1Q) Given N array elements & Q queries on same array.
For each query calculate sum of all elements in Given Range [L-R]

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| ar[10] : | -3 | 6 | 2 | 4 | 5 | 2 | 8 | -9 | 3 | 1 |

Q = 5

| L - R | Sum |
|---|---|
| 4 — 8 : | 9 |
| 2 - 7 : | 12 |
| 1 - 3 : | 12 |
| 0 - 4 : | 14 |
| 7 - 7 : | -9 |

Idea: For every query iterate on given range & print sum.

```
Q,
Read Q
for(i=1; i<=Q; i++){
    L, R; S=0
    read L, R
    for(i=L; i<=R; i++){        } N iterations
        S = S + ar[i]           } O(N)
    }
    print(S)
}
```

TC: Q * N → O(Q*N)

SC: O(1)

// Given Indian Team Score for first 10 Overs of Batting

| Overs: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Score: | 2 | 8 | 14 | 29 | 31 | 49 | 65 | 79 | 88 | 97 |

Q1: Runs scored in last Over: 97 - 88 = 9
[10 - 10] → scr[10] - scr[9]

Q2: Runs scored in last 5 Over: 97 - 31 = 66
[6 - 10] → scr[10] - scr[5]

Q3: Runs scored in 7th Over: 65 - 49 = 16
[7 - 7] → scr[7] - scr[6]

Q4: Runs scored from [3 - 6] : scr[6] - scr[2]
→ 49 - 8 = 41

Q5: Runs scored [4 - 9] : scr[9] - scr[3]
→ 88 - 14 = 74

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| ar[10] : | -3 | 6 | 2 | 4 | 5 | 2 | 8 | -9 | 3 | 1 |
| pf[10] : | -3 | 3 | 5 | 9 | 14 | 16 | 24 | 15 | 18 | 19 |

pf[i] ⇒ Sum of all array elements from [0-i]

Queries:

4 - 8 : pf[8]

4, 5, 6, 7, 8 — 0, 1, 2, 3, 4, 5, 6, 7, 8

pf[8] = pf[3] + sum[4-8]

0 1 2 3 — 4 5 6 7 8

sum[4-8] = pf[8] - pf[3]

3 - 7 : pf[7] - pf[2]

4 - 9 : pf[9] - pf[3]

(i - j): pf[j] - pf[i-1]

[0 4] : pf[4] - pf[-1] } Array out bounds

↳ pf[4]

(i, j):

if i == 0 : pf[j]

else : pf[j] - pf[i-1]

→ Construct Pf[N] Given ar[N]

Pf[0] = ar[0]

Pf[1] = ar[0] + ar[1]

Pf[2] = ar[0] + ar[1] + ar[2]

Pf[i] = ar[0] + ar[1] + ar[2] ... ar[i-1] + ar[i]

**Pf[i] = Pf[i-1] + ar[i]**

if i == 0, error

```
Pf[N]
Pf[0] = ar[0]
for(i = 1; i < N; i++){
    Pf[i] = Pf[i-1] + ar[i]
}
```

TC: O(N)

```
for(i = 1; i <= Q; i++){
    Read L, R;
    if(L == 0){
        print(Pf[R])
    }
    else{
        print(Pf[R] - Pf[L-1])
    }
}
```

TC: O(Q)

Overall TC: O(N + Q)

SC: O(N)

20) Equilibrium Index :

Given N array elements, count no: of Equilibrium Index

An index is said to be equilibrium index if

Sum of all elements Before $i^{th}$ index = Sum of all Elements after $i^{th}$ index

Sum [0, i-1] = Sum [i+1, N-1]

Note1: i==0, leftsum = 0

Note2: i==N-1, rightsum = 0

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| ar[] : | -7 | 1 | 5 | 2 | -4 | 3 | 0 |
| left : | 0 | -7 | -6 | -1 | 1 | -3 | 0 |
| right : | 7 | 6 | 1 | -1 | 3 | 0 | 0 |

cut = 2

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| ar[] : | 3 | -1 | 2 | -1 | 1 | 2 | 1 |
| left : | 0 | 3 | 2 | 4 | 3 | 4 | 6 |
| right : | 4 | 5 | 3 | 4 | 3 | 1 | 0 |

Pseudo code

→ Every index checks if it's equilibrium index or not.

```
Pf[N]
// Construct Pf[N] -> N
int c=0
i=0; i<N; i++){
    {if(i==0){ left=0 }
    else { left = sum[0, i-1] = Pf[i-1] }
    { right = sum[i+1, N-1] = Pf[N-1] - Pf[i]
    if( left == right){
        c = c+1
    }
}
return c;
```

TODO:
→ Can be done with pf[]

10:35 PM break

TC: (N+N) → O(N)

SC: O(N)

IQ) Given N array elements & Q queries, for each query

Todo!

