Subarrays → contiguous part of array

3	6	9	12	14	19	20	23	25	27

14 19 20 23 ✓

9 6 12 ✗

6 9 12 ✓

20 ✓

9 6 3 ✗

complete array is also a subarray

uniquely identify the subarray

↓

[start, end] , start <= end

end < N

0	1	2	3	4	5	6
4	2	10	3	12	-2	15

start	end
1	1
1	2
1	3
1	4
1	5
1	6

ans = 6

count all subarrays

start	# count
0	7
1	6
2	5
3	4
4	3
5	2
6	1
	28

count of subarray in array of size N

start	# cnt
0	N
1	N-1
2	N-2
3	⋮
⋮	⋮
N-1	1

$$\boxed{N*(N+1)/2}$$

```
printsubarray ( arr[], int start, int end)
{
        for( i= start ; i<=end; i++)
              print(arr[i]);
}
```

```
sum subarray ( arr[], int start, int end)
{
          int sum =0;
        for( i= start ; i<=end; i++)
              sum += arr[i];
}
```

print all subarrays

$$\begin{array}{cccc} & 0 & 1 & 2 \\ A[\,]: & 2 & 8 & 9 \end{array}$$

$$\frac{N*(N+1)}{2}$$

s	e	
0	0	[2]
0	1	[2, 8]
0	2	[2, 8, 9]
1	1	[8]
1	2	[8, 9]
2	2	[9]

To consider all subarrays, you have to consider all possible pairs of (s, e)

0	1	2	3
4	1	3	6

(0,0)	(0,1)	(0,2)	(0,3)
	(1,1)	(1,2)	(1,3)
		(2,2)	(2,3)
			(3,3)

$\boxed{T.C: O(N^3)}$

```
for( i=0; i<N; i++) → fix start
{
        for( j = i; j<N; j++) → fix end
        {
                [i, j]
                for( k = i; k<=j; k++)
                        print(arr[k])
        }
}
```

- ## Find sum of each subarray

$$A[\,]: \begin{array}{ccc} 0 & 1 & 2 \\ 2 & 8 & 9 \end{array}$$

B.F: consider all subarray & iterate.

s	e	
0	0	~~[2]~~ 2
0	1	~~[2, 8]~~ 10
0	2	~~[2, 8, 9]~~ 19
1	1	~~[8]~~ 8
1	2	~~[8, 9]~~ 17
2	2	~~[9]~~ 9

```
for( i=0; i<N; i++) → fix start
{
    for( j = i; j<N; j++) → fix end
    {
        [i, j]    int sum =0;
        for( k = i; k<=j; k++)
            sum += arr[k];

        print(sum);
    }
}
```

$$\boxed{\begin{array}{l} T.C:\ O(N^3) \\ S.C:\ O(1) \end{array}}$$

```
// build pf sum
for( i=0; i<N; i++) → fix start
{
        for( j = i; j<N; j++) → fix end
        {
                    [i, j]
                    if (i==0) sum = pf[j];
                    else  sum = pf[j] - pf[i-1];
                    print(sum);
        }
}
```

T.C: $O(N^2)$

S.C: $O(N)$

0	1	2	3	4	5	6	7	8	9
3	6	9	12	14	19	20	23	25	27

[2,2] → arr[2]

[2,3] → arr[2] + arr[3] = [2,2] + arr[3]

(2,4) → arr[2] + arr[3] + arr[4] = [2,3] + arr[4]

[i, j] → [i — j-1] + arr[j]

```
for( int i=0; i<N; i++)
{
            sum=0;
      for( int j = i; j<N; j++)
      {
                sum += arr[j]
                print(sum);
      }
}
```

0	1	2	3
4	2	-1	3

i	j	sum =0
0	0	4
	1	4+2=$\boxed{6}$
	2	6+(-1)=5
	3	5+3 =8 sum =0
1	1	2
	2	2+(-1)=1
	3	1+3 =4

T.C: $O(N^2)$

S.C: $O(1)$

- Find total sum of all subarray sums.

0	1	2	3
3	2	-1	5

s	e	sum
0	0	3
0	1	5
0	2	4
0	3	9
1	1	2
1	2	1
1	3	6
2	2	-1
2	3	4
3	3	5

38

```
int ts =0;
for( int i=0; i<N; i++)
{
                sum=0;
        for( int j = i; j<N; j++)
        {
                sum += arr[j]
                ts += sum;
        }
}
```

T.C: $O(N^2)$

S.C: $O(1)$

	0	1	2
A:	-1	3	4

0, 0	-1	arr[0]		
0, 1	2	arr[0] +	arr[1]	
0, 2	6	arr[0] +	arr[1] +	arr[2]
1, 1	3		arr[1]	
1, 2	7		arr[1] +	arr[2]
2, 2	4			arr[2]

$21 = 3 * arr[0] + 4 * arr[1] + 3 * arr[2]$

$3*(-1) + 4*3 + 3*4 = 21$

contribution of i^{th} element = No of subarrays in which a i^{th} element come * arr[i]

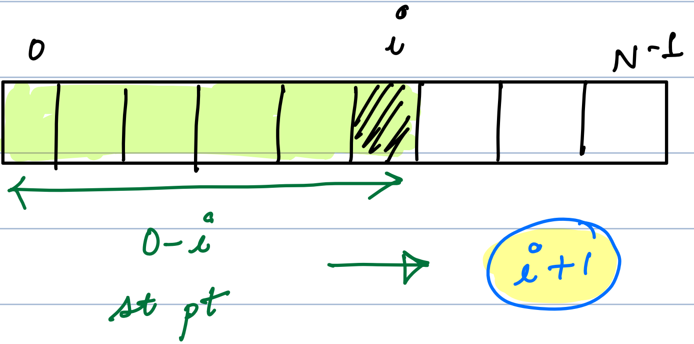
- In how many subarrays i^{th} element is coming?

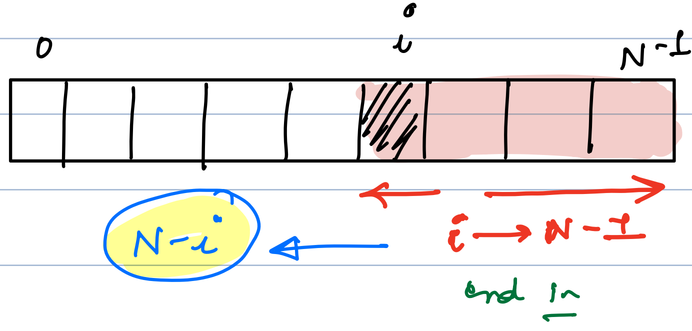
	0	1	2	3	4	5
A:	3	-2	4	1	2	6

0, 2	1, 2	2, 2
0, 3	1, 3	2, 3
0, 4	1, 4	2, 4
0, 5	1, 5	2, 5

ans = 12

= no of st pt * no of end pts





$(N-\cancel{1})-i+\cancel{1}$
$= N-i$

$$\boxed{\text{No of subarrays} = (i+1) * (N-i)}$$

cont of i^{th} ele = $(i+1) * (N-i) * arr[i]$

$$\boxed{\begin{array}{l} T.C: O(N) \\ S.C: O(1) \end{array}}$$

```
int ans = 0;
for( i=0; i<N; i++)
{
        ans += (i+1) * (N-i) * arr[i];
}
```

Pick from both sides!

B elmts

B = 5