1] Search an Element in an array

Given an integer array and another integer element. The task is to find if the given element is present in the array or not.

Example 1:

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

Explanation: There is one test case with an array as {1, 2, 3 4} and an element to be searched as 3. Since 3 is present at index 2, output is 2.

Example 2:

Input:
n = 5
arr[] = {1,2,3,4,5}
x = 5
Output: 4

Explanation: For array elements {1,2,3,4,5} element to be searched is 5 and it is at index 4. So, the output is 4.

Expected Time Complexity: O(n).
Expected Auxiliary Space: O(1).

Constraints:
1 <= n <= 10^6
0 <= arr[i] <= 10^6
0 <= x <= 10^5

2] Find minimum and maximum element in an array

Given an array A of size N of integers. Your task is to find the minimum and maximum elements in the array.

Example 1:

Input:
N = 6
A[] = {3, 2, 1, 56, 10000, 167}
Output: 1 10000

Explanation: minimum and maximum elements of array are 1 and 10000.

Example 2:

Input:
N = 5
A[] = {1, 345, 234, 21, 56789}
Output: 1 56789
Explanation: minimum and maximum elements of array are 1 and 56789.

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

Constraints:
1 <= N <= 10^5
1 <= Ai <=10^12

3] Largest Element in Array

Given an array A[] of size n. The task is to find the largest element in it.

Example 1:

Input:
n = 5
A[] = {1, 8, 7, 56, 90}
Output: 90
Explanation:
The largest element of a given array is 90.

Example 2:

Input:
n = 7
A[] = {1, 2, 0, 3, 2, 4, 5}
Output: 5
Explanation:
The largest element of a given array is 5.

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

Constraints:
1 <= n<= 10^3
0 <= A[i] <= 10^3
Arrays may contain duplicate elements.

4] Product of array elements

This is a functional problem. Your task is to return the product of array elements under a given modulo.
The modulo operation finds the remainder after the division of one number by another. For example, K(mod(m))=K%m= remainder obtained when K is divided by m

Example 1:

Input:
1
4
1 2 3 4
Output:
24

Input:
The first line of input contains T denoting the number of test cases. Then each of the T lines contains a single positive integer N denotes the number of elements in the array. The next line contains 'N' integer elements of the array.

Output:
Return the product of array elements under a given modulo.
That is, return (Array[0]*Array[1]*Array[2]...*Array[n])%modulo.

Constraints:
1<=T<=200
1<=N<=10^5
1<=ar[i]<=10^5

5] Replace all 0's with 5

You are given an integer N. You need to convert all zeros of N to 5.

Example 1:

Input:

N = 1004

Output: 1554

Explanation: There are two zeroes in 1004 on replacing all zeroes with "5", the new number will be "1554".

Example 2:

Input:

N = 121

Output: 121

Explanation: Since there are no zeroes in "121", the number remains as "121".

Expected Time Complexity: O(K) where K is the number of digits in N

Expected Auxiliary Space: O(1)

Constraints:

1 <= n <= 10000

6] Elements in the Range

Given an array arr[] containing positive elements. A and B are two numbers defining a range. The task is to check if the array contains all elements in the given range.

Example 1:

Input: N = 7, A = 2, B = 5
arr[] = {1, 4, 5, 2, 7, 8, 3}
Output: Yes
Explanation: It has elements between range 2-5 i.e 2,3,4,5

Example 2:

Input: N = 7, A = 2, B = 6
arr[] = {1, 4, 5, 2, 7, 8, 3}
Output: No
Explanation: Array does not contain 6.

Note: If the array contains all elements in the given range then driver code outputs Yes otherwise, it outputs No

Expected Time Complexity: O(N).
Expected Auxiliary Space: O(1).
Constraints:
1 ≤ N ≤ 10^7

7] Form largest number from digits

Given an array of numbers from 0 to 9 of size N. Your task is to rearrange elements of the array such that after combining all the elements of the array, the number formed is maximum.

Example 1:

Input:
N = 5
A[] = {9, 0, 1, 3, 0}
Output:
93100
Explanation:
Largest number is 93100 which can be formed from array digits.

