Department of Computer Science and Engg.- NITK Surathkal

CS-203 Data Structures and Algorithms

Assignment sheet- 1 Due Date: 5th August 2024

1.Subarray with given sum

Given an unsorted array **A** of size **N** that contains only non-negative integers, find a continuous sub-array which adds to a given number **S**.

In case of multiple subarrays, return the subarray which comes first on moving from left to right.

Example 1:

Input:

N = 5, S = 12

A[] = {1,2,3,7,5}

Output: 2 4

Explanation: The sum of elements from 2nd position to 4th position is 12.

Example 2:

Input:

N = 10, S = 15

A[] = {1,2,3,4,5,6,7,8,9,10}

Output: 1 5

Explanation: The sum of elements from 1st position to 5th position is 15.

Your Task:

The task is to complete the function **subarraySum**() which takes array, N and S as input parameters and returns an **arraylist** containing the **starting** and **ending** positions of the first such occurring subarray from the left where sum equals to S. If no such subarray is found, return an array consisting only one element that is -1.

Expected Time Complexity: O(N)

Expected Auxiliary Space: O(1)

Constraints:

$1 <= N <= 10^5$

$1 <= A_i <= 10^{9.}$

2. Count the triplets

Given an array of distinct integers. The task is to count all the triplets such that sum of two elements equals the third element.

Example 1:

Input:

N = 4

arr[] = {1, 5, 3, 2}

Output: 2

Explanation: There are 2 triplets: 1 + 2 = 3 and 3 +2 = 5

Example 2:

Input:
N = 3
arr[] = {2, 3, 4}
Output: 0
Explanation: No such triplet exits

Your Task:
Your task is to complete the function **countTriplet()** which takes the array **arr[]** and **N** as inputs and returns the triplet count

Expected Time Complexity: $O(N^2)$
Expected Auxiliary Space: O(1)

Constraints:
$1 \le N \le 10^3$
$1 \le arr[i] \le 10^5$

3. 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 = 5
A[] = {1,2,3,5}
Output: 4

Example 2:

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

Your Task :
Complete the function **MissingNumber()** that takes array and N as input parameters and returns the value of the missing number.

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

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

4. Count Inversions

Given an array of integers. Find the Inversion Count in the array.

Inversion Count: For an array, inversion count indicates how far (or close) the array is from being sorted. If array is already sorted then the inversion count is 0. If an array is sorted in the reverse order then the inversion count is the maximum.

Formally, two elements a[i] and a[j] form an inversion if a[i] > a[j] and i < j.

Example 1:

Input: N = 5, arr[] = {2, 4, 1, 3, 5}
Output: 3
Explanation: The sequence 2, 4, 1, 3, 5 has three inversions (2, 1), (4, 1), (4, 3).

Example 2:

Input: N = 5
arr[] = {2, 3, 4, 5, 6}
Output: 0
Explanation: As the sequence is already sorted so there is no inversion count.

Example 3:

Input: N = 3, arr[] = {10, 10, 10}
Output: 0
Explanation: As all the elements of array are same, so there is no inversion count.

Your Task:

Your task is to complete the function **inversionCount()** which takes the array arr[] and the size of the array as inputs and returns the inversion count of the given array.

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

Constraints:
$1 \leq N \leq 5*10^5$
$1 \leq arr[i] \leq 10^{18}$

5. Leaders in an array

Given an array A of positive integers. Your task is to find the leaders in the array. An element of 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

Your Task:
The task is to complete the function **leader**() which takes array A and n as input parameters and returns an array of leaders in order of their appearance.

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

Constraints:
$1 <= n <= 10^7$
$0 <= A_i <= 10^7$

6. Pythagorean Triplet

Given an array **arr** of **N** integers, write a function that returns true if there is a triplet (a, b, c) that satisfies $\mathbf{a^2 + b^2 = c^2}$, otherwise false.

Example 1:

Input:
N = 5
Arr[] = {3, 2, 4, 6, 5}
Output: Yes
Explanation: a=3, b=4, and c=5 forms a pythagorean triplet.

Example 2:

Input:
N = 3
Arr[] = {3, 8, 5}
Output: No
Explanation: No such triplet possible.

Your Task:
Complete the function **checkTriplet()** which takes an array **arr,** single integer **n,** as input parameters and returns boolean denoting answer to the problem.

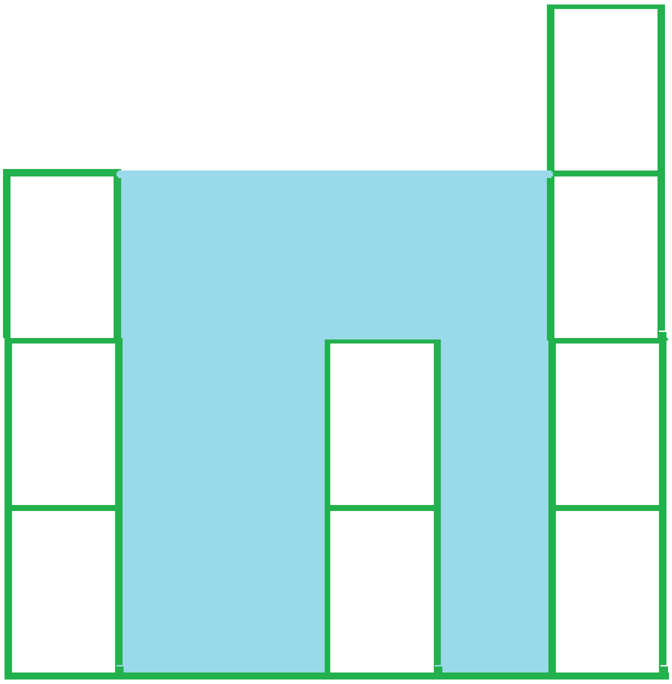
Expected Time Complexity: O(max(Arr[i])2)
Expected Auxiliary Space: O(max(Arr[i]))
Constraints:
$1 <= N <= 10^7$
1 <= Arr[i] <= 1000

7. Trapping Rain water

Given an array **arr[]** of **N** non-negative integers representing the height of blocks. If width of each block is 1, compute how much water can be trapped between the blocks during the rainy season.

