Assignment 1
Searching & Sorting

Total 60 points
Each question 10 marks (8 code , 2 points Time and Space Complexity)

Question 1: Given an array `nums` with `n` objects colored red, white, or blue, sort them **in-place** so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers `0`, `1`, and `2` to represent the color red, white, and blue, respectively.

You must solve this problem without using the library's sort function.

Example 1:

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

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

Example 2:

Input: nums = [2,0,1]

Output: [0,1,2]

Example 3:

Input: nums = [0]

Output: [0]

Example 4:

Input: nums = [1]

Output: [1]

Constraints:

- `n == nums.length`

- $1 <= n <= 300$
- nums[i] is 0, 1, or 2.

Question 2: Given an array of meeting time intervals where intervals[i] = [$start_i$, end_i], determine if a person could attend all meetings.

Example 1:

Input: intervals = [[0,30],[5,10],[15,20]]

Output: false

Example 2:

Input: intervals = [[7,10],[2,4]]

Output: true

Constraints:

- $0 <= intervals.length <= 10^4$
- intervals[i].length == 2
- $0 <= start_i < end_i <= 10^6$

Question 3: Given an integer array nums of 2n integers, group these integers into n pairs $(a_1, b_1), (a_2, b_2), ..., (a_n, b_n)$ such that the sum of $min(a_i, b_i)$ for all i is **maximized**. Return *the maximized sum*.

Example 1:

Input: nums = [1,4,3,2]

Output: 4

Explanation: All possible pairings (ignoring the ordering of elements) are:

1. (1, 4), (2, 3) -> min(1, 4) + min(2, 3) = 1 + 2 = 3

2. (1, 3), (2, 4) -> min(1, 3) + min(2, 4) = 1 + 2 = 3

3. (1, 2), (3, 4) -> min(1, 2) + min(3, 4) = 1 + 3 = 4

So the maximum possible sum is 4.

Example 2:

Input: nums = [6,2,6,5,1,2]

Output: 9

Explanation: The optimal pairing is (2, 1), (2, 5), (6, 6). min(2, 1) + min(2, 5) + min(6, 6) = 1 + 2 + 6 = 9.

Constraints:

- $1 <= n <= 10^4$
- nums.length == 2 * n
- $-10^4 <= nums[i] <= 10^4$

Question 4: Given an integer array nums sorted in **non-decreasing** order, return *an array of **the squares of each number** sorted in non-decreasing order*.

Example 1:

Input: nums = [-4,-1,0,3,10]

Output: [0,1,9,16,100]

Explanation: After squaring, the array becomes [16,1,0,9,100].

After sorting, it becomes [0,1,9,16,100].

Example 2:

Input: nums = [-7,-3,2,3,11]

Output: [4,9,9,49,121]

Constraints:

- $1 <= nums.length <= 10^4$
- $-10^4 <= nums[i] <= 10^4$
- nums is sorted in **non-decreasing** order.

Question 5: Given two strings `s` and `t`, return `true` *if* `t` *is an anagram of* `s`*, and* `false` *otherwise*.

Example 1:

Input: s = "anagram", t = "nagaram"

Output: true

Example 2:

Input: s = "rat", t = "car"

Output: false

Constraints:

- `1 <= s.length, t.length <= 5 * 10^4`
- `s` and `t` consist of lowercase English letters.

Question 6: Given an integer array `nums`, move all the even integers at the beginning of the array followed by all the odd integers.

Return ***any array*** *that satisfies this condition*.

Example 1:

Input: nums = [3,1,2,4]

Output: [2,4,3,1]

Explanation: The outputs [4,2,3,1], [2,4,1,3], and [4,2,1,3] would also be accepted.

Example 2:

Input: nums = [0]

Output: [0]

Constraints:

- `1 <= nums.length <= 5000`
- `0 <= nums[i] <= 5000`