Algorithm – Prefix Sum [Notes]

Priyanshu Shukla
MTech CSE
IIT Ropar

→ Inbuilt f → accumulate(), partial_sum()

STL

① Prefix Sum Algorithm :-

i =	0	1	2	3	4	5	6	
A =	6	3	-2	4	-1	0	-5	...n

- Calculate sum b/w [0, n) ?

pseudo code →

```
sum = 0;
for (i=0; i<n; i++) {
        sum += A[i];
}
```

→ T = O(n)

- Calculate sum b/w [start, end] ?

pseudo code →

```
sum = 0;
for (i=start ; i<end ; i++) {
        Sum += A[i];
}
```

→ T = O(n)

- Calculate sum b/w [0, 3] ?
- Calculate sum b/w [2, 3] ?
- Calculate sum b/w [4, 6] ?

⋮

'm'

There are → 'm' such queries

→ Time to perform one query → O(n)

∴ to perform 'm' queries → O(m*n)

But an efficient algo can do it in -

→ O(n) time

→ Prefix Sum algo

Algorithm :- Simple yet powerful technique that allows us to perform fast calculation on sum of elements in a given range (contiguous segments of array)

Eg :-

i =	0	1	2	3	4	5	6
A[] =	6	3	-2	4	-1	0	-5
prefix sum array, A[] =	6	9	7	11	10	10	5

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

- Calculate sum b/w [0,4] → A[4]
- Calculate sum b/w [0,6] → A[6]
- Calculate sum b/w [2,6] → A[4]

→ $A[0,6] = A[0,1] + A[2,6]$

$A[2,6] = A[0,6] - A[0,1]$

→ $A[6] - A[1]$ } → O(1)[T]

$A[3,5] = A[5] - A[3-1]$
$= A[5] - A[2]$ } → O(1)[T]

Time Complexity :- $O(n) + O(1)$
$\sim O(n)$

$$\boxed{A[i,j] = A[j] - A[i-1]}$$