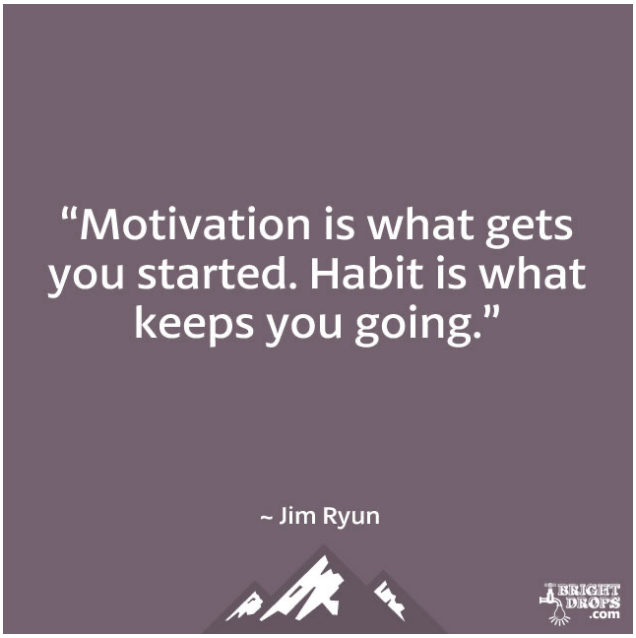
Prefix Sum



Good Evening

Todays content

01. Range query
02. Prefix sum construction
03. Equilibrium Index
04. Even numbers in given range

01. Given arr[N] elements & Q queries

For each query : Given L & R calculate & print sum of all elements in range [L R] . L & R both included

Note :- L & R are array indices such that $0 \le L \le R < N$

Constraints

$1 \le N, Q \le 10^5$

$1 \le A[i] \le 10^9$

$0 \le L \le R < N$

arr[10] =

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

Queries = 5

L	R	sum
4	8	→ 9
3	7	→ 10
1	3	→ 12
0	4	→ 14
7	7	→ -9

Brute force → For each query, iterate from L to R and print the sum

```
void rangesum ( int arr[N], int Q , int L[Q], int R[Q])
{
    for( i=0 ; i<Q ; i++){
        int s = L[i]  , int e = R[i]
        long sum=0;
        for (int j=s ; j<=e ; j++){
            sum+ = arr[j]
        }
        Print (sum);
    }
}
```

TC = O(Q*N)

SC = O(1)

TLE = TLE

Q: Given Indian Cricket Team score, for first 10 overs of Batting.

After every over, total score is given as:

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

01. Runs scored in the 10^{th} over = 97 − 88 = 9

sc[10] − sc[9]

02 Runs scored in the 7^{th} over = 65 − 49 = 16

sc[7] − sc[6]

03. Runs scored from $6^{th} - 10^{th}$ over = 97 − 31 = 66

sc[10] − sc[5]

Runs after the 5^{th} over = 31

Runs from $6^{th} - 10^{th}$ over = x

Runs after the 10^{th} over = 97

$31 + x = 97$

$x = 97 - 31$

04. Runs scored from 3 to 6 overs = 49 − 8 = 41

sc[6] − sc[2]

Runs after the 6th over = 49

Runs from 3rd − 6th over = x

Runs after the 2nd over = 8

$$8 + x = 49$$

$$x = 49 - 8 = 41$$

05. Runs scored from 1st over to 5th over = 31

Total runs from $i - j$ = sc[j] − sc[i−1]

Idea → If you have cumulative sum / prefix sum then we can get our answer

arr[10] =	-3	6	2	4	5	2	8	-9	3	1
	0	1	2	3	4	5	6	7	8	9

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

Pf [3] = sum of all ele from [0-3]

Pf [7] = sum of all ele from [0-7]

Queries = 5

L	R
4	8
3	7
1	3
0	4
7	7

→ 9

sum [4-8]

sum [0-8] = sum [0-3] + sum [4-8]

sum [0-8] − sum [0-3] = sum [4-8]

reqsum = pf [8] − pf [3]

= 18 − 9 = 9

02. sum [3-7] = pf [7] − pf [2]

03 sum [0-4] = pf [4]

Generalised

sum [i j] =

sum [0 j] = sum [0 i-1] + sum [i j]

sum [i j] = sum [0 j] − sum [0 i-1]

Pf [i-j] = pf [j] − pf [i-1]

$sum[0\ 4] = pf[4] - pf[0-1]$

$= pf[4] - pf[-1]$ ✗

Generic formula	if (i==0) pf[j]
	else pf[j] - pf[i-1]

* Construction of pf array

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

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

$Pf[0] = ar[0]$

$Pf[1] = \underbrace{ar[0]}_{Pf[0]} + ar[1]$

$Pf[2] = \underbrace{ar[0] + ar[1]}_{Pf[1]} + ar[2]$

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

$pf[3] = ar[0] + ar[1] + ar[2] + ar[3]$

$= pf[2] + ar[3]$

$pf[i] = ar[0] + ar[1] + ar[2] + \cdots + ar[i-1] + ar[i]$

$= pf[i-1] + ar[i]$

$pf[i] = pf[i-1] + ar[i]$:

