1.
The string "PAYPALISHIRING" is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)

P A H N
A P L S I I G
Y I R
And then read line by line: "PAHNAPLSIIGYIR"

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

string convert(string s, int numRows);

Example 1:

Input: s = "PAYPALISHIRING", numRows = 3
Output: "PAHNAPLSIIGYIR"
Example 2:

Input: s = "PAYPALISHIRING", numRows = 4
Output: "PINALSIGYAHRPI"
Explanation:
P I N
A L S I G
Y A H R
P I
Example 3:

Input: s = "A", numRows = 1
Output: "A"

Constraints:

1 <= s.length <= 1000
s consists of English letters (lower-case and upper-case), ',' and '.'.
1 <= numRows <= 1000

—----------

2.
Palindrome Number

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

Example 1:

Input: x = 121
Output: true
Explanation: 121 reads as 121 from left to right and from right to left.
Example 2:

Input: x = -121
Output: false
Explanation: From left to right, it reads -121. From right to left, it becomes 121-. Therefore it is not a palindrome.
Example 3:

Input: x = 10
Output: false
Explanation: Reads 01 from right to left. Therefore it is not a palindrome.
—----------

3.
Valid Parentheses

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.
Every close bracket has a corresponding open bracket of the same type.

Example 1:

Input: s = "()"
Output: true
Example 2:

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

Output: true
Example 3:

Input: s = "(]"
Output: false

Constraints:

1 <= s.length <= 104
s consists of parentheses only '()[]{}'.
—----------

4.

Merge Two Sorted Lists
You are given the heads of two sorted linked lists list1 and list2.

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

Return the head of the merged linked list.

Example 1:

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

Input: list1 = [], list2 = []
Output: []
Example 3:

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

Constraints:

The number of nodes in both lists is in the range [0, 50].
-100 <= Node.val <= 100
Both list1 and list2 are sorted in non-decreasing order.

—----------

5.

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

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

Change the array nums such that the first k elements of nums contain the unique elements in the order they were present in nums initially. The remaining elements of nums are not important as well as the size of nums.
Return k.
Custom Judge:

The judge will test your solution with the following code:

```
int[] nums = [...]; // Input array
int[] expectedNums = [...]; // The expected answer with correct length

int k = removeDuplicates(nums); // Calls your implementation

assert k == expectedNums.length;
for (int i = 0; i < k; i++) {
    assert nums[i] == expectedNums[i];
}
```

If all assertions pass, then your solution will be accepted.

Example 1:

Input: nums = [1,1,2]
Output: 2, nums = [1,2,_]
Explanation: Your function should return k = 2, with the first two elements of nums being 1 and 2 respectively.
It does not matter what you leave beyond the returned k (hence they are underscores).
Example 2:

Input: nums = [0,0,1,1,1,2,2,3,3,4]
Output: 5, nums = [0,1,2,3,4,_,_,_,_,_]
Explanation: Your function should return k = 5, with the first five elements of nums being 0, 1, 2, 3, and 4 respectively.
It does not matter what you leave beyond the returned k (hence they are underscores).

Constraints:

1 <= nums.length <= 3 * 104
-100 <= nums[i] <= 100
nums is sorted in non-decreasing order.

—----------

6.

Search in Rotated Sorted Array
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.

Example 1:

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

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

Input: nums = [1], target = 0
Output: -1

Constraints:

1 <= nums.length <= 5000
-104 <= nums[i] <= 104
All values of nums are unique.
nums is an ascending array that is possibly rotated.
-104 <= target <= 104
—----------

7.

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

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

Example 1:

Input: strs = ["eat","tea","tan","ate","nat","bat"]
Output: [["bat"],["nat","tan"],["ate","eat","tea"]]
Example 2:

Input: strs = [""]
Output: [[""]]
Example 3:

Input: strs = ["a"]
Output: [["a"]]

Constraints:

1 <= strs.length <= 104
0 <= strs[i].length <= 100
strs[i] consists of lowercase English letters.
—----------

8.

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

Example 1:

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

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

Constraints:

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

9.

Climbing Stairs

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

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

Example 1:

Input: n = 2
Output: 2
Explanation: There are two ways to climb to the top.
1. 1 step + 1 step
2. 2 steps
Example 2:

Input: n = 3
Output: 3
Explanation: There are three ways to climb to the top.
1. 1 step + 1 step + 1 step
2. 1 step + 2 steps
3. 2 steps + 1 step

Constraints:

1 <= n <= 45

—----------

10.

Sort Colors
Given an array nums with n objects colored red, white, or blue, sort them in-place so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively.