Example 1:

Input:
N = 6
arr[] = {3,0,0,2,0,4}
Output:
10
Explanation:



Bars for input {3, 0, 0, 2, 0, 4}
Total trapped water = 3 + 3 + 1 + 3 = 10

Example 2:

Input:
N = 4
arr[] = {7,4,0,9}
Output:
10
Explanation:
Water trapped by above block of height 4 is 3 units and above block of height 0 is 7 units. So, the total unit of water trapped is 10 units.

Example 3:

Input:

N = 3

arr[] = {6,9,9}

Output:

0

Explanation:

No water will be trapped.

Your Task:

The task is to complete the function **trappingWater**() which takes arr[] and N as input parameters and returns the total amount of water that can be trapped.

Expected Time Complexity: O(N)

Expected Auxiliary Space: O(N)

Constraints:

$3 \leq N \leq 10^6$

$0 \leq Ai \leq 10^8$

8. Chocolate Distribution Problem

Given an array **A[ ]** of positive integers of size **N**, where each value represents the number of chocolates in a packet. Each packet can have a variable number of chocolates. There are **M** students, the task is to distribute chocolate packets among **M** students such that :

1. Each student gets **exactly** one packet.
2. The difference between maximum number of chocolates given to a student and minimum number of chocolates given to a student is minimum.

Example 1:

Input:

N = 8, M = 5

A = {3, 4, 1, 9, 56, 7, 9, 12}

Output: 6

Explanation: The minimum difference between maximum chocolates and minimum chocolates is 9 - 3 = 6 by choosing following M packets : {3, 4, 9, 7, 9}.

Example 2:

Input:

N = 7, M = 3

A = {7, 3, 2, 4, 9, 12, 56}

Output: 2

Explanation: The minimum difference between maximum chocolates and minimum chocolates is 4 - 2 = 2 by choosing following M packets : {3, 2, 4}.

Your Task:
Your task is to complete the function **findMinDiff()** which takes array A[ ], N and M as input parameters and returns the minimum possible difference between maximum number of chocolates given to a student and minimum number of chocolates given to a student.
Expected Time Complexity: O(N*Log(N))
Expected Auxiliary Space: O(1)
Constraints:
$1 \leq T \leq 100$
$1 \leq N \leq 10^5$
$1 \leq Ai \leq 10^9$
$1 \leq M \leq N$

9. Convert array into Zig-Zag fashion

Given an array **arr** of distinct elements of size **N**, the task is to rearrange the elements of the array in a zig-zag fashion so that the converted array should be in the below form:

arr[0] < arr[1] > arr[2] < arr[3] > arr[4] < . . . . arr[n-2] < arr[n-1] > arr[n].

Example 1:

Input:
N = 7
Arr[] = {4, 3, 7, 8, 6, 2, 1}
Output: 3 7 4 8 2 6 1
Explanation: 3 < 7 > 4 < 8 > 2 < 6 > 1

Example 2:

Input:
N = 4
Arr[] = {1, 4, 3, 2}
Output: 1 4 2 3
Explanation: 1 < 4 > 2 < 3

Your Task:
Your task is to complete the function **zigZag()** which takes the array of integers **arr** and **n** as parameters and returns void. You need to modify the array itself.
NOTE: In the mentioned complexity, only a unique solution will exist.

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

Constraints:
$1 <= N <= 10^5$
$0 <= Arr[i] <= 10^6$

10. Largest Number formed from an Array

Given a list of non negative integers, arrange them in such a manner that they form the largest number possible.The result is going to be very large, hence return the result in the form of a string.

Example 1:

Input:
N = 5
Arr[] = {3, 30, 34, 5, 9}
Output: 9534330
Explanation: Given numbers are {3, 30, 34, 5, 9}, the arrangement 9534330 gives the largest value.

Example 2:

Input:
N = 4
Arr[] = {54, 546, 548, 60}
Output: 6054854654
Explanation: Given numbers are {54, 546, 548, 60}, the arrangement 6054854654 gives the largest value.

Your Task: Your task is to complete the function **printLargest()** which takes the **array of strings arr[]** as parameter and returns a string denoting the answer.

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

Constraints:
$1 \leq N \leq 10^5$
$0 \leq Arr[i] \leq 10^{18}$
Sum of all the elements of the array is greater than 0.

11.

(a) Write a program to reverse an array in O(N) time complexity, where N is the number of elements in the array.
(b) Can you propose an algorithm that doesn’t use any extra space.

12. Given an array of size **n**, generate and print all possible combinations of **r** elements in array. For example, if input array is 1, 2, 3, 4 and **r** is 2, then output should be {1, 2}, {1, 3}, {1, 4},{2, 3}, {2, 4} and {3, 4}.

13. Given a sorted array of n numbers, describe $\Theta(n)$-algorithm that, given another number **x**, determines whether or not there exist two elements in the input array whose sum is exactly **x.** Implement the above algorithm using C.