.</p> <p>Example 1: Input: nums = [2,3,-2,4] Output: 6 Explanation: [2,3] has the largest product 6.</p> <p>Example 2: Input: nums = [-2,0,-1] Output: 0 Explanation: The result cannot be 2, because [-2,-1] is not a subarray.</p> <p>Constraints: $1 \leq nums.length \leq 2 * 10^4$ $-10 \leq nums[i] \leq 10$ The product of any prefix or suffix of nums is guaranteed to fit in a 32-bit integer.</p>
146.	<p>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.</p> <p>Example 1: Input: nums = [100,4,200,1,3,2] Output: 4 Explanation: The longest consecutive elements sequence is [1, 2, 3, 4]. Therefore its length is 4.</p> <p>Example 2: Input: nums = [0,3,7,2,5,8,4,6,0,1] Output: 9</p> <p>Constraints: $0 \leq nums.length \leq 10^5$</p>

	-10 ⁹ <= nums[i] <= 10 ⁹
147.	<p>Given an integer n, return the least number of perfect square numbers that sum to n.</p> <p>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.</p> <p>Example 1: Input: n = 12 Output: 3 Explanation: 12 = 4 + 4 + 4.</p> <p>Example 2: Input: n = 13 Output: 2 Explanation: 13 = 4 + 9.</p> <p>Constraints: 1 <= n <= 104</p>
148.	<p>An additive number is a string whose digits can form an additive sequence. A valid additive sequence should contain at least three numbers. Except for the first two numbers, each subsequent number in the sequence must be the sum of the preceding two.</p> <p>Given a string containing only digits, return true if it is an additive number or false otherwise.</p> <p>Note: Numbers in the additive sequence cannot have leading zeros, so sequence 1, 2, 03 or 1, 02, 3 is invalid.</p> <p>Example 1: Input: "112358" Output: true Explanation: The digits can form an additive sequence: 1, 1, 2, 3, 5, 8. 1 + 1 = 2, 1 + 2 = 3, 2 + 3 = 5, 3 + 5 = 8</p> <p>Example 2: Input: "199100199" Output: true Explanation: The additive sequence is: 1, 99, 100, 199. 1 + 99 = 100, 99 + 100 = 199</p> <p>Constraints: 1 <= num.length <= 35 num consists only of digits.</p>
149.	A super ugly number is a positive integer whose prime factors are in the

	<p>array primes. Given an integer n and an array of integers primes, return the nth super ugly number. The nth super ugly number is guaranteed to fit in a 32-bit signed integer.</p> <p>Example 1: Input: n = 12, primes = [2,7,13,19] Output: 32 Explanation: [1,2,4,7,8,13,14,16,19,26,28,32] is the sequence of the first 12 super ugly numbers given primes = [2,7,13,19].</p> <p>Constraints: $1 \leq n \leq 10^5$ $1 \leq \text{primes.length} \leq 100$ $2 \leq \text{primes}[i] \leq 1000$ primes[i] is guaranteed to be a prime number. All the values of primes are unique and sorted in ascending order.</p>
150.	<p>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.</p> <p>Example 1: Input: coins = [1,2,5], amount = 11 Output: 3 Explanation: $11 = 5 + 5 + 1$</p> <p>Example 2: Input: coins = [2], amount = 3 Output: -1</p> <p>Constraints: $1 \leq \text{coins.length} \leq 12$ $1 \leq \text{coins}[i] \leq 231 - 1$ $0 \leq \text{amount} \leq 10^4$</p>
151.	<p>Given an integer array nums, find the Subarray with the largest sum, and return its sum.</p> <p>Example 1: Input: nums = [-2,1,-3,4,-1,2,1,-5,4] Output: 6 Explanation: The subarray [4,-1,2,1] has the largest sum 6.</p> <p>Example 2:</p>

	<p>Input: nums = [5,4,-1,7,8] Output: 23 Explanation: The subarray [5,4,-1,7,8] has the largest sum 23.</p> <p>Constraints: $1 \leq \text{nums.length} \leq 10^5$ $-10^4 \leq \text{nums}[i] \leq 10^4$</p>
152.	<p>Given strings s1, s2, and s3, find whether s3 is formed by an interleaving of s1 and s2. An interleaving of two strings s and t is a configuration where s and t are divided into n and m Substrings respectively, such that: $s = s_1 + s_2 + \dots + s_n$ $t = t_1 + t_2 + \dots + t_m$ $n - m \leq 1$ The interleaving is $s_1 + t_1 + s_2 + t_2 + s_3 + t_3 + \dots$ or $t_1 + s_1 + t_2 + s_2 + t_3 + s_3 + \dots$ Note: $a + b$ is the concatenation of strings a and b.</p> <p>Example 1:</p>  <p>Input: s1 = "aabcc", s2 = "dbbca", s3 = "aadbcbcbcac" Output: true Explanation: One way to obtain s3 is: Split s1 into s1 = "aa" + "bc" + "c", and s2 into s2 = "dbbc" + "a". Interleaving the two splits, we get "aa" + "dbbc" + "bc" + "a" + "c" = "aadbcbcbcac". Since s3 can be obtained by interleaving s1 and s2, we return true.</p>
153.	<p>You are given a perfect binary tree where all leaves are on the same level, and every parent has two children. The binary tree has the following definition:</p> <pre> struct Node { int val; Node *left; Node *right; Node *next; } </pre> <p>Populate each next pointer to point to its next right node. If there is no next right node, the next pointer should be set to NULL. Initially, all next pointers are set to NULL.</p> <p>Example 1:</p>



Input: root = [1,2,3,4,5,6,7]

Output: [1,#,2,3,#,4,5,6,7,#]

Explanation: Given the above perfect binary tree (Figure A), your function should populate each next pointer to point to its next right node, just like in Figure B. The serialized output is in level order as connected by the next pointers, with '#' signifying the end of each level.

Example 2:

Input: root = []

Output: []

Constraints:

The number of nodes in the tree is in the range $[0, 2^{12} - 1]$.

