Que 1 : Missing number in array

Given an array of size N-1 such that it only contains distinct integers in the range of 1 to N. Find the missing element.

Example 1:

Input:
N = 6
A[] = {1,2,4,5,6}
Output: 3

Example 2:

Input:
N = 11
A[] = {1,3,2,5,6,7,8,11,10,4}
Output: 9

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 10^6$
$1 \leq A[i] \leq 10^6$

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

Given an array of size N containing only 0s, 1s, and 2s; sort the array in ascending order.

Example 1:

Input:
N = 5
arr[]= {0 2 1 2 0}
Output:
0 0 1 2 2
Explanation: 0s 1s and 2s are segregated into ascending order.

Example 2:

Input:
N = 3
arr[] = {0 1 0}
Output:
0 0 1
Explanation: 0s 1s and 2s are segregated into ascending order.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
1 <= N <= 10^6
0 <= A[i] <= 2

Que 3 : Find Duplicates in an Array

Given an array of size N which contains elements from 0 to N-1, you need to find all the elements occurring more than once in the given array. Return the answer in ascending order. If no such element is found, return list containing [-1].

Note: The extra space is only for the array to be returned. Try and perform all operations within the provided array.

Example 1:

Input:
N = 4
a[] = {0,3,1,2}
Output:
-1
Explanation:
There is no repeating element in the array. Therefore output is -1.

Example 2:

Input:
N = 5
a[] = {2,3,1,2,3}
Output:
2 3
Explanation:
2 and 3 occur more than once in the given array.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
1 <= N <= 10^5
0 <= A[i] <= N-1, for each valid i

Que 4 : Count pairs with given sum

Given an array of N integers, and an integer K, find the number of pairs of elements in the array whose sum is equal to K.

Example 1:

Input:
N = 4, K = 6
arr[] = {1, 5, 7, 1}
Output: 2
Explanation:
arr[0] + arr[1] = 1 + 5 = 6
and arr[1] + arr[3] = 5 + 1 = 6.

Example 2:

Input:
N = 4, K = 2
arr[] = {1, 1, 1, 1}
Output: 6
Explanation:
Each 1 will produce sum 2 with any 1.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
$1 <= N <= 10^5$
$1 <= K <= 10^8$
$1 <= Arr[i] <= 10^6$

Que 5 : Peak Elements

An element is called a peak element if its value is not smaller than the value of its adjacent elements(if they exist).
Given an array arr[] of size N, Return the index of any one of its peak elements.
Note: The generated output will always be 1 if the index that you return is correct. Otherwise output will be 0.

Example 1:

Input:
N = 3
arr[] = {1,2,3}
Possible Answer: 2
Generated Output: 1
Explanation: index 2 is 3. It is the peak element as it is greater than its neighbor 2. If 2 is returned then the generated output will be 1 else 0.

Example 2:

Input:
N = 3
arr[] = {3,4,2}
Possible Answer: 1
Output: 1
Explanation: 4 (at index 1) is the peak element as it is greater than it's neighbor elements 3 and 2. If 1 is returned then the generated output will be 1 else 0.

Expected Time Complexity: O(log N)
Expected Auxiliary Space: O(1)

Constraints:
$1 \le N \le 10^5$
$1 \le A[] \le 10^6$

Que 6 : Second Largest

Given an array Arr of size N, print the second largest distinct element from an array.

Example 1:

Input:
N = 6
Arr[] = {12, 35, 1, 10, 34, 1}
Output: 34
Explanation: The largest element of the array is 35 and the second largest element is 34.

Example 2:

Input:
N = 3
Arr[] = {10, 5, 10}
Output: 5
Explanation: The largest element of the array is 10 and the second largest element is 5.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$2 \leq N \leq 10^5$
$1 \leq Arr_i \leq 10^5$

Que 7 : Check if array is sorted

Given an array arr[] of size N, check if it is sorted in non-decreasing order or not.