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

$i = 0$

$pf[0] = pf[0-1] + ar[0]$ ✗

```
pf[0] = ar[0] :
for(i=1 ; i<n ; i++){
    pf[i] = pf[i-1]+ ar[i] ;
}
```

→ construct our pf array

Optimized code

Step 1 → Construct the prefix array

```
long pf[] = new long[n]

pf[0] = ar[0]

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

} O(n)

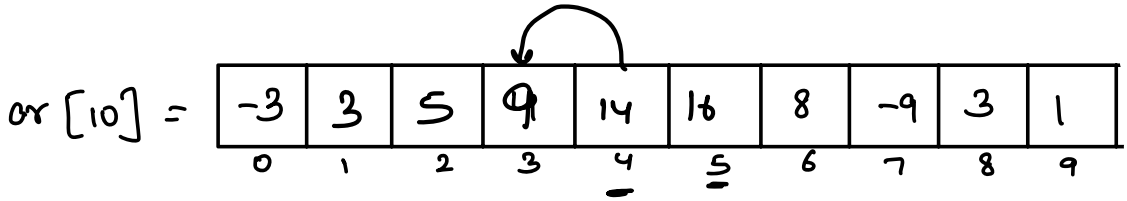
Step 2 → Answer for all queries

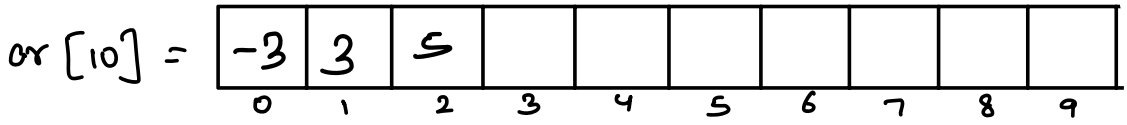
```
for (i=0; i<Q; i++){
    int s=L[i], e=R[i]
    if(s==0) {print(pf[e])};
    else {print(pf[e] - pf[s-1]);
}
```

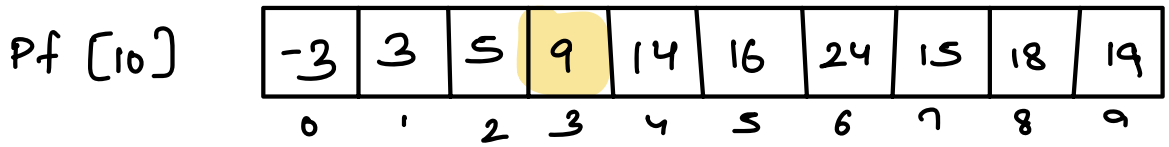
} O(Q)

TC = O(N+Q)

SC = O(N) → O(1) by updating the given array (Not possible for this eg)








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

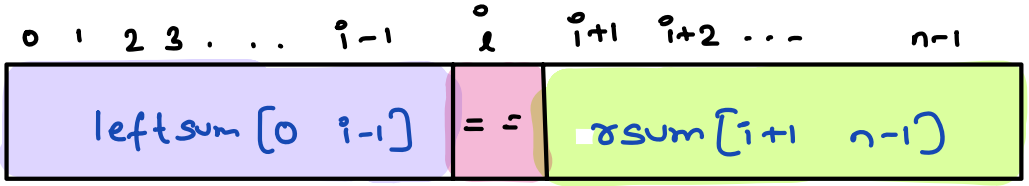
⇒ If pf & given array are of same data type then we can save the extra space

02. Equilibrium Index

Given N array elements, count no. of equilibrium idx

An idx i is known as eqm idx

sum of all elements before i idx = sum of all elements after i idx



Note :- if (i == 0) leftsum = 0

if (i == N-1) rightsum = 0

	0	1	2	3	
arr[4] =	-3	2	4	-1	Ans = 1
lsum =	0	-3	-1	3	
rsum =	5	3	-1	0	

arr[7] =	-7	1	5	2	-4	3	0	Ans = 2
lsum =	0	-7	-6	-1	1	-3	0	
rsum =	7	6	1	-1	3	0	0	

Simplest Idea → For every index, calculate the lsum & rsum

if (lsum == rsum) { increase the count }

Pseudocode →

```
int equilibrium ( int [n] ar )
    count = 0
    for ( i=0 ; i<n ; i++ )
        // lsum = [0 to i-1]
        // rsum = [i+1 to n-1]
        int lsum=0:
        for (j=0 ; j<=i-1 ;j++) { lsum+= ar[j] }
        int rsum=0
        for (j=i+1 ; j<n ; j++) { rsum+= ar[j] }
        if (lsum==rsum) {
            count++;
        }
    }
    return count;
```

$TC = O(n^2)$

$SC = O(1)$

Optimised Idea

$s \rightarrow e$

Generic formula	if (s==0) pf[e]
	else pf[e] - pf[s-1]

s e

lsum = [0 i-1] → Pf[i-1]

rsum = [i+1 n-1] = Pf[n-1] − Pf[i+1-1]

= Pf[n-1] − Pf[i]

Optimised

