📝 Revision Notes – Range of K Problem

🔷 Problem Statement (mere words mein)

Mujhe ek **array** diya gaya hai.

Mujhe ek aisa **range of K** find karna hai jisme agar main array ke har element se `K` subtract karu aur **modulus (absolute value)** le lu, to jo naye elements banenge wo **sorted (non-decreasing)** honi chahiye.

Formally:

Agar array = `{a1, a2, a3, ..., an}` hai,

to mujhe `K` aisa chahiye jisme:

$$|a1 - K| \le |a2 - K| \le |a3 - K| \le \ldots \le |an - K|$$

🔷 Maths Approach

Step 1: Pairwise Comparison

Har consecutive pair `(ai, ai+1)` ke liye condition banegi:

$$|ai - K| \le |ai+1 - K|$$

Step 2: Squaring Trick

Absolute values ko handle karne ke liye main dono side **square** kar sakta hoon (kyunki squaring monotonic hai for non-negative numbers):

$$(ai - K)^2 \le (ai+1 - K)^2$$

Step 3: Expanding Both Sides

$$(ai - K)^2 \le (ai+1 - K)^2$$
$$ai^2 - 2aiK + K^2 \le ai+1^2 - 2ai+1K + K^2$$

Cancel K^2 (dono side same):

$$ai^2 - 2aiK \le ai+1^2 - 2ai+1K$$

Step 4: Rearranging

$$2(ai+1 - ai)K \le ai+1^2 - ai^2$$

Now factorize RHS:

$$2(ai+1 - ai)K \le (ai+1 - ai)(ai+1 + ai)$$

Step 5: Cases based on (ai+1 – ai)

- **Case 1: ai ≥ ai+1 (non-increasing pair)**

  Then $(ai+1 - ai)$ ≤ 0

  Inequality flips when dividing:

  $$K \ge (ai + ai+1) / 2$$

  → Means yaha mujhe **lower bound (kmin)** milega.

- **Case 2: ai < ai+1 (increasing pair)**

  Then $(ai+1 - ai)$ > 0

  $$K \le (ai + ai+1) / 2$$

  → Yaha mujhe **upper bound (kmax)** milega.

🔷 Algorithm Summary

1. Initialize:

   ```
   kmin = -∞
   kmax = +∞
   ```

2. Har consecutive pair check karo:
   - Agar arr[i] ≥ arr[i+1] → update kmin
   - Agar arr[i] < arr[i+1] → update kmax
3. Agar kabhi `kmin > kmax` ho gaya → **No solution**
4. Otherwise, final interval = `[ceil(kmin), kmax]`

🔷 Code (final form)

```
#include<iostream>
#include<climits>
#include<algorithm>
using namespace std;

int main() {
    int arr[] = {5, 4, 2, 100, 150, 200, 300};
    int n = sizeof(arr)/sizeof(arr[0]);

    float kmin = (float)INT_MIN;
    float kmax = (float)INT_MAX;
    bool flag = true;

    for(int i = 0; i < n-1; i++) {
        if(arr[i] >= arr[i+1]) {
            kmin = max(kmin, (float)((arr[i] + arr[i+1]) / 2.0));
        } else {
            kmax = min(kmax, (float)((arr[i] + arr[i+1]) / 2.0));
        }
```

```
        if(kmin > kmax) {
            flag = false;
            break;
        }
    }

    if(!flag) {
        cout << "No limit exist!";
    } else {
        cout << "[" << (int)kmin+1 << "," << kmax << "]";
    }

    return 0;
}
```

🔷 Dry Run (for `{5, 4, 2, 100, 150, 200, 300}`)

- **Init:** `kmin = -∞`, `kmax = +∞`

1. Pair (5,4): dec → `kmin = max(-∞, (5+4)/2 = 4.5) = 4.5`
2. Pair (4,2): dec → `kmin = max(4.5, (4+2)/2 = 3.0) = 4.5`
3. Pair (2,100): inc → `kmax = min(+∞, (2+100)/2 = 51.0) = 51.0`
4. Pair (100,150): inc → `kmax = min(51.0, (100+150)/2 = 125.0) = 51.0`
5. Pair (150,200): inc → `kmax = min(51.0, (150+200)/2 = 175.0) = 51.0`
6. Pair (200,300): inc → `kmax = min(51.0, (200+300)/2 = 250.0) = 51.0`

- Final: `kmin = 4.5`, `kmax = 51.0`
- Range = `[ (int)4.5 + 1 , 51 ] = [5, 51]`

🔷 Final Answer

For array `{5, 4, 2, 100, 150, 200, 300}`,

valid range of K = **`[5, 51]`**

👉 ye notes complete hai: problem explanation, maths derivation, code, aur dry run.

Revision ke liye ekdum perfect.

---

kya mai tujhe ek **visual graph explanation** bhi bana kar de du jisme line number line pe dikhe ki `kmin` aur `kmax` kaise adjust hote hain?