Today's Content:

- → Prefix Sum Introduction
- → Problems based on prefix sum

Q) Given N array elements & Q queries on same array

For each query calculate sum of all elements in Given Range?

$0 <= [L <= R] < N$

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

Q = 6

L	R		
4	8	:	9
3	7	:	10
1	3	:	12
0	4	:	14
6	9	:	3
7	7	:	-9

Idea:

For all queries, iterate from L-R & get sum

Pseudocode:

```
// Given ar[N]          → # Input size
Read Q;                 → # No: of queries
for(int i=0; i<Q; i++){
    Read L & R
    int sum=0
    for(int j=L; j<=R; j++){
        sum = sum + ar[j]
    }
    print(sum)
}
```

TC: O(N)

Overall TC: Q * [O(N)] ⇒ TC: O(N*Q)

Overall SC: O(1)

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

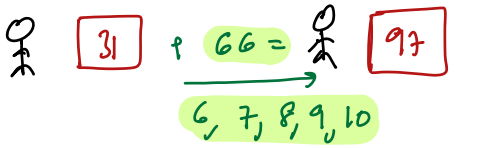
After Every Over Current Score given

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

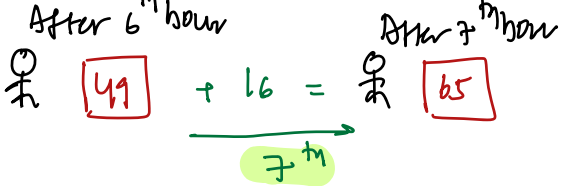
↳ Cummulative data from start / answered range queries easy

Total amount earned in Last hour = 97 - 88 = 9

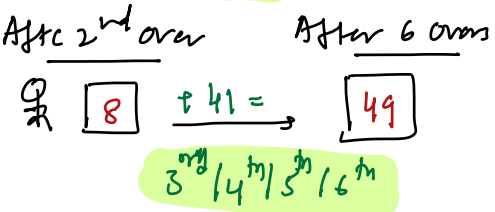
Total amount earned in Last 5 hours = amount earned in After 5th hour to After 10th hour



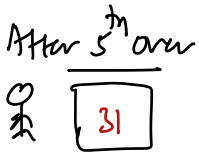
Total amount earned in 7th hour =



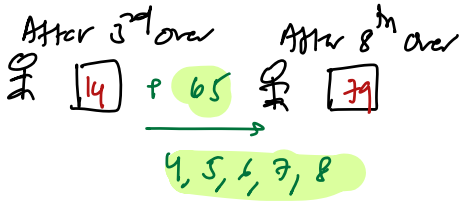
Total runs scored from [3 6] =



Total runs scored from [1 5] =



Total runs scored from [4 8] =



	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

Calculations from 0th index = prefix Sum

pf[3] = 9 = Sum of all array elements from [0-3]

pf[5] = 16 = Sum of all array elements from [0-5]

Un above pf answer below Queries?

L R

sum[4 8] = ar[4] + ar[5] + ar[6] + ar[7] + ar[8]

Add sum[0 3] in both sides

sum[0 3] + sum[4 8] = ar[4] + ar[5] + ar[6] + ar[7] + ar[8]
ar[0] + ar[1] + ar[2] + ar[3]

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

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

$Sum[3-7] = Sum[0-7] - Sum[0\ 2]$

$Sum[3-7] = Pf[7] - Pf[2]$

$Sum[5-9] = Sum[0-9] - Sum[0\ 4]$

$Sum[5-9] = Pf[9] - Pf[4]$

$Sum[2-6] = Sum[0-6] - Sum[0\ 1]$

$Sum[2-6] = Pf[6] - Pf[1]$

$l \neq r$

$Sum[l-r] = Sum[0, r] - Sum[0, l-1]$

$Sum[l-r] = Pf[r] - Pf[l-1]$ → failing if $l == 0$

$Sum[0\ \ 2] = Pf[2] - Pf[-1]$

(L R)

$L \to R \Rightarrow Sum[\underset{\text{Indices}}{l\ r}] \Rightarrow$ if $(l == 0)$: { $Pf[r]$ }
else { $Pf[r] - Pf[l-1]$ }

Given ar[N] → pf[N]

pf[0] = ar[0]

pf[1] = ar[0] + ar[1]

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

pf[2] = pf[1] + ar[2]

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

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

// pf[i] = ar[0] + ar[1] + ar[2] + ar[3] + --- ar[i-1] + ar[i]

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

pf[0] = ar[0]

pf[1] = pf[0] + ar[1] = ar[0] + ar[1]

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

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

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

ar[N] → 0 ... N-1

⇒ given ar[N]

```
int pf[N]
pf[0] = ar[0]
for(i=1; i<N; i++){
    pf[i] = pf[i-1] + ar[i]   → {i==0, fails}
}
```

i=0 ⟶ [i-1] → pf[-1] → Error

// Sum of Q queries

// Given ar[N]

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

i=0 → code fails

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

TC: O(N) + O(Q) → O(N+Q)

