L6: Searching and Sorting Practice Questions

1-Tut : Aggressive Cows Problem

Send Feedback

Farmer John has built a new long barn, with N (2 <= N <= 100,000) stalls. The stalls are located along a straight line at positions x1,...,xN (0 <= xi <= 1,000,000,000).

His C (2 <= C <= N) cows don't like this barn layout and become aggressive towards each other once put into a stall. To prevent the cows from hurting each other, FJ wants to assign the cows to the stalls, such that the minimum distance between any two of them is as large as possible. What is the largest minimum distance ?

Input Format:

The first line of input contains a number of test cases, which will be denoted by the symbol t. In the following lines, the t test cases are written.

The first line of each of the test cases contain two space separated integers: N and C. The following line contains N space separated integers, where the ith integer denotes the position of ith stall.

It is given that the sum of N over all the test cases doesn't exceed 1000, 000.

Output Format:

As described in the problem statement, the first and only line of output will print the largest minimum distance possible.

The output for each test case will be printed on a separate line.

Sample Input 1:

```
2
11 3
15 5 2 4 17 16 12 8 19 18 11
15 13
20 8 16 3 13 9 11 10 15 17 18 14 1 2 5
```

Sample Output 1:

```
8

1
```

```
#include<bits/stdc++.h>
using namespace std;

bool check(int cows,long long positions[],int n,long long distance){

        int count = 1;
        long long last_position = positions[0];

        for(int i=1;i<n;i++){
                if(positions[i] - last_position >= distance){
                        last_position = positions[i];
                        count++;
                }
```

```

            if(count == cows){
                return true;
            }
        }
        return false;
}

int main(){
        int t;
        cin >> t;
        while(t--){
            int n,c;
            cin >> n >> c;

            long long positions[n];
            for(int i=0;i<n;i++){
                cin >> positions[i];
            }
            sort(positions,positions+n);
            long long start = 0;
            long long end = positions[n-1] - positions[0];

            long long ans = -1;

            while(start<=end){
                long long mid = start + (end-start)/2;

                if(check(c,positions,n,mid)){
                    ans = mid;
                    start = mid+1;
                }else{
                    end = mid-1;
                }

            }
            cout << ans <<endl;
        }

        return 0;
}
```

2-Tut : Inversion Count

Send Feedback

For a given integer array/list 'ARR' of size 'N', find the total number of 'Inversions' that may exist.

An inversion is defined for a pair of integers in the array/list when the following two conditions are met.

A pair ('ARR[i]', 'ARR[j]') is said to be an inversion when:
1. 'ARR[i] > 'ARR[j]'
2. 'i' < 'j'

Where 'i' and 'j' denote the indices ranging from [0, 'N').

Input format :

The first line of input contains an integer 'N', denoting the size of the array.
The second line of input contains 'N' integers separated by a single space, denoting the elements of the array 'ARR'.

Output format :

Print a single line containing a single integer that denotes the total count of inversions in the input array.

Note:

You are not required to print anything, it has already been taken care of. Just implement the given function.

Constraints :

1 <= N <= 10^5
1 <= ARR[i] <= 10^9

Where 'ARR[i]' denotes the array element at 'ith' index.
Time Limit: 1 sec

Sample Input 1 :

3
3 2 1

Sample Output 1 :

3

Explanation of Sample Output 1:

We have a total of 3 pairs which satisfy the condition of inversion. (3, 2), (2, 1) and (3, 1).

Sample Input 2 :

5
2 5 1 3 4

Sample Output 2 :

4

Explanation of Sample Output 1:

We have a total of 4 pairs which satisfy the condition of inversion. (2, 1), (5, 1), (5, 3) and (5, 4).

```
#include<iostream>
using namespace std;
long long merge(long long *A,int left,int mid,int right){

        int i=left,j=mid,k=0;
```

```

    int temp[right-left+1];
    long long count = 0;
    while(i<mid && j<=right){
        if(A[i] <= A[j]){
            temp[k++] = A[i++];
        }else{
            temp[k++] = A[j++];
            count += mid - i;
        }
    }
    while(i<mid){
        temp[k++] = A[i++];
    }
    while(j<=right){
        temp[k++] = A[j++];
    }

    for(int i=left,k=0;i<=right;i++,k++){
        A[i] = temp[k];
    }
    return count;
}
long long merge_sort(long long *A,int left,int right){

    long long count = 0;
    if(right > left){
        int mid = (left + right)/2;

        long long countLeft = merge_sort(A,left,mid);
        long long countRight = merge_sort(A,mid+1,right);
        long long myCount = merge(A,left,mid+1,right);

        return myCount + countLeft + countRight;
    }
    return count;

}

long long getInversions(long long *arr, int n){
  // Write your code here.
  long long ans = merge_sort(arr,0,n-1);
    return ans;
}
```

Live Problem -

1. https://www.codechef.com/problems/CHEFRES
2. https://www.codechef.com/ZCOPRAC/problems/ZCO15002

3-Ass : **Murder**

Send Feedback

Once detective Saikat was solving a murder case. While going to the crime scene he took the stairs and saw that a number was written on every stair. He found it suspicious and decides to remember all the numbers that he has seen till now. While remembering the numbers he found that he can find some pattern in those numbers. So he decides that for each number on the stairs he will note down the sum of all the numbers previously seen on the stairs which are smaller than the present number. Calculate the sum of all the numbers written in his notes diary.

Answers may not fit in integers . You have to take a long type.

Input Format:

First line of input contains an integer T, representing the number of test cases.
For each test case, the first line of input contains integer N, representing the number of stairs.
For each test case, the second line of input contains N space separated integers, representing numbers written on the stairs.

Constraints

T<=10^5
1<=N<=10^5
All numbers will be between 0 and 10^9
Sum of N over all test cases doesn't exceed 5*10^5

Output Format

For each test case output one line giving the final sum for each test case.

Sample Input 1:

```
1
5
1 5 3 6 4
```

Sample Output 1:

```
15
```

Explanation:

For the first number, the contribution to the sum is 0.
For the second number, the contribution to the sum is 1.
For the third number, the contribution to the sum is 1.
For the fourth number, the contribution to the sum is 9 (1+5+3).
For the fifth number, the contribution to the sum is 4 (1+3).

Hence the sum is 15 (0+1+1+9+4).

```
#include<iostream>
using namespace std;
typedef long long ll;

ll merge(ll *arr, ll left, ll mid, ll right)
{
        ll i=left;
        ll k=0;
        ll j=mid;
        ll sum=0;
        ll *temp=new ll [right-left+1];
        while(i<mid && j<=right)
        {
                if(arr[i]<arr[j])
                {
                        sum+=(arr[i]*(right-j+1));
                        temp[k++]=arr[i++];
                }
                else
                {
                        temp[k++]=arr[j++];
                }
        }

        while(i<mid)
        {
                temp[k++]=arr[i++];
        }

        while(j<=right)
        {
                temp[k++]=arr[j++];
        }

        for(ll i=left, k=0; i<=right; i++, k++)
        {
                arr[i]=temp[k];
        }

        delete[]temp;
        return sum;
}

ll merge_sort(ll *arr, ll left, ll right)
{
        ll count=0;
        if(right>left)
        {
                ll mid=(right+left)/2;
                ll left_count=merge_sort(arr, left, mid);
                ll right_count=merge_sort(arr, mid+1, right);
                ll merge_count=merge(arr, left, mid+1, right);

                return left_count+right_count+merge_count;
        }
        return count;
}

```

```
ll getAns(ll *arr, ll n)
{
	ll ans=merge_sort(arr, 0, n-1);
	return ans;
}

int main()
{
	ll t;
	cin>>t;
	while(t--)
	{
		ll n;
		cin>>n;
		ll *arr=new ll [n];
		for(ll i=0; i<n; i++)
		{
			cin>>arr[i];
		}
		cout<<getAns(arr, n)<<endl;
	}
}
```

4-Ass : **Distribute Candies**

Send Feedback

Shaky has N (1<=N<=50000) candy boxes each of them contains a non-zero number of candies (between 1 and 1000000000). Shaky want to distribute these candies among his K (1<=K<=1000000000) IIIT-Delhi students. He wants to distribute them in a way such that:

1. All students get an equal number of candies.

2. All the candies which a student gets must be from a single box only.

As he wants to make all of them happy so he wants to give as many candies as possible. Help Shaky in finding out what is the maximum number of candies which a student can get.