$-1000 \leq \text{Node.val} \leq 1000$

154. You are given an integer array prices where prices[i] is the price of a given stock on the ith day.
On each day, you may decide to buy and/or sell the stock. You can only hold at most one share of the stock at any time. However, you can buy it then immediately sell it on the same day.
Find and return the maximum profit you can achieve.

Example 1:

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

Output: 7

Explanation: Buy on day 2 (price = 1) and sell on day 3 (price = 5), profit = 5-1 = 4.

Then buy on day 4 (price = 3) and sell on day 5 (price = 6), profit = 6-3 = 3.

Total profit is 4 + 3 = 7.

Example 2:

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

Output: 4

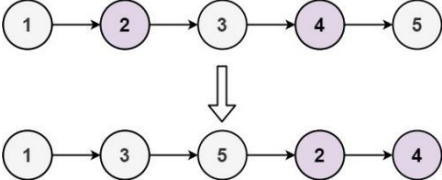
Explanation: Buy on day 1 (price = 1) and sell on day 5 (price = 5), profit = 5-1 = 4.

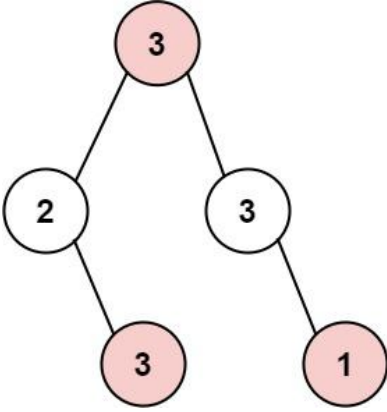
Total profit is 4.

155. There are n gas stations along a circular route, where the amount of gas at the ith station is gas[i].
You have a car with an unlimited gas tank and it costs cost[i] of gas to travel from the ith station to its next (i + 1)th station. You begin the journey with an empty tank at one of the gas stations.
Given two integer arrays gas and cost, return the starting gas station's index

	<p>if you can travel around the circuit once in the clockwise direction, otherwise return -1. If there exists a solution, it is guaranteed to be unique</p> <p>Example 1: Input: gas = [1,2,3,4,5], cost = [3,4,5,1,2] Output: 3 Explanation: Start at station 3 (index 3) and fill up with 4 unit of gas. Your tank = 0 + 4 = 4 Travel to station 4. Your tank = 4 - 1 + 5 = 8 Travel to station 0. Your tank = 8 - 2 + 1 = 7 Travel to station 1. Your tank = 7 - 3 + 2 = 6 Travel to station 2. Your tank = 6 - 4 + 3 = 5 Travel to station 3. The cost is 5. Your gas is just enough to travel back to station 3. Therefore, return 3 as the starting index.</p> <p>Constraints: n == gas.length == cost.length 1 <= n <= 105 0 <= gas[i], cost[i] <= 10⁴</p>
156.	<p>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.</p> <p>Example 1: Input: tokens = ["2","1","+","3","*"] Output: 9 Explanation: ((2 + 1) * 3) = 9</p> <p>Example 2: Input: tokens = ["4","13","5","/","+"] Output: 6 Explanation: (4 + (13 / 5)) = 6</p>
157.	<p>Given an integer n, return the number of trailing zeroes in n!. Note that n! = n * (n - 1) * (n - 2) * ... * 3 * 2 * 1.</p>

	<p>Example 1: Input: n = 3 Output: 0 Explanation: 3! = 6, no trailing zero.</p> <p>Example 2: Input: n = 5 Output: 1 Explanation: 5! = 120, one trailing zero.</p> <p>Constraints: 0 <= n <= 10⁴</p>
158.	<p>You are given an array prices where prices[i] is the price of a given stock on the ith day. Find the maximum profit you can achieve. You may complete as many transactions as you like (i.e., buy one and sell one share of the stock multiple times) with the following restrictions: After you sell your stock, you cannot buy stock on the next day (i.e., cooldown one day). Note: You may not engage in multiple transactions simultaneously (i.e., you must sell the stock before you buy again).</p> <p>Example 1: Input: prices = [1,2,3,0,2] Output: 3 Explanation: transactions = [buy, sell, cooldown, buy, sell]</p> <p>Example 2: Input: prices = [1] Output: 0</p> <p>Constraints: 1 <= prices.length <= 5000 0 <= prices[i] <= 1000</p>
159.	<p>Given an integer array nums, reorder it such that nums[0] < nums[1] > nums[2] < nums[3].... You may assume the input array always has a valid answer.</p> <p>Example 1: Input: nums = [1,5,1,1,6,4] Output: [1,6,1,5,1,4] Explanation: [1,4,1,5,1,6] is also accepted.</p> <p>Example 2: Input: nums = [1,3,2,2,3,1] Output: [2,3,1,3,1,2]</p>

	<p>Constraints:</p> <p>$1 \leq \text{nums.length} \leq 5 * 10^4$</p> <p>$0 \leq \text{nums}[i] \leq 5000$</p> <p>It is guaranteed that there will be an answer for the given input nums.</p>
160.	<p>Given the head of a singly linked list, group all the nodes with odd indices together followed by the nodes with even indices, and return the reordered list.</p> <p>The first node is considered odd, and the second node is even, and so on.</p> <p>Note that the relative order inside both the even and odd groups should remain as it was in the input.</p> <p>You must solve the problem in $O(1)$ extra space complexity and $O(n)$ time complexity.</p> <p>Example 1:</p>  <p>Input: head = [1,2,3,4,5]</p> <p>Output: [1,3,5,2,4]</p> <p>Constraints:</p> <p>The number of nodes in the linked list is in the range [0, 104].</p> <p>$-106 \leq \text{Node.val} \leq 10^6$</p>
161.	<p>Given an integer array nums, return true if there exists a triplet of indices (i, j, k) such that $i < j < k$ and $\text{nums}[i] < \text{nums}[j] < \text{nums}[k]$. If no such indices exists, return false.</p> <p>Example 1:</p> <p>Input: nums = [1,2,3,4,5]</p> <p>Output: true</p> <p>Explanation: Any triplet where $i < j < k$ is valid.</p> <p>Example 2:</p> <p>Input: nums = [5,4,3,2,1]</p> <p>Output: false</p> <p>Explanation: No triplet exists.</p> <p>Constraints:</p> <p>$1 \leq \text{nums.length} \leq 5 * 10^5$</p> <p>$-231 \leq \text{nums}[i] \leq 231 - 1$</p>
162.	<p>The thief has found himself a new place for his thievery again. There is only one entrance to this area, called root.</p> <p>Besides the root, each house has one and only one parent house. After a</p>