SC: O(N) { pf[N], extra space we are creating

ar[5] → 3 2 6 9 1

// Storing cummulative Sum in ar[]

0	1	2	3	4
3	5	11	20	21

ar[1] = ar[0] + ar[1] = 5

ar[2] = ar[1] + ar[2] = 11

ar[3] = ar[2] + ar[3] = 20

ar[4] = ar[3] + ar[4] = 21

Pseudo Code to Store Cummulative in ar[]

```
ar[0] // It won't change
for(i=1; i<N; i++){
    ar[i] = ar[i-1] + ar[i]
}
```

We have cummative in ar[] itself → O(N)

```
Read Q
for(i=0; i<Q; i++){
    Read l, r   [0, r]
    if(l==0) { print(ar[r]) }
    else {
        print( ar[r] - ar[l-1] )
    }
}
```

→ O(Q)

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

Note: If original array cannot be modified, in that case above approach won't work

// In general creating pf[], makes code more readable & understable

28) Equilibrium Index :- [Company]

Given N array elements, count no of equilibrium index?

An index i 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]

Note: if i == 0, left == 0 if i == N-1 right = 0

Ex: ar[4]

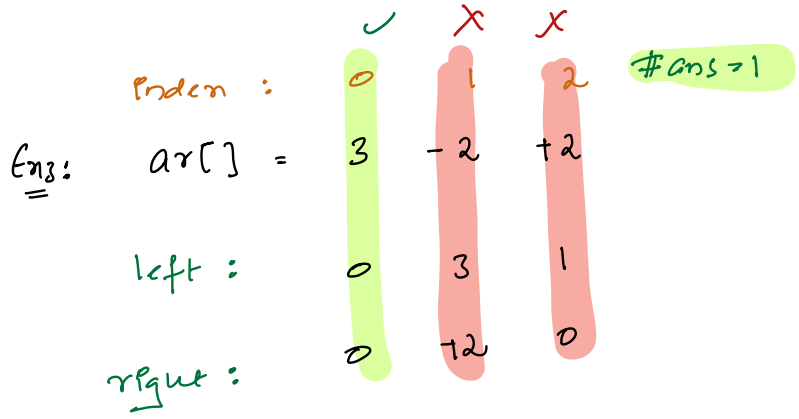
	X	X	✓	X
index:	0	1	2	3
ar[4] =	-3	2	4	-1
left =	0	-3	-1	3
right =	5	3	-1	0

ans = 1

Ex2: ar[7]

	X	X	X	✓	X	X	✓
index:	0	1	2	3	4	5	6
ar[7] =	-7	1	5	2	-4	3	0
left:	0	-7	-6	-1	1	-3	0
right:	7	6	1	-1	3	0	0

ans = 2



Idea: check if every index is equilibrium or not

```
int countEquilibrium(int ar[]) {
    int n = ar.length;
    int c = 0;
    int pf[N]; {Initialize Value TODO}
    for(int i = 0; i < N; i++) {
        // if index i is equilibrium index
        // left = [Sum [0  i-1] = pf[i-1]]
            int left = 0
            if(i > 0) { left = pf[i-1] }
                                  l   r   => pf[r] - pf[l-1]
        // right = [Sum [i+1  N-1] => pf[N-1] - pf[i]]
            int right = pf[N-1] - pf[i]
        if(left == right) { c = c+1 }
    }
    return c;
}
```

TC: O(N)

O(N)

Edge Cases

i=0:

left = pf[-1] : Error

right = pf[N-1] - pf[0]

if i==0, left = 0

i = N-1 : working

left = pf[N-2]

right = pf[N-1] - pf[N-1]
= 0

if i == N-1 right = 0

Overall TC: O(N+N) → O(N) SC: O(N) → ? {Because creating pf[]}

Important:

a) Joining class by 7:00AM, session starts by 7:05AM

↳ alarm: 6:30am, 6:35am, 6:40am, 6:45am

6:50am, 6:55am, 7:00am

b) Explaining focus

c) In Discussion time discuss ideas

d) If doubts there get clarified

2hr:30min: revision 15mins

Once every week spend 1hr on revising everything discussed

Revision

a) First skim notes

b) Again from top revise

Doubts

a) Clarify in Today's Content

b) Last Class TODO

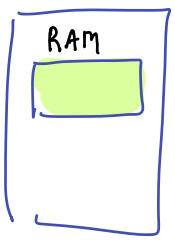
c) Previous assignments

↳ TA?

Doubts:

↳ list : 5 | 6 | 8 | 11 | 14 SC : O(N)

x x x x x



Java:

↳ ArrayList<Integer> ar = new ArrayList<Integers>(); {3 1 4 9}

0 1 2 3

printlist(ArrayList<Integer> ar){

x=3 | x=1 | x=4 | x=9

```
int n = ar.length();
for(int i=0; i<n; i=i+1){
    print(ar.get(i))
}
```

```
for(int x : ar){
    print(x)
}
```