

CSE221 Fall 2024

Lab Practice Sheet 02

Two Pointers, Greedy

Task 01:

Your little brother, Alice, is very fond of playing with integers. One day, Alice was given a sorted list of N integers in ascending order by his school teacher. Now, your brother wants to play a game with you.

Alice will give you an integer, S . You have to find if it is possible to find two values from the list (at distinct positions) whose sum is equal to S .

Now you are feeling very tired. So you decided to write a code, so that it can give you the answer very quickly.

- 1) Can you write an $O(N^2)$ Solution to solve the problem? **[Points 5]**
- 2) Come up with an $O(N)$ solution. **[Points 10]**

Input

The first line contains two integers N and S ($1 \leq N \leq 10^5$, $1 \leq S \leq 10^9$), denoting the length of the list, and the target Sum.

In the next line, there will be N integers a_1, a_2, \dots, a_N ($1 \leq a_i \leq 10^9$) separated by space.

Output

Print two integers: **the positions** of the values **[1 based indexing]**. If there are several solutions, you may print any of them. If there are no solutions, print "IMPOSSIBLE".

Sample Input 1	Sample Output 1
----------------	-----------------

4 10 1 3 5 7	2 4
Sample Input 2	Sample Output 2
6 18 1 5 8 9 9 10	3 6 [4 5 is also a valid answer] [print only one output]
Sample Input 3	Sample Output 3
4 7 2 4 6 8	IMPOSSIBLE
Sample Input 4	Sample Output 4
4 10 1 5 6 8	IMPOSSIBLE

Task 02:

Alice and Bob are two friends. Alice has a sorted list in ascending order of length **N**. On the other hand, Bob has a sorted list of length **M**. Now, they want to make a sorted list of **N+M** length in ascending order. However, they are not very good at algorithms. Hence, they asked for your help.

Since you are a computer science student, your task is to come up with an efficient algorithm. In the following, **let $n = N+M$** .

- 1) Find a solution that runs in $O(n \log n)$. **[Points 5]**
- 2) Come up with a solution that runs in $O(n)$. **[Points 10]**

Input

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

The third line contains another integer M ($1 \leq M \leq 10^5$), denoting the length of Bob's sorted list. In the next line, there will be M integers separated by space.

All the numbers given in the input will fit in a 32-bit signed integer.

It is guaranteed that the given lists will be in sorted order.

Output:

You have to make a sorted list in ascending order from the given sorted lists and show the output.

Sample Input/Output:

Sample Input 1	Sample Output 1
4 1 3 5 7 4 2 2 4 8	1 2 2 3 4 5 7 8
Sample Input 2	Sample Output 2
3 2 10 12 6 3 4 6 7 8 9	2 3 4 6 7 8 9 10 12
Sample Input 3	Sample Output 3
4 1 2 3 4 1 10	1 2 3 4 10
Sample Input 4	Sample Output 4
7 2 3 8 8 10 12 14 9 1 1 4 5 6 8 13 15 16	1 1 2 3 4 5 6 8 8 8 10 12 13 14 15 16

Task 3

You are a busy person with lots of tasks to do. You have a schedule of tasks represented by intervals of time, where each interval represents a task that you need to complete. However, you can only work on one task at a time, and you want to complete as many tasks as possible.

Given a list of N intervals of time, your task is to determine the maximum number of tasks you can complete and which tasks they are.

Come up with a solution that runs in $O(N \log N)$

Input

The input consists of a single integer N ($1 \leq N \leq 10^5$), the number of tasks, followed by N lines representing the tasks. Each task is represented by two integers S_i and E_i ($0 \leq S_i \leq E_i \leq 10^9$), the start and end times of the task, respectively.

Output

Output a single integer k , the maximum number of tasks you can complete, followed by a line with k intervals of the tasks you can complete.

If there are multiple solutions with the same maximum number of tasks, print any one of them.

Sample Input/Output:

Sample Input 1	Sample Output 1
6 1 3 2 5 3 7 4 6 6 8	3 1 3 4 6 6 8

7 9	
Sample Input 2	Sample Output 2
5 1 4 2 5 6 7 4 8 3 6	2 1 4 6 7
Sample Input 3	Sample Output 3
7 0 4 3 4 1 5 9 10 6 9 2 3 1 2	5 1 2 2 3 3 4 6 9 9 10

Task 4

Given N tasks and M people, where each task has a start time and end time, implement a greedy algorithm to find the maximum number of tasks that can be completed by M people.

Each task can only be completed by one person and a person can only be assigned one task at a time. Two tasks cannot be completed simultaneously by the same person.

In the following, **let $n = N+M$.**

- 1) Can you write an $O(n^2)$ Solution to solve the problem?
- 2) Come up with an $O(n \log n)$ solution.

Input

The input consists of two integers N and M ($1 \leq N, M \leq 10^3$), the number of activities and the number of people, respectively. This

is followed by N lines representing the activities. Each line contains two integers S_i and E_i ($0 \leq S_i \leq E_i \leq 10^9$), representing the start and end times of the activity, respectively.

Output

Output a single integer representing the maximum number of activities that can be completed.

Sample Input/Output:

Sample Input 1	Sample Output 1
5 2 1 5 3 6 2 5 8 10 6 9	4
Sample Input 2	Sample Output 2
5 2 1 4 2 5 6 7 4 8 3 6	4
Sample Input 3	Sample Output 3
6 2 1 5 4 10 8 17 12 15 9 11 14 18	5
Sample Input 4	Sample Output 4

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

Sample Input Explanation:

In sample input 2-

Person 1 will complete the tasks: 1-4, 4-8

Person 2 will complete the tasks: 2-5, 6-7