Todays Content

1. PfSum Intro
2. Range sum queries

Thursday Class

1. leaders
2. Count Triplets
3. 0/1 prefixSum

Friday class

1. Quick C++ Transition
2. Vector in C++
3. Pass by value & Pass by reference
4. Vector< Vector< > > or Array of vector.
5. String in C++

Given ar[] & s, e, calculate sum of all elements from s → e:

Eg:

0	1	2	3	4	5	6	7	8	9
3	4	6	8	9	10	2	7	4	10

ar[10] = {3 4 6 8 9 10 2 7 4 10}

s=2 e=7 : ans = 42

```
int sum(int ar[], int s, int e){ TC: O(N) SC: O(1)
    int ans=0;
    for(int i=s; i<=e; i++){
        ans = ans + ar[i];
    }
    return ans;
}
```

Q1: Given ar[N] elements & Qmat[Q][2]

In Qmat matrix, we have Q: rows & 2: columns

Each row in Qmat represents a query.

0th col in row represents : start point of query → s = Qmat[i][0]

1st col in row represents : end point of query → e = Qmat[i][1]

For every query calculate sum of elements from index s..e in ar[] & print

Constraints:

$1 \le N \le 10^5$

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

Sum of array elements:

Min: 1 Max: 10^{14} ⇒ long variable

ar[1] = {1} ar[10^5] = {10^9 10^9 10^9 .. 10^9}

$1 \le Q \le 10^5$

$0 \le s \le e < N$.

Ex:

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

Qmat[6][2]

	0:s	1:e	Output
0	4	8	9
1	3	7	10
2	1	3	12
3	7	7	-9
4	3	6	19
5	0	4	14

Ideal:

For every Query:

Iterate from s..e calculate sum & print

Expected TC: Q*N

```
void RangeSum(int ar[], int N, int Qmat[][], int Q){

    for(int i=0; i<Q; i++){
        int s = Qmat[i][0], e = Qmat[i][1];
        long sum=0; // Need long to avoid wrong answer.
        for(int j=s; j<=e; j++){ TC: O(N)
            sum=sum+ar[j];
        }
        print(sum);
    }

}
```

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

$1 <= N <= 10^5$
$1 <= Q <= 10^5$

→ $10^5 * 10^5 = 10^{10} >> 10^8$ TLE

Optimization Idea:

Say we are given csk cricket scores for first 10 overs of batting. After every over total score is given.

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

Q1: Total runs scored in 10^{th} over = score[10] - score[9] = 9

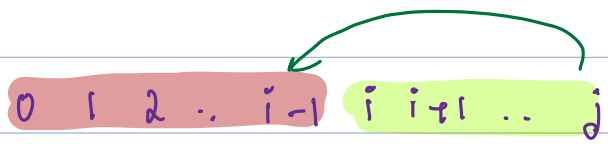
Q2: Total runs scored in 7^{th} over = score[7] - score[6] = 16

Q3: Total runs scored in 6^{th} - 10^{th} over = score[10] - score[5] = 66

Q4: Total runs scored in 3^{th} - 6^{th} over = score[6] - score[2] = 41

Q5: Total runs scored in 4^{th} - 9^{th} over = score[9] - score[3] = 74

Gen: Total runs scored in i^{th} - j^{th} over = score[j] - score[i-1]



Obs: Cummulative Sum = Total sum till that point

Cummulative Sum from 0^{th} index = pfSum

Storing pfSum information in array = Pf[]

If we have Pf[] we can answer queries.

Idea: 1. Let's create pf[N]

2. pf[i] = Sum of array elements from [0..i]

Ex:

	0	1	2	3	4	5	6	7	8	9
ar[10] = {	-3	6	2	4	5	2	8	-9	3	1 }
psum[10] = {	-3	3	5	9	14	16	24	15	18	19 }

Output: Ans using Psum[]

	0:s	1:e	psum[e] - psum[s-1]	
i=0→0	4	8	psum[8] - psum[3] = 18 - 9 =	9
i=1→1	3	7	psum[7] - psum[2] = 15 - 5 =	10
i=2→2	1	3	psum[3] - psum[0] = 9 - (-3) =	12
i=3→3	7	7	psum[7] - psum[6] = 15 - 24 =	-9
4	3	6	psum[6] - psum[2] = 24 - 5 =	19
5	0	4	psum[4] - psum[-1] : Error.	
			→ Sum[0..4] = psum[4] =	14

Query:

[s..e] =

```
if (s==0){ // [0..e]
    print( psum[e] )
}
else {
    print( psum[e] - psum[s-1] )
}
```

Construct Psum[] For Given ar[N]

Step1: To construct pfsum[] we carry forward sum from L → R

	0	1	2	3	4
ar[5] = {	3	-2	4	5	6 }
sum = 0 →	3	1	5	10	16
psum[5] = {	3	1	5	10	16 }

1. update sum
2. Store in psum[]

→ Pfsum[]

TC: O(N+Q) SC: O(N+S) = O(N)

```
void RangeSum(int ar[], int N, int Qmat[][], int Q){
    long psum[N];
    long sum=0;
    for(int i=0; i<N; i++){ → N
        sum = sum + ar[i]
        psum[i] = sum;
    }

    for(int i=0; i<Q; i++){ // iterate in each row ⇒ Query.
        int s= Qmat[i][0], e = Qmat[i][1];
        if(s==0){ // [0..e]
            print( psum[e])
        else{
            print( psum[e] - psum[s-1])
        }
    }
}
```

Note: Modifying same arr[] to Prefix arr[]

arr[5] = { 3 -2 4 5 6 }

sum = 0 → 3 → 1 → 5 → 10 → 16

arr[5] = { 3 1 5 10 16 }

Issues with above?

1. We will loose arr[] information
2. Datatype of arr[] & psum[] can be different

We can update arr[] with pfsum value

1. We are no longer using arr values
2. Datatype of arr[] & psum[] should be same.

Note: When multiple range queries given: think of prefix.