Example 1:

> **Input:**
> N = 5
> arr[] = {10, 20, 30, 40, 50}
> **Output:** 1
> **Explanation:** The given array is sorted.

Example 2:

> **Input:**
> N = 6
> arr[] = {90, 80, 100, 70, 40, 30}
> **Output:** 0
> **Explanation:** The given array is not sorted.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 10^5$
$1 \leq Arr[i] \leq 10^6$

Que 8 : Rotate Array

Given an unsorted array arr[] of size N. Rotate the array to the left (counter-clockwise direction) by D steps, where D is a positive integer.

Example 1:

Input:
N = 5, D = 2
arr[] = {1,2,3,4,5}
Output: 3 4 5 1 2
Explanation: 1 2 3 4 5 when rotated by 2 elements, it becomes 3 4 5 1 2.

Example 2:

Input:
N = 10, D = 3
arr[] = {2,4,6,8,10,12,14,16,18,20}
Output: 8 10 12 14 16 18 20 2 4 6
Explanation: 2 4 6 8 10 12 14 16 18 20 when rotated by 3 elements, it becomes 8 10 12 14 16 18 20 2 4 6.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
1 <= N <= 10^6
1 <= D <= 10^6
0 <= arr[i] <= 10^5

Que 9 : Find transition Point

Given a sorted array containing only 0s and 1s, find the transition point.

Example 1:

> **Input:**
> N = 5
> arr[] = {0,0,0,1,1}
> **Output:** 3
> **Explanation:** index 3 is the transition point where 1 begins.

Example 2:

> **Input:**
> N = 4
> arr[] = {0,0,0,0}
> **Output:** -1
> **Explanation:** Since, there is no "1", the answer is -1.

Expected Time Complexity: O(LogN)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 500000$
$0 \leq arr[i] \leq 1$

Que 10 : First Repeating Element

Given an array arr[] of size n, find the first repeating element. The element should occur more than once and the index of its first occurrence should be the smallest.

Note:- The position you return should be according to 1-based indexing.

Example 1:

Input:
n = 7
arr[] = {1, 5, 3, 4, 3, 5, 6}
Output: 2
Explanation: 5 is appearing twice and its first appearance is at index 2 which is less than 3 whose first occurring index is 3.

Example 2:

Input:
n = 4
arr[] = {1, 2, 3, 4}
Output: -1
Explanation: All elements appear only once so the answer is -1.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
$1 <= n <= 10^6$
$0 <= A_i <= 10^6$

Que 11 : Frequencies of limited Range Array Elements

Given an array A[] of N positive integers which can contain integers from 1 to P where elements can be repeated or can be absent from the array. Your task is to count the frequency of all elements from 1 to N.
Note: The elements greater than N in the array can be ignored for counting and do modify the array in-place.

Example 1:

Input:
N = 5
arr[] = {2, 3, 2, 3, 5}
P = 5
Output:
0 2 2 0 1
Explanation: Counting frequencies of each array element
We have:
1 occurring 0 times.
2 occurring 2 times.
3 occurring 2 times.
4 occurring 0 times.
5 occurring 1 time.

Example 2:

Input:
N = 4
arr[] = {3,3,3,3}
P = 3
Output:
0 0 4 0
Explanation: Counting frequencies of each array element
We have:
1 occurring 0 times.
2 occurring 0 times.
3 occurring 4 times.

4 occurring 0 times.

Can you solve this problem without using extra space (O(1) Space)?

Constraints:

$1 \leq N \leq 10^5$

$1 \leq P \leq 4*10^4$

1 <= A[i] <= P

Que 12 : Number of occurance

Given a sorted array Arr of size N and a number X, you need to find the number of occurrences of X in Arr.

Example 1:

Input:
N = 7, X = 2
Arr[] = {1, 1, 2, 2, 2, 2, 3}
Output: 4
Explanation: 2 occurs 4 times in the given array.

Example 2:

Input:
N = 7, X = 4
Arr[] = {1, 1, 2, 2, 2, 2, 3}
Output: 0
Explanation: 4 is not present in the given array.