Input

First-line contains T the number of test cases.
The first line of each test case contains N and K.
Next line contains N space-separated integers, ith of which is the number of candies in the ith box.

Output

For each test case print the required answer in a separate line.

Constraints

1<= T <= 10^5
1 <= N <= 10^5
1 <= K <= 10^9
0 <= A[i] <= 10^9
Sum of N over all test cases doesn't exceed 10^6

Sample Input:

```
2
3 2
3 1 4
4 1
3 2 3 9
```

Sample Output:

```
3

9
```

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;

bool check(ll arr[],ll n,ll k,ll range){
    ll count = 0;
    for(ll i = 0;i<n;i++){
        count += arr[i]/range;  //possibility of box can accommodate more than one student
        if(count >=k){
            return true;
        }
    }
    return false;
}
int main() {
    int t;
    cin >> t;
    while(t--){
        ll n,k;
        cin >> n >> k;
        ll arr[n];
        for(ll i=0;i<n;i++){
            cin >> arr[i];
        }
        sort(arr,arr+n);
        ll start = 0;
        ll end = arr[n-1];
        ll ans = -1;
        while(start <= end){
            ll mid = (start + end )/2;
            if(check(arr,n,k,mid)){
                ans = mid;
                start = mid+1;
            }else{
                end = mid -1;
            }
        }
        cout << ans << endl;
    } }
```

5-Ass : **Momos Market**

Send Feedback

Shreya loves to eat momos. Her mother gives her money to buy vegetables but she manages to save some money out of it daily. After buying vegetables, she goes to "Momo's Market", where there are ‘n’ number of momos' shops of momos. Each of the shops of momos has a rate per momo. She visits the market and starts buying momos (one from each shop) starting from the first shop. She will visit the market for ‘q’ days. You have to tell that how many momos she can buy each day if she starts buying from the first shop daily. She cannot use the remaining money of one day on some other day. But she will save them for other expenses in the future, so, you also need to tell the sum of money left with her at the end of each day.

Input Format:

First-line will have an integer ‘n’ denoting the number of shops in the market.
The next line will have ‘n’ numbers denoting the price of one momo of each shop.
The next line will have an integer ‘q’ denoting the number of days she will visit the market.
Next ‘q’ lines will have one integer ‘X’ denoting the money she saved after buying vegetables.

Constraints:

1 <= n <= 10^5
1 <= q <= 10^4
1 <= X <= 10^9
1 <= rate per momo <=10^9 (for each shop)

Output:

There will be ‘q’ lines of output each having two space separated integers denoting number of momos she can buy and amount of money she saved each day.

Sample Input:

```
4
2 1 6 3
1
11
```

Sample Output:

```
3 2
```

Explanation:

Shreya visits the "Momo's Market" for only one day. She has 11 INR to spend. She can buy 3 momos, each from the first 3 shops. She would 9 INR (2 + 1 + 6) for the same and hence, she will save 2 INR.

```
#include<iostream>

using namespace std;
typedef long long int ll;
bool checker(ll *arr, ll n, ll mid, ll target)
{
        if(arr[mid]<=target)        {
                return true;
        }
```

