Q1.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].

Q2.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

Q3.Given an array of integers nums which is sorted in ascending order, and an integer target, write a function to search target in nums. If the target exists, then return its index. Otherwise, return -1.

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

Example 1:

Input: nums = [-1,0,3,5,9,12], target = 9 Output: 4 Explanation: 9 exists in nums and its index is 4

Q4.Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.

If the target is not found in the array, return [-1, -1].

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

Example 1:

Input: nums = [5,7,7,8,8,10], target = 8 Output: [3,4]
Example 2:

Input: nums = [5,7,7,8,8,10], target = 6 Output: [-1,-1]

Q5.We are playing the Guess Game. The game is as follows:

I pick a number from 1 to n. You have to guess which number I picked.

Every time you guess wrong, I will tell you whether the number I picked is higher or lower than your guess.

You call a pre-defined API int guess(int num), which returns three possible results:

-1: Your guess is higher than the number I picked (i.e. num > pick).1: Your guess is lower than the number I picked (i.e. num < pick).0: your guess is equal to the number I picked (i.e. num == pick).
Return the number that I picked.

Example 1:

Input: n = 10, pick = 6 Output: 6
Example 2:

Input: n = 1, pick = 1 Output: 1
Example 3:

Input: n = 2, pick = 1 Output: 1

Constraints:

1 <= n <= 231 - 11 <= pick <= n

Q6.Given an integer array nums and an integer k, split numbers into k non-empty subarrays such that the largest sum of any subarray is minimized.

Return the minimized largest sum of the split.

A subarray is a contiguous part of the array.

Example 1:

Input: nums = [7,2,5,10,8], k = 2 Output: 18

Q7.Given an integer array nums, move all 0's to the end of it while maintaining the relative order of the non-zero elements.

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

Example 1:

Input: nums = [0,1,0,3,12] Output: [1,3,12,0,0]
Example 2:

Input: nums = [0] Output: [0]

Q8.Given two integer arrays nums1 and nums2, return an array of their intersection. Each element in the result must be unique and you may return the result in any order.

Example 1:

Input: nums1 = [1,2,2,1]

Q9.Given an array of integers nums, calculate the pivot index of this array.

The pivot index is the index where the sum of all the numbers strictly to the left of the index is equal to the sum of all the numbers strictly to the index's right.

If the index is on the left edge of the array, then the left sum is 0 because there are no elements to the left. This also applies to the right edge of the array.

Return the leftmost pivot index. If no such index exists, return -1.

Example 1:

Input: nums = [1,7,3,6,5,6] Output: 3

Q10.Given an integer array nums sorted in non-decreasing order, return an array of the squares of each number sorted in non-decreasing order.

Example 1:

Input: nums = [-4,-1,0,3,10] Output: [0,1,9,16,100] Explanation: After squaring, the array becomes [16,1,0,9,100]. After sorting, it becomes [0,1,9,16,100].

Q11.Given an integer array nums, find the subarray with the largest sum, and return its sum.

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.
Example 2:

Input: nums = [1] Output: 1 Explanation: The subarray [1] has the largest sum 1.
Example 3:

Input: nums = [5,4,-1,7,8] Output: 23 Explanation: The subarray [5,4,-1,7,8] has the largest sum 23.
Option 1

Q12.Given an integer array nums and an integer k, return the kth largest element in the array.

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

Can you solve it without sorting?

Example 1:

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

Q13.You are given an integer array nums. You are initially positioned at the array's first index, and each element in the array represents your maximum jump length at that position.

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

Example 1:

Input: nums = [2,3,1,1,4] Output: true Explanation: Jump 1 step from index 0 to 1, then 3 steps to the last index.

Q14.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.

Example 1:

Input: nums = [2,3,-2,4] Output: 6 Explanation: [2,3] has the largest product 6

Q15.Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.

If the target is not found in the array, return [-1, -1].

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

Q16.Given an array of integers nums and an integer k, return the total number of continuous subarrays whose sum equals to k.

Example:
Input: nums = [1,2,3], k = 3
Output: 2
(Explanation: Subarrays [1,2] and [3] sum to 3)

Q17.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.

Example:
Input: nums = [100, 4, 200, 1, 3, 2]
Output: 4
(Explanation: The longest consecutive sequence is [1, 2, 3, 4])

Q18.You are given an n x n 2D matrix. Rotate the matrix 90 degrees clockwise in-place.

