Q1 Given an array of size N & Q queries of the format start & end. Return the sum of elements from index start to end.

	0	1	2	3	4	5	6	7	8	9
A :	-3	6	2	4	5	2	8	-9	3	1

→ Q : 4

	start	end	
1	1	3	→ 12 ✓
2	2	7	→ 12 ✓
3	4	8	→ 9 ✓
4	0	2	→ 5 ✓

3 Min

```
for(i = 0; i < Q; i++) {          // Q
    scan(start, end);
    sum = 0
    for(j = start; j <= end; j++){ } // N
        sum = sum + arr[j];
}
print(sum);
}
```

# of iterations = Q * N

TC : O(Q*N)
SC : O(1)

Given the track of runs scored for first 10 overs of a match.

⇒ Cumulative sum of scores

1	2	3	4	5	6	7	8	9	10
2,	8,	14,	29,	31,	49,	65,	79,	88,	97

Runs scored in [1, 2] ⇒ 6

① Runs scored in 10th over → 97 − 88

[10, 10] (s, e) ⟶ ⇒ R[10] − R[9]

R[e] − R[s−1]

② Runs scored in 9th over = 88 − 79

[9, 9] (s, e) ⟶ ⇒ R[9] − R[8]

③ Runs scored from 6th to 10th over → 97 − 31

[6, 10] (s, e) ⟶ ⇒ R[10] − R[5]

④ Runs scored from 3rd to 5th over =

[3, 5] ⟶ ⇒ R[5] − R[2]

⑤ Runs scored in 7th over → R[7] − R[6]

[7, 7]

⇒ R[i] = Total sum of runs scored till i^{th} over

Prefix Array ⇒ Every index should store sum of all elements from start till that index.

PS[i] = Sum of elements from index 0 to index i.

	0	1	2	3	4	5	6	7	8	9
A :	-3	6	2	4	5	2	8	-9	3	1
PS :	-3	3	5	9	14	16	24	15	18	19

PS[0] = A[0] = ?

PS[1] = A[0] + A[1] = PS[0] + A[1]

PS[2] = A[0] + A[1] + A[2] = PS[1] + A[2]

PS[3] = A[0] + A[1] + A[2] + A[3] = PS[2] + A[3]

PS[i] = A[0] + A[1] + ... + A[i-1] + A[i] = PS[i-1] + A[i]

(A[0] + A[1] + ... + A[i-1] = PS[i-1])

$$PS[i] = PS[i-1] + A[i] \quad i \neq 0$$

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

N

# of iterations = N

TC : O(N)

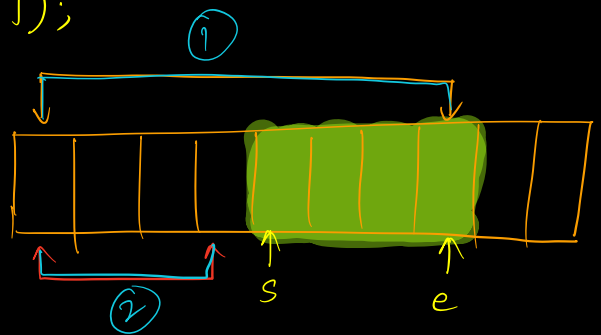
SC : O(N)

Q+N

TC: → O(Q+N)

SC: → O(N)

```
→ for(i=0; i<Q; i++){
    scan(start, end);
    if(start != 0)
→     print(PS[end] - PS[start-1];
    else
      print(PS[end]);
  }
```



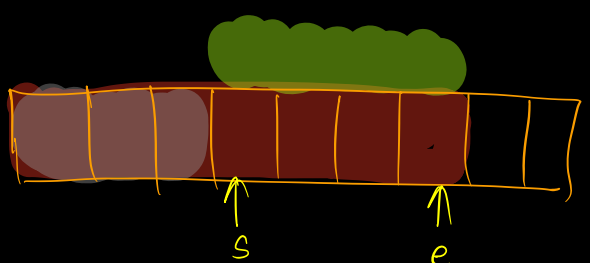
$sum(s-e) = sum(0-e) - sum(0-s-1)$

$= PS[e] - PS[s-1]$

Range sum [s, e]

⇓

Prefix Array.