	<p>tour, the smart thief realized that all houses in this place form a binary tree. It will automatically contact the police if two directly-linked houses were broken into on the same night.</p> <p>Given the root of the binary tree, return the maximum amount of money the thief can rob without alerting the police.</p> <p>Example 1:</p>  <pre> graph TD A((3)) --- B((2)) A --- C((3)) B --- D((3)) C --- E((1)) </pre> <p>Input: root = [3,2,3,null,3,null,1] Output: 7 Explanation: Maximum amount of money the thief can rob = 3 + 3 + 1 = 7.</p> <p>Constraints: The number of nodes in the tree is in the range [1, 10⁴]. 0 ≤ Node.val ≤ 10⁴</p>
163.	<p>Given an integer n, break it into the sum of k positive integers, where k ≥ 2, and maximize the product of those integers. Return the maximum product you can get.</p> <p>Example 1: Input: n = 2 Output: 1 Explanation: 2 = 1 + 1, 1 × 1 = 1.</p> <p>Example 2: Input: n = 10 Output: 36 Explanation: 10 = 3 + 3 + 4, 3 × 3 × 4 = 36.</p> <p>Constraints: 2 ≤ n ≤ 58</p>
164.	<p>Given an integer array nums and an integer k, return the k most frequent elements. You may return the answer in any order.</p> <p>Example 1: Input: nums = [1,1,1,2,2,3], k = 2 Output: [1,2]</p>

	<p>Example 2: Input: nums = [1], k = 1 Output: [1]</p> <p>Constraints: 1 <= nums.length <= 10⁵ -10⁴ <= nums[i] <= 10⁴ k is in the range [1, the number of unique elements in the array]. It is guaranteed that the answer is unique.</p>
165.	<p>Your task is to calculate $ab \bmod 1337$ where a is a positive integer and b is an extremely large positive integer given in the form of an array.</p> <p>Example 1: Input: a = 2, b = [3] Output: 8</p> <p>Example 2: Input: a = 2, b = [1,0] Output: 1024</p> <p>Example 3: Input: a = 1, b = [4,3,3,8,5,2] Output: 1</p> <p>Constraints: 1 <= a <= 231 - 1 1 <= b.length <= 2000 0 <= b[i] <= 9 b does not contain leading zeros.</p>
166.	<p>You are given two integer arrays nums1 and nums2 sorted in non-decreasing order and an integer k.</p> <p>Define a pair (u, v) which consists of one element from the first array and one element from the second array.</p> <p>Return the k pairs (u1, v1), (u2, v2), ..., (uk, vk) with the smallest sums.</p> <p>Example 1: Input: nums1 = [1,7,11], nums2 = [2,4,6], k = 3 Output: [[1,2],[1,4],[1,6]] Explanation: The first 3 pairs are returned from the sequence: [1,2],[1,4],[1,6],[7,2],[7,4],[11,2],[7,6],[11,4],[11,6]</p> <p>Example 2: Input: nums1 = [1,1,2], nums2 = [1,2,3], k = 2 Output: [[1,1],[1,1]] Explanation: The first 2 pairs are returned from the sequence: [1,1],[1,1],[1,2],[2,1],[1,2],[2,2],[1,3],[1,3],[2,3]</p>

	<p>Constraints:</p> <p>$1 \leq \text{nums1.length}, \text{nums2.length} \leq 10^5$</p> <p>$-10^9 \leq \text{nums1}[i], \text{nums2}[i] \leq 10^9$</p> <p>nums1 and nums2 both are sorted in non-decreasing order.</p> <p>$1 \leq k \leq 10^4$</p>
167.	<p>A wiggle sequence is a sequence where the differences between successive numbers strictly alternate between positive and negative. The first difference (if one exists) may be either positive or negative. A sequence with one element and a sequence with two non-equal elements are trivially wiggle sequences.</p> <p>For example, [1, 7, 4, 9, 2, 5] is a wiggle sequence because the differences (6, -3, 5, -7, 3) alternate between positive and negative.</p> <p>In contrast, [1, 4, 7, 2, 5] and [1, 7, 4, 5, 5] are not wiggle sequences. The first is not because its first two differences are positive, and the second is not because its last difference is zero.</p> <p>A subsequence is obtained by deleting some elements (possibly zero) from the original sequence, leaving the remaining elements in their original order.</p> <p>Given an integer array nums, return the length of the longest wiggle subsequence of nums.</p> <p>Example 1:</p> <p>Input: nums = [1,7,4,9,2,5]</p> <p>Output: 6</p> <p>Explanation: The entire sequence is a wiggle sequence with differences (6, -3, 5, -7, 3).</p> <p>Example 2:</p> <p>Input: nums = [1,17,5,10,13,15,10,5,16,8]</p> <p>Output: 7</p> <p>Explanation: There are several subsequences that achieve this length. One is [1, 17, 10, 13, 10, 16, 8] with differences (16, -7, 3, -3, 6, -8).</p> <p>Constraints:</p> <p>$1 \leq \text{nums.length} \leq 1000$</p> <p>$0 \leq \text{nums}[i] \leq 1000$</p>
168.	<p>Given an array of distinct integers nums and a target integer target, return the number of possible combinations that add up to target.</p> <p>The test cases are generated so that the answer can fit in a 32-bit integer.</p> <p>Example 1:</p> <p>Input: nums = [1,2,3], target = 4</p> <p>Output: 7</p> <p>Explanation:</p> <p>The possible combination ways are:</p> <p>(1, 1, 1, 1)</p>

