subarrays — contiguous part of an array

0	1	2	3	4	5	6	7	8
4	1	2	3	-1	6	9	8	12

whole array → subarray

single element → subarray

2 3 -1 6	✓	2,5
4 12	✗	
1 2 6	✗	
9	✓	6,6

(start, end) → uniquely represent subarray

start <= end

sequence, what are the elements!

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

start = 1

s	e
1	1
1	2
1	3
1	4
1	5
1	6

$\boxed{ans = 6}$

Total no of

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

count of total subarray in an array of size N

subarray starting from 0 → N
from 1 → N-1
2 → N-2
3 → N-3
⋮ ⋮
N-1 → 1

count of subarray ⇒ $N*(N+1)/2$

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

}
```

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

- print all possible subarrays

	0	1	2
A:	2	8	9

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

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)

```
// first fix starting pt.

for( i=0; i<n; i++)
{
    for( j=i; j<n; j++)
    {
        // i, j
        for(k=i; k<=j; k++)
            print(arr[k]);
    }
}
```

T.C: $O(N^3)$

can't reduce further

Find sum of each & every subarray

0	1	2	3
3	2	-1	5

$O(\cancel{N}) \approx N^2$

		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

```
for( i=0; i<n; i++)
{
    for( j=i; j<n; j++)
    {      int sum=0;
           // i,j
           for(K=i; K<=j; K++)
               sum += arr[K];
           print(sum);
    }
}
```

prefix sum — gives you a range sum — O(1)

```
for( i=0; i<n; i++)
{
    for( j=i; j<n; j++)
    {
        // i,j
        if( i==0) sum = pf[j];
        else  sum = pf[j] - pf[i-1];
    }
}
```

$$\boxed{\begin{array}{l} T\cdot C: N + N^2 \\ \quad \approx O(N^2) \\ S\cdot C: O(N) \end{array}}$$

sum of subarrays starting from index = 2

0	1	2	3	4	5	6
7	3	2	-1	5	6	8

sum = 0;

i j

[2, 2] = arr[2]

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

[2, 4] = arr[2] + arr[3] + arr[4]

[2, 5] = arr[2] + arr[2] + arr[4] + arr[5]

carry forward

```
for( i=0; i<n; i++)
{
    sum=0;
    for( j=i; j<n; j++)
    {
        // i, j
        sum += arr[j];
        print(sum);
    }
}
```

0	1	2	3
4	2	1	-3

i	j	sum
0	0	0+4
0	1	4+2=6
0	2	6+1=7
0	3	7+(-3)=4
1	1	0+2
1	2	2+1
1	3	3-3

sum

4 6 7 4 2 3 0

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

S.C: $O(1)$

10:28

Q Find total sum of all subarray sums!

Google
FB

0	1	2	3
3	2	-1	5

		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

```
totalsum = 0;
for( i=0; i<n; i++)
{
    sum = 0;
    for( j = i; j<n; j++)
    {
        // i, j
        sum += arr[j];
        totalsum += sum;
    }
}
```

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

	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]

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

$-3 + 12 + 12 = 21$

count of subarray in which a particular element comes

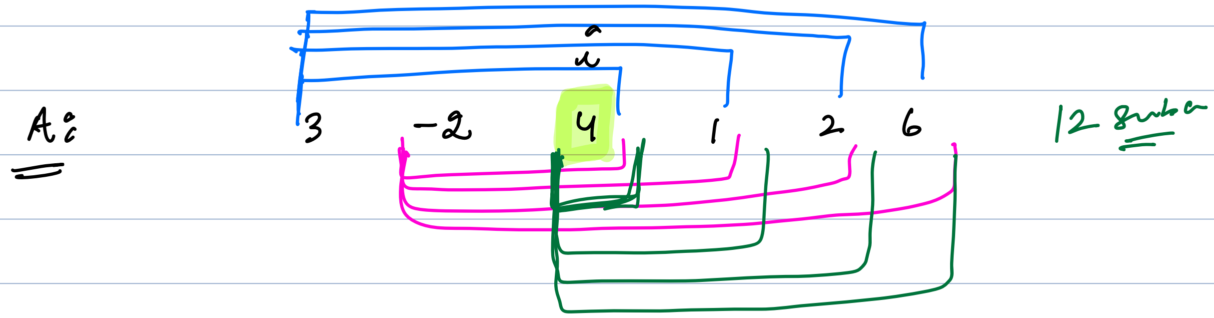
0	1	2	3
4	-1	2	3

0	0	4				= 4
0	1	4	-1			3
0	2	4	-1	2		5
0	3	4	-1	2	3	8
1	1		-1			-1
1	2		-1	2		1
1	3		-1	2	3	4
2	2			2		2
2	3			2	3	5
3	3				3	3

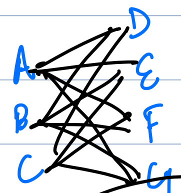
$4*4 + 6*(-1) + 6*2 + 4*3$

$16 - 6 + 12 + 12 = 34$

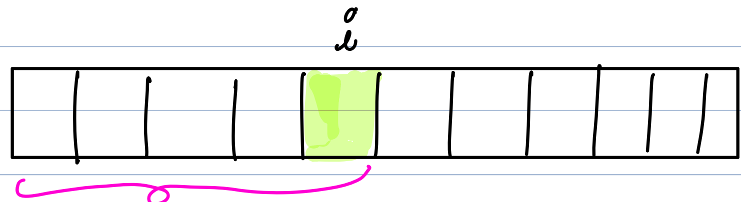
In how many subarrays i^{th} element is present?



start, end



$3 * 4 = 12$

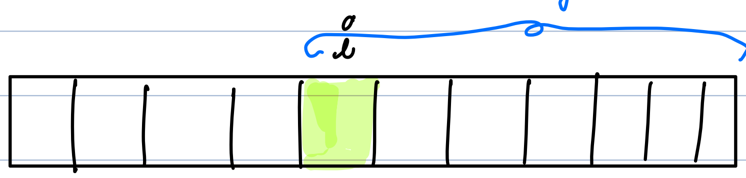


start point

$0 - i$

$i \to n-1$

ending pt



start pt $\Rightarrow$ $i+1$ $(0 - i)$

end pt $\Rightarrow$ $n - i$ $(i \to n-1)$

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

$$\boxed{\# \text{ cnt of subarr} = (i+1) * (n-i)}$$

$$\boxed{\text{final contribution of } i^{th} \text{ element} = (i+1) * (n-i) * arr[i]}$$

ans = 0

```
for( i=0; i<n; i++)
        ans += (i+1) * (n-i) * arr[i]
```

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