Example 2:

Input:
N = 3
A[] = {1, 2, 3}
Output:
321

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

Constraints:
1 <= N <= 10^7
0 <= Ai <= 9

8] Even occurring elements

Given an array Arr of N integers that contains an odd number of occurrences for all numbers except for a few elements which are present even number of times. Find the elements which have even occurrences in the array.

Example 1:

Input:
N = 11
Arr[] = {9, 12, 23, 10, 12, 12,
15, 23, 14, 12, 15}
Output: 12 15 23

Example 2:

Input:
N = 5
Arr[] = {23, 12, 56, 34, 32}
Output: -1
Explanation:
Every integer is present odd number of times.

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

Constraints:
1 ≤ N ≤ 10^5
0 ≤ Arr[i] ≤ 63

9] Remove an Element at Specific Index from an Array

Given an array of a fixed length. The task is to remove an element at a specific index from the array.

Examples 1:

> **Input:** arr[] = { 1, 2, 3, 4, 5 }, index = 2
> **Output:** arr[] = { 1, 2, 4, 5 }

Examples 2:

> **Input:** arr[] = { 4, 5, 9, 8, 1 }, index = 3
> **Output:** arr[] = { 4, 5, 9, 1 }

10] Max Odd Sum

Given an array of integers, check whether there is a subsequence with odd sum and if yes, then find the maximum odd sum. If no subsequence contains an odd sum, print -1.

Example 1:

> **Input:**
> **N**=4
> **arr[]** = {4, -3, 3, -5}
> **Output:** 7
> **Explanation:**
> The subsequence with maximum odd
> sum is 4 + 3 = 7

Example 2:

> **Input:**
> **N**=5
> **arr**[] = {2, 5, -4, 3, -1}
> **Output**: 9
> **Explanation:**
> The subsequence with maximum odd
> sum is 2 + 5 + 3 + -1 = 9

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

Constraints:
2 ≤ N ≤ 10^7
-10^3 <= arr[i] <= 10^3

11] Product of maximum in first array and minimum in second

Given two arrays of A and B respectively of sizes N1 and N2, the task is to calculate the product of the maximum element of the first array and minimum element of the second array.

Example 1:

Input : A[] = {5, 7, 9, 3, 6, 2},
B[] = {1, 2, 6, -1, 0, 9}
Output : -9
Explanation:
The first array is 5 7 9 3 6 2.
The max element among these elements is 9.
The second array is 1 2 6 -1 0 9.
The min element among these elements is -1.
The product of 9 and -1 is 9*-1=-9.

Example 2:

Input : A[] = {0, 0, 0, 0},
B[] = {1, -1, 2}
Output : 0

Expected Time Complexity: O(N + M).
Expected Auxiliary Space: O(1).

Output:
For each test case, output the product of the max element of the first array and the minimum element of the second array.

Constraints:
1 ≤ N, M ≤ 10^6

$-10^8 \le A_i, B_i \le 10^8$

12] First and last occurrences of X

Given a sorted array having N elements, find the indices of the first and last occurrences of an element X in the given array.

Note: If the number X is not found in the array, return '-1' as an array.

Example 1:

Input:
N = 4 , **X** = 3
arr[] = { 1, 3, 3, 4 }
Output:
1 2
Explanation:
For the above array, first occurance of X = 3 is at index = 1 and last occurrence is at index = 2.

Example 2:

Input:
N = 4, **X** = 5
arr[] = { 1, 2, 3, 4 }
Output:
-1
Explanation:
As 5 is not present in the array, so the answer is -1.

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

Constraints:
1 <= N <= 10^5

0 <= arr[i], X <= 10^9

13] Find unique element

Given an array of size n which contains all elements occurring in multiples of K, except one element which doesn't occur in multiple of K. Find that unique element.

Example 1:

Input :
n = 7, **k** = 3
arr[] = {6, 2, 5, 2, 2, 6, 6}
Output :
5
Explanation:
Every element appears 3 times except 5.

Example 2:

Input :
n = 5, **k** = 4
arr[] = {2, 2, 2, 10, 2}
Output :
10
Explanation:
Every element appears 4 times except 10.

Expected Time Complexity: O(N. Log(A[i]))
Expected Auxiliary Space: O(Log(A[i]))

Constraints:
3<= N<=2*10^5
2<= K<=2*10^5
1<= A[i]<=10^9

14] Maximum repeating number

Given an array Arr of size N, the array contains numbers in range from 0 to K-1 where K is a positive integer and K <= N. Find the maximum repeating number in this array. If there are two or more maximum repeating numbers return the element having least value.

