August 3, 2025

# BINARY SEARCH APPLICATIONS

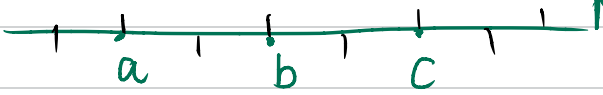
## Ques 1. Points on a line. https://codeforces.com/problemset/problem/251/A (BS on every start)

Little Petya likes points a lot. Recently his mom has presented him $n$ points lying on the line $OX$. Now Petya is wondering in how many ways he can choose three distinct points so that the distance between the two farthest of them doesn't exceed $d$.

Note that the order of the points inside the group of three chosen points doesn't matter.

### Approach / Logic Used:

N points are given. We have to figure out in how many ways we can choose 3 distinct points from those n points so that distance b/w the first point and the third point $\leq d$.



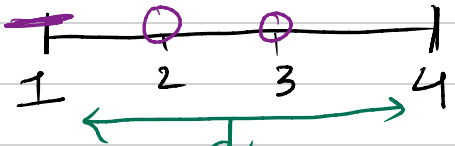

N points, a, b, c are 3 distinct points.

Let us take 'a' as the first point (fixed), then our job simply becomes to find out how many triplets can be created $\leq d$. (a, _, _)
Then, we can continue to find such triplets for every point.

Example: N = 4, d = 3, Points are: 1, 2, 3, 4

[1, 2, 3, 4]

→ 1 is the first point, the farthest we can go from 1 is till 4 (4 − 1 = 3 = d)




No. of points b/w 1 & 4 = 2.
So, for a[0] = 1, we can form triplets in $^3C_2$ ways.
And so on for other cases.

```
void solve(){
    int n,d;
    cin>>n>>d;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }
    long long ans = 0;
    for(int i=0;i<n;i++){
        // i [i+1....x] idx = (x+1)  arr[x+1] > arr[i] + d
        int idx = upper_bound(arr,arr+n,arr[i]+d)-arr;
        long long cnt = idx - i - 1;
        ans += cnt*(cnt-1)/2;
    }
    cout<<ans<<endl;
}
```

Find ele > arr[i]+d

count no. of points b/w i & i+d

$^xC_2$

↳ cumulate all points for each index

If current element is at i, the farthest point we can pick is i+d. Then we are simply finding out no. of points from i+1 ... x such that arr[x] <= arr[i] + d.

AlgoZenith

# Ques 2. Factory Machines https://cses.fi/problemset/task/1620

A factory has $n$ machines which can be used to make products. Your goal is to make a total of $t$ products.

For each machine, you know the number of seconds it needs to make a single product. The machines can work simultaneously, and you can freely decide their schedule.

What is the shortest time needed to make $t$ products? Minimise Time to make t products

Such class of problems where we have to minimize/maximise the output comes under Binary Search on Answer.

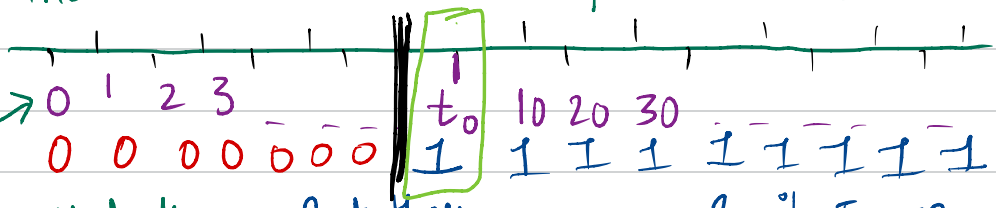
First thing is to identify, what is the output? In this case, it is Time. Time is our answer space here i.e. where our answer will lie. Let us define the constraints lo & hi of our answer space.

lo = 0

hi = K · min($a_i$)

Max val in my answer space

The numbers on this axis represents time.




→ Can we make t products in 0s? No

→ Can we make t products in 1s? No

→ What about 3? No.

But there comes an optimal time $t_0$ in which we can make t products.

So, if I can make t products in $t_0$ time. Then I can surely make it in $t_0+1$ time resulting 1's.

As you can see above, we used our search space (answer space) to compute the logic of check which shall give us 0's followed by 1's. Our problem is now reduced to 1. Building this check fx^n^
2. finding the first 1 which shall represent shortest time to build t products.

Note that we are given an array of time ($a_i$ → time needed for a machine to complete a product)

So, No. of product that machine i will generate in x time = $\lfloor \frac{x}{a_i} \rfloor$ products.

For all the given machines, it will be

$$\Rightarrow \sum \left\lfloor \frac{x}{a_i} \right\rfloor$$

check(x) : Can we generate K products in x time.

This is the total products that will be generated for N Machines given for a random x in our answer space.

So, In order to have 0's, 1's space, all we have to check is if this total is greater than or equal to K. If it is then we shall return 1 else 0.

