Question Number 06:

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 atleast 2 times whereas at index = 6, 7 has occurred atleast 2 times.

Your Task:
You don't need to read input or print anything. Your task is to complete the function firstElementKTime() which takes the array A[], its size N, and an integer K as inputs and returns the required answer. If the answer is not present in the array, return -1.

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

Constraints:
1 <= N <= 104
1 <= K <= 100
1<= A[i] <= 200
Output: 4

Question Number 07:

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].

NOTE: If your transformation is correct, the output will be 1 else the output will be 0.

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:
You don't need to read input or print anything. 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.

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

Constraints:
1 <= N <= 106
0 <= Arri <= 109

Question Number 08:

Subarray with Given Sum

Given an unsorted array A of size N that contains only positive integers, find a continuous sub-array that adds to a given number S and return the left and right index(1-based indexing) of that subarray.

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

Note:- You have to return an ArrayList consisting of two elements left and right. In case no such subarray exists return an array consisting of element -1.

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:

You don't need to read input or print anything. The task is to complete the function subarraySum() which takes arr, 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. The two indexes in the array should be according to 1-based indexing. If no such subarray is found, return an array consisting of only one element that is -1.

Expected Time Complexity: O(N)

Expected Auxiliary Space: O(1)

Constraints:

1 <= N <= 105

1 <= Ai <= 109

0<= S <= 1014

Question Number 09:

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:

You don't need to read input or print anything. 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 <= 107

0 <= Ai <= 107

Question Number 10:

Largest Subarray with 0 Sum

Given an array having both positive and negative integers. The task is to compute the length of the largest subarray with sum 0.

Example 1:

Input:

N = 8

A[] = {15,-2,2,-8,1,7,10,23}

Output: 5

Explanation: The largest subarray with
sum 0 will be -2 2 -8 1 7.

Your Task:

You just have to complete the function maxLen() which takes two arguments an array A and n, where n is the size of the array A and returns the length of the largest subarray with 0 sum.

Expected Time Complexity: O(N).

Expected Auxiliary Space: O(N).

Constraints:

1 <= N <= 105

-1000 <= A[i] <= 1000, for each valid i