Expected Time Complexity: O(logN)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 10^5$
$1 \leq Arr[i] \leq 10^6$
$1 \leq X \leq 10^6$

Que 13 : Minimum distance between two number

You are given an array A, of N elements. Find minimum index based distance between two elements of the array, x and y such that (x!=y).

Example 1:

Input:
N = 4
A[] = {1,2,3,2}
x = 1, y = 2
Output: 1
Explanation: x = 1 and y = 2. There are two distances between x and y, which are 1 and 3 out of which the least is 1.

Example 2:

Input:
N = 7
A[] = {86,39,90,67,84,66,62}
x = 42, y = 12
Output: -1
Explanation: x = 42 and y = 12. We return -1 as x and y don't exist in the array.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
1 <= N <= 10^5
0 <= A[i], x, y <= 10^5

Que 14 : Intersection of two Array

Given two arrays a[] and b[] respectively of size n and m, the task is to print the count of elements in the intersection (or common elements) of the two arrays.

For this question, the intersection of two arrays can be defined as the set containing distinct common elements between the two arrays.

Example 1:

> **Input:**
> n = 5, m = 3
> a[] = {89, 24, 75, 11, 23}
> b[] = {89, 2, 4}
> **Output:** 1
> **Explanation:**
> 89 is the only element in the intersection of two arrays.

Example 2:

> **Input:**
> n = 6, m = 5
> a[] = {1, 2, 3, 4, 5, 6}
> b[] = {3, 4, 5, 6, 7}
> **Output:** 4
> **Explanation:**
> 3 4 5 and 6 are the elements in the intersection of two arrays.

Expected Time Complexity: O(n + m).
Expected Auxiliary Space: O(min(n,m)).

Constraints:
$1 \leq n, m \leq 10^5$
$1 \leq a[i], b[i] \leq 10^5$

Que 15 : Union of two sorted Array

Union of two arrays can be defined as the common and distinct elements in the two arrays.
Given two sorted arrays of size n and m respectively, find their union.

Example 1:

Input:
n = 5, arr1[] = {1, 2, 3, 4, 5}
m = 3, arr2 [] = {1, 2, 3}
Output: 1 2 3 4 5
Explanation: Distinct elements including both the arrays are: 1 2 3 4 5.

Example 2:

Input:
n = 5, arr1[] = {2, 2, 3, 4, 5}
m = 5, arr2[] = {1, 1, 2, 3, 4}
Output: 1 2 3 4 5
Explanation: Distinct elements including both the arrays are: 1 2 3 4 5.

Example 3:

Input:
n = 5, arr1[] = {1, 1, 1, 1, 1}
m = 5, arr2[] = {2, 2, 2, 2, 2}
Output: 1 2
Explanation: Distinct elements including both the arrays are: 1 2.

Expected Time Complexity: O(n+m).
Expected Auxiliary Space: O(n+m).

Constraints:
1 <= n, m <= 10^5
1 <= arr[i], brr[i] <= 10^6

Que 16 : Rotation

Given an ascending sorted rotated array Arr of distinct integers of size N. The array is right rotated K times. Find the value of K.

Example 1:

Input:
N = 5
Arr[] = {5, 1, 2, 3, 4}
Output: 1
Explanation: The given array is 5 1 2 3 4. The original sorted array is 1 2 3 4 5. We can see that the array was rotated 1 time to the right.

Example 2:

Input:
N = 5
Arr[] = {1, 2, 3, 4, 5}
Output: 0
Explanation: The given array is not rotated.

Expected Time Complexity: O(log(N))
Expected Auxiliary Space: O(1)

Constraints:
$1 <= N <=10^5$
$1 <= Arr_i <= 10^7$

Que 17 : Find all Pairs with given Sum

Given two unsorted arrays A of size N and B of size M of distinct elements, the task is to find all pairs from both arrays whose sum is equal to X.

