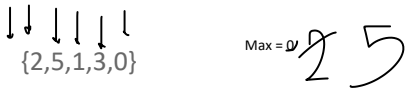
Easy
Thursday, 1 May 2025 4:42 PM

Find the Largest element in an array

Problem Statement: Given an array, we have to find the largest element in the array.

Example 1:
Input: arr[] = {-55, -54, -24,-7};
Output: 5
Explanation: 5 is the largest element in the array.
Example2:
Input: arr[] = {8,10,5,7,9}; = > {5, 7, 8, 9, 10}
Output: 10
Explanation: 10 is the largest element in the array.



Find Second Smallest and Second Largest Element in an array

Problem Statement: Given an array, find the second smallest and second largest element in the array. Print '-1' in the event that either of them doesn't exist.
Example 1:
Input: [1,2,4,7,7,5]
Output: Second Smallest : 2
Second Largest : 5
Explanation: The elements are as follows 1,2,3,5,7,7 and hence second largest of these is 5 and second smallest is 2
Example 2:
Input: [1]
Output: Second Smallest : -1
Second Largest : -1
Explanation: Since there is only one element in the array, it is the largest and smallest element present in the array. There is no second largest or second smallest element present.

Check if an Array is Sorted

Problem Statement: Given an array of size **n**, write a program to check if the given array is sorted in **(ascending / Increasing / Non-decreasing)** order or not. If the array is sorted then return True, Else return False.

Note: Two consecutive equal values are considered to be sorted.
Example 1:

Input: N = 5, array[] = {1,2,3,4,5}
Output: True.
Explanation: The given array is sorted i.e Every element in the array is smaller than or equals to its next values, So the answer is True.

Example 2:
Input: N = 5, array[] = {5,4,6,7,8}
Output: False.
Explanation: The given array is Not sorted i.e Every element in the array is not smaller than or equal to its next values, So the answer is False.
Here element 5 is not smaller than or equal to its future elements.

Remove Duplicates in-place from Sorted Array

Problem Statement: Given an integer array 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.

If there are k elements after removing the duplicates, then the first k elements of the array should hold the final result. It does not matter what you leave beyond the first k elements.
Note: Return k after placing the final result in the first k slots of the array.

Example 1:
Input: arr[1,1,2,2,2,3,3]
Output: arr[1,2,3,_,_,_,_]
Explanation: Total number of unique elements are 3, i.e[1,2,3] and Therefore return 3 after assigning [1,2,3] in the beginning of the array.
Example 2:
Input: arr[1,1,1,2,2,3,3,3,3,4,4]
Output: arr[1,2,3,4,_,_,_,_,_,_,_]
Explanation: Total number of unique elements are 4, i.e[1,2,3,4] and Therefore return 4 after assigning [1,2,3,4] in the beginning of the array.

Left Rotate the Array by One

Problem Statement: Given an array of **N integers**, left rotate the array by one place.
Example 1:
Input: N = 5, array[] = {1,2,3,4,5}
Output: 2,3,4,5,1
Explanation:
Since all the elements in array will be shifted toward left by one so '2' will now become the first index and and '1' which was present at first index will be shifted at last.

Example 2:
Input: N = 1, array[] = {3}
Output: 3
Explanation: Here only element is present and so the element at first index will be shifted to last index which is also by the way the first index.

Rotate array by K elements

Rotate array by K elements

Problem Statement: Given an array of integers, rotating array of elements by k elements either left or right.

Example 1:
Input: N = 7, array[] = {1,2,3,4,5,6,7} , k=2 , right
Output: 6 7 1 2 3 4 5
Explanation: array is rotated to right by 2 position .
Example 2:
Input: N = 6, array[] = {3,7,8,9,10,11} , k=3 , left
Output: 9 10 11 3 7 8
Explanation: Array is rotated to right by 3 position.

