Subarrays

- Contiguous part of an array
- Complete array is a subarray of its own.
- A single element is a subarray of a given array
- Empty array is a sub-array

  But we will only consider non-empty sub arrays.

A : 3, 4, $\overline{5, 6, -2}$, 8, 10

[5, 6, -2] ✓

[3, 4, 6, -2] ✗

[10] ✓

[ ] ✓ (theoretically)

[4, 6, 5] ✗

No of sub arrays of

[$\overset{0}{4}$, $\overset{1}{2}$, $\overset{2}{10}$, $\overset{3}{3}$, $\overset{4}{12}$, $\overset{5}{-2}$, $\overset{6}{15}$]

s	e	
0	0	→ [4]
0	1	→ [4, 2]
0	2	→ [4, 2, 10]
0	3	→ [4, 2, 10, 3]
⋮		⋮
0	6	→ [4, 2, 10, 3, 12, -2, 15]

No of sub arrays

# subarrays starting at index 0	⟶	N
# subarrays starting at index 1	⟶	$N-1$
# subarrays starting at index 2	⟶	$N-2$
# subarrays starting at index 3	⟶	$N-3$
⋮		⋮
# subarrays starting at index N-1	⟶	1
Total # subarrays	⟶	$\frac{N(N+1)}{2} \approx O(N^2)$

Q Print all values of a subarray starting at s & ending at e.

```
void printSubarray ( A[], s, e ) {
    for ( i= s; i<=e; i++){
            Print(A[i]);
    }
}
```

TC: $O(N)$

Q Return the sum of a given subarray.

```
int addSubarray ( A[], s, e ) {
sum = 0;
for ( i = s; i <= e; i++){
        Sum = sum + A[i];
}
return sum;
}
```

TC: O(N)

Q Print all sub-arrays of a given array A (size N)

```
Void printAllSubarrays ( A[] ) {

    for (i=0; i<N; i++){
        for ( j= i; j<N; j++) {
            Print (A[j]);
        }
        Print ln();
    }
}
```

wrong →

0	1	2	3
1	2	3	4

i	OutPut
0	1, 2, 3, 4
1	2, 3, 4
2	3, 4
3	4

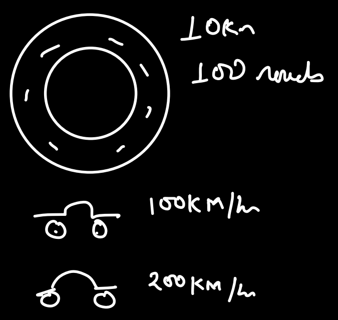
1
1, 2
1, 2, 3
2
2, 3
⋮

0	1	2	3
1	2	3	4

S	e	
0	0	1
0	1	1, 2
0	2	1, 2, 3
0	3	1, 2, 3, 4
1	1	2
1	2	2, 3
1	3	2, 3, 4
2	2	3
2	3	3, 4
3	3	4

```
void    printAllSubarrays (A[]) {
        // i is start
        for (i=0;  i<N;  i++) {
                // j is end
                for (j=i; j<N;  j++) {          } O(N^3)
                        // Print elements of subarray from i to j
                        printSubarray (A, i, j); ---> O(N)
                }
        }
}
```

TC : $O(N^3)$



Amazon

Q Print the sum of every single subarray of the given array.

A : $\overset{0}{3}$, $\overset{1}{2}$, $\overset{2}{-1}$, $\overset{3}{4}$

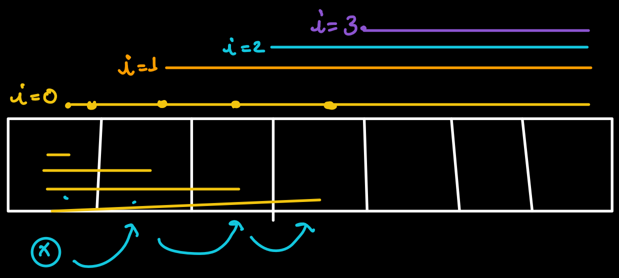
0	1	2	3
1	2	3	4

S	C	Subarray	Sum
0	0	3	3
0	1	3, 2	5
0	2	3, 2, -1	4
0	3	3, 2, -1, 4	8
⋮			⋮

```
Void    add All Subarrays ( A[] ) {
        // i is start
        for ( i= 0;  i<N ;  i ++) {
                // j is end
                for ( j = i ;  j < N ;   j++) {          } O(N²)
                                                           Iterating over all subarray
Replace by   {  // Print elements of subarray from i to j
RS (O(1))    {  Print(add Subarray ( A, i, j));   → O(N)
                }
        }
}
```

