Q1 Given n array element and a starting index and a ending index. Find sum in that range.

Ex 1

$$\begin{array}{cccccc} & 0 & 1 & 2 & 3 & 4 \\ arr & [2 & 4 & 1 & 6 & 5] \end{array}$$

$l = 2$

$r = 4$ $[l, r]$

Approach 1 : Loop from l to r, calculate sum.

TC: $O(n)$

Q2 Given n array element and a starting index and a ending index. Find sum in that range. You are given Q queries

Approach 2 :

```
for (int i=0 ; i<q ; i++) {
    int s=0;
    Take l, r as input.

    for (j=l; j<=r ;j++) {
        s = s + arr[j];
    }

    print s;
}
```

TC: $O(Qn)$

SC: $O(1)$

Q3 Given are Indian Score after every over.

Over =	1	2	3	4	5	6	7	8	9	10
S[ ] =	2	8	14	29	31	49	65	79	88	97

(i) Runs scored in 10^{th} over : [10, 10]

97 − 88

$\Rightarrow S[10] - S[9]$

(ii) Runs scored in last 5 overs : [6, 10]

$S[10] - S[5]$

$97 - 31 = 66$

(iii) Runs scored in the 7^{th} over = [7, 7]

$S[7] - S[6]$

$\Rightarrow 65 - 49 = 16$

Cumulative Sum from the start $\Rightarrow$ prefix Sum

prefix Sum array $\Rightarrow$

pf[i] $\Rightarrow$ Sum of all element from start till i^{th} index

Sum [0 .... i]

$$
\begin{array}{lccccccc}
 & & 0 & 1 & 2 & 3 & 4 & \\
arr[\,] = & [& -9 & 10 & 2 & 1 & 8 &]
\end{array}
$$

pf[ ] = [-9 1 3 4 12]

(i) Sum(2, 4)

Sum(0, 4) = Sum(0, 1) + Sum(2, 4)

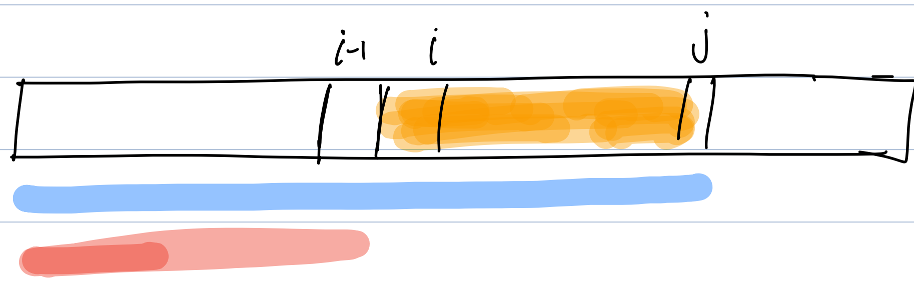
Sum(2, 4) $\Rightarrow$ Sum(0, 4) − Sum(0, 1)

Sum(2, 4) $\Rightarrow$ pf[4] − pf[1]

(ii) Sum(1, 3) = pf[3] − pf[0]

(iii) $Sum(2,3) = pf[3] - pf[1]$

$$Sum(i,j) = pf[j] - pf[i-1]$$



\# $Sum(0,3) = pf[3] - pf[-1]$

$i=0$

$\Rightarrow$ $Sum(i,j) =$

```
if (i==0)
    return pf[j];
else
    return pf[j] - pf[i-1]
```

Q3 Given n array element and a starting index and a ending index. Find sum in that range. You are given Q queries

$\Rightarrow$

```
pf[n];

pf[0] = arr[0];

for (int i=1 ; i<n; i++) {
    pf[i] = pf[i-1] + arr[i];
}
```

$O(n)$

```
for (int i=0 ; i<Q ; i++) {

    // input L, R;
    if (L==0)
        print pf[R];
    else
        print pf[R] - pf[L-1];
}
```

$O(Q)$

TC $\Rightarrow$ $O(N+Q)$

SC = $O(N)$

Save space in the above problem.

```
for (int i = 1 ; i < n ; i++) {

    arr[i] = arr[i-1] + arr[i];

}
```

SC: $O(1)$

Equilibrium Index

⇒ arr[ ], find count of Equilibrium Index.

Equi Index ⇒ Sum of all element to left is equal. Sum of all element to right.

		0	1	2	3
arr[]	=	-3	2	4	-1
left	=	0	-3	-1	3
right	=	5	3	-1	0

= ans = 1

for any index i

left = pf [i-1]

right ⇒ [i+1, n-1] = pf (n-1) - pf (i)

// Construct Pf sum array. ⇒ O(N)

```
c = 0

for (int i=0 ; i<n ; i++) {
        if (i==0)
            left = 0
        else
            left = pf[i-1];

        right = pf[n-1] - pf[i];

        if (left == right)
            c++
}

return c;
```

Tc: O(N)

SC = O(1)

↓

If original array can be modified.

else O(N)

$Sum(i,j) = pf[j] - pf[i-1]$

Q Given N array element. & Q queries.
Each Q consists of i, j.
Return the number of even valued items
in [i, j]

		0	1	2	3	4	5	
Ex1	arr[] = [	2	3	6	8	1	10	]

(I) [0, 3] $\Rightarrow$ 3

even[i] $\Rightarrow$ no of even valued terms till i^{th} index

(II) [4, 5] $\Rightarrow$ 1

↳ even[5] - even[3]

Brute Force : For each Q loop from i to j

TC : O(QN)
SC : O(1)

		0	1	2	3	4	5	
arr[] =	[	2	3	6	8	1	10	]
=	[	1	0	1	1	0	1	]
even[i] =	[	1	1	2	3	3	4	]

(i, j) $\Rightarrow$ even[j] - even[i-1]

TC : $O(N+Q)$

SC : $O(1)$ $\Rightarrow$ if original array is modified

$O(N)$ $\Rightarrow$ if new array created.

arr[10]

4	5	6	7	8	10	12	15	1	15

4	5	7	6	8	9	10