Move all Zeros to the end of the array

Problem Statement: You are given an array of integers, your task is to move all the zeros in the array to the end of the array and move non-negative integers to the front by maintaining their order.
Example 1:
Input: 1 ,0 ,2 ,3 ,0 ,4 ,0 ,1
Output: 1 ,2 ,3 ,4 ,1 ,0 ,0 ,0
Explanation: All the zeros are moved to the end and non-negative integers are moved to front by maintaining order
Example 2:
Input: 1,2,0,1,0,4,0
Output: 1,2,1,4,0,0,0
Explanation: All the zeros are moved to the end and non-negative integers are moved to front by maintaining order

Linear Search

Problem Statement: Given an array, and an element num the task is to find if num is present in the given array or not. If present print the index of the element or print -1.
Examples:
Example 1:
Input: arr[]= 1 2 3 4 5, num = 3
Output: 2
Explanation: 3 is present in the 2nd index
Example 2:
Input: arr[]= 5 4 3 2 1, num = 5
Output: 0
Explanation: 5 is present in the 0th index

Union of Two Sorted Arrays

Problem Statement: Given two sorted arrays, **arr1,** and **arr2** of size **n** and **m**. Find the union of two sorted arrays.
The union of two arrays can be defined as the common and distinct elements in the two arrays.**NOTE:** Elements in the union should be in ascending order.
Example 1:
Input:
n = 5,m = 5.
arr1[] = {1,2,3,4,5}

arr1[] = {1,2,3,4,5}
arr2[] = {2,3,4,4,5}
Output:
{1,2,3,4,5}
Explanation:
Common Elements in arr1 and arr2 are: 2,3,4,5
Distnict Elements in arr1 are : 1
Distnict Elemennts in arr2 are : No distinct elements.
Union of arr1 and arr2 is {1,2,3,4,5}
Example 2:
Input:
n = 10,m = 7.
arr1[] = {1,2,3,4,5,6,7,8,9,10}
arr2[] = {2,3,4,4,5,11,12}
Output: {1,2,3,4,5,6,7,8,9,10,11,12}
Explanation:
Common Elements in arr1 and arr2 are: 2,3,4,5
Distnict Elements in arr1 are : 1,6,7,8,9,10
Distnict Elemennts in arr2 are : 11,12
Union of arr1 and arr2 is {1,2,3,4,5,6,7,8,9,10,11,12}

Find the missing number in an array

Problem Statement: Given an **integer N** and an array of size **N-1** containing N−1 numbers between 1 to N. Find the number(*between 1 to N*), that is not present in the given array.
Example 1:
Input Format: N = 5, array[] = {1,2,4,5}
Result: 3
Explanation: In the given array, number 3 is missing. So, 3 is the answer.
Example 2:
Input Format: N = 3, array[] = {1,3}
Result: 2
Explanation: In the given array, number 2 is missing. So, 2 is the answer.

Count Maximum Consecutive One's in the array

Problem Statement: Given an array that contains **only 1 and 0** return the count of **maximum consecutive** ones in the array.
Examples:
Example 1:
Input: prices = {1, 1, 0, 1, 1, 1}
Output: 3
Explanation: There are two consecutive 1's and three consecutive 1's in the array out of which maximum is 3.

Input: prices = {1, 0, 1, 1, 0, 1}
Output: 2
Explanation: There are two consecutive 1's in the array.

Find the number that appears once, and the other numbers twice

Problem Statement: Given a non-empty array of integers **arr**, every element appears twice except for one. Find that single one.

Example 1:
Input Format: arr[] = {2,2,1}
Result: 1
Explanation: In this array, only the element 1 appear once and so it is the answer.
Example 2:
Input Format: arr[] = {4,1,2,1,2}
Result: 4
Explanation: In this array, only element 4 appear once and the other elements appear twice. So, 4 is the answer.

Longest Subarray with given Sum K(Positives)

Problem Statement: Given an array and a sum k, we need to print the length of the longest subarray that sums to k.
Example 1:
Input Format: N = 3, k = 5, array[] = {2,3,5}
Result: 2
Explanation: The longest subarray with sum 5 is {2, 3}. And its length is 2.
Example 2:
Input Format: N = 5, k = 10, array[] = {2,3,5,1,9}
Result: 3
Explanation: The longest subarray with sum 10 is {2, 3, 5}. And its length is 3.

Two Sum : Check if a pair with given sum exists in Array

Problem Statement: Given an array of integers arr[] and an integer target.
Return indices of the two numbers such that their sum is equal to the target. Otherwise, we will return {-1, -1}.