TC : O(N³) $\xrightarrow{\text{using PS}}$ O(N²)

SC : O(1)   O(N)
(Extra)



0	1	2	3
1,	2,	3,	4

i	j			Sum	
0	0	1	⟶	1	
0	1	1, 2	⟶	3	(1+2)
0	2	1, 2, 3	⟶	6	(3+3)
0	3	1, 2, 3, 4	⟶	10	(6+4)
			⟶	sum = 0	
1	1	2	⟶	2	
1	2	2, 3	⟶	5	(2+3)
1	3	2, 3, 4	⟶	9	(5+4)

Carry Forward

```
for ( i=0; i<N ; i++) {
    sum = 0;
    for ( j= i ; j<N; j++) {
        sum = sum + A[j];
        Print (sum),
    }
}
```

TC: $O(N^2)$

SC: $O(1)$

0	1	2	3
3	2	1	7

i	j	Sum
0	0	3
0	1	5
0	2	6
0	3	13 ⟶ 0
1	1	2
1	2	3
1	3	10 ⟶ 0

Google
Facebook

Q Given an array. Find the sum of all possible subarray sums.

0	1	2	3
1	2	3	4

s	e		
0	0	1	→ 1
0	1	1, 2	→ 3
0	2	1, 2, 3	→ 6
0	3	1, 2, 3, 4	→ 10
1	1	2	→ 2
1	2	2, 3	→ 5
1	3	2, 3, 4	→ 9
2	2	3	→ 3
2	3	3, 4	→ 7
3	3	4	→ 4

Sum of all subarrays sums ⟶ 50

Iterate over all subarrays
& keep adding the sums

Brute Force
TC : $O(N^3)$
SC : $O(1)$

Prefix Sum
TC : $O(N^2)$
SC : $O(N)$

Carry forward
TC : $O(N^2)$
SC : $O(1)$

0	1	2	3
1	2	3	4

s	e		
0	0	1	→ 1
		+	
0	1	1 + 2	→ 3
		+	
0	2	1 + 2 + 3	→ 6
		+	
0	3	1 + 2 + 3 + 4	→ 10
		+	
1	1	2	→ 2
		+	
1	2	2 + 3	→ 5
		+	
1	3	2 + 3 + 4	→ 9
		+	
2	2	3	→ 3
		+	
2	3	3 + 4	→ 7
		+	
3	3	4	→ 4

1 + 1+2 + 1+2+3 + 1+2+3+4
2 + 2+3 + 2+3+4
3 + 3+4
4

⟶

1 + 1 + 1 + 1
2 + 2+2+ 2+2 +2 → 6 ×2
3 + 3+3 + 3+3 +3
4 +4 +4 +4

4 ×1
+
6 ×2
+
6 ×3
+
4 × 4

50

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

Q No of subarrays containing index 3.

0	1	2	3	4	5
3,	−2,	4,	−1,	2,	6

s	e	
0	3	3, −2, 4, −1
0	4	3, −2, 4, −1, 2
0	5	3, −2, 4, −1, 6
1	3	−2, 4, −1
1	4	−2, 4, −1, 2
1	5	−2, 4, −1, 2, 6
2	3	4, −1
2	4	4, −1, 2
2	5	4, −1, 2, 6
3	3	−1
3	4	−1, 2
3	5	−1, 2, 6

$|s| = 4$

$|e| = 3$

$= 4 * 3$

$$\boxed{\begin{array}{l} e \geq i \\ s \leq i \end{array}}$$

Element at index i will be present in all subarrays for which

$s <= i \longrightarrow [0, i] \longrightarrow i+1$ options

& $e >= i \longrightarrow [i, N-1] \longrightarrow N-i$ options

⇒ Count of subarrays in which i^{th} element is present $= (i+1) * (N-i)$

		0	1	2	3	4	5
A :		3,	−2,	4,	−1,	2,	6
$\lvert S\rvert = (i+1)$	⟶	1	2	3	4	5	6
$\lvert e\rvert = (N-i)$	⟶	6	5	4	3	2	1
# subarrays in which ith element is prest	⟶	6	10	12	12	10	6

$$\Sigma \text{subarray} = \sum_{\forall i} A[i] \times \text{Count}(A[i])$$

$$= (3 \times 6) + (-2 \times 10) + (4 \times 12) + (-1 \times 12) + (2 \times 10) + (6 \times 6)$$

$$= 90$$

```
sum = 0;
for( i=0; i<N; i++){
        Count = (i+1) x (N-i);
        Sum = Sum + A[i] x Count;
}
```

TC : O(N)

SC : O(1)

Contribution Tech

$[a, b] \rightarrow b-a+1$

$[0, i] \rightarrow i-0+1 \Rightarrow i+1$