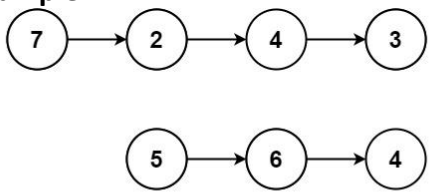
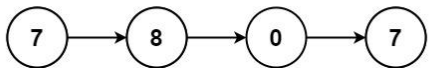
	<p>(1, 1, 2) (1, 2, 1) (1, 3) (2, 1, 1) (2, 2) (3, 1)</p> <p>Note that different sequences are counted as different combinations.</p> <p>Example 2: Input: nums = [9], target = 3 Output: 0</p> <p>Constraints: $1 \leq \text{nums.length} \leq 200$ $1 \leq \text{nums}[i] \leq 1000$ All the elements of nums are unique. $1 \leq \text{target} \leq 1000$</p>
169.	<p>You have a list arr of all integers in the range [1, n] sorted in a strictly increasing order. Apply the following algorithm on arr: Starting from left to right, remove the first number and every other number afterward until you reach the end of the list. Repeat the previous step again, but this time from right to left, remove the rightmost number and every other number from the remaining numbers. Keep repeating the steps again, alternating left to right and right to left, until a single number remains. Given the integer n, return the last number that remains in arr.</p> <p>Example 1: Input: n = 9 Output: 6 Explanation: arr = [1, 2, 3, 4, 5, 6, 7, 8, 9] arr = [2, 4, 6, 8] arr = [2, 6] arr = [6]</p> <p>Example 2: Input: n = 1 Output: 1</p> <p>Constraints: $1 \leq n \leq 10^9$</p>
170.	<p>Given a string s and an integer k, return the length of the longest substring of s such that the frequency of each character in this substring is greater than or equal to k. if no such substring exists, return 0.</p>

	<p>Example 1: Input: s = "aaabb", k = 3 Output: 3 Explanation: The longest substring is "aaa", as 'a' is repeated 3 times.</p> <p>Example 2: Input: s = "ababbc", k = 2 Output: 5 Explanation: The longest substring is "ababb", as 'a' is repeated 2 times and 'b' is repeated 3 times.</p> <p>Constraints: $1 \leq s.length \leq 10^4$ s consists of only lowercase English letters. $1 \leq k \leq 10^5$</p>
171.	<p>Given a positive integer n, you can apply one of the following operations: If n is even, replace n with $n / 2$. If n is odd, replace n with either $n + 1$ or $n - 1$. Return the minimum number of operations needed for n to become 1.</p> <p>Example 1: Input: n = 8 Output: 3 Explanation: $8 \rightarrow 4 \rightarrow 2 \rightarrow 1$</p> <p>Example 2: Input: n = 7 Output: 4 Explanation: $7 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$ or $7 \rightarrow 6 \rightarrow 3 \rightarrow 2 \rightarrow 1$</p> <p>Example 3: Input: n = 4 Output: 2</p> <p>Constraints: $1 \leq n \leq 2^{31} - 1$</p>
172.	<p>Given an integer n, return the nth digit of the infinite integer sequence [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ...].</p> <p>Example 1: Input: n = 3 Output: 3</p> <p>Example 2: Input: n = 11 Output: 0 Explanation: The 11th digit of the sequence 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ... is a 0, which is part of the number 10.</p>

	Constraints: $1 \leq n \leq 2^{31} - 1$
173.	<p>Given string num representing a non-negative integer num, and an integer k, return the smallest possible integer after removing k digits from num.</p> <p>Example 1: Input: num = "1432219", k = 3 Output: "1219" Explanation: Remove the three digits 4, 3, and 2 to form the new number 1219 which is the smallest.</p> <p>Example 2: Input: num = "10200", k = 1 Output: "200" Explanation: Remove the leading 1 and the number is 200. Note that the output must not contain leading zeroes.</p> <p>Constraints: $1 \leq k \leq \text{num.length} \leq 10^5$ num consists of only digits. num does not have any leading zeros except for the zero itself.</p>
174.	<p>An integer array is called arithmetic if it consists of at least three elements and if the difference between any two consecutive elements is the same.</p> <p>For example, [1,3,5,7,9], [7,7,7,7], and [3,-1,-5,-9] are arithmetic sequences.</p> <p>Given an integer array nums, return the number of arithmetic subarrays of nums.</p> <p>A subarray is a contiguous subsequence of the array.</p> <p>Example 1: Input: nums = [1,2,3,4] Output: 3 Explanation: We have 3 arithmetic slices in nums: [1, 2, 3], [2, 3, 4] and [1,2,3,4] itself.</p> <p>Example 2: Input: nums = [1] Output: 0</p> <p>Constraints: $1 \leq \text{nums.length} \leq 5000$ $-1000 \leq \text{nums}[i] \leq 1000$</p>
175.	<p>Given an integer array nums, return true if you can partition the array into two subsets such that the sum of the elements in both subsets is equal or false otherwise.</p>

	<p>Example 1: Input: nums = [1,5,11,5] Output: true Explanation: The array can be partitioned as [1, 5, 5] and [11].</p> <p>Example 2: Input: nums = [1,2,3,5] Output: false Explanation: The array cannot be partitioned into equal sum subsets.</p> <p>Constraints: $1 \leq \text{nums.length} \leq 200$ $1 \leq \text{nums}[i] \leq 100$</p>
176.	<p>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.</p> <p>Example 1: Input: s = "ABAB", k = 2 Output: 4 Explanation: Replace the two 'A's with two 'B's or vice versa.</p> <p>Example 2: Input: s = "AABABBA", k = 1 Output: 4 Explanation: Replace the one 'A' in the middle with 'B' and form "AABBBBA". The substring "BBBB" has the longest repeating letters, which is 4. There may exists other ways to achieve this answer too.</p> <p>Constraints: $1 \leq \text{s.length} \leq 10^5$ s consists of only uppercase English letters. $0 \leq k \leq \text{s.length}$</p>
177.	<p>Given two strings s and p, return an array of all the start indices of p's anagrams in s. You may 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.</p> <p>Example 1: Input: s = "cbaebabacd", p = "abc" Output: [0,6] Explanation: The substring with start index = 0 is "cba", which is an anagram of "abc". The substring with start index = 6 is "bac", which is an anagram of "abc".</p> <p>Example 2:</p>