Example 1:

> **Input:**
> **N** = 4, **K** = 3
> **Arr**[] = {2, 2, 1, 2}
> **Output**: 2
> **Explanation**: 2 is the most frequent element.

Example 2:

> **Input:**
> **N** = 6, **K** = 3
> **Arr[]** = {2, 2, 1, 0, 0, 1}
> **Output:** 0
> **Explanation:** 0, 1 and 2 all have the same frequency of 2.But since 0 is smallest, you need to return 0.

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

Constraints:
1 <= N <= 10^7
1 <= K <= N
0 <= Arri <= K - 1

15] Sum of distinct elements

You are given an array Arr of size N. Find the sum of distinct elements in an array.

Example 1:

Input:
N = 5
Arr[] = {1, 2, 3, 4, 5}
Output: 15
Explanation: Distinct elements are 1, 2, 3
4, 5. So the sum is 15.

Example 2:

Input:
N = 5
Arr[] = {5, 5, 5, 5, 5}
Output: 5
Explanation: Only Distinct element is 5.
So the sum is 5.

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

Constraints:
1 ≤ N ≤ 10^7
0 ≤ A[i] ≤ 10^4

16] Last index of One

Given a string S consisting only '0's and '1's, find the last index of the '1' present in it.

Example 1:

Input:
S = 00001
Output:
4
Explanation:
Last index of 1 in the given string is 4.

Example 2:

Input:
0
Output:
-1
Explanation:
Since, 1 is not present, so output is -1.

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

Constraints:
1 <= |S| <= 10^6
S = {0,1}

17] Product of maximum in first array and minimum in second

Given two arrays of A and B respectively of sizes N1 and N2, the task is to calculate the product of the maximum element of the first array and minimum element of the second array.

Example 1:

Input : A[] = {5, 7, 9, 3, 6, 2},
B[] = {1, 2, 6, -1, 0, 9}
Output : -9
Explanation:
The first array is 5 7 9 3 6 2.
The max element among these elements is 9.
The second array is 1 2 6 -1 0 9.
The min element among these elements is -1.
The product of 9 and -1 is 9*-1=-9.

Example 2:

Input : A[] = {0, 0, 0, 0},
B[] = {1, -1, 2}
Output : 0

Expected Time Complexity: O(N + M).
Expected Auxiliary Space: O(1).

Output:
For each test case, output the product of the max element of the first array and the minimum element of the second array.

Constraints:
1 ≤ N, M ≤ 10^6

-10^8 ≤ Ai, Bi ≤ 10^8

18] Find Subarray with given sum | Set 1 (Non-negative Numbers)

Given an array arr[] of non-negative integers and an integer sum, find a subarray that adds to a given sum.

Note: There may be more than one subarray with sum as the given sum, print first such subarray.

Examples:

Input: arr[] = {1, 4, 20, 3, 10, 5}, sum = 33
Output: Sum found between indexes 2 and 4
Explanation: Sum of elements between indices 2 and 4 is 20 + 3 + 10 = 33

Input: arr[] = {1, 4, 0, 0, 3, 10, 5}, sum = 7
Output: Sum found between indexes 1 and 4
Explanation: Sum of elements between indices 1 and 4 is 4 + 0 + 0 + 3 = 7

Input: arr[] = {1, 4}, sum = 0
Output: No subarray found
Explanation: There is no subarray with 0 sum

19] Find common elements in three sorted arrays

Given three Sorted arrays in non-decreasing order, print all common elements in these arrays.

Examples:

Input:
ar1[] = {1, 5, 10, 20, 40, 80}
ar2[] = {6, 7, 20, 80, 100}
ar3[] = {3, 4, 15, 20, 30, 70, 80, 120}
Output: 20, 80

Input:
ar1[] = {1, 5, 5}
ar2[] = {3, 4, 5, 5, 10}
ar3[] = {5, 5, 10, 20}
Output: 5, 5

20] Check if pair with given Sum exists in Array (Two Sum)

Given an array A[] of n numbers and another number x, the task is to check whether or not there exist two elements in A[] whose sum is exactly x.