```
            return false;
}
void momos(ll *arr, ll n, ll target)
{
            ll start=0;
            ll end=n-1;
            ll mid;
            while(start<end)
            {
                        mid=(start+end)/2;
                        if(checker(arr, n, mid, target))
                        {
                                    start=mid;
                        }
                        else
                        {
                                    end=mid;
                        }
            if(end-start==1)
                        {
                                    if(checker(arr, n, end, target))
                                    {
                                                cout<<end+1<<" "<<target-arr[end]<<endl;
                                                return;
                                    }
                                    else
                                    {
               if(!checker(arr, n, start, target))
               {
                   cout<<0<<" "<<target<<endl;
                   return;
               }
                                                cout<<start+1<<" "<<target-arr[start]<<endl;
                                                return;
                                    }
                        }
            }
    cout<<mid+1<<" "<<target-arr[mid]<<endl;
    return;
}
int main()
{
            ll n;
            cin>>n;
            ll *prices=new ll [n];
            for(ll i=0; i<n; i++)
            {
                        cin>>prices[i];
            }
            ll q;//number of days.
            cin>>q;
            ll *money=new ll [q];
            for(ll i=0; i<q; i++)
            {
                        cin>>money[i];
            }
            ll *prefix_sum_prices=new ll [n];
            ll sum=0;
            for(ll i=0; i<n; i++)
            {
            sum+=prices[i];
                        prefix_sum_prices[i]=sum;
            }
            for(ll i=0; i<q; i++)
            {
            momos(prefix_sum_prices, n, money[i]);
            }
}
```

6-Ass : Taj Mahal Entry

Send Feedback

Taj Mahal is one of the seven wonders of the world. Aahad loves to travel places and wants to visit the Taj Mahal. He visited Agra to view the Taj Mahal. There is a ticketing system at Taj Mahal. There are total ‘n’ windows which provide the tickets to get entry into Taj Mahal. There are ‘Ai’ people already present at each window to get the tickets. Each window gives a ticket to one person in one minute. Initially, Aahad stands in front of the first window. After each minute, if he didn’t get the ticket, he moves on to the next window to get the ticket. If he is at window 1, he will move to 2. If at 2nd, he will move to 3rd. If he is at last window, he will move to 1st again and so on. Find the window number at which he will get the ticket.

Input Format:

First line contains a single integer ‘n’ denoting the no. of windows.
Next line contains ‘n’ space separated integers denoting the no. of people already standing in front of the ith window. (1 <= i <= n)

Output Format:

Print a single integer denoting the window number that Aahad will get the ticket from.

Constraints:

1 <= n <= 10^5
1 <= Ai <= 10^9

Sample Input:

```
4
2 3 2 0
```

Sample Output:

```
3
```

Explanation:

Aahad at Window 1: [2, 3, 2, 0]

Aahad at Window 2: [1, 2, 1, 0]
Aahad at Window 3: [0, 1, 0, 0]

So, when Aahad is at window 3, he got zero people before him. Hence, he will get the ticket at window 3.

```
#include<iostream>
using namespace std;
typedef long long int ll;
int main()
{
        int n;
        cin>>n;
        int *arr=new int [n];
        for(int i=0; i<n; i++)
        {
                cin>>arr[i];
        }
        int*t=new int [n];
        for(int i=0; i<n; i++)
        {
                if((arr[i]-i)%n==0)
                {
                        t[i]=(arr[i]-i)/n;
                }
     else if(arr[i]-i<0)
     {
        t[i]=0;
     }
                else
                {
                        t[i]=((arr[i]-i)/n)+1;
                }
        }
        int *ans=new int [n];
        int minimum=9999999999;
        int minimum_index=-1;
        for(int i=0; i<n; i++)
        {
                ans[i]=i+t[i]*n;
                if(ans[i]<minimum)
                {
                        minimum=ans[i];
                        minimum_index=i;
                }
        }
        cout<<minimum_index+1;

}
```

Method-2 :

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n;
    cin>>n;
    int w[n];
    for(int i=0;i<n;i++){
        cin>>w[i];
    }
    int del=0;
    for(int i=0;;i++){
        i=i%n;
        w[i]-=del;
        del++;
        if(w[i]<=0){
            cout<<i+1<<endl;
            break;
        }
    }
         return 0;
}
```

7-Ass : **Sorting the Skills**

Send Feedback

There is a company named James Peterson & Co. The company has ‘n’ employees. The employees have skills, which is denoted with the help of an integer. The manager of James Peterson & Co. wants to sort the employees on the basis of their skills in ascending order. He is only allowed to swap two employees which are adjacent to each other. He is given the skills of employees in an array of size ‘n’. He can swap the skills as long as the absolute difference between their skills is 1. You need to help the manager out and tell whether it is possible to sort the skills of employees or not.