$sum(0-e) = sum(0-s-1) + sum(s-e)$

$sum(s-e) = sum(0-e) - sum(0-s-1)$

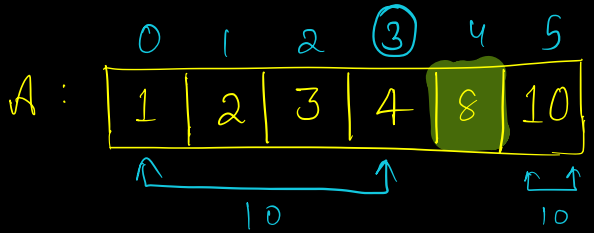
$PS[e] - PS[s-1]$

Directi

Q2 Given an array. Return true if there exists an equilibrium index in the array.

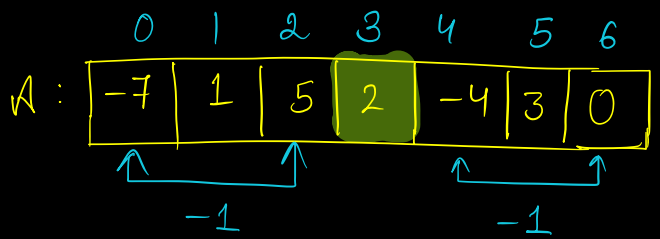
Equilibrium Index ⇒ index for which,

Sum of all the nos on left side = Sum of all the nos on right side



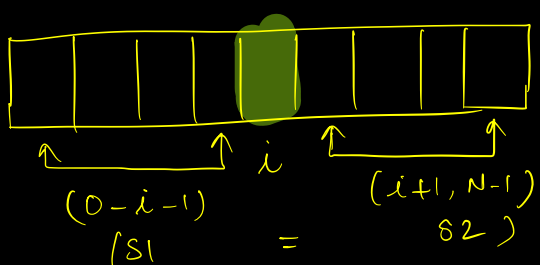
index = 4 is an EI

⇒ True



index = 3 is an EI.

⇒ True



for every index i,

check if $\Sigma[0 \cdots (i-1)] == \Sigma[(i+1), (N-1)]$

ret true;

```
// Build PS → O(N)
for (i = 0; i < N; i++) {
    // SL = sum from 0 to (i-1)   // Range sum query
    // SR = sum from (i+1) to (N-1)   // Range sum query
    SL = PS[i-1]; // if i == 0, SL = 0
    SR = PS[N-1] - PS[i]; // if i == N-1, SR = 0

    if (SL == SR)
        return true;
}
return false;
```

TC : Build PS + Find EI

⇒ O(N) + O(N)

⇒ **O(N)**

SC : O(N)

Q. Given an array & Q queries of following type :-

① start, end, O ⇒ Sum of all odd indexed elements } from start to end.

② start, end, E ⇒ sum of all even indexed elements }

0	1	2	3	4	5	6	7
2	3	1	-1	0	8	5	4

PSE ⇒ We will only consider values whose indices are even.

PSO ⇒ We will only consider values whose indices are odd.

Q : 2

start	end	O/E	
3	6	O	→ A[3] + A[5] = 7
1	5	E	→ A[2] + A[4] = 1 + 0 = 1

0	1	2	3	4	5
2	3	1	6	4	5

PSe : 2 2 3 3 7 7

2 5 'E'

PSe[5] - PSe[1]

0	1	2	3	4
2	4	3	1	5

PSo : 0 4 4 5 5

PSe : 2 2 5 . . . .

PSo → O(N)
PSe → O(N)

PSo[0] = 0

s e O ⇒ PSo[e] - PSo[s-1]

s e E ⇒ PSe[E] - PSe[s-1]

→ PSo[0] = 0; PSe[0] = A[0];

```
for(i=1; i<N; i++){
  if(i%2==0){
```

Calculating Prefix sum for odd indices.

```
{
    PSO[i] = PSO[i-1];
} else {
    PSO[i] = PSO[i-1] + A[i];
}
}
```

```
if(i % 2 == 1){
    PS[i] = PS[i-1] + A[i];
}
else
    PS[i] = PS[i-1];
```

Q Given an array. Count the number of Special indexes in the array.

