SOLUTION DAY 23/10

1. SEQ

Trước khi đến với bài này thì ta phải làm được bài toán nếu chỉ cần tìm 1 đoạn có độ dài chia hết cho 3 và tổng trọng số lớn nhất.

Ta sẽ có 1 mảng f gồm 3 phần tử là f[0], f[1], f[2]

Với f[i] là phần tử có giá trị nhỏ nhất sau khi cộng dồn và chỉ số của phần tử đó chia 3 dư i.

Khởi tạo f[0]=0, f[1]=INT_MAX, f[2]=INT_MAX.

Với mỗi i: res=max(res,a[i]-f[i%3]).

f[i%3]=min(f[i%3],a[i])

Quay lại với vấn đề chính, với hai dãy ta sẽ giải quyết như sau:

- Ý tưởng trâu: tìm điểm cắt chia dãy này ra làm hai, bên trái và bên phải, tìm kết quả của dãy bên trái như bài trên (tìm một dãy mà có độ dài chia hết cho 3) và trọng số lớn nhất gọi đó là res1. Tương tự, tìm kết quả bên phải, gọi đó là res2

Như vậy: res=max(res,res1+res2)

Độ phức tạp: $O(n^2)$.

- Ý tưởng tối ưu:

Ta sẽ gọi f1[i] là kết quả của dãy từ 1→i có một đoạn độ dài chia hết cho 3 và có trọng số lớn nhất, f2[i] là kết quả của dãy từ n→i có một đoạn độ dài chia hết cho 3 và có trọng số lớn nhất. Ta cũng làm như bài trên đưa về 1 dãy.

```
f[0]=0, f[1]=INT_MAX, f[2]=INT_MAX, res=INT_MIN
For(i,1,n)
{
        res=max(res,a[i]-f[i%3]);
        f1[i]=res;
        f[i%3]=min(a[i],f[i%3]);
} // mảng a là mảng cộng dồn
```

Thêm 1 mảng b dùng để lưu các giá trị của dãy a ban đầu theo chiều ngược lại tức là b[1]=a[n], b[2]=a[n-1],…,b[n]=a[1] //lưu ý mảng a ban đầu trước khi cộng dồn.

Sau đó ta thực hiện cộng dồn mảng b.

```
For(i,n,1)
{
    res=max(res,b[i]-f[i%3]);
    f2[i]=res;
    f[i%3]=min(b[i],f[i%3]);
}
```

Đảo ngược mảng f2, kết quả chính là: res=max(res,f1[i]+f2[i+1])

Độ phức tạp: O(n)

2. Evensub

Subtask 1: Với subtask này ta có thể làm thuật toán $O(n^2)$ đơn giản là duyệt từng đoạn (i, j) và tìm max min bằng cách cố định i và chạy j.

Subtask 2: Vì chỉ có 3 giá trị của a_i là 1, 2, 3, những dãy thỏa mãn là những dãy con liên tiếp hoặc chỉ gồm những số giống nhau, hoặc chỉ chứa những số có giá trị là 1 và 3. Như vậy những dãy không thỏa mãn phải chứa hoặc là 1 và 2, hoặc là 2 và 3. Nếu đếm được số dãy con liên tiếp có phải chứa hai số 1 và 2 là s_{12}, số dãy con liên tiếp phải chứa hai số 2 và 3 là s_{23}, thì kết quả số dãy thỏa mãn là:

$$ans = \frac{n \times (n+1)}{2} - s_{12} - s_{23}$$

Vậy giờ ta cần tìm cách tính được s_{12} và s_{23}. Xét tính s_{12}:

- Gọi s_1 là số dãy con liên tiếp chỉ chứa số 1, s_2 là số dãy con liên tiếp chỉ chứa số 2.
  - Ta có thể tính được s_1 bằng cách tìm các đoạn $[i, j]$ mà $\forall i \leq k \leq j$ có $a_k == 1$ và $a_{i-1} \neq 1$, $a_{j+1} \neq 1$ (1), số dãy chỉ chứa số 1 trong đoạn $[i, j]$ là:

$$d(1, i, j) = \frac{(j - i + 1) \times (j - i + 2)}{2}$$

  - Vậy số lượng s_1 sẽ là tổng của các $d(1, i, j)$
  - Tương tự tính được s_2
- Gọi $\overline{s_{12}}$ là số dãy con liên tiếp có thể chứa hoặc số 1 hoặc số 2 hoặc cả 1 và 2. Tương tự ta cũng có thể tính được $\overline{s_{12}}$ như tính s_1. Sau khi tính được $\overline{s_{12}}$, s_1 và s_2 ta có thể tính được s_{12}:

$$s_{12} = \overline{s_{12}} - s_1 - s_2$$

Tương tự tính được s_{23}. Vì ta có thể tìm được các đoạn $[i, j]$ thỏa mãn (1) trong $O(n)$, và số lần cần tìm sẽ là khi tính s_1, s_2, s_3, $\overline{s_{12}}$ và $\overline{s_{23}}$ nên thuật toán chỉ là $O(n)$.

