CSE221
Lab Assignment 03
Spring 2024

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_Spring2024.zip** [Example:**LabSection01_21101XXX_CSE221LabAssignment03_Spring2024.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: [Points 10]

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

Pseudocode of Merge Sort Algorithm:

```
def merge(a, b):
      # write your code here
      # Here a and b are two sorted list
      # merge function will return a sorted list after merging a and b


def mergeSort(arr):
    if len(arr) <= 1:
        return arr
```

```
    else:
        mid = len(arr)//2
        a1 = mergeSort(.............)  # write the parameter
        a2 = mergeSort(.............)  # write the parameter
        return merge(a1, a2)          # complete the merge function above
```

Note: You already have coded the merge function. Do you know which task?

Input

The first line contains an integer N (1 <= N <= 10^5), denoting the length of Alice's sorted list. In the next line, there will be N integers separated by space.

Output:

You have to sort the number using the Merge Sort algorithm 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 02 [10 Points]:

Alice and you are playing with a list of **N** non negative integers. Today you will try to find out the maximum number of the list. Alice writes the following code to find the maximum number.

```
maxValue = arr[0]
for i in range(1,N):
  if maxValue < arr[i]:
    maxValue = arr[i]
```

Recently you have learned merge sort. Now, you are thinking if you can use the divide and conquer approach to find out the maximum from the given list.

Please note, you are not allowed to sort the given list. The motive for this task is not sorting but to use the concepts of divide and conquer approach.

Input

The first line contains an integer N ($1 <= N <= 10^5$), denoting the length of Alice's list. In the next line, there will be N integers separated by space.

Output:

You have to find out the maximum value from the list using the divide and conquer approach.

Sample Input/Output:

Sample Input 1	Sample Output 1
8 1 7 13 4 5 7 13 12	13
Sample Input 2	Sample Output 2
7	15

5 15 2 3 10 1 9	
Sample Input 3	Sample Output 3
1 9	9
Sample Input 4	Sample Output 4
6 5 2 3 10 1 9	10

What is the time complexity of your code?

Task 03 [10 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 <= 10^6 - the number of total aliens.

The next line contains N integers $H_1, H_2, \ldots\ldots\ldots, H_n$**(1 ≤ H_i ≤ 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 $H_i > H_j$.

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 04 [10 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 log N).

Input

The first line contains a single integer $1 <= N <= 10^6$ - the length of the list.

The next line contains N integers $A_1, A_2, \ldots\ldots\ldots, A_n$ ($-10^8 \leq A_i \leq 10^8$) 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 05 [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:

```
QUICKSORT(A, p, r)
1  if p < r
2      q = PARTITION(A, p, r)
3      QUICKSORT(A, p, q − 1)
4      QUICKSORT(A, q + 1, r)
```

```
PARTITION(A, p, r)
1  x = A[r]
2  i = p − 1
3  for j = p to r − 1
4      if A[j] ≤ x
5          i = i + 1
6          exchange A[i] with A[j]
7  exchange A[i + 1] with A[r]
8  return i + 1
```

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

Input

The first line contains an integer N (1 <= N <= 10^5), 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 06 [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 ($1 <= N <= 10^6$), denoting the length of the list.

The next line contains N integers $A_1, A_2, \ldots\ldots, A_n$ ($1 \leq A_i \leq 10^6$) 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 ($1 \leq K \leq N$).

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