	<p>Input: s = "abab", p = "ab"</p> <p>Output: [0,1,2]</p> <p>Explanation:</p> <p>The substring with start index = 0 is "ab", which is an anagram of "ab".</p> <p>The substring with start index = 1 is "ba", which is an anagram of "ab".</p> <p>The substring with start index = 2 is "ab", which is an anagram of "ab".</p> <p>Constraints:</p> <p>1 ≤ s.length, p.length ≤ 3 * 10⁴</p> <p>s and p consist of lowercase English letters.</p>
178.	<p>Given an array of characters chars, compress it using the following algorithm:</p> <p>Begin with an empty string s. For each group of consecutive repeating characters in chars:</p> <p>If the group's length is 1, append the character to s.</p> <p>Otherwise, append the character followed by the group's length.</p> <p>The compressed string s should not be returned separately, but instead, be stored in the input character array chars. Note that group lengths that are 10 or longer will be split into multiple characters in chars.</p> <p>After you are done modifying the input array, return the new length of the array.</p> <p>You must write an algorithm that uses only constant extra space.</p> <p>Example 1:</p> <p>Input: chars = ["a","a","b","b","c","c","c"]</p> <p>Output: Return 6, and the first 6 characters of the input array should be: ["a","2","b","2","c","3"]</p> <p>Explanation: The groups are "aa", "bb", and "ccc". This compresses to "a2b2c3".</p> <p>Example 2:</p> <p>Input: chars = ["a"]</p> <p>Output: Return 1, and the first character of the input array should be: ["a"]</p> <p>Explanation: The only group is "a", which remains uncompressed since it's a single character.</p> <p>Constraints:</p> <p>1 ≤ chars.length ≤ 2000</p> <p>chars[i] is a lowercase English letter, uppercase English letter, digit, or symbol.</p>
179.	<p>You are given two non-empty linked lists representing two non-negative integers. The most significant digit comes first and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.</p> <p>You may assume the two numbers do not contain any leading zero, except</p>

	<p>the number 0 itself.</p> <p>Example 1:</p>  <hr/>  <p>Input: l1 = [7,2,4,3], l2 = [5,6,4] Output: [7,8,0,7]</p> <p>Example 2: Input: l1 = [2,4,3], l2 = [5,6,4] Output: [8,0,7]</p> <p>Constraints: The number of nodes in each linked list is in the range [1, 100]. $0 \leq \text{Node.val} \leq 9$ It is guaranteed that the list represents a number that does not have leading zeros.</p>
180.	<p>Given four integer arrays nums1, nums2, nums3, and nums4 all of length n, return the number of tuples (i, j, k, l) such that:</p> $0 \leq i, j, k, l < n$ $\text{nums1}[i] + \text{nums2}[j] + \text{nums3}[k] + \text{nums4}[l] == 0$ <p>Example 1: Input: nums1 = [1,2], nums2 = [-2,-1], nums3 = [-1,2], nums4 = [0,2] Output: 2 Explanation: The two tuples are:</p> <ol style="list-style-type: none"> (0, 0, 0, 1) -> $\text{nums1}[0] + \text{nums2}[0] + \text{nums3}[0] + \text{nums4}[1] = 1 + (-2) + (-1) + 2 = 0$ (1, 1, 0, 0) -> $\text{nums1}[1] + \text{nums2}[1] + \text{nums3}[0] + \text{nums4}[0] = 2 + (-1) + (-1) + 0 = 0$ <p>Example 2: Input: nums1 = [0], nums2 = [0], nums3 = [0], nums4 = [0] Output: 1</p> <p>Constraints: $n == \text{nums1.length}$ $n == \text{nums2.length}$ $n == \text{nums3.length}$ $n == \text{nums4.length}$</p>

	$1 \leq n \leq 200$ $-2^{28} \leq \text{nums1}[i], \text{nums2}[i], \text{nums3}[i], \text{nums4}[i] \leq 2^{28}$
181.	<p>Given a string s, sort it in decreasing order based on the frequency of the characters. The frequency of a character is the number of times it appears in the string.</p> <p>Return the sorted string. If there are multiple answers, return any of them.</p> <p>Example 1: Input: s = "tree" Output: "eert" Explanation: 'e' appears twice while 'r' and 't' both appear once. So 'e' must appear before both 'r' and 't'. Therefore "eetr" is also a valid answer.</p> <p>Example 2: Input: s = "cccaaa" Output: "aaaccc" Explanation: Both 'c' and 'a' appear three times, so both "cccaaa" and "aaaccc" are valid answers. Note that "cacaca" is incorrect, as the same characters must be together.</p> <p>Constraints: $1 \leq \text{s.length} \leq 5 * 10^5$ s consists of uppercase and lowercase English letters and digits.</p>
182.	<p>Given an array of n integers nums, a 132 pattern is a subsequence of three integers nums[i], nums[j] and nums[k] such that $i < j < k$ and $\text{nums}[i] < \text{nums}[k] < \text{nums}[j]$.</p> <p>Return true if there is a 132 pattern in nums, otherwise, return false.</p> <p>Example 1: Input: nums = [1,2,3,4] Output: false Explanation: There is no 132 pattern in the sequence.</p> <p>Example 2: Input: nums = [3,1,4,2] Output: true Explanation: There is a 132 pattern in the sequence: [1, 4, 2].</p> <p>Example 3: Input: nums = [-1,3,2,0] Output: true Explanation: There are three 132 patterns in the sequence: [-1, 3, 2], [-1, 3, 0] and [-1, 2, 0].</p> <p>Constraints: $n == \text{nums.length}$ $1 \leq n \leq 2 * 10^5$ $-10^9 \leq \text{nums}[i] \leq 10^9$</p>