Subtask 3: Có thể có nhiều cách làm để đạt được full điểm bài này, ở đây tôi đề xuất thuật toán chia để trị vào giải quyết được bài toán này:

– và để tránh xét thiếu các j mà $maxL[i] < maxR[j]$ thì ta chỉ cần làm thêm là xét từng j và đếm số lượng i thỏa mãn, để tránh hai lần tính bị trùng lặp các cặp (i, j) thì ta chỉ xét những giá trị i với $maxL[i] < maxR[j]$.

- Với i đang xét ta sẽ tìm được j_m là giá trị lớn nhất có $maxL[i] \geq maxR[j_m]$, ta biết min của các số bên trái là $minL[i]$, vậy giá trị j xác định min trong đoạn $[i, j]$ sẽ hoặc chính là $minL[i]$ hoặc là $minR[j]$:
  - Gọi j_s là vị trí lớn nhất mà $minL[i] \leq minR[j_s]$, vậy những $mid < j \leq j_s$ thì min của đoạn $[i, j]$ sẽ là $minL[i]$, vậy nếu $maxL[i] - minL[i]$ là số chẵn thì số lượng j thỏa mãn chính là $j_s - mid$, ngược lại thì 0 có giá trị j nào thỏa mãn.
  - Với $j_s < j \leq j_m$ thì min của đoạn $[i, j]$ sẽ là $minR[j]$, vậy các giá trị j thỏa mãn sẽ phụ thuộc vào $maxL[i]$ là chẵn hay lẻ. Do đó ta cần một mảng cnt để đếm số lượng giá trị $minR[j]$ là chẵn hay lẻ. Từ đó có thể đếm được có bao nhiêu giá trị $j_s < j \leq j_m$ thỏa mãn.
  - mảng cnt sẽ được cập nhật theo j_m và j_s, khi ta duyệt i giảm dần thì j_m và j_s sẽ tăng dần nên ta sẽ cần cập nhật cnt với tất cả i trong $r - mid$ lần.
- tương tự ta cũng có thể đếm i khi duyệt j theo cách trên.

- Giả sử ta đang tính đến đoạn $[l, r]$ và chia ra thành 2 phần là $[l, mid]$ và $[mid + 1, r]$, giờ ta cần đếm số cặp (i, j) với $l \leq i \leq mid < j \leq r$ sao cho thỏa mãn.
- Ta xây dựng 4 mảng sau:
  - $maxL[i]$ là số lớn nhất trong đoạn $[i, mid]$
  - $minL[i]$ là số nhỏ nhất trong đoạn $[i, mid]$
  - $maxR[i]$ là số lớn nhất trong đoạn $[mid + 1, r]$
  - $minR[i]$ là số nhỏ nhất trong đoạn $[mid + 1, r]$
- Nếu ta duyệt i và với mỗi i đếm xem có bao nhiêu giá trị j thỏa mãn, tuy nhiên ta thấy rằng nếu xét tất cả $mid < j \leq r$ thì rất khó xác định được max và min của đoạn $[i, j]$:
  - do đó ta chỉ xét j với $maxL[i] \geq maxR[j]$ để dễ dàng xác định được max và min của đoạn $[i, j]$

3. MEDSUM

Alg1.

+ Duyệt 2 vòng for sinh ra n^2 dãy con mỗi dãy 2xL phần tử

+ Sắp xếp N^2 dãy này

+ Res = (res + c[L]) % 10^9

Độ phức tạp: O(N^2.(2L)log(2L))

Alg2.

+ Duyệt 2 vòng for sinh ra N^2 dãy con, mỗi dãy con có độ dài 2xL phần tử

+ Duyệt 2 con trỏ tìm phần tử thứ L của mỗi dãy con

+ Res = (res + c[L]) % 10^9

Độ phức tạp: O(N^3)

4. EQUAL

SUB 1

Đầu tiên ta thấy rằng con số để biến đổi cả mảng về sẽ là số lớn hơn hoặc bằng số nhỏ nhất của mảng và bé hơn hoặc bằng số lớn nhất của mảng. Quan sát này dễ nhìn thấy thử nhìn vào mảng đã được sắp xếp, việc chọn số nằm ngoài khoảng từ số nhỏ nhất đến số lớn nhất của mảng sẽ làm "khoảng cách" giữa số được chọn và các số trong mảng sẽ dần tăng lên khi ta càng tiến ra xa.

