Arrays : Prefix Sum

Agenda

- Prefix Sum Introduction
- Equilibrium Index
- 0-1 Prefix Sum

Q1 Given N array elements & Q queries on same array.
For each query calculate sum of all elements in given range - [L, R]
Note: L & R are indices such that L <= R

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

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

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

Q = 6

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

Idea
For every query, calculate the sum from L to R in a loop

```
solve ( int [] arr ) {
        Q ← input
Q  →    while ( Q > 0 ) {
            Q -= 1
            L, R ← input
            sum = 0
   N {      for (i=L ; i<=R ; i++)
                  sum += arr[i]
            print (sum)
        }
}
```

Total iterations
= Q * N

Time - O(Q N)
Space - O(1)

Worst Case
$N = 10^5$
$Q = 10^5$ } TLE

Q2

Given Indian Cricket Team scores 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
Scores :	2	8	14	29	31	49	65	79	88	97

Cumulative sum

Total runs scored in last over: 97 - 88 = 9 runs

Total runs scored in 7th over: **Quiz 1**

score [7] - score [6]

65 - 49 = 16 runs

Total runs scored in overs 6th to 10th: **Quiz 2**

score [10] - score [6]

[~~1,2,3,4,5,~~6, 7,8,9,10] - [~~1,2,3,4,5,6~~] = [7,8,9,10]

score [10] - score [5]

[~~1,2,3,4,5~~,6, 7,8,9,10] - ~~[1,2,3,4,5]~~ = [6,7,8,9,10]

97 - 31 = 66 runs

Total runs scored in overs 3rd to 6th:

score [6] - score [2]

49 - 8 = 41 runs

Total runs scored in overs i^{th} to j^{th} = score [j] - score [i-1]

Idea - Store the cumulative sum of your array

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

$pf[i] = sum[0 \quad i]$

[4, 8] : $pf[8] - pf[3]$

$= 18 - 9$

$= 9$

[3, 7] : **Quiz 3**

$pf[7] - pf[2]$

$= 15 - 5$

$= 10$

[i, j] :
(i <= j)

$$\begin{cases} \text{if } i > 0, & pf[j] - pf[i-1] \\ \text{if } i == 0, & pf[j] \end{cases}$$

[0, 3] : $pf[3] - \underline{pf[0-1]}$ ↪ Wrong answer / Error

$pf[3]$

How to construct prefix array ?

arr - -3, 6, 2, -9, 5

pf - -3, 3, 5, -4, 1

$pf[0] = \underline{arr[0]}$

$\underline{pf[1]} = \boxed{\underbrace{arr[0]}_{pf[0]} + arr[1]}$

$= pf[0] + arr[1]$

$pf[2] = \underbrace{arr[0] + arr[1]}_{pf[1]} + arr[2]$

$= pf[1] + arr[2]$

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

$pf[i] = pf[i-1] + arr[i]$, $i > 0$

```
arr ← input
pf [N]
pf [0] = arr [0]
for ( i=1 ; i<N ; i++ ) {
        pf[i] = pf[i-1] + arr[i]

}
```

Pseudocode for Q1

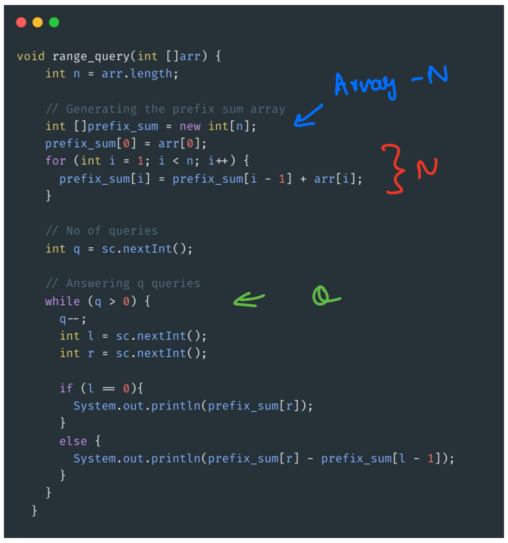
```
solve (int [] arr) {
    // Construct the prefix array
    Q ← input
    while ( Q > 0 ) {
        Q -= 1
        L, R ← input
        // sum [L   R]
        if (L > 0) {
            ans = pf[R] - pf[L-1]
        } else {
            ans = pf[R]
        }
        print (ans)
    }
}
```