```
int equilibrium (int [] ar)
{
    01. Construct pf array → {TODO} ⟶ n

        c = 0
    for (i=0; i<n; i++){
        // lsum = [0 i-1]
        int lsum = 0
        if (i!=0) { lsum = pf[i-1]; }
        int rsum = 0
        rsum = pf[n-1] - pf[i];

        if (lsum == rsum) { c = c+1 };
    }                                   } n
    return c;
}
```

TC = O(n) SC = O(n)

03. Given ar[N] elements & Q queries.

For each query : Given L & R calculate & print no. of even numbers in given range [L R]

Note :- L & R are array indices such that $0 \le L \le R < N$

Constraints
$1 \le N, Q \le 10^5$
$1 \le A[i] \le 10^9$
$0 \le L \le R < N$

Eg:- ar[10] :

2	4	3	7	9	8	6	3	4	9
0	1	2	3	4	5	6	7	8	9

Queries : 4

L[4]	R[4]	count
4	8	3
3	9	3
2	7	2
0	4	2

BF → For a given query, calculate the no. of even numbers from L to R

```
void countEven(int ar[N], int Q, int L[Q], int R[Q])
{
    for(i=0; i<Q; i++){
        int s=L[i], e=R[i], c=0
        for(int j=s; j<=e; j++){
            if(ar[j]%2==0){c=c+1};
        }
        print(c);
    }
```

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

Optimisation → Create a prefix array of sum of count of even no.

Eg:- ar[10] :

2	4	3	7	9	8	6	3	4	9
0	1	2	3	4	5	6	7	8	

ar[10] =

1	1	0	0	0	1	1	0	1	0
0	1	2	3	4	5	6	7	8	9

even = 1

odd = 0

pf [10] =

1	2	2	2	2	3	4	4	5	5
0	1	2	3	4	5	6	7	8	9

pf [6] = count of all the even no. from [0-6]

Queries : 4

L[4]	R[4]
4	8
3	9
2	7
0	4

count

= pf [8] - pf [3] = 5 - 2 = 3

= pf [9] - pf [2] = 5 - 2 = 3

= pf [7] - pf [1] = 4 - 2 = 2

= pf [4] = 2

```
void rangecount (int []ar, int Q, int L[Q], int R[Q])
{
  01. Modify the given arr

    for (i=0 ; i<n; i++){
      if (ar[i] % 2 ==0) { ar[i] =1 }
      else { ar[i] =0 }
    }                                          } N
  02. Create pf array.

    Pf [0] = ar[0] :

    for (i=1 ; i<n ; i++) {
      pf [i] = pf [i-1]+ ar[i] :
    }                                          } N

  03. Answer all queries

    for (i=0 ; i<Q ; i++) {
      int s= L[i] , e= R[i]
      if (s==0) { print (pf [e] )} :
      else { print (pf[e]- pf[s-1]);
    }                                          } Q

}
```

$TC = O(n+n+Q)$

$= O(2n+Q)$

$TC = O(n+q) = O(n)$

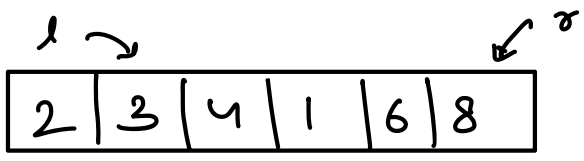
$SC = O(n)$

Prefix Array approach → Continous segment of the array

Doubts

Pick ele from both sides

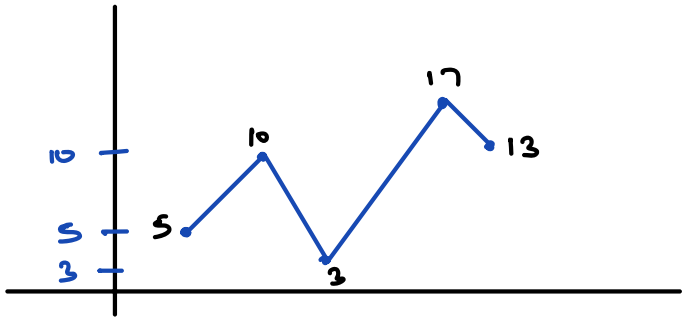
max =



B=3

	left	right		
Possible scen =	3	0	=	2+3+4 + 0 → ans
	2	1	=	ans − 4 + 8 = ans
play with ptrs	1	2	=	ans − 3 + 6 = ans
	0	3	=	ans − 2 + 1 = ans

Buy & Sell stock Question (1 iteration)



int minbp = ~~∞~~ ~~5~~ 3

int pist = ~~0~~ 5

int maxop = ~~0~~ ~~5~~ 14

minbp = 5

```
for (i=0; i<n; i++)
    if( arr[i] < minbp){
    |   minbp = arr[i]
    3
pist = arr[i] - minbp
if ( pist > op ){
|   op = pist;
3
```