Examples:

Input: arr[] = {0, -1, 2, -3, 1}, x= -2
Output: Yes
Explanation: If we calculate the sum of the output,1 + (-3) = -2

Input: arr[] = {1, -2, 1, 0, 5}, x = 0
Output: No

21] First element to occur k times

Given an array of N integers. Find the first element that occurs at least K number of times.

Example 1:

Input :
N = 7, K = 2
A[] = {1, 7, 4, 3, 4, 8, 7}
Output :
4
Explanation:
Both 7 and 4 occur 2 times.
But 4 is first that occurs 2 times
As at index = 4, 4 has occurred
at least 2 times whereas at index = 6,
7 has occurred at least 2 times.

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

Constraints:
1 <= N <= 10^4
1 <= K <= 100
1<= A[i] <= 200

22] Exceptionally odd

Given an array of N positive integers where all numbers occur even number of times except one number which occurs odd number of times. Find the exceptional number.

Example 1:

Input:
N = 7
Arr[] = {1, 2, 3, 2, 3, 1, 3}
Output: 3
Explanation: 3 occurs three times.

Example 2:

Input:
N = 7
Arr[] = {5, 7, 2, 7, 5, 2, 5}
Output: 5
Explanation: 5 occurs three times.

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

Constraints:
1 ≤ N ≤ 10^5
1 ≤ arr[i] ≤ 10^6

23] Find the smallest and second smallest element in an array

Given an array of integers, your task is to find the smallest and second smallest element in the array. If smallest and second smallest do not exist, print -1.

Example 1:

Input :
5
2 4 3 5 6
Output :
2 3
Explanation:
2 and 3 are respectively the smallest
and second smallest elements in the array.

Example 2:

Input :
6
1 2 1 3 6 7
Output :
1 2
Explanation:
1 and 2 are respectively the smallest
and second smallest elements in the array.

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

Constraints:
1<=N<=10^5
1<=A[i]<=10^5

24] Ceil The Floor

Given an unsorted array Arr[] of N integers and an integer X, find floor and ceiling of X in Arr[0..N-1].
Floor of X is the largest element which is smaller than or equal to X. Floor of X doesn't exist if X is smaller than the smallest element of Arr[].
Ceil of X is the smallest element which is greater than or equal to X. Ceil of X doesn't exist if X is greater than the greatest element of Arr[].

Example 1:

Input:
N = 8, **X** = 7
Arr[] = {5, 6, 8, 9, 6, 5, 5, 6}
Output: 6 8
Explanation:
Floor of 7 is 6 and ceil of 7 is 8.

Example 2:

Input:
N = 8, **X** = 10
Arr[] = {5, 6, 8, 9, 6, 5, 5, 6}
Output: 9 -1

Explanation:
Floor of 10 is 9 but ceil of 10 is not possible.

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

Constraints :
1 ≤ N ≤ 10^5

1 ≤ Arr[i], X ≤ 10^6

25] Maximum product of two numbers

Given an array Arr of size N with all elements greater than or equal to zero. Return the maximum product of two numbers possible.

Example 1:

Input:
N = 6
Arr[] = {1, 4, 3, 6, 7, 0}
Output: 42

Example 2:

Input:
N = 5
Arr = {1, 100, 42, 4, 23}
Output: 4200

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

Constraints:
2 ≤ N ≤ 10^7
0 ≤ Arr[i] ≤ 10^4

26] Positive and negative elements

Given an array arr[ ] containing equal number of positive and negative elements, arrange the array such that every positive element is followed by a negative element.
Note- The relative order of positive and negative numbers should be maintained.

Example 1:

> **Input**:
> **N** = 6
> **arr**[] = {-1, 2, -3, 4, -5, 6}
> **Output:**
> 2 -1 4 -3 6 -5
> **Explanation:** Positive numbers in order are 2, 4 and 6. Negative numbers in order are -1, -3 and -5. So the arrangement we get is 2, -1, 4, -3, 6 and -5.

Example 2:

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

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

Constraints:
1 ≤ N ≤ 10^6
1 ≤ arr[i] ≤ 10^9

27] Count pair sum

Given two sorted arrays(arr1[] and arr2[]) of size M and N of distinct elements. Given a value Sum. The problem is to count all pairs from both arrays whose sum is equal to Sum.
Note: The pair has an element from each array.

