Searching (Binary) and Sorting – Week 2

Total 90 points
Each question 10 marks (8 points for code , 2 points for Time and Space Complexity)

Note: For questions related to Searching, please use Binary Search

1. Given an array of integers nums and an integer target, return indices of the two numbers such that they add up to target. You may assume that each input would have exactly one solution, and you may not use the same element twice.
You can return the answer in any order.

Example 1:
Input: nums = [2,7,11,15], target = 9
Output: [0,1]
Output: Because nums[0] + nums[1] == 9, we return [0, 1].

Example 2:
Input: nums = [3,3], target = 6
Output: [0,1]

Constraints:

- 2 <= nums.length <= 104
- -109 <= nums[i] <= 109
- -109 <= target <= 109
- Only one valid answer exists.

2. Given an array of meeting time intervals intervals where intervals[i] = [starti, endi], return the minimum number of conference rooms required.

Example 1:
Input: intervals = [[0,30],[5,10],[15,20]]
Output: 2

Example 2:
Input: intervals = [[7,10],[2,4]]
Output: 1

Constraints:

- 1 <= intervals.length <= 104
- 0 <= starti < endi <= 106

3. Given two integer arrays nums1 and nums2, return an array of their intersection. Each element in the result must be unique and you may return the result in any order.

Example 1:
Input: nums1 = [1,2,2,1], nums2 = [2,2]
Output: [2]

Example 2:
Input: nums1 = [4,9,5], nums2 = [9,4,9,8,4]
Output: [9,4]
Explanation: [4,9] is also accepted.

Constraints:

- 1 <= nums1.length, nums2.length <= 1000
- 0 <= nums1[i], nums2[i] <= 1000

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

5. Given an array of integers nums sorted in ascending order, find the starting and ending position of a given target value. If target is not found in the array, return [-1, -1].

Example 1:
Input: nums = [5,7,7,8,8,10], target = 8
Output: [3,4]

Example 2:
Input: nums = [5,7,7,8,8,10], target = 6
Output: [-1,-1]

Example 3:
Input: nums = [], target = 0
Output: [-1,-1]

Constraints:

- 0 <= nums.length <= 105
- -109 <= nums[i] <= 109
- nums is a non-decreasing array.
- -109 <= target <= 109

6. Given a m x n matrix grid which is sorted in non-increasing order both row-wise and column-wise, return the number of negative numbers in grid.

Example 1:
Input: grid = [[4,3,2,-1],[3,2,1,-1],[1,1,-1,-2],[-1,-1,-2,-3]]
Output: 8
Explanation: There are 8 negatives number in the matrix.

Example 2:
Input: grid = [[3,2],[1,0]]
Output: 0

Example 3:
Input: grid = [[1,-1],[-1,-1]]
Output: 3

Example 4:
Input: grid = [[-1]]
Output: 1

Constraints:

- m == grid.length
- n == grid[i].length
- 1 <= m, n <= 100
- -100 <= grid[i][j] <= 100

7. A peak element is an element that is strictly greater than its neighbors. Given an integer array nums, find a peak element, and return its index. If the array contains multiple peaks, return the index to any of the peaks.
You may imagine that nums[-1] = nums[n] = -infinity

Example 1:
Input: nums = [1,2,3,1]
Output: 2

Explanation: 3 is a peak element and your function should return the index number 2.

Example 2:
Input: nums = [1,2,1,3,5,6,4]
Output: 5
Explanation: Your function can return either index number 1 where the peak element is 2, or index number 5 where the peak element is 6.

Constraints:

- 1 <= nums.length <= 1000
- -231 <= nums[i] <= 231 - 1
- nums[i] != nums[i + 1] for all valid i.

8. Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive. There is only one repeated number in nums, return this repeated number.

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

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

Example 3:
Input: nums = [1,1]
Output: 1

Example 4:
Input: nums = [1,1,2]
Output: 1

Constraints:

- 1 <= n <= 105
- nums.length == n + 1
- 1 <= nums[i] <= n
- All the integers in nums appear only once except for precisely one integer which appears two or more times.

9. Given an array arr of positive integers sorted in a strictly increasing order, and an integer k. Find the kth positive integer that is missing from this array.

Example 1:
Input: arr = [2,3,4,7,11], k = 5
Output: 9
Explanation: The missing positive integers are [1,5,6,8,9,10,12,13,...]. The 5th missing positive integer is 9.

Example 2:
Input: arr = [1,2,3,4], k = 2
Output: 6
Explanation: The missing positive integers are [5,6,7,...]. The 2nd missing positive integer is 6.

Constraints:

- 1 <= arr.length <= 1000
- 1 <= arr[i] <= 1000
- 1 <= k <= 1000
- arr[i] < arr[j] for 1 <= i < j <= arr.length