**CSE221**

**Lab Assignment 03**

**Summer 2023**

**Submission Guidelines:**

1. You can code all of them either in Python, CPP, or Java. But you should choose a specific language for all tasks.
2. For each task write separate python files like task1.py, task2.py, and so on.
3. For each problem, take input from files called **"inputX.txt"** and output at **"outputX.txt"**, where X is the task number.
4. Add a hand written explanation of 3-4 lines for each of your solutions in a separate document. You may compile all of your explanations in a single file.
5. Finally zip all the files and rename this zip file as per this format:**LabSectionNo\_ID\_CSE221LabAssignmentNo\_Summer2023.zip** [Example:**LabSection01\_21101XXX\_CSE221LabAssignment03\_Summer2023.zip**]
6. Don't copy from your friends.
7. You MUST follow all the guidelines, naming/file/zipping convention stated above.

*Failure to follow instructions will result in a straight 50% mark deduction.*

**Task 01 [15 Points]:**

Somewhere in the universe, the Biannual Regional Alien Competition is taking place.

There are **N** aliens standing in a line. You will be given a permutation of N, which denotes the height of each alien. A sequence of N numbers is called a permutation if it contains all integers from 1 to N exactly once. For example, the sequences [3,1,4,2], [1] and [2,1] are permutations, but [1,2,1], [0,1] and [1,3,4] — are not.

In the competition, for each alien, the judge wants to count how many aliens are standing on its right side with a strictly smaller height. Then the judge wants to add up all the counts. To do this, the judge writes the following piece of code.







| **count** = 0 **for** i **in** **range**(**n**):  **for** j **in** **range**(i+1,**n**):  **if** H[i] > H[j]:  **count**+=1 |
| --- |

However, their algorithm wasn’t efficient at all. Hence, the alien calls you to write a better solution for the program.

More formally, you have to count how many pairs of aliens are standing in the line such that H[i] > H[j] and i < j. Here, A is a permutation of the aliens’ heights. And i,j denote the Aliens’ positions.

**Input**

The first line contains a single integer 1 <= N <= 106 - the number of total aliens.

The next line contains N integers H1,H2,…………,Hn(1 ≤ Hi ≤ N)- the height of the i-th alien. It is guaranteed that the given heights will be a permutation of N.

**Output**

Print a single integer, which denotes the total number of pairs (i, j) such that i < j and Hi >Hj.

**Sample Input/Output:**

| Sample Input 1 | Sample Output 1 |
| --- | --- |
| 5  1 2 3 4 5 | 0 |
| Sample Input 2 | Sample Output 2 |
| 5  5 4 3 2 1 | 10 |
| Sample Input 3 | Sample Output 3 |
| 8  2 7 4 1 5 6 8 3 | 11 |

**Sample Input 3 Explanation:**

In the sample input 3, the following pairs on alien’s heights satisfy the condition: (2,1), (7,4), (7,1), (7,5), (7,6), (7,3), (4,1), (4,3), (5,3), (6,3), (8,3)

**Task 02 [15 points]**

You are given a list A of N integers. You have to choose two indices i and j such that 1 <= i < j <= N and A[i] + A[j]2 is maximum possible. Here, we are considering 1-based indexing.

Write a code which will find the maximum value of A[i] + A[j]2 in O(N) or O(N log N).

**Input**

The first line contains a single integer 1 <= N <= 106 - the length of the list.

The next line contains N integers A1,A2,…………,An (-108 ≤ Ai ≤ 108) separated by a space.

**Output**

Print a single integer - which denotes the maximum possible value of A[i] + A[j]2.

**Sample Input/Output:**

| Sample Input 1 | Sample Output 1 |
| --- | --- |
| 5  9 6 5 8 2 | 73 |
| Sample Input 2 | Sample Output 2 |
| 8  5 10 4 -3 1 6 -10 2 | 110 |
| Sample Input 3 | Sample Output 3 |
| 7  -5 -2 -6 -7 -1 8 2 | 63 |

**Task 03 [10 Points]**

In this problem, you will be given a list of numbers. You have to sort the list using the Quick Sort algorithm in ascending order.

**Pseudocode of Quick Sort Algorithm:**

****

****

**[The code snippet has been taken from the book: Introduction to Algorithms]**

**Input**

The first line contains an integer N (1 <= N <= 105), denoting the length of Alice’s list. In the next line, there will be N integers separated by space.

**Output:**

You have to sort the number using the Quick Sort algorithm in ascending order and show the sorted list.

**Sample Input/Output:**

| Sample Input 1 | Sample Output 1 |
| --- | --- |
| 8  9 5 4 6 1 3 2 9 | 1 2 3 4 5 6 9 9 |
| Sample Input 2 | Sample Output 2 |
| 1  10 | 10 |
| Sample Input 3 | Sample Output 3 |
| 6  8 1 4 2 1 3 | 1 1 2 3 4 8 |
| Sample Input 4 | Sample Output 4 |
| 7  7 6 5 4 3 2 1 | 1 2 3 4 5 6 7 |

**Task 04 [10 Points]**

In this problem, you will be given a list of numbers. You have to find the **k**-th smallest value from the list without sorting using the Partition function of Quick sort.

We will consider the 1-based indexing of the list.

**Input**

The first line contains an integer N (1 <= N <= 106), denoting the length of the list.

The next line contains N integers A1,A2,…………,An( 1 ≤ Ai ≤ 106) separated by a space.

The third line contains a single integer Q (1 <= Q <= 100) - which denotes the number of queries you have to answer.

Each of the next Q lines will contain a single integer K (1 ≤ K≤ N).

**Output:**

For each query, you have to find the K-th smallest number from the given list.

**Sample Input/Output:**

| Sample Input 1 | Sample Output 1 |
| --- | --- |
| 9 // Total Elements  10 11 10 6 7 9 8 15 2  4 // Total queries  5  3  2  7 | 9  7  6  10 |