Sample Problems

Sample Problem #1:

Today you decided to go to the gym. You currently have energy equal to E units. There are N exercises in the gym. Each of these exercises drains Ai amount of energy from your body.

You feel tired if your energy reaches 0 or below. Calculate the minimum number of exercises you have to perform such that you become tired. Every unique exercise can only be performed at most 2 times as others also have to use the machines.

If performing all the exercises does not make you feel tired, return -1.

Parameters:

E :: INTEGER
The first line contains an integer, E, denoting the Energy.
E :: 1 -> 10^5

N :: INTEGER
The next line contains an integer, N, denoting the number of exercises.
N :: 1 -> 10^5

A :: INTEGER ARRAY
Each line i of the N subsequent lines (where 0 ≤ i < N) contains an integer describing the amount of energy drained by *i*-th exercise.
A[i] :: 1 -> 10^5

Test Cases

Case#: 1

Input:
6
2
1
2
Output:
4

E = 6
Do 1st exercise 2 times
Do 2nd exercise 2 times
Hence, total exercises done 4.

Case#: 2

Input:
10
2
1
2
Output:
-1

E = 10
By doing both the exercises 2 times, you won't feel tired.

Case#: 3

Input:
2
3
1
5
2
Output:
1

E = 2
Use 3rd exercise 1 time.
Hence, total exercise done 1.

Sample Problem #2:

There is a battle between heroes and villains going on. You have M heroes, all of them have the same health H. There are N villains, health of the *i*-th villain is Vi.

When a hero, with health H battles a villain with health Vi, one of the three scenarios can happen:

if H > Vi: The villain is defeated, and the health of the hero is decreased by Vi
if H < Vi: The villain wins, his health is not affected, and the hero is no longer able to fight.
if H = Vi: Both are considered defeated, and neither can fight.
The heroes start fighting villains one by one in the same order, first villain 1 then villain 2 and so on. It might be possible that before defeating all the villains, all the heroes are defeated. Therefore, to ensure the victory of the heroes, you want to remove some villains from the front.

Your task is to find the minimum number of villains you need to remove from the front such that the victory of the heroes is guaranteed.

Note: If in the last battle, both the hero and villain are defeated and no more heroes or villains remain, it would still be considered a victory since all the villains are defeated.

Parameters:

N :: INTEGER
The first line contains an integer, N, denoting the number of villains
N :: 1 -> 2*10^5

M :: INTEGER
The next line contains an integer, M, denoting the number of heroes
M :: 1 -> 2*10^5

H :: INTEGER
The next line contains an integer, H, denoting the health of each of the heroes
H :: 1 -> 10^9

array :: INTEGER ARRAY
Each line i of the N subsequent lines (where $0 \leq i < N$) contains an integer describing the health of each of the villains.
array[i] :: 1 -> 10^9

Test Cases

Case#: 1

Input:
4
4
3
3
1
3
3
Output:
0

[3, 1, 3, 3]. We have 4 heroes will health 3. The heroes 1 will fight villain 1. Both get defeated. The hero 2 fights villain 2. It wins the battle and now his health is 2. He fights the third villain and loses, the villain still has health 3. The hero 3 fights villain 3 and both get defeated. Hero 4 fights villain 4 and both get defeated. So, no need to remove any villain.

Case#: 2

Input:
5
3

3
1
2
3
1
1
Output:
0

The fight will take place and hero 1 will defeat villain 1 and 2. Hero 2 will defeat villain 2. Hero 3 will defeat villain 3 and 4

Case#: 3

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

Only 1 hero is present with health 4. Since you can only remove villain from the front, you will have to remove the first 3 villains to ensure victory. The hero can fight the last 2 villain of health 1 and 3 respectively and win the battle.

Sample Problem #3:

You need to build a road in a rugged terrain. You know the sea level of each segment of the rugged terrain, i.e., the *i*-th segment is Li meters from sea level.

You need to transform the terrain into a strictly downward sloping terrain for the road, i.e., for each i-th segment where 2 <= i <= N, resultant Li-1 > Li. To do so, you employ a powerful digging team to help you dig and reduce the sea level of the segments. On day D, the team can reduce the sea level for each segment that you scheduled that day by 2D-1 meters each.

