Data Structures and Algorithms (CS F211)
Second Semester 2018-2019
Lab Sheet 5

1. Captain America and Iron Man are fighting but instead of collecting other superheroes, they plan on collecting ranges. You have ***n*** ranges and have been assigned the task of dividing the ranges between the two teams. Every range ***i*** has a left limit l_i and a right limit r_i. You have to divide the groups in such a way that, there is no pair of ranges, each from a different group which have a common point. However, ranges assigned to the same group may have a common point. Each range should belong to exactly one group. You have to print ***IM*** if you assign a range to Team Iron Man and print ***CA*** if you assign the range to Team Captain America. If an assignment is impossible, print -1. There is no restriction on the number of ranges that you can assign to a team. But the assignment cannot be too skewed also, meaning that out ***n*** ranges, you can't give (***n*** -1) ranges to one team and the remaining 1 range to the other team. Note that for a given set of ranges, multiple assignments are possible. Determine any one of them. If you see that the only possible assignment of ranges is to give (***n*** - 1) range to one team and the remaining range to the other team, then you can report it to be your output. But if some alternate assignment is also possible, then the alternate one should be reported. Duplicate ranges can be present also.

Input:
The first line contains one integer ***T*** ($1 \le T \le 50000$) — the number of queries. Each query contains description of the set of ranges. Queries are independent. First line of each query contains a single integer ***n*** ($2 \le n \le 105$) — the number of segments. It is guaranteed that $\sum n$ over all queries does not exceed 105. Each of the next ***n*** lines contains two integers l_i, r_i ($1 \le l_i \le r_i \le 2 \cdot 105$) for range ***i***.

Example
Input:
3
2
5 5
2 3
3
3 5
2 3
2 3
3
3 3
4 4
5 5

Comment: Here there are total 3 queries, i.e., there are 3 sets of ranges. The first set contains 2 ranges - (5, 5) and (2, 3). The second set contains 3 ranges - (3, 5), (2, 3) and (2, 3). The third set contains 3 ranges - (3, 3), (4, 4), (5, 5).

Output:
CA IM
-1
IM IM CA

Explanation:
For the first query, since only two ranges (***n*** = 2) are present, you have to assign one range to team IM and one to team CA. For the second query, no assignment is possible because of the presence of the common point. For the last query, since ***n*** = 3, you have to give two ranges, (i.e, ***n*** - 1) to one team and the remaining to the other. For the first and last query, no other assignment was possible.

2. Naruto and Sasuke are given two arrays ***A1*** and ***A2*** of size ***N*** and ***M*** respectively. Since they are in a team, they need to be in sync. Kakashi, as their Sensei, decided to sort ***A1*** in such a way that the relative of the elements will be same as that in ***A2***. If there are some elements left in ***A2***, their friend Sakura will append them at last in sorted order. It is also given that the number of elements in ***A2*** is smaller than or equal to number of elements in ***A1*** and ***A2*** has all distinct elements. ***A1*** can have duplicate elements. Moreover, ***A2*** can have a few elements which are not present in ***A1***.

Input:
N M
A1
A2

Output:
The final array.

Constraints:
1 ≤ ***N***, ***M*** ≤ 106
|***A1***|<= 106, |***A2***| <= 106

Example:
Input:
11 4
2 1 2 5 7 1 9 3 6 8 8
2 1 8 3

Output:
2 2 1 1 8 8 3 5 6 7 9

Input:
10 5
3 1 4 6 2 1 8 5 3 6
6 1 7 9 8

Output:
6 6 1 1 8 2 3 3 4 5

3. Aroma is a very unique restaurant. They don't ask for money, but instead request you to carry out certain operations on their behalf. They ask you to insert an element into an already sorted list, delete an element from the sorted list, swap pairs of elements of the list and finally sort the list after swapping(s). You are also required to print the list after every operation. Note that you are not allowed to have vacant positions in the list after deletions and no insertion will be done after swapping (since after every insertion the list needs to remain sorted). Insertions are possible after deletions. After swapping(s), deletions are however, possible. At any point of time, the list will contain only unique elements, assuming that you will not be asked to insert a duplicate element in the list at any point of time. Moreover, you do not know apriori, how many insertion and deletions you will be asked to carry out.

Input:
The first line contains the elements for the existing list in sorted (ascending) order.
The second line contains the number ***N*** of operations to be carried out.