Example:
Input:

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

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

Q19.Problem: Given a string s, return the longest palindromic substring in s.

Example:
Input: "babad" → Output: "bab" or "aba"

Q20.Problem: Given an array of strings, group the anagrams together.

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

Q21.Merge Two Sorted Arrays Problem Statement

Given two sorted integer arrays ARR1 and ARR2 off size M and N, respectively, merge them into ARR1 as one sorted array. Assume that ARR1 has a size of M+ N to hold all elements of ARR2.

Input

The first line contains an integer T representing the number of test cases. Each test case contains:

-Two space-separated integers M and N.

- A line with M integers followed by N zeros in ARR1.

- A line with N integers in ARR2.

Output:

For each test case, return the merged ARRL as one sorted array.

Example:

Input:

ARR1 = [3, 6, 9, 0, 0) ARR2 = [4, 10]

Output:

ARR1 [3, 4, 6, 9, 10]

Merge two sorted arrays into one sorted array efficiently.

Use two pointers to traverse ARR1 and ARR2 from the end to the beginning.

Start filling ARR1 from the last position to avoid overwriting elements.

Example: For ARR1 = [3, 6, 9, 0, 0] and ARR2 = [4, 10], start merging from the end.

Time complexity is O(M + N) where M and N are the sizes of ARR1 and ARR2.

Q22.Form a Triangle Problem Statement

You are given an array of integers ARR with a length of N. Your task is to determine whether it's possible to construct at least one non-degenerate triangle using the values from the array as the sides of the triangle. If possible, return true; otherwise, return false.

Input:

2

3

345

4

1 10 12 30

Output:

YES

NO

Q23.Questions

Maximum Subarray Sum Problem Statement Given an array of integers, determine the maximum possible sum of any contiguous subarray within the array.

Example:

Probl...

Input:

array = [34, -50, 42, 14, -5,86]

Output:

4137

Q24.Given two 2-D arrays which represent intervals. Each 2-D array represents a list of intervals. Each list of intervals is disjoint and sorted in increasing order. Find the intersection or set of ranges that are common to both the lists.

Note: Disjoint means no element is common in a list

Examples:

Input arr1] = {{0, 4}, {5, 10}, {13, 20}, {24, 25}} arr2[][] = {{1, 5}, {8, 12}, {15, 24}, {25, 26}}

Output {{1, 4}, {5, 5}, {8, 10}, {15, 20}, {24, 24}, {25, 25}}

Explanation {1, 4} is the overlap of {0, 4} and {1, 5}. Similarly, (24, 24} is the overlap of {24, 25} and {15, 24}.

Input arr1[][] = {{0, 2}, {5, 10}, {12, 22}, {24, 25}} arr2[][] = {{1, 4}, {9, 12}, {15, 24}, {25, 26}}

Output {{1, 2}, {9, 10}, {12, 12}, {15, 22}, {24, 24}, {25, 25}}

Explanation {1, 2} is the overlap of {0, 2} and {1, 4}. Similarly, (12, 12} is the overlap of {12, 22} and {9, 12}.

Q25.Given an array of positive numbers, calculate the number of possible contiguous subarrays having product lesser than a given number K.

Examples:

Input: arr[] = [1, 2, 3, 4] K = 10

Output: 7

The subarrays are {1}, {2}, {3}, {4}, {1, 2}, {1, 2, 3} and {2, 3}

Input: arr[] = [1, 9, 2, 8, 6, 4, 3] K = 100

Output: 16

Input: arr[] = [10, 5, 2, 6] K = 100

Output: 8

Q26.Given an array of integers arr[] representing a permutation (i.e., all elements are unique and arranged in some order), find the next lexicographically greater permutation by rearranging the elements of the array.

If such a permutation does not exist (i.e., the array is the last possible permutation), rearrange the elements to form the lowest possible order (i.e., sorted in ascending order).

Examples:

Input: arr[] = [2, 4, 1, 7, 5, 0]

Output: [2, 4, 5, 0, 1, 7]

Explanation: The next lexicographically greater arrangement of the elements in the array arr[] is [2, 4, 5, 0, 1, 7].

Input: arr[] = [3, 2, 1]

Output: [1, 2, 3]

Explanation: This is the last permutation, so we return the lowest possible permutation (ascending order).