Note: You are not allowed to use the same element twice. Example: If the target is equal to 6 and num[1] = 3, then nums[1] + nums[1] = target is not a solution.

Examples:
Example 1:
Input Format: N = 5, arr[] = {2,6,5,8,11}, target = 14
Result: [1, 3]
Explanation: arr[1] + arr[3] = 14. So, the answer is [1, 3].
Example 2:
Input Format: N = 5, arr[] = {2,6,5,8,11}, target = 15
Result: [-1, -1]
Explanation: There exist no such two numbers whose sum is equal to the target.

Sort an array of 0s, 1s and 2s

Problem Statement: Given an array consisting of only 0s, 1s, and 2s. Write a program to in-place sort the array without using inbuilt sort functions. (Expected: Single pass-O(N) and constant space)

Examples

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

Find the Majority Element that occurs more than N/2 times

7757515755775555

Problem Statement: Given an array of **N integers**, write a program to return an element that occurs more than **N/2** times in the given array. You may consider that such an element always exists in the array.
Example 1:
Input Format: N = 3, nums[] = {3,2,3}
Result: 3
Explanation: When we just count the occurrences of each number and compare with half of the size of the array, you will get 3 for the above solution.
Example 2:
Input Format: N = 7, nums[] = {2,2,1,1,1,2,2}
Result: 2
Explanation: After counting the number of times each element appears and comparing it with half of array size, we get 2 as result.
Example 3:
Input Format: N = 10, nums[] = {4,4,2,4,3,4,4,3,2,4}
Result: 4

Maximum Subarray Sum in an Array

Problem Statement: Given an integer array arr, find the contiguous subarray (containing at least one number) which has the largest sum and returns its sum and prints the subarray.

Examples

Example 1:
Input: arr = [-2,1,-3,4,-1,2,1,-5,4]
Output: 6
Explanation: [4,-1,2,1] has the largest sum = 6.
Examples 2:
Input: arr = [1]
Output: 1
Explanation: Array has only one element and which is giving positive sum of 1.

Stock Buy And Sell

Problem Statement: You are given an array of prices where prices[i] is the price of a given stock on an 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.

Examples

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.

Rearrange Array Elements by Sign

Problem Statement:
There's an array 'A' of size 'N' with an equal number of positive and negative elements. Without altering the relative order of positive and negative elements, you must return an array of alternately positive and negative values.
Note: Start the array with positive elements.
Examples:
Example 1:
Input:
arr[] = {1,2,-4,-5}, N = 4
Output:
1 -4 2 -5
Explanation:
Positive elements = 1,2
Negative elements = -4,-5
To maintain relative ordering, 1 must occur before 2, and -4 must occur before -5.
Example 2:
Input:
arr[] = {1,2,-3,-1,-2,3}, N = 6
Output:
1 -3 2 -1 3 -2
Explanation:
Positive elements = 1,2,3
Negative elements = -3,-1,-2
To maintain relative ordering, 1 must occur before 2, and 2 must occur before 3.
Also, -3 should come before -1, and -1 should come before -2.

Majority Elements(N/3 times) | Find the elements that appears more than N/3 times in the array

Problem Statement: Given an array of N integers. Find the elements that appear more than **N/3** times in the array. If no such element exists, return an empty vector.

Example 1:
Input Format: N = 5, array[] = {1,2,2,3,2}
Result: 2
Explanation: Here we can see that the Count(1) = 1, Count(2) = 3 and Count(3) = 1 Therefore, the count of 2 is

Explanation: Here we can see that the Count(1) = 1, Count(2) = 3 and Count(3) = 1.Therefore, the count of 2 is greater than N/3 times. Hence, 2 is the answer.
Example 2:
Input Format: N = 6, array[] = {11,33,33,11,33,11}
Result: 11 33
Explanation: Here we can see that the Count(11) = 3 and Count(33) = 3. Therefore, the count of both 11 and 33 is greater than N/3 times. Hence, 11 and 33 is the answer.