Time - O(N+Q)

Space - O(N)

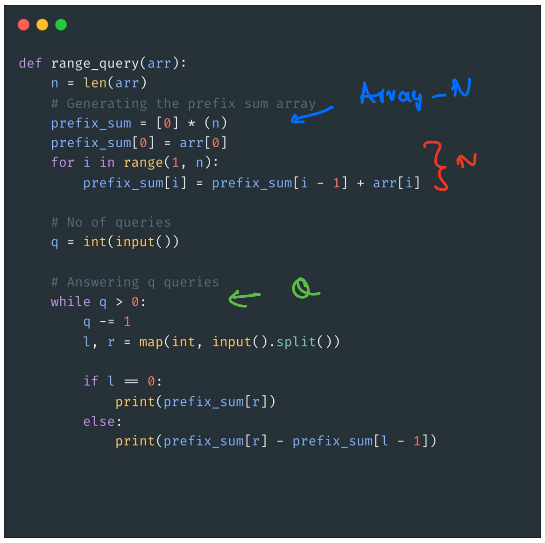
Constraints:
$1 \leqslant N, Q \leqslant 10^5$

Java



```
void range_query(int []arr) {
    int n = arr.length;

    // Generating the prefix sum array
    int []prefix_sum = new int[n];
    prefix_sum[0] = arr[0];
    for (int i = 1; i < n; i++) {
      prefix_sum[i] = prefix_sum[i - 1] + arr[i];
    }

    // No of queries
    int q = sc.nextInt();

    // Answering q queries
    while (q > 0) {
      q--;
      int l = sc.nextInt();
      int r = sc.nextInt();

      if (l == 0){
        System.out.println(prefix_sum[r]);
      }
      else {
        System.out.println(prefix_sum[r] - prefix_sum[l - 1]);
      }
    }
  }
```

Python



```
def range_query(arr):
    n = len(arr)
    # Generating the prefix sum array
    prefix_sum = [0] * (n)
    prefix_sum[0] = arr[0]
    for i in range(1, n):
        prefix_sum[i] = prefix_sum[i - 1] + arr[i]

    # No of queries
    q = int(input())

    # Answering q queries
    while q > 0:
        q -= 1
        l, r = map(int, input().split())

        if l == 0:
            print(prefix_sum[r])
        else:
            print(prefix_sum[r] - prefix_sum[l - 1])
```

Total = N + Q

Worst

Originally, TC = $\underline{QN}$ $\qquad 10^5 \times 10^5 = 10^{10}$

Now, TC = Q + N $\qquad 2 \times 10^5$

Modifying the original array

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

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

Benefit

No extra space

Space - O(1)

Drawback

Lost the original data

Break till 10:21 PM

Q2 Equilibrium Index

Adobe amazon hike

Given N array elements, count no of equilibrium index.

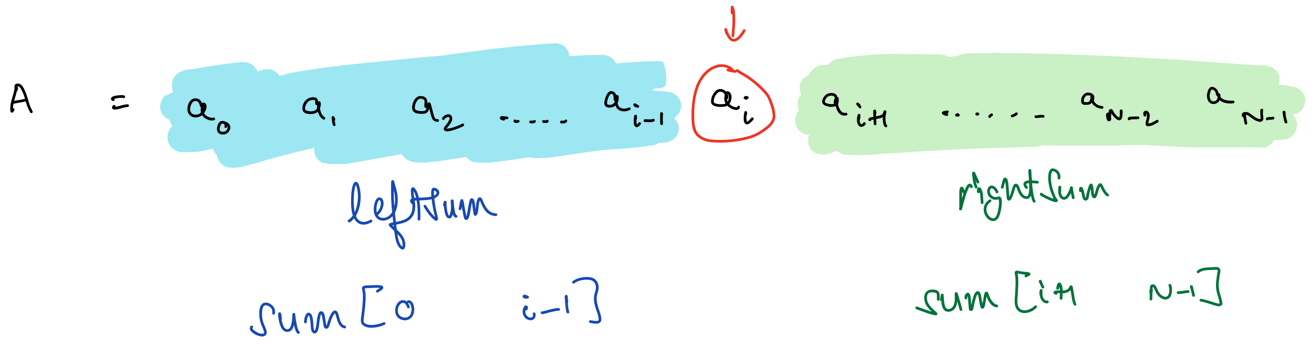
An index i is said to be equilibrium index if:

Sum of all elements left of i^{th} index **=** Sum of all elements right of i^{th} index

Sum[0, i-1] Sum[i+1, N-1]

Note:

- if i == 0, leftSum = 0
- If i == N-1, rightSum = 0



$A = a_0 \quad a_1 \quad a_2 \quad \ldots \quad a_{i-1} \quad a_i \quad a_{i+1} \quad \ldots \quad a_{N-2} \quad a_{N-1}$
leftSum
rightSum
sum [0 i-1]
sum [i+1 N-1]

Example

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

Equilibrium index = 2

Count = 1

Example

		0	1	2
ar [3]	=	3	-2	2
left	:	0	3	1
right	:	0	2	0

Count = 1

Quiz 4

Options
2 | [✓ 3] | 4 | 5

Example

		0	1	2	3	4	5	6
arr[7]	:	-7	1	5	2	-4	3	0
left	:	0	-7	-6	-1	1	-3	0
right	:	7	6	1	-1	3	0	0

Count = 2

Example

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

Logic & Pseudocode

→ for every index i, just check if leftSum == rightSum

```
int equilibriumIndex(int []arr) {
        count = 0
        // Construct prefix array        ← N iterations
      for (i=0; i<N; i++) {            ← N iterations
            // Check if i is equilibrium
            leftSum = sum [0  i-1]   → pf [i-1]
            rightSum = sum [i+1  N-1]  → pf [N-1] - pf [i]
            if ( leftSum == rightSum)
                count ++
      }
      return count
}
```

```
if L>0
  pf [R] - pf [L-1]
else
  pf [R]
```

Time - O(N)
Space - O(N)

Total iterations
= N + N ≈ 2N

Extra space for prefix array

Q3   0-1 Prefix Sum

Given N array elements & Q queries containing l & r each. Find no of even numbers in given range.

Example

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

Q = 3

l	r	ans
0	4	2
4	8	3
3	9	3

Brute Force

for every query, iterate l to r and count even

```
solve ( int [] arr) {
        Q <- input
        while ( Q > 0 ) {
            Q -= 1
            l, r <- input
            count = 0
            for(i=l ; i<=r ; i++)
                if ( arr[i] % 2 == 0 )
                    count += 1
            print(count)
        }.
}
```

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

Optimised Approach

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

↑ Count of even numbers

pf [i] = evenCount [0   i]

even → 1
odd → 0

```
evenCount ( int [] arr) {
        // Construct prefix array
        pf [N]
        if (arr[0] is even):
            pf [0] = 1
        else
            pf[0] = 0
        for (i=1; i<N ; i++) {
            pf [i] = pf[i-1] + ______   even → 1
                                        odd → 0     } N iterations
        }
        // Answering queries
        Q ← input                                  ← Q iterations
        while (Q>0) {
            Q - =1
            l, r ← input
            // Print   evenCount [ l   r ]
        if (l == 0)
            print ( pf[r])
        else
            print( pf[R] - pf[L-1])
    }
}
```