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

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

Q27.Given an integer array arr[] sorted in ascending order, along with three integers: A, B, and C. The task is to transform each element x in the array using the quadratic function A^ * (x ^ 2) +B^ * x+C After applying this transformation to every element, return the modified array in sorted order.

Examples:

Input: arr[] = [-4, -2, 0, 2, 4], A = 1 , B = 3, C = 5

Output: [3, 5, 9, 15, 33]

Explanation: After applying f(x) = 7^ * X ^ 2 +3^ * X+5 to each x, we get [9, 3, 5, 15, 33]. After sorting this array, the array becomes [3, 5, 9, 15, 33].

Input: arr[] = [-3, -1, 2, 4], A = - 1 , B = 0 C = 0

Output: [-16, -9, -4, -1]

Explanation: After applying f(x) = -1^ * x ^ 2 to each x, we get [-9, -1, -4, -16]. After sorting this array, the array becomes [-16, -9, -4, -1].

Input: arr[] = [-1, 0, 1, 2, 3, 4], A = - 1 B = 2, C = - 1

Output: [-9, -4, -4, -1, -1, 0]

Q28.Given an integer array arr[], the task is to divide the array into three non-empty contiguous segments with equal sum. In other words, we need to return an index pair [i, j], such that sum(arr[0...i]) = sum(arr[i+1...j]) = sum(arr[j+1...n-1]).

Note: If it is impossible to divide the array into three non-empty contiguous segments, return [-1, -1].

Examples:

Input: arr[] = [1, 3, 4, 0, 4]

Output: [1, 2]

Explanation: 3 equal sum segments are: arr[0...1], arr[2...2] and arr[3...4] each having sum = 4.

Input: arr[] = [2, 3, 4]

Output: [-1, -1]

Explanation: No three segments exist which has equal sum.

Input: arr[] = [1, -1, 1, -1, 1, -1, 1, -1]

Output: [1, 3]

Explanation: 3 equal sum segments are: arr[0...1], arr[2...3] and arr[4...7] each having sum = 0.

Q29.Given an array arr[] consisting of positive, negative, and zero values, find the maximum product that can be obtained from any contiguous subarray of arr[].

Examples:

Input: arr[] = [-2, 6, -3, -10, 0, 2]

Output: 180

Explanation: The subarray with maximum product is [6, -3, -10] with product = 6 * (-3) * (-10) = 180.

Input: arr[] = [-1, -3, -10, 0, 6]

Output: 30

Explanation: The subarray with maximum product is [-3, -10] with product = (-3) * (-10) = 30.

Input: arr[] = [2, 3, 4]

Output: 24

Explanation: For an array with all positive elements, the result is the product of all elements.

Q30.Given an array of integers arr[] representing a permutation (i.e., all elements are unique and arranged in some order), find the next lexicographically greater permutation by rearranging the elements of the array.
If such a permutation does not exist (i.e., the array is the last possible permutation), rearrange the elements to form the lowest possible order (i.e., sorted in ascending order).
Examples:
Input: arr[] = [2, 4, 1, 7, 5, 0]
Output: [2, 4, 5, 0, 1, 7]
Explanation: The next lexicographically greater arrangement of the elements in the array arr[] is [2, 4, 5, 0, 1, 7].
Input: arr[] = [3, 2, 1]
Output: [1, 2, 3]
Explanation: This is the last permutation, so we return the lowest possible permutation (ascending order).
Input: arr[] = [1, 3, 5, 4, 2]
Output: [1, 4, 2, 3, 5]

Q31.Given two non-empty strings s1 and s2 of lowercase letters, determine if they are anagrams - i.e., if they contain the same characters with the same frequencies.
Examples:
Input: s1 = "geeks" s2 = "kseeg"
Output: true
Explanation: Both the strings have the same characters with the same frequency. So, they are anagrams.
Input: s1 = "allergy", s2 = "allergy"
Output: false

Explanation: Although the characters are mostly the same, s2 contains an extra 'y' character. Since the frequency of characters differs, the strings are not anagrams.
Input: s1 = "listen", s2 = "lists"
Output: false
Explanation: The characters in the two strings are not the same some are missing or extra. So, they are not anagrams.