183.	<p>Given a string queryIP, return "IPv4" if IP is a valid IPv4 address, "IPv6" if IP is a valid IPv6 address or "Neither" if IP is not a correct IP of any type.</p> <p>A valid IPv4 address is an IP in the form "x1.x2.x3.x4" where $0 \leq x_i \leq 255$ and x_i cannot contain leading zeros. For example, "192.168.1.1" and "192.168.1.0" are valid IPv4 addresses while "192.168.01.1", "192.168.1.00", and "192.168@1.1" are invalid IPv4 addresses.</p> <p>A valid IPv6 address is an IP in the form "x1:x2:x3:x4:x5:x6:x7:x8" where:</p> <ul style="list-style-type: none"> ● $1 \leq x_i.length \leq 4$ ● x_i is a hexadecimal string which may contain digits, lowercase English letter ('a' to 'f') and upper-case English letters ('A' to 'F'). ● Leading zeros are allowed in x_i. <p>For example, "2001:0db8:85a3:0000:0000:8a2e:0370:7334" and "2001:db8:85a3:0:0:8A2E:0370:7334" are valid IPv6 addresses, while "2001:0db8:85a3::8A2E:037j:7334" and "02001:0db8:85a3:0000:0000:8a2e:0370:7334" are invalid IPv6 addresses.</p> <p>Example 1: Input: queryIP = "172.16.254.1" Output: "IPv4" Explanation: This is a valid IPv4 address, return "IPv4".</p> <p>Example 2: Input: queryIP = "2001:0db8:85a3:0:0:8A2E:0370:7334" Output: "IPv6" Explanation: This is a valid IPv6 address, return "IPv6".</p> <p>Example 3: Input: queryIP = "256.256.256.256" Output: "Neither" Explanation: This is neither a IPv4 address nor a IPv6 address.</p> <p>Constraints: queryIP consists only of English letters, digits and the characters '.' and ':'.</p>
184.	<p>We define the string base to be the infinite wraparound string of "abcdefghijklmnopqrstuvwxyz", so base will look like this: "...zabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcd....".</p> <p>Given a string s, return the number of unique non-empty substrings of s are present in base.</p> <p>Example 1: Input: s = "a" Output: 1 Explanation: Only the substring "a" of s is in base.</p> <p>Example 2:</p>

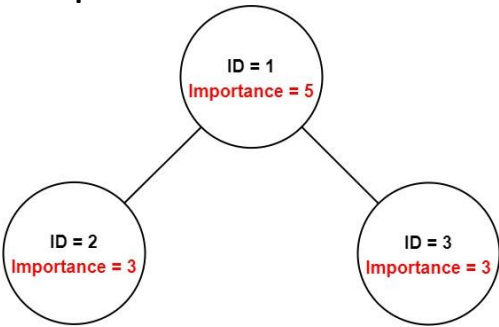
	<p>Input: s = "cac"</p> <p>Output: 2</p> <p>Explanation: There are two substrings ("a", "c") of s in base.</p> <p>Example 3:</p> <p>Input: s = "zab"</p> <p>Output: 6</p> <p>Explanation: There are six substrings ("z", "a", "b", "za", "ab", and "zab") of s in base.</p> <p>Constraints:</p> <p>1 <= s.length <= 10⁵</p> <p>s consists of lowercase English letters.</p>
185.	<p>Winter is coming! During the contest, your first job is to design a standard heater with a fixed warm radius to warm all the houses.</p> <p>Every house can be warmed, as long as the house is within the heater's warm radius range.</p> <p>Given the positions of houses and heaters on a horizontal line, return the minimum radius standard of heaters so that those heaters could cover all houses.</p> <p>Notice that all the heaters follow your radius standard, and the warm radius will the same.</p> <p>Example 1:</p> <p>Input: houses = [1,2,3], heaters = [2]</p> <p>Output: 1</p> <p>Explanation: The only heater was placed in the position 2, and if we use the radius 1 standard, then all the houses can be warmed.</p> <p>Example 2:</p> <p>Input: houses = [1,2,3,4], heaters = [1,4]</p> <p>Output: 1</p> <p>Explanation: The two heaters were placed at positions 1 and 4. We need to use a radius 1 standard, then all the houses can be warmed.</p> <p>Example 3:</p> <p>Input: houses = [1,5], heaters = [2]</p> <p>Output: 3</p> <p>Constraints:</p> <p>1 <= houses.length, heaters.length <= 3 * 10⁴</p> <p>1 <= houses[i], heaters[i] <= 10⁹</p>
186.	<p>A magical string s consists of only '1' and '2' and obeys the following rules:</p> <p>The string s is magical because concatenating the number of contiguous occurrences of characters '1' and '2' generates the string s itself.</p> <p>The first few elements of s is s = "122112122122121122.....". If we group the consecutive 1's and 2's in s, it will be "1 22 11 2 1 22 1 22 11 2 11</p>

	<p>22". and the occurrences of 1's or 2's in each group are "1 2 2 1 1 2 1 2 2 1 2 2". You can see that the occurrence sequence is s itself.</p> <p>Given an integer n, return the number of 1's in the first n number in the magical string s.</p> <p>Example 1: Input: n = 6 Output: 3 Explanation: The first 6 elements of magical string s is "122112" and it contains three 1's, so return 3.</p> <p>Example 2: Input: n = 1 Output: 1</p> <p>Constraints: $1 \leq n \leq 10^5$</p>
187.	<p>You are given an integer array nums and an integer target.</p> <p>You want to build an expression out of nums by adding one of the symbols '+' and '-' before each integer in nums and then concatenate all the integers.</p> <p>For example, if nums = [2, 1], you can add a '+' before 2 and a '-' before 1 and concatenate them to build the expression "+2-1".</p> <p>Return the number of different expressions that you can build, which evaluates to target.</p> <p>Example 1: Input: nums = [1,1,1,1,1], target = 3 Output: 5 Explanation: There are 5 ways to assign symbols to make the sum of nums be target 3.</p> <p>-1 + 1 + 1 + 1 + 1 = 3 +1 - 1 + 1 + 1 + 1 = 3 +1 + 1 - 1 + 1 + 1 = 3 +1 + 1 + 1 - 1 + 1 = 3 +1 + 1 + 1 + 1 - 1 = 3</p> <p>Example 2: Input: nums = [1], target = 1 Output: 1</p> <p>Constraints: $1 \leq \text{nums.length} \leq 20$ $0 \leq \text{nums}[i] \leq 1000$ $0 \leq \text{sum}(\text{nums}[i]) \leq 1000$ $-1000 \leq \text{target} \leq 1000$</p>