Calculate sum of all {even indices} in given range

0 is even number

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| ar[8] : | 3 | 4 | -2 | 8 | 6 | 2 | 1 | 3 |
| Pfeven[N]: | 3 | 3 | 1 | 1 | 7 | 7 | 8 | 8 |

5 Queries Sum

[2 5] : 4

[3 7] : 7

[0 7] : 8

[7 7] : 0

[4 4] : 6

Pfeven[i] = Sum of all even index [0 i]

Pfeven[0] = Sum even [0 0] = ar[0]

Pfeven[1] = Sum even [0 1] = ar[0]

```
Pfeven[N];
Pfeven[0] = ar[0]
i = 1; i < N; i++){
    if(i%2==1){
        Pfeven[i] = Pfeven[i-1]
    }
    else{
        Pfeven[i] = Pfeven[i-1] + ar[i]
    }
}
```

[3 7]:

Pfeven[7] - Pfeven[2]

Sumeven[0 7] - sumeven[0 2]

Sumeven[3 7]

→ Queries ✓

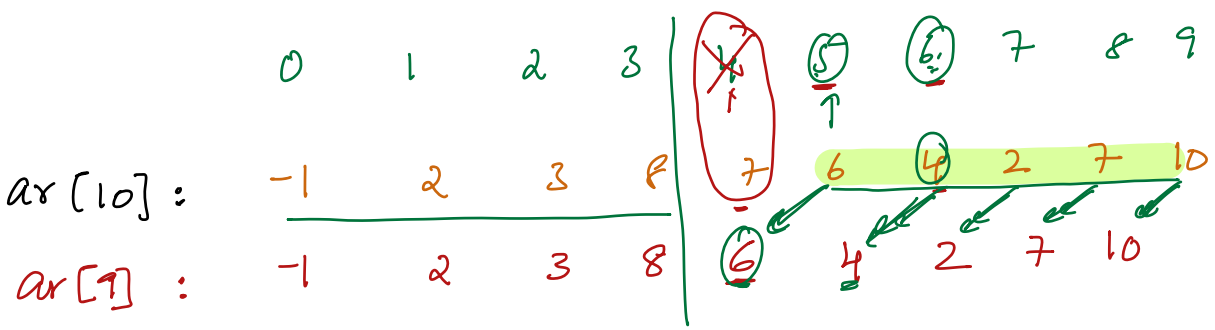
Similar manner =>

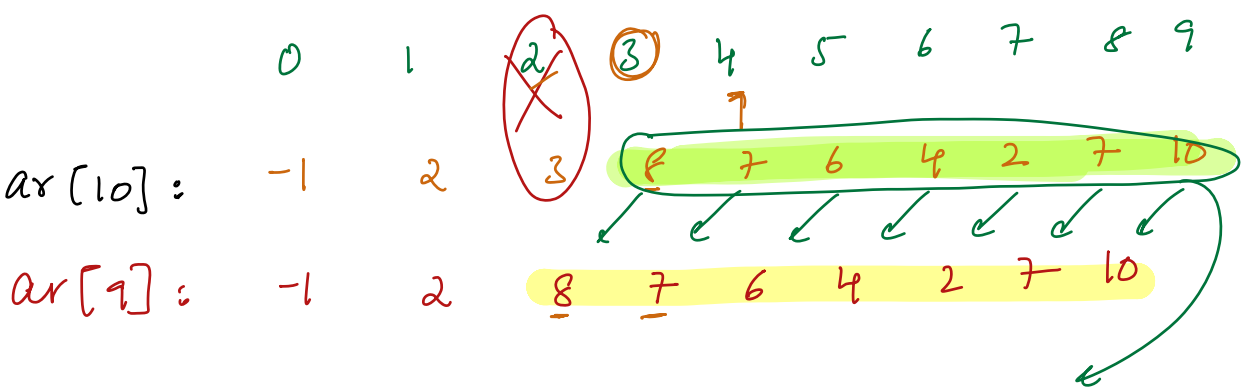
Pfodd[i] = Sum of all odd indices from [0-i]