Q32.Given two strings s1 and s2 of equal length, determine whether s2 is a rotation of s1. A string is said to be a rotation of another if it can be obtained by shifting some leading characters of the original string to its end without changing the order of characters.
Examples:
Input: s1 = "abcd", s2 = "cdab"
Output: true
Explanation: After 2 right rotations, s1 will become equal to s2.
Input: s1 = "aab", s2 = "aba"
Output: true
Explanation: After 1 left rotation, s1 will become equal to s2.
Input: s1 = "abcd", s2 = "acbd"
Output: false
Explanation: Strings are not rotations of each other.

Q33.Given a binary array arr[] consisting of only 0s and 1s, find the length of the longest contiguous sequence of either 1s or Os in the array.
Examples:
Input: arr[] = [0, 1, 0, 1, 1, 1, 1]
Output: 4
Explanation: The maximum number of consecutive 1's in the array is 4 from index 3-6.
Input: arr[] = [0, 0, 1, 0, 1, 0]
Output: 2
Explanation: The maximum number of consecutive O's in the array is 2 from index 0-1.
Input: arr[] = [0, 0, 0, 0]
Output: 4
Explanation: The maximum number of consecutive O's in the array is 4.

Q34.Given an array arr[] of size n, find the element that appears more than Ln/2」 times. If no such element exists, return -1.
Examples:
Input: arr[] = [1, 1, 2, 1, 3, 5, 1]
Output: 1
Explanation: Element 1 appears 4 times. Since L7/2] = 3, and 4 > 3, it is the majority element.
Input: arr[] = [7]
Output: 7
Explanation: Element 7 appears once. Since L1/2] = 0, and 1 > 0, it is the majority element.
Input: arr[] = [2, 13]

Output: -1
Explanation: No element appears more than L2/2] = 1 time, so there is no majority element.

Q36.Given an array arr[] and an integer k, find the array after reversing every subarray of consecutive k elements in place. If the last subarray has fewer than k elements, reverse it as it is. Modify the array in place, do not return anything.
Examples:
Input: arr[] = [1, 2, 3, 4, 5, 6, 7, 8], k = 3
Output: [3, 2, 1, 6, 5, 4, 8, 7]
Explanation: Elements is reversed: [1, 2, 3]→ [3, 2, 1], [4, 5, 6]→ [6, 5, 4], and the last group [7, 8](size < 3) is reversed as [8, 7].
Input: arr[] = [1, 2, 3, 4, 5], k = 3
Output: [3, 2, 1, 5, 4]
Explanation: First group consists of elements 1, 2, 3. Second group consists of 4, 5.
Input: arr[] = [5, 6, 8, 9], k = 5
Output: [9, 8, 6, 5]
Explanation: Since k is greater than array size, the entire array is reversed.

Q37.Given three sorted arrays in non-decreasing order, print all common elements in non-decreasing order across these arrays. If there are no such elements, return an empty array. In this case, the output will be -1.
Note: In case of duplicate common elements, print only once.
Examples:
Input: arr1[] = [1, 5, 10, 20, 30], arr2[] = [5, 13, 15, 20], arr3[] = [5, 20]
Output: 5 20
Explanation: 5 and 20 are common in all the arrays.
Input: arr1[] = [2, 5, 10, 30], arr2[] = [5, 20, 34], arr3[] = [5, 13, 19]
Output: 5
Explanation: 5 is common in all the arrays.

Q38.Given an integer array arr[] and an integer ele the task is to remove all occurrences of ele from arr[] in-place and return the number of elements which are not equal to ele. If there are k numbers of elements which are not equal to ele then the input array arr should be modified such that the first k elements should contain the elements which are not equal to ele and then the remaining elements.
Note: The order of first k elements may be changed.
Examples:
Input: arr[] = [3, 2, 2, 3], ele = 3
Output: 2
Explanation: The answer is 2 because there are 2 elements which are not equal to 3 and arr[] will be modified such that the first 2 elements contain the elements which are not equal to 3 and remaining elements can contain any element. So, modified arr[] = [2, 2,,]
Input: arr[] = [0, 1, 3, 0, 2, 2, 4, 2], ele = 2

Q39.Given a sorted array arr[] of size n, the goal is to rearrange the array so that all distinct elements appear at the beginning in sorted order. Additionally, return the length of this distinct sorted subarray.
Note: The elements after the distinct ones can be in any order and hold any value, as they don't affect the result.
Examples:
Input: arr[] = [2, 2, 2, 2, 2]
Output: [2]
Explanation: All the elements are 2, So only keep one instance of 2.
Input: arr[] = [1, 2, 2, 3, 4, 4, 4, 5, 5]
Output: [1, 2, 3, 4, 5]
Input: arr[] = [1, 2, 3]
Output: [1, 2, 3]
Explanation: No change as all elements are distinct.

