Todays Content

1. Equilibrium index
2. ABA pairs
3. Triplet pairs.

Carryforward Idea:

When we calculate same data from L→R or R→L multiple times use carry forward & calculate data in only 1 iteration.

Note: We iterate from direction we are calculating

Equilibrium Index:

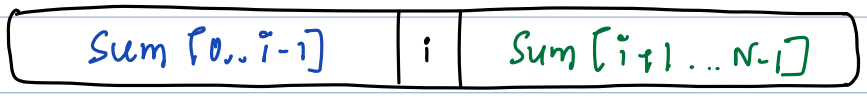
Given ar[N] elements count no: of equilibrium indices.

An element ar[i] is said to be equilibrium

if Sum of elements in left [0..i-1] = right [i+1...N-1] sum of elements in right

Note: if i==0: No elements in left can assume =0

if i==N-1: No elements in right can assume =0



Constraints:

$1 <= N <= 10^5$

$1 <= A[i] <= 10^9$

Ex1: ar[4] = { -3 2 4 -1 } ans = 1

	0	1	2	3
leftSum =	0	-3	-1	3
RightSum =	5	3	-1	0

Ex2: ar[7] = { -7 1 5 2 -4 3 0 } ans = 2

	0	1	2	3	4	5	6
leftSum =	0	-7	-6	-1	1	-3	0
RightSum =	7	6	1	-1	3	0	0

Ideal: for every ar[i]: Estimated TC: $O(N^2)$

check if equilibrium.

Iterate & calculate lsum = [0..i-1]

Iterate & calculate rsum = [i+1..N-1]

if[lsum == rsum]{

} ar[i] is equilibrium

```
int equilibrium(int ar[], int N){
    int c=0;
    for(int i=0; i<N; i++){ TC: O(N^2)  SC: O(1)
        int lsum=0, rsum=0;
        for(int j=0; j<i; j++){ -> [0..i-1] = i
            lsum = lsum + ar[j];
        }
                                                  a   b
        for(int j=i+1; j<N; j++){ -> [i+1..N-1] = N-i-1
            rsum = rsum + ar[j]
        }                                          b-a+1
                                                   N-1-(i+1)+1 = N-1-i-1+1
        if(lsum==rsum){
            c++;
        }
    }
    return c;
}
```

= N-1 ≈ N

Traling:

	0	1	2	3	4	5	6
ar[7] = {	-7	1	5	2	-4	3	0 }
LSum : →	0	-7	-6	-1	1	-3	0
RSum :	7	6	1	-1	3	0	0

Hint: For every ar[i]:

Check if equilibrium.

1. Iterate & calculate lsum = [0.. i-1] // Carry forward Left → Right
2. Doubt: How to calculate rsum =

lsum = [0.. i-1]

rsum = [i+1.. N-1] = Totalsum − lsum − ar[i]
[0.. i-1]

obs:

1. Keep a loop and calculate Totalsum
2. Lsum carry forward
3. Rsum calculate using Lsum

Dry Run:

	0	1	2	3	4	5	6	
ar[7] = {	-7	1	5	2	-4	3	0	}
Total = 0								
Lsum = 0	0	-7	-6	-1	1	-3	0	0
Rsum =	7	6	1	-1	3	0	0	

Total − Lsum − ar[i]

2. Count ABA Pairs:

Given a char[], calculate no: of triplet indices = i j k such that

i < j < k && ch[i] == 'a' && ch[j] == 'b' && ch[k] = 'a'

Constraints:

$1 \le N \le 10^5$

'a' <= ch[i] <= 'z'

Eg1:

	0	1	2	3	4	5	6	7
ch[8] =	d	a	b	a	d	b	a	c

Triplets: i < j < k ch[i] = 'a' && ch[j] = 'b' && ch[k] = 'a'

1	2	3
1	2	6
3	5	6
1	5	6
6	5	3

Ideal: Generate all triplets & check if they form valid pair.

Estimated: O(N3)

```
int triplets ( char s[], int N){ TC: O(N3)  SC: O(1)
    int c=0;
    for(int i=0; i<N; i++){
        for(int j= i+1; j<N; j++){
            for(int k=j+1; k<N; k++){
                if( s[i]=='a' && s[j]=='b' && s[k]=='a'){
                    c++;
                }
            }
        }
    }
    return c;
}
```

Idea2: In Triplet based Questions:

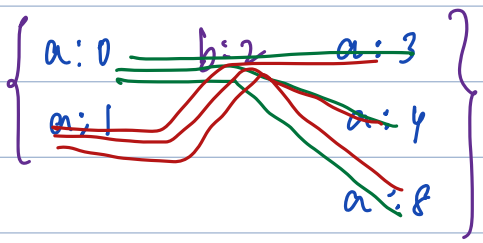
Try to fix center element of the problem.

i < j < k

a b a

Dry Run:

j		0	1	2	3	4	5	6	7	8
ch[8] =		a	a	b	a	a	b	e	b	a
				↓			↓		↓	
				la=2			la=4		la=4	
				ra=3			ra=1		ra=1	
				tr=6			tr=4		tr=4	Total = 14.



Steps: Iterate on ch[]

if ch[j] == 'b'

1. Iterate on left & calculate a's : la
2. Iterate on right & calculate a's : ra
3. ans += la*ra

```
int pairs(char ch[], int N){ TC: O(N^2)  SC: O(1)
    int ans=0;
    for(int j=0; j<N; j++){
        if(ch[j]=='b'){
            int la=0, ra=0;
            for(int i=0; i<j; i++){          [0.. j-1] j [j+1  N-1]
                if(ch[i]=='a'){la++;}
            for(int k=j+1; k<N; k++){
                if(ch[k]=='a'){ra++;}
        } ans = ans + la*ra;
    }
    return ans;
}
```

Ideas:

Iterate in ch[]

if ch[j] == 'b'

0. Iterate & calculate Total a's
1. Iterate in left & calculate a's : la // Optimize with carry forward
2. How to calculate a's in right?

   ra = Total a's - la's

3. ans += la * ra

Dry Run:

	0	1	2	3	4	5	6	7	8
ch[8] =	a	a	b	a	a	b	e	b	a
Total a's = 5									
la = 0 →	1	2	2	3	4	4	4	4	5
ra = Ta - la			3			1		1	
Triplets			6			4		4	= 14.

```
int pairs(char ch[], int N){ TC: O(N+N) = O(N)  SC: O(1)
    int ans=0, la=0, totala=0, ra=0;
    for(int i=0; i<N; i++){
        if(ch[i]=='a'){
            totala++;
        }
    }
    for(int i=0; i<N; i++){
        if(ch[i]=='b'){
            ra = total-la;
            ans = ans + la*ra;
        }else if(ch[i]=='a'){
            la++;
        }
    }
    return ans;
}
```

la ra

b

Q3

Count of Triplets

Given ar[N], calculate no: of triplets $i < j < k$ & $ar[i] < ar[j] < ar[k]$

Constraints:

$1 <= N <= 10^3$

Ex: ar[5] = { 2 6 9 4 10 }

	0	1	2	3	4
	2	6	9	4	10

i j k ar[i] < ar[j] < ar[k]

Ideal:

Idea2:

	0	1	2	3	4	5	6	7
fn: ar[8] = {	3	2	6	4	7	10	9	12 }

i j k ar[i] < ar[j] < ar[k]

	0	1	2	3	4	5	6	7
fn: ar[8] = {	3	2	6	4	7	10	9	12 }

Count less :

Count more :