Example 1:

Input:
M=4, **N**=4 , **Sum** = 10
arr1[] = {1, 3, 5, 7}
arr2[] = {2, 3, 5, 8}
Output: 2
Explanation: The pairs are: (5, 5) and (7, 3).

Example 2:

Input:
N=4, **M**=4, **sum**=5
arr1[] = {1, 2, 3, 4}
arr2[] = {5, 6, 7, 8}
Output: 0

Expected Time Complexity: O(M+N).
Expected Auxiliary Space: O(1).

Constraints:
1 ≤ M, N ≤ 10^5

28] Remove Duplicates from unsorted array

Given an array of integers which may or may not contain duplicate elements. Your task is to remove duplicate elements, if present.

Example 1:

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

Example 2:

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

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

Constraints:
1<=N<=10^5
1<=A[i]<=10^5

29] Last index of One

Given a string S consisting only of '0's and '1's, find the last index of the '1' present in it.

Example 1:

> **Input:**
> **S** = 00001
> **Output:**
> 4
> **Explanation**:
> Last index of 1 in the given string is 4.

Example 2:

> **Input:**
> 0
> **Output:**
> -1
> **Explanation:**
> Since, 1 is not present, so output is -1.

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

Constraints:
1 <= |S| <= 10^6
S = {0,1}

30] Find the closest number

Given an array of sorted integers. The task is to find the closest value to the given number in the array. Arrays may contain duplicate values.

Note: If the difference is same for two values print the value which is greater than the given number.

Example 1:

Input : Arr[] = {1, 3, 6, 7} and K = 4
Output : 3
Explanation:
We have an array [1, 3, 6, 7] and the target is 4. If we look at the absolute difference of target with every element of an array we will get [|1-4|, |3-4|, |6-4|, |7-4|].
So, the closest number is 3.

Example 2:

Input : Arr[] = {1, 2, 3, 5, 6, 8, 9}, **K** = 4
Output : 5

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

Constraints:
1 ≤ N ≤ 10^5
1 ≤ K ≤ 10^5
1 ≤ A[i] ≤ 10^5

31] Sum of f(a[i], a[j]) over all pairs in an array of n integers

Given an array A of n integers, find the sum of f(a[i], a[j]) of all pairs (i, j) such that (1 <= i < j <= n).

```
f(a[i], a[j]):    if abs(a[j]-a[i]) > 1
                  f(a[i], a[j]) = a[j] - a[i]
                  else  if abs(a[j]-a[i]) <= 1
                  f(a[i], a[j]) = 0
```

Example 1:

Input : arr[ ] = {6, 6, 4, 4}
Output : -8
Explanation:
All pairs are:
(6 - 6) + (4 - 6) + (4 - 6) +
(4 - 6) + (4 - 6) + (4 - 4) = -8
return -8.

Example 2:

Input : arr[ ] = {1, 2, 3, 1, 3}
Output: 4
Explanation: All pairs are:
(3-1) + (1-3) + (3-1) + (3-1) = 4
(1-2),(3-2),(1-1),(2-1),(3-3),(3-2) these pairs will give zero as their absolute difference is <= 1

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

Constraints:
1 ≤ N ≤ 10^5

1 ≤ a[i] ≤ 10^7

32] Maximize sum(arr[i]*i) of an Array

Given an array A of N integers. Your task is to write a program to find the maximum value of ∑arr[i]*i, where i = 0, 1, 2,., n 1.
You are allowed to rearrange the elements of the array.
Note: Since output could be large, hence module 109+7 and then print answer.

Example 1:

Input : Arr[] = {5, 3, 2, 4, 1}
Output : 40
Explanation:
If we arrange the array as 1 2 3 4 5 then we can see that the minimum index will multiply with minimum number and maximum index will multiply with maximum number.
So 1*0+2*1+3*2+4*3+5*4=0+2+6+12+20 = 40 mod(109+7) = 40

Example 2:

Input : Arr[] = {1, 2, 3}
Output : 8

Expected Time Complexity: O(nlog(n)).
Expected Auxiliary Space: O(1).

Constraints:
1 ≤ N ≤ 10^7
1 ≤ Ai ≤ N

33] Multiply left and right array sum.