Note: All pairs should be printed in increasing order of u. For eg. for two pairs (u1,v1) and (u2,v2), if u1 < u2 then
(u1,v1) should be printed first else second.

Example 1:

> **Input:**
> A[] = {1, 2, 4, 5, 7}
> B[] = {5, 6, 3, 4, 8}
> X = 9
> **Output:**
> 1 8
> 4 5
> 5 4
> **Explanation:** (1, 8), (4, 5), (5, 4) are the pairs which sum to 9.

Example 2:

> **Input:**
> A[] = {-1, -2, 4, -6, 5, 7}
> B[] = {6, 3, 4, 0}
> X = 8
> **Output:**
> 4 4
> 5 3

Expected Time Complexity: O(NLog(N))
Expected Auxiliary Space: O(N)

Constraints:
$1 \leq N, M \leq 10^6$
$-10^6 \leq X, A[i], B[i] \leq 10^6$

Que 18 : Move all zero two the end of Array

Given an array arr[] of N positive integers. Push all the zeros of the given array to the right end of the array while maintaining the order of non-zero elements.

Example 1:

Input:
N = 5
Arr[] = {3, 5, 0, 0, 4}
Output: 3 5 4 0 0
Explanation: The non-zero elements preserve their order while the 0 elements are moved to the right.

Example 2:

Input:
N = 4
Arr[] = {0, 0, 0, 4}
Output: 4 0 0 0
Explanation: 4 is the only non-zero element and it gets moved to the left.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 105$
$0 \leq arr_i \leq 105$

Que 19 : Elements with left side smaller and right side greater

Given an unsorted array of size N. Find the first element in the array such that all of its left elements are smaller and all right elements to it are greater than it.

Note: Left and right side elements can be equal to required elements. And extreme elements cannot be required.

Example 1:

Input:
N = 4
A[] = {4, 2, 5, 7}
Output:
5
Explanation:
Elements on the left of 5 are smaller than 5 and on the right of it are greater than 5.

Example 2:

Input:
N = 3
A[] = {11, 9, 12}
Output:
-1

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
$3 <= N <= 10^6$
$1 <= A[i] <= 10^6$

Que 20 : Bitonic point

Given an array Arr of n elements that is first strictly increasing and then maybe strictly decreasing, find the maximum element in the array.
Note: If the array is increasing then just print the last element will be the maximum value.

Example 1:

Input:
n = 9
arr[] = {1,15,25,45,42,21,17,12,11}
Output: 45
Explanation: Maximum element is 45.

Example 2:

Input:
n = 5
arr[] = {1, 45, 47, 50, 5}
Output: 50
Explanation: Maximum element is 50.

Expected Time Complexity: O(logn)
Expected Auxiliary Space: O(1)

Constraints:
$3 \leq n \leq 10^6$
$1 \leq arr_i \leq 10^6$

Que 21 : Non Repeating Element

Find the first non-repeating element in a given array Arr of N integers.
Note: Array consists of only positive and negative integers and not zero.

Example 1:

Input : arr[] = {-1, 2, -1, 3, 2}
Output : 3
Explanation: -1 and 2 are repeating whereas 3 is the only number occuring once.
Hence, the output is 3.

Example 2:

Input : arr[] = {1, 1, 1}
Output : 0

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
$1 <= N <= 10^7$
$-10^{16} <= A_i <= 10^{16}$
$\{A_i != 0 \}$

Que 22 : Left most and right most index

Given a sorted array with possibly duplicate elements. The task is to find indexes of first and last occurrences of an element X in the given array.

Note: If the element is not present in the array return {-1,-1} as a pair.

Example 1:

Input:
N = 9
v[] = {1, 3, 5, 5, 5, 5, 67, 123, 125}
X = 5
Output:
2 5
Explanation:
Index of the first occurrence of 5 is 2 and the index of the last occurrence of 5 is 5.