Leaders in an Array

Problem Statement: Given an array, print all the elements which are leaders. A Leader is an element that is greater than all of the elements on its right side in the array.
Example 1:
Input:
arr = [4, 7, 1, 0]
Output:
7 1 0
Explanation:
Rightmost element is always a leader. 7 and 1 are greater than the elements in their right side.
Example 2:
Input:
arr = [10, 22, 12, 3, 0, 6]
Output:
22 12 6
Explanation:
6 is a leader. In addition to that, 12 is greater than all the elements in its right side (3, 0, 6), also 22 is greater than 12, 3, 0, 6.

Count Subarray sum Equals K

Problem Statement: Given an array of integers and an integer k, return the total number of subarrays whose sum equals k.
A subarray is a contiguous non-empty sequence of elements within an array.

Examples

Example 1:
Input Format: N = 4, array[] = {3, 1, 2, 4}, k = 6
Result: 2
Explanation: The subarrays that sum up to 6 are [3, 1, 2] and [2, 4].

Example 2:
Input Format: N = 3, array[] = {1,2,3}, k = 3
Result: 2
Explanation: The subarrays that sum up to 3 are [1, 2], and [3].

3 Sum : Find triplets that add up to a zero

Problem Statement: Given an array of N integers, your task is to find unique triplets that add up to give a sum of zero. In short, you need to return *an array of all the unique* triplets [arr[a], arr[b], arr[c]] such that i!=j, j!=k, k!=i, and their sum is equal to zero.

Examples

Example 1:

Input: nums = [-1,0,1,2,-1,-4]
Output: [[-1,-1,2],[-1,0,1]]
Explanation: Out of all possible unique triplets possible, [-1,-1,2] and [-1,0,1] satisfy the condition of summing up to zero with i!=j!=k
Example 2:
Input: nums=[-1,0,1,0]
Output: Output: [[-1,0,1]]
Explanation: Out of all possible unique triplets possible, [-1,0,1] satisfy the condition of summing up to zero with i!=j!=k

Transpose of a matrix

Rotate Image by 90 degree

Problem Statement: Given a matrix, your task is to rotate the matrix 90 degrees clockwise.

Examples

Example 1:
Input: [[1,2,3],[4,5,6],[7,8,9]]
Output: [[7,4,1],[8,5,2],[9,6,3]]
Explanation: Rotate the matrix simply by 90 degree clockwise and return the matrix.
Example 2:
Input: [[5,1,9,11],[2,4,8,10],[13,3,6,7],[15,14,12,16]]
Output:[[15,13,2,5],[14,3,4,1],[12,6,8,9],[16,7,10,11]]
Explanation: Rotate the matrix simply by 90 degree clockwise and return the matrix

Spiral Traversal of Matrix

Problem Statement: Given a Matrix, print the given matrix in spiral order.
Examples:
Example 1:
Input: Matrix[][] = { { 1, 2, 3, 4 },
{ 5, 6, 7, 8 },
{ 9, 10, 11, 12 },
{ 13, 14, 15, 16 } }
Outhput: 1, 2, 3, 4, 8, 12, 16, 15, 14, 13, 9, 5, 6, 7, 11, 10.
Explanation: The output of matrix in spiral form.
Example 2:
Input: Matrix[][] = { { 1, 2, 3 },
{ 4, 5, 6 },
{ 7, 8, 9 } }

Output: 1, 2, 3, 6, 9, 8, 7, 4, 5.
Explanation: The output of matrix in spiral form.

Set Matrix Zero

Problem Statement: Given a matrix if an element in the matrix is 0 then you will have to set its entire column and row to 0 and then return the matrix.

Examples

Examples 1:
Input: matrix=[[1,1,1],[1,0,1],[1,1,1]]
Output: [[1,0,1],[0,0,0],[1,0,1]]
Explanation: Since matrix[2][2]=0.Therfore the 2nd column and 2nd row wil be set to 0.

Input: matrix=[[0,1,2,0],[3,4,5,2],[1,3,1,5]]
Output:[[0,0,0,0],[0,4,5,0],[0,3,1,0]]
Explanation: Since matrix[0][0]=0 and matrix[0][3]=0. Therefore 1st row, 1st column and 4th column will be set to 0