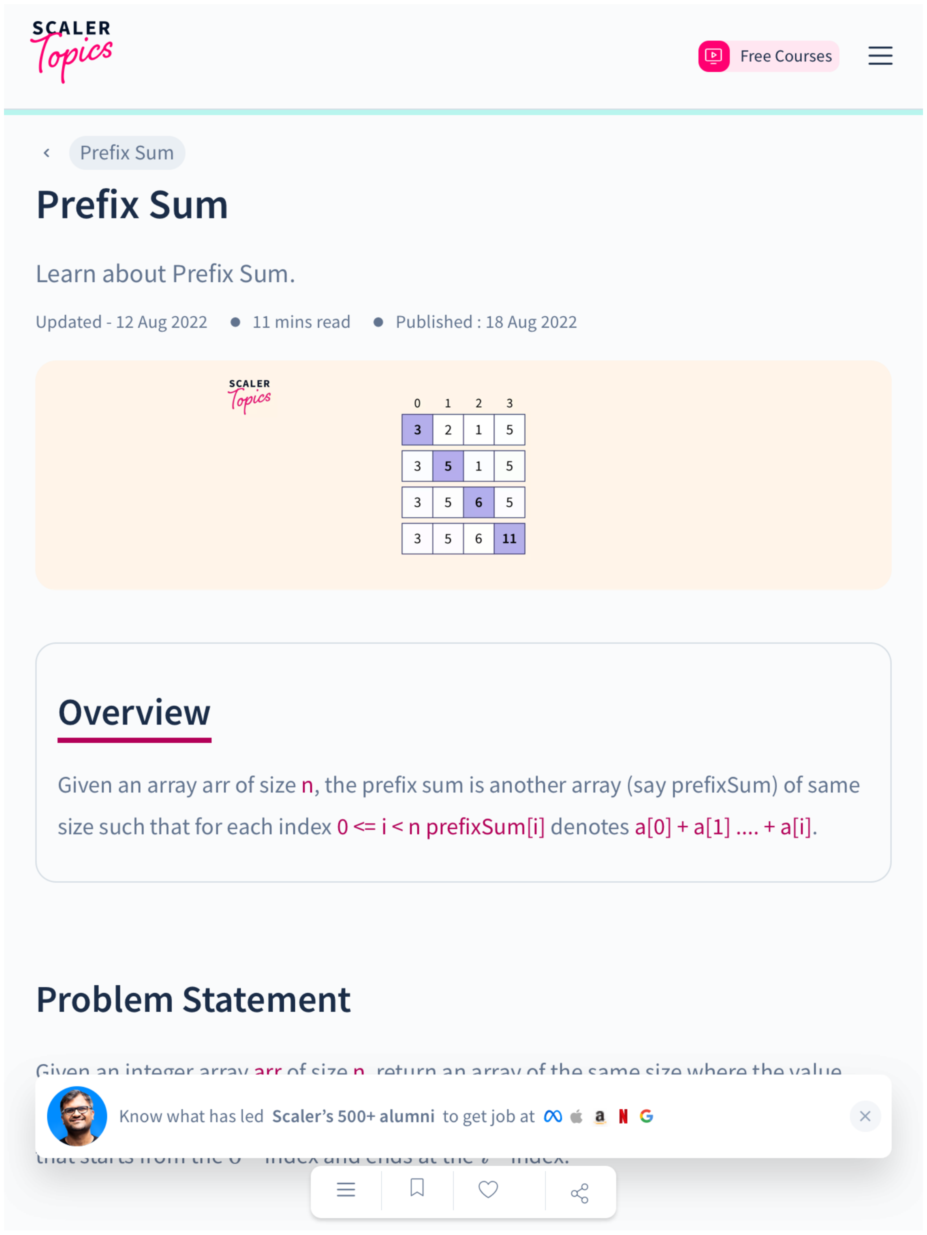
Total iterations = $N + Q$

$$\boxed{\begin{array}{l} TC : O(N+Q) \\ SC : O(N) \end{array}}$$

SC can be optimised to O(1) if you modify the original array.

Bonus Reading Material

https://www.scaler.com/topics/prefix-sum/



SCALER Topics

Free Courses

Prefix Sum

Prefix Sum

Learn about Prefix Sum.

Updated - 12 Aug 2022 ● 11 mins read ● Published : 18 Aug 2022

0	1	2	3
3	2	1	5
3	5	1	5
3	5	6	5
3	5	6	11

Overview

Given an array arr of size n, the prefix sum is another array (say prefixSum) of same size such that for each index 0 <= i < n prefixSum[i] denotes a[0] + a[1] .... + a[i].

Problem Statement

Know what has led Scaler's 500+ alumni to get job at

Doubts

Thank You

5 9 2

5 14 16

In Q1 $1 <= N, Q <= 10^5$

worst case, $N = 10^5$

$Q = 10^5$

Brute force $\rightarrow$ TC: $O(QN)$

worst $Q * N \rightarrow 10^5 \times 10^5 = \underbrace{10^{10} \text{ iterations}}_{\text{TLE}}$

Find max

Approach 1 - Run a loop - $O(N)$

2 - Sort the array - $O(N \log N)$

pick arr[0]

Example

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

Count = 2

Good Night

Thank You

Wednesday