Example 2:

Input:
N = 9
v[] = {1, 3, 5, 5, 5, 5, 7, 123, 125}
X = 7
Output:
6 6

Expected Time Complexity: O(Log(N))
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 10^5$
$1 \leq v[i], X \leq 10^{18}$

Que 23 : Three Great Candidates

The hiring team aims to find 3 candidates who are great collectively. Each candidate has his or her ability expressed as an integer. 3 candidates are great collectively if product of their abilities is maximum. Given abilities of N candidates in an array arr[], find the maximum collective ability from the given pool of candidates.

Example 1:

Input:
N = 5
Arr[] = {10, 3, 5, 6, 20}
Output: 1200
Explanation:
Multiplication of 10, 6 and 20 is 1200.

Example 2:

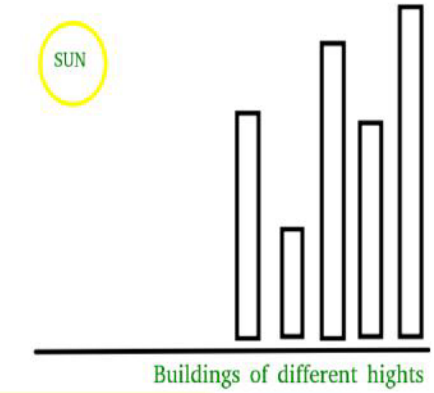
Input:
N = 5
Arr[] = {-10, -3, -5, -6, -20}
Output: -90
Explanation:
Multiplication of -3, -5 and -6 is -90.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$3 \leq N \leq 10^7$
$-10^5 \leq Arr[i] \leq 10^5$

Que 24 : Facing the Sun

Given an array H representing heights of buildings. You have to count the buildings which will see the sunrise (Assume : Sun rises on the side of the array starting point).
Note : Height of building should be strictly greater than height of buildings in left in order to see the sun.



Example 1:

Input:
N = 5
H[] = {7, 4, 8, 2, 9}
Output: 3
Explanation: As 7 is the first element, it can see the sunrise. 4 can't see the sunrise as 7 is hiding it. 8 can see. 2 can't see the sunrise. 9 also can see the sunrise.

Example 2:

Input:
N = 4
H[] = {2, 3, 4, 5}
Output: 4
Explanation: As 2 is the first element, it can see the sunrise. 3 can see the sunrise as 2 is not hiding it. Same for 4 and 5, they also can see the sunrise.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 10^6$
$1 \leq Hi \leq 10^8$

Que 25 : Even and Odd

Given an array arr[] of size N containing an equal number of odd and even numbers. Arrange the numbers in such a way that all the even numbers get the even index and odd numbers get the odd index.
Note: There are multiple possible solutions, Print any one of them. Also, 0-based indexing is considered.

Example 1:

Input:
N = 6
arr[] = {3, 6, 12, 1, 5, 8}
Output:
1
Explanation:
6 3 12 1 8 5 is a possible solution. The output will always be 1 if your rearrangement is correct.

Example 2:

Input:
N = 4
arr[] = {1, 2, 3, 4}
Output :
1

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 10^5$
$1 \leq arr[i] \leq 10^5$

Que 26 : Leaders in an Array

Given an array A of positive integers. Your task is to find the leaders in the array. An element of an array is leader if it is greater than or equal to all the elements to its right side. The rightmost element is always a leader.

Example 1:

> **Input:**
> n = 6
> A[] = {16,17,4,3,5,2}
> **Output:** 17 5 2
> **Explanation:** The first leader is 17 as it is greater than all the elements to its right. Similarly, the next leader is 5. The right most element is always a leader so it is also included.

Example 2:

> **Input:**
> n = 5
> A[] = {1,2,3,4,0}
> **Output:** 4 0
> **Explanation:** The first leader is 4 as all the other numbers aren't greater than the other elements to their right side. The second leader is 0 as there are no elements at the right side so it's greater itself.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
1 <= n <= 10^5
0 <= Ai <= 10^9

Que 27 : Remove Duplicate Elements from sorted Array

Given a sorted array A[] of size N, delete all the duplicate elements from A[]. Modify the array such that if there are X distinct elements in it then the first X positions of the array should be filled with them in increasing order and return the number of distinct elements in the array.