Pitsy needs help with the given task by her teacher. The task is to divide an array into two sub-array (left and right) containing n/2 elements each and do the sum of the subarrays and then multiply both the subarrays.

Note: If the length of the array is odd then the right half will contain one element more than the left half.

Example 1:

> **Input** : arr[ ] = {1, 2, 3, 4}
> **Output** : 21
> **Explanation**:
> Sum up an array from index 0 to 1 = 3. Sum up an array from index 2 to 3 = 7. Their multiplication is 21.

Example 2:

> **Input** : arr[ ] = {1, 2}
> **Output** : 2

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

Constraints:
$1 \leq T \leq 100$
$1 \leq N \leq 1000$
$1 \leq A[i] \leq 100$

34] Print an array in Pendulum Arrangement

Given an array arr of size n. Arrange the elements of the array in a way similar to the to-and-fro movement of a Pendulum.

The minimum element out of the list of integers, must come in the center position of the array. If there are even elements, then minimum element should be moved to (n-1)/2 index (considering that indexes start from 0)
The next number (next to minimum) in the ascending order, goes to the right, the next to next number goes to the left of the minimum number and it continues like a Pendulum.
As higher numbers are reached, one goes to one side in a to-and-fro manner similar to that of a Pendulum

Example 1:

Input :
n = 5
arr[] = {1, 3, 2, 5, 4}
Output :
5 3 1 2 4
Explanation:
The minimum element is 1, so it is moved to the middle. The next higher element 2 is moved to the right of the middle element while the next higher element 3 is moved to the left of the middle element and this process is continued.

Example 2:

Input :
n = 5
arr[] = {11, 12, 31, 14, 5}
Output :
31 12 5 11 14

Expected Time Complexity: O(n. Log(n))
Expected Auxiliary Space: O(n)

Constraints:

1 <= n <= 10^5
0 <= arr[i] <= 10^5

35] Minimum Product of k Integers

Given an array of N positive integers. You need to write a program to print the minimum product of k integers of the given array.

Note: Since output could be large, hence module 10^9+7 and then print answer.

Example 1:

Input : Arr[] = {1, 2, 3, 4, 5}, K = 2
Output : 2
Explanation:
If we have an array [1, 2, 3, 4, 5]. We will get the minimum product after multiplying 1 and 2 that is 2.
So, the answer is 2.

Example 2:

Input : Arr[] = {9, 10, 8}, K = 3
Output : 720

Expected Time Complexity: O(nlogn).
Expected Auxiliary Space: O(1).

Constraints:
1 ≤ K, N ≤ 10^5

36] Find a peak element which is not smaller than its neighbors

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.

Examples:

Input: array[]= {5, 10, 20, 15}
Output: 20
Explanation: The element 20 has neighbors 10 and 15, both of them are less than 20.

Input: array[] = {10, 20, 15, 2, 23, 90, 67}
Output: 20 or 90
Explanation: The element 20 has neighbors 10 and 15, both of them are less than 20, similarly 90 has neighbors 23 and 67.

37] Move all negative numbers to beginning and positive to end with constant extra space

An array contains both positive and negative numbers in random order. Rearrange the array elements so that all negative numbers appear before all positive numbers.

Examples :

Input: -12, 11, -13, -5, 6, -7, 5, -3, -6
Output: -12 -13 -5 -7 -3 -6 11 6 5

38] Immediate Smaller Element

Given an integer array Arr of size N. For each element in the array, check whether the right adjacent element (on the next immediate position) of the array is smaller. If the next element is smaller, update the current index to that element. If not, then -1.

Example 1:

Input:
N = 5
Arr[] = {4, 2, 1, 5, 3}
Output:
2 1 -1 3 -1
Explanation: Array elements are 4, 2, 1, 5, 3. Next to 4 is 2 which is smaller, so we print 2. Next of 2 is 1 which is smaller, so we print 1. Next of 1 is 5 which is greater, so we print -1. Next of 5 is 3 which is smaller, so we print 3. Note that for the last element, output is always going to be -1 because there is no element on the right.