You are allowed to assign the team to dig on multiple segments and/or dig on the same segments for multiple days.

Your task is to find the minimum number of days needed to transform the terrain as per your requirements.

Parameters:

N :: INTEGER
The first line contains an integer, N, denoting the number of elements in L.
N :: 1 -> 10^5

L :: INTEGER ARRAY
Each line i of the N subsequent lines (where 0 < i ≤ N) contains an integer describing Li, the sea level of the *i*-th segment.
L[i] :: -10^9 -> 10^9

Test Cases

Case#: 1

Input:
2
3
3
Output:
1

We can dig on the 2nd segment, reducing it from 3-meter sea level to 2. Resulting in {3, 2} which is strictly decreasing.

Case#: 2

Input:
2
5
-3
Output:
0

It is already strictly decreasing before start.

Case#: 3

Input:
4
-1
1
1
1
Output:
3

One of the possible ways:
On day 1, we can dig on 1st and 4th segment, resulting in {-2, 1, 1, 0}
On day 2, we can dig on 3rd and 4th segments, resulting in {-2, 1, -1, -2}
On day 3, we can dig on 2nd, 3rd, and 4th segments, resulting in {-2, -3, -5, -6}

Sample Problem #4:

You are given an array of size N. You need to change this array into a mountain. By mountain we mean, the either ends of the array should have equal elements. Then as we move towards the middle from both ends, the next element is just one more than the previous one. So, it would have a peak in the middle and decreases if you go towards either end, just like a mountain.

Examples of mountains are [1, 2, 3, 2, 1] or [6, 7, 8, 8, 7, 6]. But the array [1, 2, 4, 2, 1] is not a mountain because from 2 to 4 the difference is 2. The array [1, 2, 3, 1] is also not a mountain because the elements 2 and 3 are not equal from both ends.

You need to find the minimum number of elements that should be changed to make the array a mountain. You can make the elements negative or zero as well.

Parameters:

N :: INTEGER
The first line contains an integer, N, denoting the number of elements in array.
N :: 1 -> 10^5

array :: INTEGER ARRAY
Each line i of the N subsequent lines (where $0 \leq i < N$) contains an integer describing i-th element of array.
array[i] :: 1 -> 10^6

Test Cases

Case#: 1

Input:
5
1
2
3
4
5
Output:
2

array = [1, 2, 3, 4, 5]. We can change 4 and 5 to make it [1, 2, 3, 2, 1]

Case#: 2

Input:
9
1
1
1
2
3
2
1
1
1
Output:
4

array = [1, 1, 1, 2, 3, 2, 1, 1, 1]. We can change the array to [-1, 0, 1, 2, 3, 2, 1, 0, -1]

Case#: 3

Input:
6
3
3
4
4
5
5
Output:
3

array = [3, 3, 4, 4, 5, 5]. We can change the array to [2, 3, 4, 4, 3, 2]

Sample Problem #5:

You have an interesting string S of length N. It is interesting because you can rearrange the characters of this string in any order. You want to cut this string into some contiguous pieces such that after cutting, all the pieces are equal to one another.

You can't rearrange the characters in the cut pieces or join the pieces together. You want to make the number of pieces as large as possible. What is the maximum number of pieces you can get?

Note: You can observe that you may not want to cut the string at all, therefore the number of pieces is 1. Hence, the answer always exists.

Parameters:

S :: STRING
The first line contains a string, S, denoting the string.
length(S) :: 1 -> 2 * 10^5

Test Cases

Case#: 1

Input:
zzzzz
Output:
5

You can cut it into 5 pieces "z" + "z" + "z" + "z" + "z"

Case#: 2

Input:
ababcc
Output:
2

Rearrange the string as abcabc. you can cut it into "abc" + "abc", hence the answer is 2

Case#: 3

Input:
abccdcabacda
Output:
2

Rearrange the string as "dcbaca" + "dcbaca", the answer is 2.