Following ***N*** lines contain the details of the operations

- For insertion - I Number to be inserted
- For deletion - D Number to be deleted
- For swapping - SW Numbers to be swapped
- For sorting - SO

Output:
The intermediate array after every operation

Example
Input:
3 5 6 8 10 15
13
I 1
I 7
I 18
D 5
D 10
I 2
I 4
I 9
SW 4 18
SW 1 6
D 1
D 7
SO

Output:
1 3 5 6 8 10 15
1 3 5 6 7 8 10 15
1 3 5 6 7 8 10 15 18
1 3 6 7 8 10 15 18
1 3 6 7 8 15 18
1 2 3 6 7 8 15 18
1 2 3 4 6 7 8 15 18
1 2 3 4 6 7 8 9 15 18
1 2 3 18 6 7 8 9 15 4
6 2 3 18 1 7 8 9 15 4
6 2 3 18 7 8 9 15 4
6 2 3 18 8 9 15 4
2 3 4 6 8 9 15 18

4. Aman, Ketan and Shreya always hangout on marine drive. But whenever they ask Apoorva to hang out, she is busy doing homework. They want to help her finish the homework faster. Can you help them understand Apoorva's homework so that she can hang out with them?
Consider an array of ***arr*** of ***n*** distinct integers. Any two elements of the array can be swapped any number of times. An array is beautiful if the sum of all ***|arr[i] - arr[i-1]|*** (absolute value) for ***1 <= i <= (n -1)*** is minimal. Given the array ***arr***, determine the minimum number of swaps that should be performed in order to make the array beautiful. Note that you are not allowed to make any extra swaps other than what you are displaying as output.

Input:
The first line contains the number of elements of the array.

The second line contains the elements of the array

Output:
The number of swaps required to make the array beautiful.

Example
Input:
4
7 15 12 3

Output:
2

Explanation:
After first swap → 3 15 12 7
After second swap → 3 7 12 15

Input:
4
2 5 3 1

Output:
2

Explanation:
After first swap → 2 1 3 5
After second swap → 1 2 3 5

5. Klay Thompson is one the best 3-pointer shooters in the history of NBA. He is so much obsessed with the number 3 that his team mates gave him a task. They gave him an array of numbers 0-9 (not necessarily containing all the 10 digits and not necessarily containing only unique digits in sorted order) and asked him to come up with the largest number that is divisible by 3 using these digits. If no such number is possible, he will have to say "No such number". The number of digits present in the output can vary between 1 and the size of the array. You are allowed to use any element of the input array only once.

Input:
The first line contains the size of the array.
The second line contains the elements of the array.

Output:
The largest number divisible by 3.

Example
Input:
5
5 4 3 1 1

Output:
4311 (Not possible to construct a number divisible by 3 consisting of all the 5 digits. You cannot create a number consisting of more than one 3 or 4 since these digits are only present once in the input array. However, 1 is present twice in the output since the input array contained two 1s.)

Input:
5
0 5 5 5 7

Output:
5550 (Not possible to construct a number divisible by 3 consisting of all the 5 digits. Also, you cannot use more than three 5s.)

6. You are given an unsorted array ***a***. You have to find out all the unordered pairs in the array. An unordered pair is defined as a pair ***(a[i], a[j])***, such that ***i*** < ***j*** but ***a[i]*** > ***a[j]***. For example, if the array ***a*** contains 2, 3, 1, 4, then the unordered pairs are (2, 1) and (3, 1).

Input:
The first line contains ***n*** (the size of the array).
The second line contains ***n*** different integers as elements of ***a***.

Output:
The number of unordered pairs in less than O(n^2) time.

Example:
Input:
5
2 4 1 3 5

Output:
3

7. You are given an undirected, unweighted, connected graph without self-loops consisting of ***n*** vertices. For convenience, we will assume that the graph vertices are indexed using integers from 1 to ***n***. One day, you counted the shortest distances from one of the graph vertices to all others and wrote them in an array ***d***. Thus, element ***d[i]*** of the array shows the shortest distance from the vertex you chose to vertex number ***i***. Then you lost the graph and also forgot which vertex you chose as the source. Form the graph using the array ***d***.

Input:
One integer ***n***.
The array ***d*** of size ***n***.