Example 2:

Input:
N = 6
Arr[] = {5, 6, 2, 3, 1, 7}
Output:
-1 2 -1 1 -1 -1
Explanation: Next to 5 is 6 which is greater, so we print -1.Next of 6 is 2 which is smaller, so we print 2. Next of 2 is 3 which is greater, so we print -1. Next of 3 is 1 which is smaller, so we print 1. Next of 1 is 7 which is greater, so we print -1.
Note that for the last element, output is always going to be -1 because there is no element on the right.

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

Constraints:
1 ≤ N ≤ 10^7
1 ≤ Arr[i] ≤ 10^5

39] Leaders in an array

Write a program to print all the LEADERS in the array. An element is a leader if it is greater than all the elements to its right side. And the rightmost element is always a leader.

For example:

Input: arr[] = {16, 17, 4, 3, 5, 2},
Output : 17, 5, 2

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

40] Fibonacci in the array

Given an array arr of size N, the task is to count the number of elements of the array which are Fibonacci numbers

Example 1:

Input: N = 9, arr[] = {4, 2, 8, 5, 20, 1,
40, 13, 23}
Output: 5
Explanation: Here, Fibonacci series will be 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55. Numbers that are present in array are 2, 8, 5, 1, 13

Example 2:

Input: N = 4, arr[] = {4, 7, 6, 25}
Output: 0
Explanation: No Fibonacci number in this array.

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

Constraints:
1 ≤ N ≤ 10^6

41]Countries at war

The two countries of A and B are at war against each other. Both countries have N number of soldiers. The power of these soldiers are given by A[i]...A[N] and B[i]....B[N].
These soldiers have a peculiarity. They can only attack their counterpart enemies, like A[i] can attack only B[i] and not anyone else. A soldier with higher power can kill the enemy soldier. If both soldiers have the same power, they both die. You need to find the winning country.

Example 1:

> **Input** : a[ ] = {2, 2}, b[ ] = {5, 5}
> **Output** : B
> **Explanation**:
> Both countries have 2 soldiers.
> B[0] kills A[0], B[1] kills A[1].
> A has 0 soldiers alive at the end.
> B has both soldiers alive at the end.
> Return "B" as a winner.

Example 2:

> **Input** : a[ ] = {9}, b[ ] = {8}
> **Output** : A

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

Constraints:
1 ≤ N ≤ 10^5
0 ≤ Ai ≤ 10^7
0 ≤ Bi ≤ 10^7

42] Count the number of elements between two given elements in an array. Given an unsorted array and two elements num1 and num2.

The task is to count the number of elements occurring between the given elements (excluding num1 and num2). If there are multiple occurrences of num1 and num2, we need to consider the leftmost occurrence of num1 and rightmost occurrence of num2.

Example 1:

Input : Arr[] = {4, 2, 1, 10, 6}

num1 = 4 and num2 = 6

Output : 3

Explanation:

We have an array [4, 2, 1, 10, 6] and num1 = 4 and num2 = 6. So, the leftmost index of num1 is 0 and the rightmost index of num2 is 4. So, the total number of elements between them is [2, 1, 10] So, the function will return 3 as an answer.

Example 2:

Input : Arr[] = {3, 2, 1, 4}

num1 = 2 and **num2** = 4

Output : 1

Expected Time Complexity: O(N).

Expected Auxiliary Space: O(1).

Constraints:

2 ≤ N ≤ 10^5

1 ≤ A[i], num1, num2 ≤ 10^5

43] Count number of elements between two given elements in array

Given an unsorted array and two elements num1 and num2. The task is to count the number of elements occurring between the given elements (excluding num1 and num2). If there are multiple occurrences of num1 and num2, we need to consider the leftmost occurrence of num1 and rightmost occurrence of num2.

Example 1:

Input : Arr[] = {4, 2, 1, 10, 6}
num1 = 4 and num2 = 6
Output : 3
Explanation:
We have an array [4, 2, 1, 10, 6] and
num1 = 4 and **num2** = 6.
So, the leftmost index of num1 is 0 and the rightmost index of num2 is 4.
So, the total number of elements between them is [2, 1, 10] So, the function will return 3 as an answer.

Example 2:

Input : Arr[] = {3, 2, 1, 4}
num1 = 2 and num2 = 4
Output : 1

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

Constraints:
2 ≤ N ≤ 10^5
1 ≤ A[i], num1, num2 ≤ 10^5