Pfodd[0] = 0

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| ar[] = | 2 | 4 | 3 | 1 | 5 |
| Pfodd = | 0 | 4 | 4 | 5 | 5 |

// Construct Pfodd[N] on your own (TODO) ✓



Google

**Special Index** : HARD

An index is said to be special index, if after deleting that index **Sum of even index value = Sum of all Odd index values**

Count how many special index are there?

Ex:

| 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 4 | 3 | 2 | 7 | 6 | -2 |

**Delete 0** ✓

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr[]: | 3 | 2 | 7 | 6 | -2 |

SumevenIndex = 8

SumoddIndex = 8

**Delete 1** ✗

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr[]: | 4 | 2 | 7 | 6 | -2 |

SumevenIndex = 9

SumoddIndex = 8

**Delete 2** ✓

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| | 4 | 3 | 7 | 6 | -2 |

SumevenIndex = 9

SumoddIndex = 9

**Delete 3** ✗

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr[]: | 4 | 3 | 2 | 6 | -2 |

SumevenIndex = 4

SumoddIndex = 9

**Delete 4** ✗

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr[]: | 4 | 3 | 2 | 7 | -2 |

SumevenIndex = 4

SumoddIndex = 10

**Delete 5** ✗

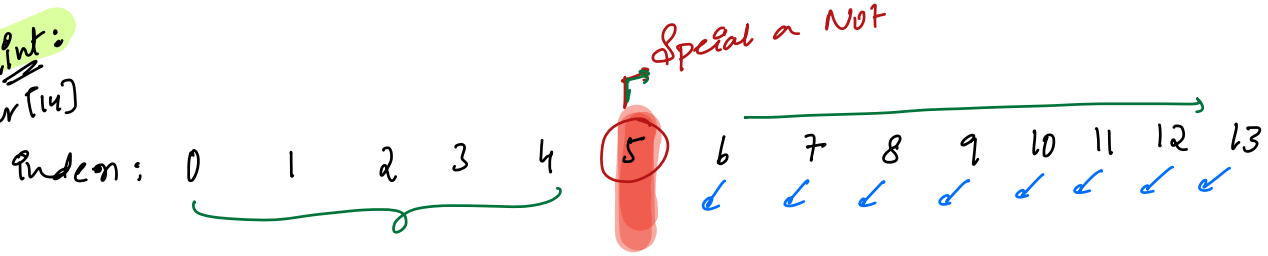
| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| arr[]: | 4 | 3 | 2 | 7 | 6 |

SumevenIndex = 12

SumoddIndex = 10

Hint:

ar[14]




Sumeven [0 4]

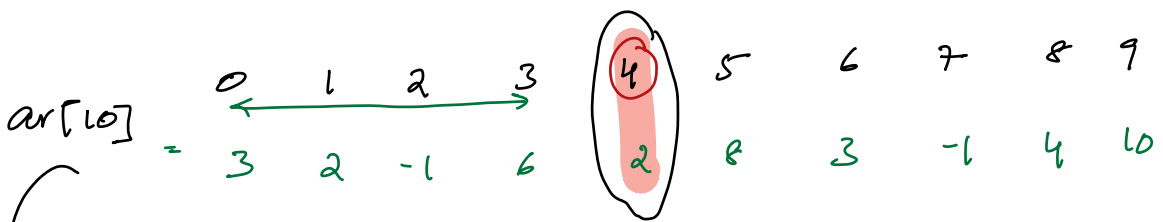
Sumodd [0 4]

Sumeven [6 13]

Sumodd [6 13]

Total sumeven = sumeven[0 4] + sumodd[6 13]

Total sumodd = sumodd [0 4] + sumeven [6 13]