Q40.Given a string S which consists of only lowercase English alphabets, the task is to remove the first repeating character, reverse it, and repeat until there are no repeating characters. Return the final string.
Examples:
Input: S = "abab"
Output: ba
Explanation: In 1st operation: The first non repeating character is a. After removing the first character, S = "bab". After reversing the string, S = "bab".
In 2nd operation: The first non repeating character is b. After removing the first character, S = "ab". After reversing the string, S = "ba". Now the string S does not contain any repeating character.

Q41.Given an array of integers, you have to find three numbers such that the sum of two elements equals the third element.
Examples:
Input: arr[] = [1, 2, 3, 4, 5]
Output: True
Explanation: The pair (1, 2) sums to 3.
Input: arr[] = [3, 4, 5]
Output: True
Explanation: No triplets satisfy the condition.
Input: arr[] = [2, 7, 9, 15]
Output: True
Explanation: The pair (2, 7) sums to 9.

Q.42 Given an array arr[] of integers and a number x, the task is to find the smallest subarray with a sum strictly greater than X.
Examples:
Input: x = 51, arr[] = [1, 4, 45, 6, 0, 19]

Output: 3
Explanation: Minimum length subarray is [4, 45, 6]
Input: x = 100, arr[] = [1, 10, 5, 2, 7]
Output: 0
Explanation: No subarray exist
Q43.Given an array arr[] of n integers and an integer target, find a pair of elements from the array such that the sum of the pair is closest to the given target.
Note:
Return the pair in sorted order.
If multiple pairs have the same closest sum, return the one with the maximum absolute difference (i.e., |a - bl is largest).
If no valid pair exists (i.e., array has fewer than 2 elements), return an empty array.
Examples:
Input: arr = [10, 30, 20, 5], target = 25
Output: [5, 20]
Explanation: Out of all the pairs, [5, 20] has sum = 25 which is closest to 25.
Input: arr[] = [5, 2, 7, 1, 4], target = 10
Output: [2, 7]

Q44.Questions
Maximum Subarray Sum Problem Statement Given an array of integers, determine the maximum possible sum of any contiguous subarray within the array.
Example:
Probi...
Input:
array = [34, -50, 42, 14, -5, 86]
Output:
137
Nate
comment
0400
ENG IN
10:05 AM
22-Sep-25

Q45.Editing
Nth Fibonacci Number Problem Statement
Calculate the Nth term in the Fibonacci sequence, where the sequence is defined as follows: F(n) = F(n-1) + F(n-2), with initial conditions F(1) = F(2) = 1.
Input:
N = 5
Output:
5

The Fibonacci sequence up to the 5th term is: 1, 1, 2, 3, 5. Hence, the 5th Fibonacci number is 5.
Constraints:
+110000 Input Nh is a positive integer

Q46.Minimum Number of Platforms Needed Problem Statement
You are given the arrival and departure times of N trains at a railway station for a particular day. Your task is to determine the minimum number of platforms required so that no train has to wait, meaning that every arriving train has an available platform immediately.
Input:
T-1
N=3
Arrivals = [900, 940, 950]
Departures = [910, 1200, 1120]
Output:
2
Explanation:
In the given example, between 940 and 950, you have two overlapping schedules which require two platforms
Constraints:
100 1000
an departure 2350
Time representation is in HHMM format

Q47.Input: s = "bbabcbcab"
Output: 7
Explanation: Subsequence "babcbab" is the longest subsequence which is also a palindrome.
Input: s = "abcd"
Output: 1
Explanation: "a", "b", "c" and "d" are palindromic and all have a length 1.

Q48.Given an array arr[] of size n, containing elements from the range 1 to n, and each element appears at most twice, return an array of all the integers that appears twice.
Note: You can return the elements in any order but the driver code will print them in sorted order.
Examples:
Input: arr[] = [2, 3, 1, 2, 3]
Output: [2, 3]
Explanation: 2 and 3 occur more than once in the given array.
Input: arr[] = [3, 1, 2]
Output: []
Explanation: There is no repeating element in the array, so the output is empty.