Input Format:

First Line will have an integer ‘t’ denoting the no. of test cases.
First line of each test case contains an integer ‘n’ denoting the no. of employees in the company.
Second line of each test case contains ‘n’ integers.

Output Format:

For each test case, print “Yes” if it is possible to sort the skills otherwise “No”.

Constraints:

1 <= t <= 10^4
1 <= n <= 10^5
Sum of n over all test cases doesn't exceed 10^6

Sample Input:

```
2
4
```

```
1 0 3 2
3
2 1 0
```

Sample Output:

```
Yes
No
```

Explanation:

In first T.C., [1, 0, 3, 2] -> [0, 1, 3, 2] -> [0, 1, 2, 3]

In second T.C., [2, 1, 0] -> [1, 2, 0] OR [2, 1, 0] -> [2, 0, 1] So, it is impossible to sort.

```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
        int t;
    cin >> t;
        while(t--)
        {
                int n;
        cin >> n;
                vector<int> a(n);
                for(int i = 0 ; i < n ; ++i) {
                        cin >> a[i];
                }

                string ans = "Yes";

                for(int i = 0 ; i < n-1 ; ++i) {
                        if(a[i] > a[i+1]) {
                                if(a[i] - a[i+1] == 1) {
                                        swap(a[i], a[i+1]);
                                }
                                else {
                                        ans = "No";
                                }
                        }
                }

                cout << ans << '\n';
        }
        return 0;
}
```

8-Ass : Collecting the balls

Send Feedback

There are ‘n’ numbers of balls in a container. Mr. Sharma and Singh want to take balls out from the container. At each step, Mr. Sharma took ‘k’ balls out of the box and Mr. Singh took one-tenth of the remaining balls. Suppose there are 29 balls at the moment and k=4. Then, Mr. Sharma will take 4 balls and Mr. Singh will take 2 balls (29-4 = 25; 25/10 = 2). If there are less than ‘k’ balls remaining at some moment, then Mr. Sharma will take all the balls which will get the container empty. The process will last until the container becomes empty. Your task is to choose a minimal ‘k’ for Mr. Sharma such that Mr. Sharma will take at least half of the balls from the container.

Input Format:

The first line of input will contain T (number of test cases).
The next n lines of input contain a single integer ‘n’.

Output Format:

For every test case print a single integer denoting the minimal value of ‘k’ in a newline.

Constraints:

1 <= T <= 10^4
1 <= n <= 10^18
Time Limit: 1 second

Sample Input:

1
68

Sample Output:

3

Explanation:

68-3 = 65; 65/10 = 6; 65-6 = 59
59-3 = 56; 56/10 = 5; 56-5 = 51
51-3 = 48; 48/10 = 4; 48-4 = 44
44-3 = 41; 41/10 = 4; 41-4 = 37
…..
…..
…..
6-3 = 3; 3/10 = 0; 3-0 = 3

3-3 = 0; 0/10 = 0; 0-0 = 0

```
#include <bits/stdc++.h>
using namespace std;
#define int long long
#define double long double

bool check(const int& k, const int& n) {
        int sharma = 0, singh = 0;

        int x = n, step = 1;
        while(x) {
```

```
        // sharma's turn
        sharma += min(k, x);
        x -= min(k, x);

        // singh's turn
        singh -= (x / 10);
        x -= (x / 10);
    }
    return (2*sharma >= n);
}

int32_t main()
{
    ios::sync_with_stdio(0); cin.tie(0); cout.tie(0);

    int t; cin >> t;
    while(t--)
    {
        int n; cin >> n;

        int ans = n;

        int left = 1, right = n;
        while(left <= right) {
            int mid = left + (right - left) / 2;
            if(check(mid, n)) {
                ans = mid;
                right = mid - 1;
            }
            else {
                left = mid + 1;
            }
        }
        cout << ans << '\n';
    }
    return 0;
}
```

Note : when replacing with int it is giving TLE for some test cases