Như vậy thuật toán của chúng ta sẽ đơn giản là duyệt từng giá trị trong $[\min(a_i), \max(a_i)]$ và một dòng for nữa để tính chi phí. Sau khi tính xong chi phí cập nhật vào biến đáp án và tiếp tục chạy cho đến khi nào duyệt qua được hết tất cả các giá trị. Độ phức tạp là $O(nA)$ với $A = \max(|a_i|)$.

Tag: Duyệt.

SUB 2

Ta thấy $(a_i - x)^2 = a_i^2 + x^2 - 2a_ix$. Ta có thể tính tổng a_i^2 và $2a_i$ ngay từ ban đầu. Như vậy khi for từng x có giá trị trong $[\min(a_i), \max(a_i)]$, thì khi cập nhật đáp án, tổng x^2 chỉ là $n * x^2$ và tổng $2a_ix$ chỉ cần nhân x vào tổng $2a_i$ (đặt nhân tử chung là x). Độ phức tạp là $O(A)$ với $A = \max(|a_i|)$.

Tag: Duyệt, Toán.

BONUS

Ngoài ra vẫn còn một cách giải nữa mà mình sẽ để như một cách biết thêm. Như công thức ta đã phân tích ở subtask trước, ta có tổng chi phí có công thức:

$$nx^2 - 2x(a_1 + a_2 + \dots + a_n) + (a_1^2 + a_2^2 + \dots + a_n^2)$$

Phần tổng phía sau không dính gì tới x nên ta sẽ bỏ qua và sẽ còn lại là $nx^2 + 2xs$ với $s = a_1 + a_2 + \dots + a_n$. Đó là phương trình bậc hai có thể được biểu diễn bằng một parabola hướng lên, giao với trục x (nghiệm phương trình bằng 0) tại $x = 0$ và $x = 2s/n$, như vậy giá trị nhỏ nhất sẽ tại $x = s/n$ hay nói chính xác hơn là khi x bằng trung bình cộng của mảng a. Như vậy chúng ta sẽ làm như subtask 1 nhưng chỉ kiểm tra x tại giá trị làm tròn xuống của trung bình cộng và làm tròn lên của trung bình cộng. Độ phức tạp là $O(n)$.

Tag: Toán, Duyệt.

Vẫn còn một cách nữa đó là tìm kiếm nhị phân/tìm kiếm tam phân, khá là "không não". Ta thấy bài toán chính là tìm tổng nhỏ nhất của $|a_i - x|$ với mọi i và với mọi x. Như đã nhận xét từ trước là x cho ra đáp án nhỏ nhất chỉ nằm từ số nhỏ nhất đến số lớn nhất của a và công thức tính toán lại chính là một parabola, do đó ta cứ việc cài chặt vào mà tìm trong O(n log A) với $A = \max(|a_i|)$.

Tag: Tìm kiếm nhị phân, Tìm kiếm tam phân.

5. Primehigher

Sub1. N<20

Duyệt quay lui sinh tất cả các dạy con, kiểm tra dãy nào thỏa điều kiện thì cập nhật kết quả

Sub2: N<1000

Quy hoạch động dãy con tăng liên tiếp dài nhất (LIS bản dễ)

Sub3: N<2. 10^5

Quy hoạch động dãy con tăng liên tiếp dài nhất

Sub4: N<2.10^5

Quy hoạch động dãy con tăng có thể không liên tiếp dài nhất (LIS bản khó) – quy hoạch động kết hợp chặt nhị phân.

6. PALINY

Sub 1: Duyệt trâu, thử tất cả các độ dài của xâu, duyệt kiểm tra từng xâu để tìm được xâu con đối xứng dài nhất. Độ phức tạp: $O(n^3)$

Sub 2: Quy hoạch động dp[i][j] = true/false để biết xâu S[i..j] có là đối xứng hay không.

Ta có: dp[i][i]=true

dp[i][j]= dp[i+i][j-1] nếu S[i]=S[j]

dp[i][j]=false nếu $S[i] \neq S[j]$

Độ phức tạp $O(n^2)$.

Sub 3: Sử dụng kĩ thuật so khớp chuỗi với hash; kết hợp tìm kiếm nhị phân. Gọi F[i] là độ dài xâu con đối xứng dài nhất mà có tâm là S[i].

Khi đó độ phức tạp sẽ là O(nlogn).