Special Index :- An index after removing which,

Sum of all odd indexed elements == Sum of all even indexed elements

(In Resulting array)

A :

4	3	②	7	6	-2

Remove

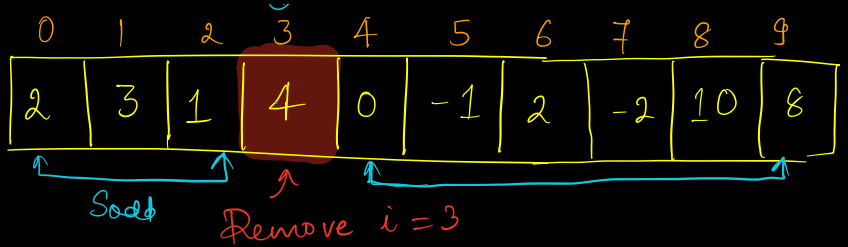
	0	1	2	3	4	
i = 0	3	2	7	6	-2	$\rightarrow S_E = 8,\ S_O = 8$ ✓
i = 1	4	2	7	6	-2	$\rightarrow S_E = 9,\ S_O = 8$ ✗
i = 2	4	3	7	6	-2	$\rightarrow S_E = 9,\ S_O = 9$ ✓

Quiz

0	1	2	3	4	5
4	1	5	3	7	10

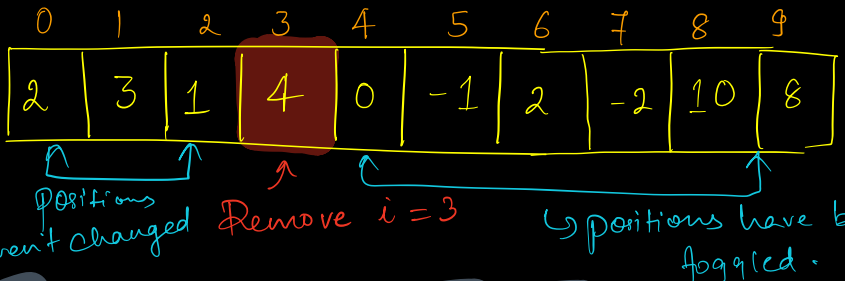
$S_{odd} = 8$

Quiz :-



$$S_{odd} = S_{odd}[0-2] + S_{even}[4-9]$$

Quiz :-



$$S_{even} = S_{even}[0-2] + S_{odd}[4-9]$$

After removal of index i,

s, e, E

s, e, O

$$S_E = S_E[0, i-1] + S_O[i+1, N-1]$$

$$S_O = S_O[0, i-1] + S_E[i+1, N-1]$$

① $S_O[0 \cdots i-1] = PS_O[i-1]$.

② $S_E[i+1 \cdots N-1] = PS_E[N-1] - PS_E[i]$

③ $S_O[i+1, N-1] \Rightarrow PS_O[N-1] - PS_O[i]$

④ $S_E[0 \cdots i-1] \Rightarrow PS_E[i-1]$

```
{ // Build PSe  ------> N
{ // Build PSo  ------> N
  cnt = 0                                   i = 0
  for(i = 0; i < N; i++) {  ------> N
    // SE = sum of even indexed elements after removal of index i
    SE = PSE[i-1] + (PSO[N-1] - PSO[i]);
    // SO = sum of odd indexed elements after removal of i
    SO = PSO[i-1] + (PSE[N-1] - PSE[i])
    if(SE == SO){
      // i is a special index.
      cnt = cnt + 1;
    }
  }
  return cnt;
```

⇒ 3N

{ TC : O(N)
  SC : O(N)

// Doubts

	0	1	2	3	4	5	6	7
A:	1	2	3	4	6	7	8	9
PSE :-	1	1	4	4	10	10	18	18

```
PSE[0] = A[0];
for(i=1; i<N; i++){
    if(i%2==0){
        PS[i] = PS[i-1] + A[i];
    }else{
        PS[i] = PS[i-1];
    }
}
```

Build PSE

```
PSo[0] = 0;
if(i%2==1){
    PS[i] = PS[i-1]
            + A[i];
}else{
    PS[i] = PS[i-1]
}
```

Build PSo.