**First half - Questions List -**

**Question1 - You are given an m x n integer matrix matrix with the following two properties:**

* **Each row is sorted in non-decreasing order.**
* **The first integer of each row is greater than the last integer of the previous row.**

**Given an integer target, return true *if the target* *is in* matrix *or* false *otherwise*.**

**You must write a solution in O(log(m \* n)) time complexity.**

Example 1:

Input: matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 3

Output: true

Example 2:

Input: matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 13

Output: false

**Question2 - Given an array of string words, return *all strings in* words *that is a substring of another word*. You can return the answer in any order.**

**A substring is a contiguous sequence of characters within a string**

Example 1:

Input: words = ["mass","as","hero","superhero"]

Output: ["as","hero"]

Explanation: "as" is substring of "mass" and "hero" is substring of "superhero".

["hero","as"] is also a valid answer.

Example 2:

Input: words = ["leetcode","et","code"]

Output: ["et","code"]

Explanation: "et", "code" are substrings of "leetcode".

Example 3:

Input: words = ["blue","green","bu"]

Output: []

Explanation: No string of words is a substring of another string.

**Question3 -**

**You are given an integer array nums sorted in non-decreasing order.**

**Build and return *an integer array* result *with the same length as* nums *such that* result[i] *is equal to the summation of absolute differences between* nums[i] *and all the other elements in the array.***

**In other words, result[i] is equal to sum(|nums[i]-nums[j]|) where 0 <= j < nums.length and j != i (0-indexed).**

**Example 1:**

**Input:** nums = [2,3,5]

**Output:** [4,3,5]

**Explanation:** Assuming the arrays are 0-indexed, then

result[0] = |2-2| + |2-3| + |2-5| = 0 + 1 + 3 = 4,

result[1] = |3-2| + |3-3| + |3-5| = 1 + 0 + 2 = 3,

result[2] = |5-2| + |5-3| + |5-5| = 3 + 2 + 0 = 5.

**Example 2:**

**Input:** nums = [1,4,6,8,10]

**Output:** [24,15,13,15,21]

**Constraints:**

* 2 <= nums.length <= 105
* **1 <= nums[i] <= nums[i + 1] <= 104**

**Question 4 - Given an integer array of size n, find all elements that appear more than ⌊ n/3 ⌋ times.**

**Example 1:**

**Input:** nums = [3,2,3]

**Output:** [3]

**Example 2:**

**Input:** nums = [1]

**Output:** [1]

**Example 3:**

**Input:** nums = [1,2]

**Output:** [1,2]

**Constraints:**

* 1 <= nums.length <= 5 \* 104
* -109 <= nums[i] <= 109

**Question5 -**

**Given two integer arrays arr1 and arr2, and the integer d, *return the distance value between the two arrays*.**

**The distance value is defined as the number of elements arr1[i] such that there is not any element arr2[j] where |arr1[i]-arr2[j]| <= d.**

Example 1:

Input: arr1 = [4,5,8], arr2 = [10,9,1,8], d = 2

Output: 2

Explanation:

For arr1[0]=4 we have:

|4-10|=6 > d=2

|4-9|=5 > d=2

|4-1|=3 > d=2

|4-8|=4 > d=2

For arr1[1]=5 we have:

|5-10|=5 > d=2

|5-9|=4 > d=2

|5-1|=4 > d=2

|5-8|=3 > d=2

For arr1[2]=8 we have:

|8-10|=2 <= d=2

|8-9|=1 <= d=2

|8-1|=7 > d=2

|8-8|=0 <= d=2

Example 2:

Input: arr1 = [1,4,2,3], arr2 = [-4,-3,6,10,20,30], d = 3

Output: 2

Example 3:

Input: arr1 = [2,1,100,3], arr2 = [-5,-2,10,-3,7], d = 6

Output: 1

**Question 6 -**

**Given an array of integers nums, sort the array in ascending order and return it.**

**You must solve the problem without using any built-in functions in O(nlog(n)) time complexity and with the smallest space complexity possible.**

Example 1:

Input: nums = [5,2,3,1]

Output: [1,2,3,5]

Explanation: After sorting the array, the positions of some numbers are not changed (for example, 2 and 3), while the positions of other numbers are changed (for example, 1 and 5).

Example 2:

Input: nums = [5,1,1,2,0,0]

Output: [0,0,1,1,2,5]

Explanation: Note that the values of nums are not necessairly unique.

**Constraints:**

* **1 <= nums.length <= 5 \* 104**
* **-5 \* 104 <= nums[i] <= 5 \* 104**

**Question 7 -**

**Given two arrays of strings list1 and list2, find the common strings with the least index sum.**

**A common string is a string that appeared in both list1 and list2.**

**A common string with the least index sum is a common string such that if it appeared at list1[i] and list2[j] then i + j should be the minimum value among all the other common strings.**

**Return *all the common strings with the least index sum*. Return the answer in any order.**

Example 1:

Input: list1 = ["Shogun","Tapioca Express","Burger King","KFC"], list2 = ["Piatti","The Grill at Torrey Pines","Hungry Hunter Steakhouse","Shogun"]

Output: ["Shogun"]

Explanation: The only common string is "Shogun".

Example 2:

Input: list1 = ["Shogun","Tapioca Express","Burger King","KFC"], list2 = ["KFC","Shogun","Burger King"]

Output: ["Shogun"]

Explanation: The common string with the least index sum is "Shogun" with index sum = (0 + 1) = 1.

Example 3:

Input: list1 = ["happy","sad","good"], list2 = ["sad","happy","good"]

Output: ["sad","happy"]

Explanation: There are three common strings:

"happy" with index sum = (0 + 1) = 1.

"sad" with index sum = (1 + 0) = 1.

"good" with index sum = (2 + 2) = 4.

The strings with the least index sum are "sad" and "happy".

**Question 8 -**

**Given two strings s and t, *determine if they are isomorphic*.**

**Two strings s and t are isomorphic if the characters in s can be replaced to get t.**

**All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character, but a character may map to itself.**

Example 1:

Input: s = "egg", t = "add"

Output: true

Example 2:

Input: s = "foo", t = "bar"

Output: false

Example 3:

Input: s = "paper", t = "title"

Output: true