You must solve this problem without using the library's sort function.

Example 1:

Input: nums = [2,0,2,1,1,0]
Output: [0,0,1,1,2,2]
Example 2:

Input: nums = [2,0,1]
Output: [0,1,2]

Constraints:

n == nums.length
1 <= n <= 300
nums[i] is either 0, 1, or 2.

Follow up: Could you come up with a one-pass algorithm using only constant extra space?

—----------

11.

Pascal's Triangle
Given an integer numRows, return the first numRows of Pascal's triangle.

In Pascal's triangle, each number is the sum of the two numbers directly above it as shown:

Example 1:

Input: numRows = 5
Output: [[1],[1,1],[1,2,1],[1,3,3,1],[1,4,6,4,1]]
Example 2:

Input: numRows = 1
Output: [[1]]

Constraints:

1 <= numRows <= 30

—----------

12.

Best Time to Buy and Sell Stock
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.

Example 1:

Input: prices = [7,1,5,3,6,4]
Output: 5
Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.
Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.
Example 2:

Input: prices = [7,6,4,3,1]
Output: 0
Explanation: In this case, no transactions are done and the max profit = 0.

Constraints:

1 <= prices.length <= 105
0 <= prices[i] <= 104
—----------

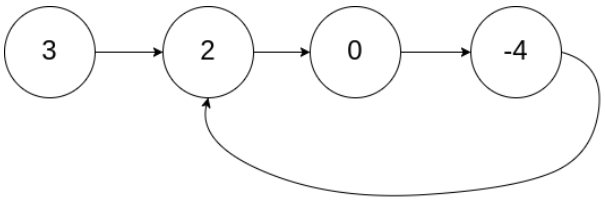
13.

Linked List Cycle
Solved
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.

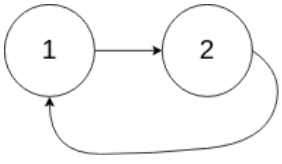
Return true if there is a cycle in the linked list. Otherwise, return false.

Example 1:



Input: head = [3,2,0,-4], pos = 1
Output: true
Explanation: There is a cycle in the linked list, where the tail connects to the 1st node (0-indexed).

Example 2:



Input: head = [1,2], pos = 0
Output: true
Explanation: There is a cycle in the linked list, where the tail connects to the 0th node.
Example 3:

Input: head = [1], pos = -1
Output: false
Explanation: There is no cycle in the linked list.

Constraints:

The number of the nodes in the list is in the range [0, 104].
-105 <= Node.val <= 105
pos is -1 or a valid index in the linked-list.

Follow up: Can you solve it using O(1) (i.e. constant) memory?
—----------

14.

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

Implement the MinStack class:

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

Example 1:

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

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

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

Constraints:

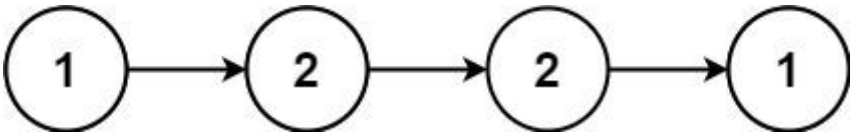
-231 <= val <= 231 - 1
Methods pop, top and getMin operations will always be called on non-empty stacks.
At most 3 * 104 calls will be made to push, pop, top, and getMin.
—----------

15.

Palindrome Linked List

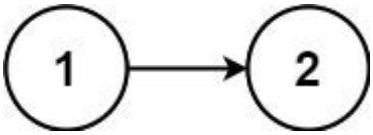
Given the head of a singly linked list, return true if it is a
palindrome
 or false otherwise.

Example 1:



Input: head = [1,2,2,1]
Output: true

Example 2:



Input: head = [1,2]
Output: false

Constraints:

The number of nodes in the list is in the range [1, 105].
0 <= Node.val <= 9

Follow up: Could you do it in O(n) time and O(1) space?

—----------

16.

Valid Anagram
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.

Example 1:

Input: s = "anagram", t = "nagaram"
Output: true
Example 2:

Input: s = "rat", t = "car"
Output: false

Constraints:

1 <= s.length, t.length <= 5 * 104
s and t consist of lowercase English letters.

Follow up: What if the inputs contain Unicode characters? How would you adapt your solution to such a case?

—----------

17.

Sum of Two Integers
Given two integers a and b, return the sum of the two integers without using the operators + and -.