Q49.Write a code to convert a given number into Words Example
i/p : 4,3,8,2,3,77,6,4 and
o/p : forty three crore eighty two lacs thirty seven thousand seven hundred sixty four

Q50.Given two binary string S1 and S2 the task is to written that some the input string may contain reading zeros but the output string should not have any reading zero
i/p: S1 = 00100
S2 = 010
o/p: 110

Q51 Given a string **s** containing three types of brackets {}, () and []. Determine whether the expressions are balanced or not.
An expression is balanced if each opening bracket has a corresponding closing bracket of the same type, the pairs are properly ordered and no bracket closes before its matching opening bracket.

- **Balanced:** "[()()]{}" → every opening bracket is closed in the correct order.
- **Not balanced:** "([{]})" → the ']' closes before the matching '{' is closed, breaking the nesting rule.

Q52.Given an integer array **arr[]** and an integer **ele** the task is to the remove all occurrences of **ele** from arr[] **in-place** and return the number of elements which are not equal to **ele**. If there are **k** number of elements which are not equal to **ele** then the input array arr[] should be modified such that the first **k** elements should contain the elements which are not equal to **ele** and then the remaining elements.
Note: The order of first **k** elements may be changed.

Examples:

Input: *arr[] = [3, 2, 2, 3], ele = 3*
Output: *2*
Explanation: *The answer is 2 because there are 2 elements which are not*

equal to 3 and arr[] will be modified such that the first 2 elements contain the elements which are not equal to 3 and remaining elements can contain any element. So, modified arr[] = [2, 2, _, _]

Input: *arr[] = [0, 1, 3, 0, 2, 2, 4, 2], ele = 2*
Output: *5*
Explanation: *The answer is 5 because there are 5 elements which are not equal to 2 and arr[] will be modified such that the first 5 elements contain the elements which are not equal to 2 and remaining elements can contain any element. So, modified arr[] = [0, 1, 3, 0, 4, _, _, _]*

Q52.Given two sorted arrays **a[]** and **b[]** of size **n** and **m** respectively, merge both the arrays and rearrange the elements such that the smallest **n** elements are in **a[]** and the remaining **m** elements are in **b[]**. All elements in a[] and b[] should be in sorted order.
Examples:

Input: *a[] = [2, 4, 7, 10], b[] = [2, 3]*
Output: *a[] = [2, 2, 3, 4], b[] = [7, 10]*
Explanation*: Combined sorted array = [2, 2, 3, 4, 7, 10], array a[] contains smallest 4 elements: 2, 2, 3 and 4, and array b[] contains remaining 2 elements: 7, 10.*

Input: *a[] = [1, 5, 9, 10, 15, 20], b[] = [2, 3, 8, 13]*
Output: *a[] = [1, 2, 3, 5, 8, 9], b[] = [10, 13, 15, 20]*
Explanation*: Combined sorted array = [1, 2, 3, 5, 8, 9, 10, 13, 15, 20], array a[] contains smallest 6 elements: 1, 2, 3, 5, 8 and 9, and array b[] contains remaining 4 elements: 10, 13, 15, 20.*

Input: *a[] = [0, 1], b[] = [2, 3]*
Output: *a[] = [0, 1], b[] = [2, 3]*
Explanation*: Combined sorted array = [0, 1, 2, 3], array a[] contains smallest 2 elements: 0 and 1, and array b[] contains remaining 2 elements: 2 and 3.*

Q68.Given the ***n*** size array, find its mean.
Examples:

Input : *{1, 3, 4, 2, 6, 5, 8, 7}*
Output : *Mean = 4.5*
Explanation: *Sum of the elements is 1 + 3 + 4 + 2 + 6 + 5 + 8 + 7 = 36, Mean = 36/8 = 4.5*

Input : *{4, 4, 4, 4, 4}*
Output : *Mean = 4*

Q54.Given a queue, the task is to sort it using recursion without using any loop. We can only use the following functions of the queue:

- **empty(q):** Tests whether the queue is empty or not.
- **push(q):** Adds a new element to the queue.
- **pop(q):** Removes front element from the queue.
- **size(q):** Returns the number of elements in a queue.
- **front(q):** Returns the value of the front element without removing it.