Condition for our check fxn.

$$\Rightarrow \sum \left\lfloor \frac{x}{a_i} \right\rfloor \geq K$$

O(N) check.

```
int check(int x){ O(N)
    int product = 0;
    for(int i=0;i<n;i++){
        product += x/arr[i];
        if(product>k)return 1;
    }
    return 0;
}
void solve(){
    cin>>n>>k;
    int minelem = 1e9;
    for(int i=0;i<n;i++){
        cin>>arr[i];
        minelem = min(minelem,arr[i]);
    }
    int lo = 0;
    int hi = minelem*k;
    int ans = -1;
    while(lo<=hi){
        int mid = (lo+hi)/2;
        if(check(mid)){
            ans = mid;
            hi = mid-1;
        }else{
            lo = mid+1;
        }
    }
    cout<<ans<<endl;
```

→ calculating product generated by i Machine

→ return b/c K products can be generated.

## Ques 3. Painter Partition Problem

https://maang.in/problems/Famous-Painter-Partition-Problem-472

Same problem as earlier one. Axis of answer = Time.

Check(x): Can we paint in x time using ≤ K painters

```
int check(int x){
    int painter = 0;
    int timeleft = 0;
    for(int i=0;i<n;i++){
        if(timeleft >= arr[i]){
            timeleft -= arr[i];
        }else{
            painter++;
            timeleft = x-arr[i];
        }
    }
    if(painter<=k)return 1;
    else return 0;
}
void solve(){
    cin>>n>>k;
    int maxelem = 0;
    int sum = 0;
    for(int i=0;i<n;i++){
        cin>>arr[i];
        maxelem = max(maxelem,arr[i]);
        sum += arr[i];
    }
    int lo = maxelem;
    int hi = sum;
    int ans = -1;
    while(lo<=hi){
        int mid = (lo+hi)/2;
        if(check(mid)){
            ans = mid;
            hi = mid-1;
        }
        else lo = mid+1;
    }
    cout<<ans;
}
```

O(N)

Calculate no. of painters needed to finish a job in x time.

If that num ≤ K, that means it can be done in x time with K painters or less, return 1 it is a possible answer.

≈ O(N log N)

# Ques 4. Multiplication Table https://codeforces.com/problemset/problem/448/D

Bizon the Champion isn't just charming, he also is very smart.

While some of us were learning the multiplication table, Bizon the Champion had fun in his own manner. Bizon the Champion painted an $n \times m$ multiplication table, where the element on the intersection of the $i$-th row and $j$-th column equals $i \cdot j$ (the rows and columns of the table are numbered starting from 1). Then he was asked: what number in the table is the $k$-th largest number? Bizon the Champion always answered correctly and immediately. Can you repeat his success?

Consider the given multiplication table. If you write out all $n \cdot m$ numbers from the table in the non-decreasing order, then the $k$-th number you write out is called the $k$-th largest number.

Example:

- N=2, M=2, K=2

| | 1 | 2 |
|---|---|---|
| 1 | 1 | 2 |
| 2 | 2 | 4 |

→ Ans

1, 2, 2, 4

k=2

- N=2, M=3, K=4

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 1 | 2 | 3 |
| 2 | 2 | 4 | 6 |

→ 1, 2, 2, 3, 4, 6 (positions 1 2 3 4 5 6)

Ans = 3.

Here, in this problem, the axis of answer is a number.

lo = 1 smallest

hi = n*m (largest val in our answer space)

Check(x): [[(No. of value ≤ x)]] ≥ k (if true return 1 else return 0)

In ex 2: 1, 2, 2, 3, 4, 6

check(2) → #ele≤2 = 3 [is 3 ≥ k(4) ⇒ NO]

return 0

lo = mid+1

Check(3) → #ele≤3 = 4 [is 4 ≥ k, yes]

Return 1, found possible ans.

Check(4) → 1

Check(5) → 1

finding the first 1 will solve our problem.

For an i-th row, multiples of $i \le x$ can be represented as $\left\lfloor \frac{x}{i} \right\rfloor$

We can use min(x/i, m) to calculate #ele≤x in a row i across n*m

```
int n,m,k;
int check(int x){  O(N)
    // (# of val <=x) >=k
    int cnt= 0 ;
    for(int i=1;i<=n;i++){
        cnt += min(x/i,m);
    }
    if(cnt >= k)return 1;
    else return 0;
}
```

```
void solve(){
    cin>>n>>m;

    int lo = 1;
    int hi = n*m;
    int ans = -1;

    while(lo<=hi){
        int mid = (lo+hi)/2;
        if(check(mid)){
            ans = mid;
            hi = mid-1;
        }else{
            lo = mid+1;
        }
    }
    cout<<ans<<endl;
```

O(NlogN)

AlgoZenith