Example 1:

Input: a = 1, b = 2
Output: 3
Example 2:

Input: a = 2, b = 3
Output: 5

Constraints:

-1000 <= a, b <= 1000

—----------

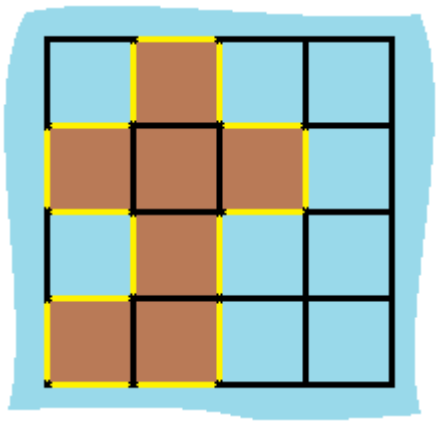
18.

Island Perimeter
You are given row x col grid representing a map where grid[i][j] = 1 represents land and grid[i][j] = 0 represents water.

Grid cells are connected horizontally/vertically (not diagonally). The grid is completely surrounded by water, and there is exactly one island (i.e., one or more connected land cells).

The island doesn't have "lakes", meaning the water inside isn't connected to the water around the island. One cell is a square with side length 1. The grid is rectangular, width and height don't exceed 100. Determine the perimeter of the island.

Example 1:



Input: grid = [[0,1,0,0],[1,1,1,0],[0,1,0,0],[1,1,0,0]]
Output: 16
Explanation: The perimeter is the 16 yellow stripes in the image above.
Example 2:

Input: grid = [[1]]
Output: 4
Example 3:

Input: grid = [[1,0]]
Output: 4

Constraints:

row == grid.length
col == grid[i].length
1 <= row, col <= 100
grid[i][j] is 0 or 1.
There is exactly one island in grid.
—----------

19.

Daily Temperatures
Given an array of integers temperatures represents the daily temperatures, return an array answer such that answer[i] is the number of days you have to wait after the ith day to get a warmer temperature. If there is no future day for which this is possible, keep answer[i] == 0 instead.

Example 1:

Input: temperatures = [73,74,75,71,69,72,76,73]
Output: [1,1,4,2,1,1,0,0]
Example 2:

Input: temperatures = [30,40,50,60]
Output: [1,1,1,0]
Example 3:

Input: temperatures = [30,60,90]
Output: [1,1,0]

Constraints:

1 <= temperatures.length <= 105
30 <= temperatures[i] <= 100

—----------

20.

Boats to Save People

You are given an array people where people[i] is the weight of the ith person, and an infinite number of boats where each boat can carry a maximum weight of limit. Each boat carries at most two people at the same time, provided the sum of the weight of those people is at most limit.

Return the minimum number of boats to carry every given person.

Example 1:

Input: people = [1,2], limit = 3
Output: 1
Explanation: 1 boat (1, 2)
Example 2:

Input: people = [3,2,2,1], limit = 3
Output: 3
Explanation: 3 boats (1, 2), (2) and (3)
Example 3:

Input: people = [3,5,3,4], limit = 5
Output: 4
Explanation: 4 boats (3), (3), (4), (5)

Constraints:

1 <= people.length <= 5 * 104
1 <= people[i] <= limit <= 3 * 104

—----------

21.

Find the Town Judge

In a town, there are n people labeled from 1 to n. There is a rumor that one of these people is secretly the town judge.

If the town judge exists, then:

The town judge trusts nobody.
Everybody (except for the town judge) trusts the town judge.

There is exactly one person that satisfies properties 1 and 2.
You are given an array trust where trust[i] = [ai, bi] representing that the person labeled ai trusts the person labeled bi. If a trust relationship does not exist in trust array, then such a trust relationship does not exist.

Return the label of the town judge if the town judge exists and can be identified, or return -1 otherwise.

Example 1:

Input: n = 2, trust = [[1,2]]
Output: 2
Example 2:

Input: n = 3, trust = [[1,3],[2,3]]
Output: 3
Example 3:

Input: n = 3, trust = [[1,3],[2,3],[3,1]]
Output: -1

Constraints:

1 <= n <= 1000
0 <= trust.length <= 104
trust[i].length == 2
All the pairs of trust are unique.
ai != bi
1 <= ai, bi <= n
—----------

22.

Remove All Adjacent Duplicates In String

You are given a string s consisting of lowercase English letters. A duplicate removal consists of choosing two adjacent and equal letters and removing them.