188.	<p>Given an integer array <code>nums</code> and an integer <code>k</code>, return <code>true</code> if <code>nums</code> has a good subarray or <code>false</code> otherwise.</p> <p>A good subarray is a subarray where: its length is at least two, and the sum of the elements of the subarray is a multiple of <code>k</code>.</p> <p>Note that: A subarray is a contiguous part of the array. An integer <code>x</code> is a multiple of <code>k</code> if there exists an integer <code>n</code> such that $x = n * k$. 0 is always a multiple of <code>k</code>.</p> <p>Example 1: Input: <code>nums = [23,2,4,6,7]</code>, <code>k = 6</code> Output: <code>true</code> Explanation: <code>[2, 4]</code> is a continuous subarray of size 2 whose elements sum up to 6.</p> <p>Example 2: Input: <code>nums = [23,2,6,4,7]</code>, <code>k = 6</code> Output: <code>true</code> Explanation: <code>[23, 2, 6, 4, 7]</code> is an continuous subarray of size 5 whose elements sum up to 42. 42 is a multiple of 6 because $42 = 7 * 6$ and 7 is an integer.</p> <p>Example 3: Input: <code>nums = [23,2,6,4,7]</code>, <code>k = 13</code> Output: <code>false</code></p> <p>Constraints: $1 \leq \text{nums.length} \leq 10^5$ $0 \leq \text{nums}[i] \leq 10^9$ $0 \leq \text{sum}(\text{nums}[i]) \leq 2^{31} - 1$ $1 \leq k \leq 2^{31} - 1$</p>
189.	<p>Given a binary array <code>nums</code>, return the maximum length of a contiguous subarray with an equal number of 0 and 1.</p> <p>Example 1: Input: <code>nums = [0,1]</code> Output: 2 Explanation: <code>[0, 1]</code> is the longest contiguous subarray with an equal number of 0 and 1.</p> <p>Example 2: Input: <code>nums = [0,1,0]</code> Output: 2 Explanation: <code>[0, 1]</code> (or <code>[1, 0]</code>) is a longest contiguous subarray with equal number of 0 and 1.</p> <p>Constraints: $1 \leq \text{nums.length} \leq 10^5$</p>

	<p>nums[i] is either 0 or 1.</p>
190.	<p>Given an array of integers nums and an integer k, return the number of unique k-diff pairs in the array.</p> <p>A k-diff pair is an integer pair (nums[i], nums[j]), where the following are true:</p> <ul style="list-style-type: none"> ● $0 \leq i, j < \text{nums.length}$ ● $i \neq j$ ● $\text{nums}[i] - \text{nums}[j] == k$ <p>Notice that val denotes the absolute value of val.</p> <p>Example 1: Input: nums = [3,1,4,1,5], k = 2 Output: 2 Explanation: There are two 2-diff pairs in the array, (1, 3) and (3, 5). Although we have two 1s in the input, we should only return the number of unique pairs.</p> <p>Example 2: Input: nums = [1,2,3,4,5], k = 1 Output: 4 Explanation: There are four 1-diff pairs in the array, (1, 2), (2, 3), (3, 4) and (4, 5).</p> <p>Example 3: Input: nums = [1,3,1,5,4], k = 0 Output: 1 Explanation: There is one 0-diff pair in the array, (1, 1).</p> <p>Constraints: $1 \leq \text{nums.length} \leq 10^4$ $-10^7 \leq \text{nums}[i] \leq 10^7$ $0 \leq k \leq 10^7$</p>
191.	<p>A complex number can be represented as a string on the form "real+imaginaryi" where:</p> <ul style="list-style-type: none"> ● real is the real part and is an integer in the range [-100, 100]. ● imaginary is the imaginary part and is an integer in the range [-100, 100]. <p>$i^2 == -1$.</p> <p>Given two complex numbers num1 and num2 as strings, return a string of the complex number that represents their multiplications.</p> <p>Example 1: Input: num1 = "1+1i", num2 = "1+1i" Output: "0+2i" Explanation: $(1 + i) * (1 + i) = 1 + i^2 + 2 * i = 2i$, and you need convert it to the form of 0+2i.</p> <p>Example 2: Input: num1 = "1+-1i", num2 = "1+-1i"</p>

	<p>Output: "0+-2i"</p> <p>Explanation: $(1 - i) * (1 - i) = 1 + i^2 - 2 * i = -2i$, and you need convert it to the form of 0+-2i.</p> <p>Constraints: num1 and num2 are valid complex numbers.</p>
192.	<p>There is only one character 'A' on the screen of a notepad. You can perform one of two operations on this notepad for each step:</p> <p>Copy All: You can copy all the characters present on the screen (a partial copy is not allowed).</p> <p>Paste: You can paste the characters which are copied last time.</p> <p>Given an integer n, return the minimum number of operations to get the character 'A' exactly n times on the screen.</p> <p>Example 1: Input: n = 3 Output: 3 Explanation: Initially, we have one character 'A'. In step 1, we use Copy All operation. In step 2, we use Paste operation to get 'AA'. In step 3, we use Paste operation to get 'AAA'.</p> <p>Example 2: Input: n = 1 Output: 0</p> <p>Constraints: $1 \leq n \leq 1000$</p>
193.	<p>Given a sorted integer array arr, two integers k and x, return the k closest integers to x in the array. The result should also be sorted in ascending order.</p> <p>An integer a is closer to x than an integer b if: $a - x < b - x$, or $a - x == b - x$ and $a < b$</p> <p>Example 1: Input: arr = [1,2,3,4,5], k = 4, x = 3 Output: [1,2,3,4]</p> <p>Example 2: Input: arr = [1,2,3,4,5], k = 4, x = -1 Output: [1,2,3,4]</p> <p>Constraints: $1 \leq k \leq \text{arr.length}$ $1 \leq \text{arr.length} \leq 10^4$ arr is sorted in ascending order. $-10^4 \leq \text{arr}[i], x \leq 10^4$</p>