Output:
If there is no such graph print -1. Otherwise output the integer ***m*** which is the number of edges. In each of the next ***m*** lines, print two integers, a_i and b_i, separated by a space, denoting the edge that connects vertices with numbers a_i and b_i. The graph shouldn't contain self-loops and should have only one edge between a pair of vertices. If there are multiple possible answers, print any one of them. ***m*** >= |***d***| meaning you are allowed to add a few extra edges apart from the information derived from the array ***d*** keeping in mind the constraint that you have to create exactly one cycle and that cycle should consist of the minimum number of edges. In other words, if multiple cycles are possible, you are allowed to include the one containing the minimum number of edges. If there is a tie in this regard, you can select any one of the cycles. You are not allowed to alter the shortest distances of the original graph, i.e., in the graph that you will construct (containing the cycle), the shortest distances of every vertex from the source should be the same as those present in the array ***d***. In case, no graph can be constructed using the given input and also keeping in mind the above mentioned constraints, report -1 as output.

Example:
Input:
3
0 1 1

Output:
3
1 2
1 3
3 2

Explanation:
Here only one cycle is possible.

Input:
5
1 0 2 1 1

Output:
5
1 2
1 3
2 4
2 5
1 4

Explanation: (1 4) is the additional edge. Instead of (1 4), (2 3), (1 5), (4 5) could also have been selected. However, (3 4) would have been a wrong choice.

8. It is Christmas time and you have offered to be the Santa Clause for your village. You want to gift the children XBOX or PS4. For this, you will have to buy PS4s and XBOXs, but you want to spend as little as possible. Now, you know the demand of each child. Some of them are Sony fanatics and only want PS4, some of them love Microsoft products and only want XBOXs and some of them are happy with either of them, i.e., you can gift them any. You have found a shop which sells both PS4s and XBOXs at variable prices. Different types of PS4s come in different prices and so do different types of XBOXs. The total number of devices sold by the shop is ***m***. You want to buy a set of XBOXs and PS4s such that you want to maximize the number of children you can gift (it is not guaranteed that you can gift all the children) and minimize the amount you will be spending. However, your priority is to gift as many children as possible. A child does not care about the variety or cost of a PS4 or an XBOX.

Input:
First line contains ***a***, ***b***, ***c***. ***a*** denotes the number of children who want a PS4, ***b*** denotes the number of children who want an XBOX and ***c*** denotes the number of children will do with either ($0 \le a, b, c \le 10^5$).
The next line contains ***m*** (the number of devices the shop sells). ($0 \le m \le 10^5$).
Each of the next ***m*** lines describes the device. The i^{th} line contains first integer ***val[i]*** ($1 \le val[i] \le 100$) — the cost of the i^{th} device, then the type of device (PS4 or XBOX).

Output:
Output two integers. The number of children you will be able to gift (maximize the number) and the cost you will be spending (minimize the cost).

Example:
Input:
2 1 1
4
5 PS4
6 XBOX
3 XBOX
7 XBOX

Output:
3 14

Explanation: 2 children want PS4 but the shop only has 1 PS4. So 1 child from ***a*** category will be left unsatisfied. Only 1 wants XBOX and only 1 will do with either of them. So you will buy 1 PS4 worth 5 INR (to satisfy ***a***). Then, you will get 1 XBOX worth 3 INR (to satisfy ***b***). Then, you will get another XBOX worth 6 INR (to satisfy ***c***). Total 3 devices worth 5+3+6 = 14 INR.

9. You are performing a Physics experiment. You have recorded ***n*** measurements in your notebook. After that, you remembered that the largest and the smallest measurements should differ by more than two times. However, you do not have enough time to record all the measurements and want to somehow manage with the ***n*** measurements that you have. You will erase some of the recorded measurements, so as to make the largest and the smallest results of the remaining measurements differ in no more than two times. In other words, if the remaining measurements (after removal) have the smallest value ***x***, and the largest value ***y***, then the inequality $y \leq 2 \cdot x$ must be fulfilled. Also, the largest and the smallest measurements should appear only once in the measurement set. If that is not the case with your recorded set of measurements, you will have to ensure that as well. Of course, to avoid the teacher's suspicion, you want to remove the minimum number of measurements.

Input:
First line contains ***n***, the size of array.
The second line contains ***n*** integers ***c1***, ***c2***, ..., ***cn*** ($1 \leq ci \leq 5000$) — the recorded measurements.

Output:
Print a single integer — the minimum number of measurements you will have to remove

Example:
Input:
6
4 5 3 8 3 7 8

Output:
3

Explanation: Two 3 and one 8 is removed. Now the largest is 8 and the smallest is 4.

****