Binary and Ternary Search

3 4 7 9

3 7 14 23

Q. Given an array of n positive integers. Your task is to divide it into k contiguous non-empty segments so that the maximum sum on the segment is the minimum possible.
Link:
https://codeforces.com/edu/course/2/lesson/6/3/practice/contest/285083/problem/B

k<=n

2 3 4 6

Ans >= 6 ans Ans <= 15

Left = 6 and right = 15

Between = 6 + 15/2 = 10

If we can divide our segments such that the maximum sum of our segments is <= 10.

2 3 4 // 9
6 // 6

Left = 6 and Right 10

Between = (6+10)/2= 8

2 3 // 5
4 // 4

6 // 6

Not Possible

Left = 9 Right = 10
Middle = 19/2 = 9

2 3 4 // 9
6 // 6

Left = 9 right = 9
Break ans = 9

Counter case: 1 1 1 1 1 1 1 1 1 1 10

```
int count(vector<int> &v, int lim)
{
    int c = 1, cursum = 0;
    for (auto num : v)
    {
        if (cursum + num > lim)
        {
            c++;
            cursum = num;
        }
        else
            cursum += num;
    }
    return c;
}

int maxsum(vector<int> &v, int k)
{
    int n = v.size();
    int sum = 0, mx = 0;
    for (auto num : v)
    {
        mx = max(mx, num);
        sum += num;
    }
    int lb = mx, ub = sum, ans = sum;
    while (lb <= ub)
```

```
        {
            int md = lb + (ub - lb) / 2;
            if (count(v, md) <= k)
            {
                ans = md;
                ub = md - 1;
            }
            else
            {
                lb = md + 1;
            }
        }
        return ans;
    }
```

Aggressive Cows -
https://www.spoj.com/problems/AGGRCOW/

```
    double eps = 1e-9;
    while (ub - lb > eps)
    {
        double md = lb + (ub - lb) / 2;
        if (count(v, md) <= k)
        {
            ans = md;
            ub = md;
        }
        else
        {
            lb = md;
        }
    }

    double eps = 1e-9;
    double lb = 0, ub = 1e9;
    while (abs(ub - lb) > eps)
    {
        double md = lb + (ub - lb) / 2;
        if (md * md <= n)
        {
```

```
            ans = md;
            lb = md;
        }
        else
        {
            ub = md;
        }
    }
```

O(log((ub-lb)/eps))

Q. There are n people on a straight line. They need to gather at one point. The i-th person's current position is x[i] and his maximum speed v[i]. Find out the minimum time they require to gather at the same point.
Link:
https://codeforces.com/edu/course/2/lesson/6/3/practice/contest/285083/problem/A

1 2 3 4
2 1 6 3
x = 2
0.5 0 0.16 0.67

[x[i]-md*v[i],x[i]+md*v[i]]

```
    bool check(double t)
    {
        double l[n], r[n];
        for (int i = 0; i < n; i++)
        {
            l[i] = x[i] - t * v[i];
            r[i] = x[i] + t * v[i];
        }
        double li = l[0], ri = r[0];
        for (int i = 0; i < n; i++)
        {
            li = max(li, l[i]);
```

```
            ri = min(ri, r[i]);
        }
        if (ri - li > eps || abs(ri - li) <= eps)
            return true;
        else
            return false;
    }

    double eps = 1e-9;
    double lb = 0, ub = 2e9;
    while (abs(ub - lb) > eps)
    {
        double md = lb + (ub - lb) / 2;
        if (check(md))
        {
            ans = md;
            ub = md;
        }
        else
        {
            lb = md;
        }
    }
```

if(ri>li)

if(ri+eps>li)

for(int i=0;i<;i++)
{

}
if(r>l-eps)

1 4
3 7 3 4
5 10 5 4

https://atcoder.jp/contests/zone2021/tasks/zone2021_c

if(ri-li>-eps)

ri>=li

1e-6

1e-9

2 - 1e-20

2 + 1e-20

```
10  0  0  0  0
0  10  0  0  0
0  0  10  0  0
0  0  0  10  0
0  0  0  0  10
```