194.	<p>You have a data structure of employee information, including the employee's unique ID, importance value, and direct subordinates' IDs. You are given an array of employees <code>employees</code> where:</p> <ul style="list-style-type: none"> ● <code>employees[i].id</code> is the ID of the <i>i</i>th employee. ● <code>employees[i].importance</code> is the importance value of the <i>i</i>th employee. ● <code>employees[i].subordinates</code> is a list of the IDs of the direct subordinates of the <i>i</i>th employee. <p>Given an integer <code>id</code> that represents an employee's ID, return the total importance value of this employee and all their direct and indirect subordinates.</p> <p>Example 1:</p>  <pre> graph TD E1((ID = 1 Importance = 5)) --- E2((ID = 2 Importance = 3)) E1 --- E3((ID = 3 Importance = 3)) </pre> <p>Input: <code>employees = [[1,5,[2,3]],[2,3,[]],[3,3,[]]]</code>, <code>id = 1</code> Output: 11 Explanation: Employee 1 has an importance value of 5 and has two direct subordinates: employee 2 and employee 3. They both have an importance value of 3. Thus, the total importance value of employee 1 is $5 + 3 + 3 = 11$.</p> <p>Constraints:</p> <ul style="list-style-type: none"> ● $1 \leq \text{employees.length} \leq 2000$ ● $1 \leq \text{employees}[i].\text{id} \leq 2000$ ● All <code>employees[i].id</code> are unique. ● $-100 \leq \text{employees}[i].\text{importance} \leq 100$ ● One employee has at most one direct leader and may have several subordinates. ● The IDs in <code>employees[i].subordinates</code> are valid IDs.
195.	<p>Given two strings <code>s1</code> and <code>s2</code>, return the lowest ASCII sum of deleted characters to make two strings equal.</p> <p>Example 1: Input: <code>s1 = "sea"</code>, <code>s2 = "eat"</code> Output: 231 Explanation: Deleting "s" from "sea" adds the ASCII value of "s" (115) to the sum. Deleting "t" from "eat" adds 116 to the sum.</p>

	<p>At the end, both strings are equal, and $115 + 116 = 231$ is the minimum sum possible to achieve this.</p> <p>Example 2: Input: $s1 = \text{"delete"}, s2 = \text{"leet"}$ Output: 403 Explanation: Deleting "dee" from "delete" to turn the string into "let", adds $100[d] + 101[e] + 101[e]$ to the sum. Deleting "e" from "leet" adds $101[e]$ to the sum. At the end, both strings are equal to "let", and the answer is $100+101+101+101 = 403$. If instead we turned both strings into "lee" or "eet", we would get answers of 433 or 417, which are higher.</p> <p>Constraints: $1 \leq s1.length, s2.length \leq 1000$ $s1$ and $s2$ consist of lowercase English letters.</p>
196.	<p>Given an array of integers $nums$ and an integer k, return the number of contiguous subarrays where the product of all the elements in the subarray is strictly less than k.</p> <p>Example 1: Input: $nums = [10,5,2,6], k = 100$ Output: 8 Explanation: The 8 subarrays that have product less than 100 are: $[10], [5], [2], [6], [10, 5], [5, 2], [2, 6], [5, 2, 6]$ Note that $[10, 5, 2]$ is not included as the product of 100 is not strictly less than k.</p> <p>Example 2: Input: $nums = [1,2,3], k = 0$ Output: 0</p> <p>Constraints: $1 \leq nums.length \leq 3 * 10^4$ $1 \leq nums[i] \leq 1000$ $0 \leq k \leq 10^6$</p>
197.	<p>We are given an array $asteroids$ of integers representing asteroids in a row. For each asteroid, the absolute value represents its size, and the sign represents its direction (positive meaning right, negative meaning left). Each asteroid moves at the same speed.</p> <p>Find out the state of the asteroids after all collisions. If two asteroids meet, the smaller one will explode. If both are the same size, both will explode. Two asteroids moving in the same direction will never meet.</p> <p>Example 1: Input: $asteroids = [5,10,-5]$</p>

	<p>Output: [5,10] Explanation: The 10 and -5 collide resulting in 10. The 5 and 10 never collide. Example 2: Input: asteroids = [8,-8] Output: [] Explanation: The 8 and -8 collide exploding each other. Example 3: Input: asteroids = [10,2,-5] Output: [10] Explanation: The 2 and -5 collide resulting in -5. The 10 and -5 collide resulting in 10.</p> <p>Constraints: $2 \leq \text{asteroids.length} \leq 10^4$ $-1000 \leq \text{asteroids}[i] \leq 1000$ $\text{asteroids}[i] \neq 0$</p>
198.	<p>You are standing at position 0 on an infinite number line. There is a destination at position target. You can make some number of moves numMoves so that: On each move, you can either go left or right. During the ith move (starting from $i == 1$ to $i == \text{numMoves}$), you take i steps in the chosen direction. Given the integer target, return the minimum number of moves required (i.e., the minimum numMoves) to reach the destination.</p> <p>Example 1: Input: target = 2 Output: 3 Explanation: On the 1st move, we step from 0 to 1 (1 step). On the 2nd move, we step from 1 to -1 (2 steps). On the 3rd move, we step from -1 to 2 (3 steps). Example 2: Input: target = 3 Output: 2 Explanation: On the 1st move, we step from 0 to 1 (1 step). On the 2nd move, we step from 1 to 3 (2 steps).</p> <p>Constraints: $-10^9 \leq \text{target} \leq 10^9$ $\text{target} \neq 0$</p>
199.	<p>Given a string s, rearrange the characters of s so that any two adjacent characters are not the same. Return any possible rearrangement of s or return "" if not possible.</p>

	<p>Example 1: Input: s = "aab" Output: "aba"</p> <p>Example 2: Input: s = "aaab" Output: ""</p> <p>Constraints: 1 <= s.length <= 500 s consists of lowercase English letters.</p>
200.	<p>Given a string s, you can transform every letter individually to be lowercase or uppercase to create another string. Return a list of all possible strings we could create. Return the output in any order.</p> <p>Example 1: Input: s = "a1b2" Output: ["a1b2","a1B2","A1b2","A1B2"]</p> <p>Example 2: Input: s = "3z4" Output: ["3z4","3Z4"]</p> <p>Constraints: 1 <= s.length <= 12 s consists of lowercase English letters, uppercase English letters, and digits.</p>