Note:
1. Don't use set or HashMap to solve the problem.
2. You must return the number of distinct elements(X) in the array, the driver code will print all the elements of the modified array from index 0 to X-1 as output of your code.

Example 1:

> **Input:**
> N = 5
> Array = {2, 2, 2, 2, 2}
> **Output:** 2
> **Explanation:** After removing all the duplicates only one instance of 2 will remain i.e. {2} so modify array will contain 2 at first position and you should return 1 after modify the array.

Example 2:

> **Input:**
> N = 4
> Array = {1, 2, 2, 4}
> **Output:** 1 2 4
> **Explanation:** After removing all duplicates, the modified array will contain {1, 2, 4} at first 3 positions so you should return 3 after modifying the array.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(1)

Constraints:
$1 \leq N \leq 10^4$
$1 \leq A[i] \leq 10^6$

Que 28 : Product Array Puzzle

Given an array nums[] of size n, construct a Product Array P (of same size n) such that P[i] is equal to the product of all the elements of nums except nums[i].

Example 1:

Input:
n = 5
nums[] = {10, 3, 5, 6, 2}
Output:
180 600 360 300 900
Explanation:
For i=0, P[i] = 3*5*6*2 = 180.
For i=1, P[i] = 10*5*6*2 = 600.
For i=2, P[i] = 10*3*6*2 = 360.
For i=3, P[i] = 10*3*5*2 = 300.
For i=4, P[i] = 10*3*5*6 = 900.

Example 2:

Input:
n = 2
nums[] = {12,0}
Output:
0 12

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
1 <= n <= 1000
0 <= $nums_i$ <= 200
Arrays may contain duplicates.

Que 29 : Key Pair

Given an array Arr of N positive integers and another number X. Determine whether or not there exist two elements in Arr whose sum is exactly X.

Example 1:

Input:
N = 6, X = 16
Arr[] = {1, 4, 45, 6, 10, 8}
Output: Yes
Explanation: Arr[3] + Arr[4] = 6 + 10 = 16

Example 2:

Input:
N = 5, X = 10
Arr[] = {1, 2, 4, 3, 6}
Output: Yes
Explanation: Arr[2] + Arr[4] = 4 + 6 = 10

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
$1 \leq N \leq 10^5$
$1 \leq Arr[i] \leq 10^5$

Que 30 : Alternate Positive and Negative number

Given an unsorted array Arr of N positive and negative numbers. Your task is to create an array of alternate positive and negative numbers without changing the relative order of positive and negative numbers.
Note: Array should start with a positive number and 0 (zero) should be considered a positive element.

Example 1:

Input:
N = 9
Arr[] = {9, 4, -2, -1, 5, 0, -5, -3, 2}
Output:
9 -2 4 -1 5 -5 0 -3 2
Explanation : Positive elements : 9,4,5,0,2
Negative elements : -2,-1,-5,-3
As we need to maintain the relative order of positive elements and negative elements we will pick each element from the positive and negative and will store them. If any of the positive and negative numbers are completed. we will continue with the remaining signed elements.The output is 9,-2,4,-1,5,-5,0,-3,2.

Example 2:

Input:
N = 10
Arr[] = {-5, -2, 5, 2, 4, 7, 1, 8, 0, -8}
Output:
5 -5 2 -2 4 -8 7 1 8 0
Explanation : Positive elements : 5,2,4,7,1,8,0
Negative elements : -5,-2,-8
As we need to maintain the relative order of positive elements and negative elements we will pick each element from the positive and negative and will store them. If any of the positive and negative numbers are completed. we will continue with the remaining signed elements.The output is 5,-5,2,-2,4,-8,7,1,8,0.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
$1 \leq N \leq 10^7$
$-10^6 \leq Arr[i] \leq 10^7$