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

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

Example 1:

Input: s = "abbaca"
Output: "ca"
Explanation:
For example, in "abbaca" we could remove "bb" since the letters are adjacent and equal, and this is the only possible move. The result of this move is that the string is "aaca", of which only "aa" is possible, so the final string is "ca".
Example 2:

Input: s = "azxxzy"
Output: "ay"

Constraints:

1 <= s.length <= 105
s consists of lowercase English letters.

—----------

23.

Maximum Product of Two Elements in an Array

Given the array of integers nums, you will choose two different indices i and j of that array. Return the maximum value of (nums[i]-1)*(nums[j]-1).

Example 1:

Input: nums = [3,4,5,2]
Output: 12
Explanation: If you choose the indices i=1 and j=2 (indexed from 0), you will get the maximum value, that is, (nums[1]-1)*(nums[2]-1) = (4-1)*(5-1) = 3*4 = 12.
Example 2:

Input: nums = [1,5,4,5]
Output: 16
Explanation: Choosing the indices i=1 and j=3 (indexed from 0), you will get the maximum value of (5-1)*(5-1) = 16.
Example 3:

Input: nums = [3,7]
Output: 12

Constraints:

2 <= nums.length <= 500
1 <= nums[i] <= 10^3

—----------

24.

Check if Array Is Sorted and Rotated

Given an array nums, return true if the array was originally sorted in non-decreasing order, then rotated some number of positions (including zero). Otherwise, return false.

There may be duplicates in the original array.

Note: An array A rotated by x positions results in an array B of the same length such that A[i] == B[(i+x) % A.length], where % is the modulo operation.

Example 1:

Input: nums = [3,4,5,1,2]
Output: true
Explanation: [1,2,3,4,5] is the original sorted array.
You can rotate the array by x = 3 positions to begin on the the element of value 3: [3,4,5,1,2].
Example 2:

Input: nums = [2,1,3,4]
Output: false
Explanation: There is no sorted array once rotated that can make nums.
Example 3:

Input: nums = [1,2,3]
Output: true
Explanation: [1,2,3] is the original sorted array.
You can rotate the array by x = 0 positions (i.e. no rotation) to make nums.

Constraints:

1 <= nums.length <= 100
1 <= nums[i] <= 100

—----------

25.

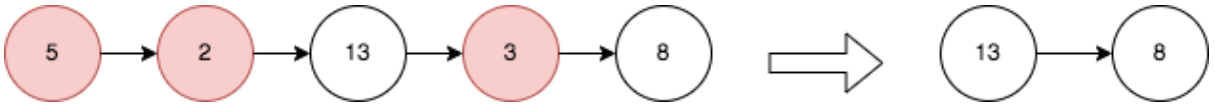
Remove Nodes From Linked List

You are given the head of a linked list.

Remove every node which has a node with a greater value anywhere to the right side of it.

Return the head of the modified linked list.

Example 1:



Input: head = [5,2,13,3,8]
Output: [13,8]
Explanation: The nodes that should be removed are 5, 2 and 3.
- Node 13 is to the right of node 5.
- Node 13 is to the right of node 2.
- Node 8 is to the right of node 3.
Example 2:

Input: head = [1,1,1,1]
Output: [1,1,1,1]
Explanation: Every node has value 1, so no nodes are removed.

Constraints:

The number of the nodes in the given list is in the range [1, 105].
1 <= Node.val <= 105
—----------

26.
Remove All Occurrences of a Substring

Given two strings s and part, perform the following operation on s until all occurrences of the substring part are removed:

Find the leftmost occurrence of the substring part and remove it from s.
Return s after removing all occurrences of part.

A substring is a contiguous sequence of characters in a string.

Example 1:

Input: s = "daabcbaabcbc", part = "abc"
Output: "dab"
Explanation: The following operations are done:
- s = "daabcbaabcbc", remove "abc" starting at index 2, so s = "dabaabcbc".
- s = "dabaabcbc", remove "abc" starting at index 4, so s = "dababc".
- s = "dababc", remove "abc" starting at index 3, so s = "dab".
Now s has no occurrences of "abc".
Example 2:

Input: s = "axxxxyyyyb", part = "xy"
Output: "ab"
Explanation: The following operations are done:
- s = "axxxxyyyyb", remove "xy" starting at index 4 so s = "axxxyyyb".
- s = "axxxyyyb", remove "xy" starting at index 3 so s = "axxyyb".
- s = "axxyyb", remove "xy" starting at index 2 so s = "axyb".
- s = "axyb", remove "xy" starting at index 1 so s = "ab".
Now s has no occurrences of "xy".

Constraints:

1 <= s.length <= 1000
1 <= part.length <= 1000
s and part consists of lowercase English letters.

—----------