Examples:

Input: *queue = {10, 7, 16, 9, 20, 5}*
Output: *5 7 9 10 16 20*

Explanation : *After sorting the elements of the Queue the order becomes 5 7 9 10 16 20*

Input: *queue = {0, -2, -1, 2, 3, 1}*
Output: *-2 -1 0 1 2 3*
Explanation : *After sorting the elements of the Queue the order becomes -2 -1 0 1 2 3.*

Q55. In the given arr the elements is isn ascending order the task is to find the interval from lowest interval to highest including the value that is present b/w the elements as well.
Ex. i/p: arr = [14,15,20,30,31,45]
Lower = 10
Higher = 50
o/p : [10,13,],[16,19],[21,29],[32,44],[46,50]

Q56.Given an integer array, find a maximum product of a triplet in the array.
Examples:

Input: *arr[ ] = [10, 3, 5, 6, 20]*
Output: *1200*
Explanation: *Multiplication of 10, 6 and 20*

Input: *arr[ ] = [-10, -3, -5, -6, -20]*
Output: *-90*

Input: *arr[ ] = [1, -4, 3, -6, 7, 0]*
Output: *168*

Q57Given an integer array, find a maximum product of a triplet in the array.

Examples:

Input: *arr[ ] = [10, 3, 5, 6, 20]*
Output: *1200*
Explanation: *Multiplication of 10, 6 and 20*

Input: *arr[ ] = [-10, -3, -5, -6, -20]*
Output: *-90*

Input: *arr[ ] = [1, -4, 3, -6, 7, 0]*
Output: *168*

Q58.Given a signed 32-bit integer x, return x *with its digits reversed*. If reversing x causes the value to go outside the signed 32-bit integer range $[-2^{31}, 2^{31} - 1]$, then return 0.
Assume the environment does not allow you to store 64-bit integers (signed or unsigned).

Example 1:

Input: x = 123
Output: 321

Example 2:

Input: x = -123
Output: -321

Example 3:

Input: x = 120
Output: 21

Constraints:

- $-2^{31} <= x <= 2^{31} - 1$

Q59.Given a signed 32-bit integer x, return x *with its digits reversed*. If reversing x causes the value to go outside the signed 32-bit integer range $[-2^{31}, 2^{31} - 1]$, then return 0.
Assume the environment does not allow you to store 64-bit integers (signed or unsigned).

Example 1:

Input: x = 123
Output: 321

Example 2:

Input: x = -123
Output: -321

Example 3:

Input: x = 120
Output: 21

Constraints:

- $-2^{31} <= x <= 2^{31} - 1$

Q60.Given a signed 32-bit integer x, return x *with its digits reversed*. If reversing x causes the value to go outside the signed 32-bit integer range $[-2^{31}, 2^{31} - 1]$, then return 0.
Assume the environment does not allow you to store 64-bit integers (signed or unsigned).

Example 1:

Input: x = 123
Output: 321

Example 2:

Input: x = -123
Output: -321

Example 3:

Input: x = 120
Output: 21

Constraints:

- $-2^{31} <= x <= 2^{31} - 1$

Q61.Preorder Traversal (Root-Left-Right): Visit the root node, Recursively traverse the left subtree, and Recursively traverse the right subtree.

Q62. Inorder Traversal (Left-Root-Right): Recursively traverse the left subtree, Visit the root node, and Recursively traverse the right subtree.

Q63.Postorder traversal visits the node in the order: **Left -> Right -> Root**

Q64.Given a number **n**, check whether it is a prime number or not.
Note: A prime number is a number greater than 1 that has no positive divisors other than 1 and itself.

Input: `n = 7`
Output: `true`
Explanation: *7 is a prime number because it is greater than 1 and has no divisors other than 1 and itself.*

Input: `n = 25`
Output: `false`
Explanation: *25 is not a prime number because it is divisible by 5 (25 = 5 × 5), so it has divisors other than 1 and itself.*

Input: *n = 1*
Output: *false*
Explanation: *1 has only one divisor (1 itself), which is not sufficient for it to be considered prime.*

Q65.Given a number N, the task is to print the prime numbers from 1 to N.
Examples:

Input: *N = 10*
Output: *2, 3, 5, 7*
Explanation *: The output "2, 3, 5, 7" for input N = 10 represents the list of the prime numbers less than or equal to 10.*

Input: *N = 5*
Output: *2, 3, 5*
Explanation *: The output "2, 3, 5" for input N = 5 represents the list of the prime numbers less than or equal to 5.*

Q66.Given the non-negative integers **n** , compute the factorial of a given number.
Note: Factorial of **n** is defined as **n * (n -1) * (n - 2) * ... * 1**, for n = 0, factorial is 1.