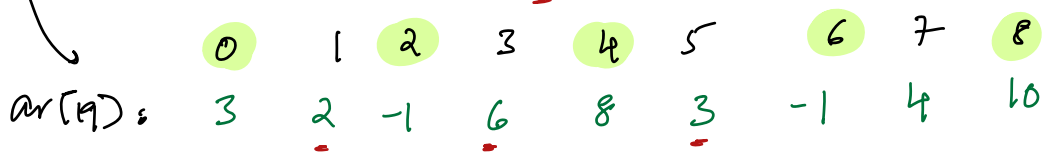



Sumeven [0 3] : 2

Sumodd [0 3] : 8

Sumeven [5 9] : 7

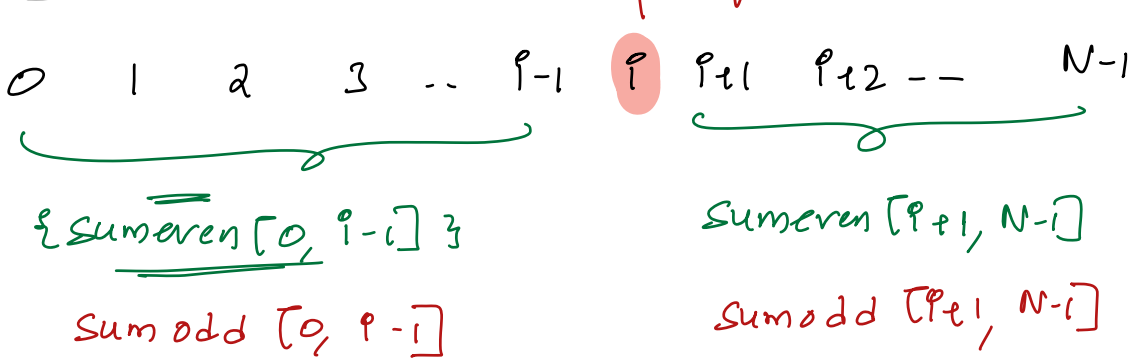
Sumodd [5 9] : 17




Total sumeven = 19 = sumeven [0 3] + sumodd [5 9]

Total odd sum = 15 = sumodd [0 3] + sumeven [5 9]

// Generalise:




Total Even sum = Sumeven [0, i-1] + Sum odd [i+1, N-1)

Total Odd sum = Sum odd [0, i-1] + Sumeven [i+1, N-1]

Pseudocode

1 → PfEven[] } → O(N) → Loop

2 → Pfodd[] } → O(N) → Loop

cut = 0

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

// check if i is special index

Total Even sum = sumeven[0, i-1] + sumodd[i+1, N-1]

Total Odd sum = sumodd[0, i-1] + sumeven[i+1, N-1]

if(Total Even sum == Total Odd sum){

cut++

}

}

O(N)

Solve pfeven[] query

if i=0 error

if i=N-1 error

pfodd[] is

pfodd[]

pfeven[]

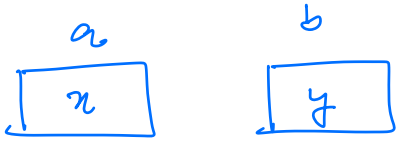
pfodd[]

Note: if i==0, i==N-1, handle Edge Cases.

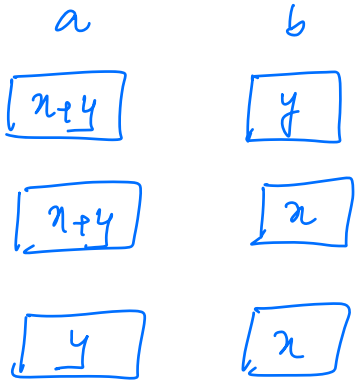
TC: O(N + N + N) → O(N)  SC: O(N+N) → O(N)

Doubts:

```
int a = x
int b = y
```




```
a = a + b
b = a - b
a = a - b
```




TODO:

```
max = ar[0], cnt = 0

for(i = 0; i < N; i++){
    if(max == ar[i]){
        cnt = cnt + 1
    }
    else if(max < ar[i]){
        max = ar[i]
        cnt = 1
    }
}
```

Q4)

→ Notes : ar[N] : [0 → N-1)

↳ Working Hard > Talent