Examples:

Input: *n = 5*
Output: *120*
Explanation: *5! = 5 * 4 * 3 * 2 * 1 = 120*

Input: *n = 4*
Output: *24*
Explanation: *4! = 4 * 3 * 2 * 1 = 24*

Q67.Given an array, **arr[]** of **n** integers, and an integer element **x**, find whether element **x** is **present** in the array. Return the **index** of the first occurrence of **x** in the array, or **-1** if it doesn't exist.

***Input**: arr[] = [1, 2, 3, 4], x = 3*
***Output**: 2*
***Explanation**: There is one test case with array as [1, 2, 3 4] and element to be searched as 3. Since 3 is present at index 2, the output is 2.*

***Input**: arr[] = [10, 8, 30, 4, 5], x = 5*
***Output**: 4*
***Explanation**: For array [10, 8, 30, 4, 5], the element to be searched is 5 and it is at index 4. So, the output is 4.*

***Input**: arr[] = [10, 8, 30], x = 6*
***Output**: -1*
***Explanation**: The element to be searched is 6 and its not present, so we return -1.*

Q68.Given an array **arr[]** of length **N**, The task is to find the maximum and the minimum number in the array.

Examples:*Input: arr[] = {1, 2, 3, 4, 5}*
***Output:** Maximum is: 5*
Minimum is: 1
Explanation: The maximum of the array is 5
and the minimum of the array is 1.

***Input:** arr[] = {5, 3, 7, 4, 2}*
***Output:** Maximum is: 7*
Minimum is: 2

Q69.Write a function to find the longest common prefix string amongst an array of strings.

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

Example 1:

Input: strs = ["flower","flow","flight"]

Output: "fl"

Example 2:

Input: strs = ["dog","racecar","car"]

Output: ""

Explanation: There is no common prefix among the input strings.

Q70.Write a program to print the elements of an array in **prefix** and **suffix** manner.

Description:

Given an array of integers, compute and print two sequences:

1. **Prefix sum array** – Each element at index *i* is the sum of all elements from index `0` to `i`.

2. **Suffix sum array** – Each element at index *i* is the sum of all elements from index `i` to the end of the array.

Example:

Input:

```
arr = [1, 2, 3, 4]
```

Output:

```
Prefix: 1, 3, 6, 10
Suffix: 10, 9, 7, 4
```

Q71.Given a number n, our task is to print first n terms of the Fibonacci Series.

Input : n = 5
Output : 0 1 1 2 3

Input n = 1
Output : 0

Q72. Write a program for swapping two arrays.

Q73. Write a program for swapping two strings.

Q74. Write a program to convert the string from upper case to lower case.

Q75. Write a program to convert the string from lower case to upper case.

Q76. Write a program to delete all consonants from a given string.

Q77. Write a program to count the different types of characters in a given string.

Q78. Write a program to sort the characters of a string.

Q79. Write a program for addition of two matrices.

Q80. Write a program for subtraction of two matrices.

Q81. Write a program for multiplication of two matrices.

Q82. Write a program to find out the sum of diagonal elements of a matrix.

Q83. All basic operations of linked-list along with other operations like reverse, size of linked list , middle element of linked list and print all alternative elements of LL and do not use prebuilt functions.

Q84. All basic operations of stack operations and do not use prebuilt functions.

Q85. All basic operations of queue operations and do not use prebuilt functions.

Q86.**Move all alphabets to the front and digits to the end of a string**

Description:
Write a program to rearrange the characters of a given string such that all **alphabetic characters** appear at the beginning and all **digits** appear at the end, while maintaining their original order within each group.

Example:
Input:

```
s = "A1B2C3"
```

Output:

```
ABC123
```

Explanation:
The alphabets `A`, `B`, and `C` are moved to the front, and the digits `1`, `2`, and `3` are moved to the end.