Subarray

↳ continous part of an array

$$A = \begin{array}{ccccccc} [2 & 3 & 9 & 4 & -5 & 6 & 7] \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

[9 4 -5 6] → ✓

[9 4 6 7] → X

i) A single ele is also a subarray

ii) A complete array is also a subarray

iii) order of elements matters.

$$A = \begin{array}{ccccccc} [2 & 3 & 9 & 4 & -5 & 6 & 7] \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

[4] → ✓

[4 -5 6] ✓

[4 9 3] X

$A = [3 \quad 9 \quad 4 \quad 10 \quad -5 \quad 12 \quad 7]$

9 4 10 −5 ✓

10 4 9 × (order)

10 ✓

10 −5 12 ✓

$A = [3 \quad 9 \quad 4 \quad 10 \quad -5 \quad 12 \quad 7]$

3 ✓

9 4 10 ✓

9 4 −5 12 × (discontinous)

12 7 ✓

$$A = \begin{array}{ccccccc} [3 & 9 & 4 & 10 & -5 & 12 & 7] \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{array}$$

A particular subarray can be defined using start & end

$s = 2$ $[4 \quad 10 \quad -5 \quad 12]$

$e = 5$

$len = e - s + 1$

$$A = \begin{array}{cccc} [3 & 1 & 4 & 5] \\ 0 & 1 & 2 & 3 \end{array}$$

s	e	sa
0	0	3
0	1	3 1
0	2	3 1 4
0	3	3 1 4 5

s	e	sa
1	1	1
1	2	1 4
1	3	1 4 5

s	e	sa
2	2	4
2	3	4 5

s	e	sa
3	3	5

$A = [\, a_0 \quad a_1 \quad a_2 \quad \ldots\ldots \quad a_{n-1} \,]$ length $= n$

no. of subarrays starting from 0th index $= n$

no. of subarrays starting from 1st index $= n-1$

$\vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots$

no. of subarrays starting from $(n-1)$th index $= 1$

$$n + (n-1) + (n-2) + \ldots + 2 + 1$$

$$\boxed{\text{total subarray} = \frac{n(n+1)}{2}}$$

Q.1 Given an A[] and s & e. Print the subarray from s to e.

s = 1
e = 3

$$\begin{array}{lcccccc} A = & [3 & 4 & 5 & 9 & 7 & 8] \\ & 0 & 1 & 2 & 3 & 4 & 5 \end{array}$$

```
void print (int[] A, int s, int e) {

        for (int k = s; k <= e; k++) {
                SOP(A[k]);

        }

}
```

Q.2 Given an array, print all subarrays.

$$\begin{array}{lccc} A = & [3 & -1 & 2] \\ & 0 & 1 & 2 \end{array}$$

(0,0) → 3

(0,1) → 3 -1

(0,2) → 3 -1 2

(1,1) → -1

(1,2) → -1 2

(2,2) → 2

```
void   printAll (int [] A) {
    int  n= A.length;
    for( int  s=0;  s<n;  s++) {
        for (int  e=s;  e<n;  e++) {
            // to  print  subarray  from  s to e
            for ( int  k = s;  k <= e;  k++) {
                      SOP (A [k] + " ");
            }
            SOPln();
        }
    }
}
```

A =	3	1	5
	0	1	2

s	e	k
0	0	0,0 → 3
	1	0,1 → 3 1
	2	0,2 → 3 1 5
1	1	1,1 → 1
	2	1,2 → 1 5
2	2	2,2 → 5

TC : $O(n^3)$

SC : $O(1)$

Q.3 Given an A[], print sum of every subarray.

$$\begin{array}{cccc} A = [& 3 & -1 & 4] \\ & 0 & 1 & 2 \end{array}$$

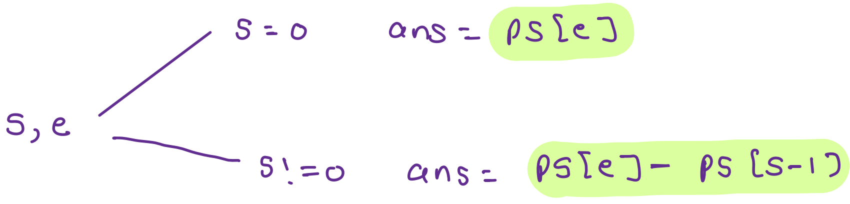
s	e	sa	sum
0	0	[3]	3
0	1	[3 -1]	2
0	2	[3 -1 4]	6
1	1	[-1]	-1
1	2	[-1 4]	3
2	2	[4]	4

```
void solve (int[] A) {
    int n= A.length;
    for(int s=0; s<n; s++) {
        for (int e=s; e<n; e++) {
            int sum=0;
            for(int k=s; k<=e; k++) {
                    sum+= A[k];
            }
            Sopln(sum);
        }
    }
}
```

TC: $O(N^3)$

SC: $O(1)$

Improvise → Prefix sum



```
void  solve (int [] A) {

    int  n= A.length;

    int [] ps= prefixsum (A);

    for (int  s=0;  s<n;  s++) {

        for (int e=s; e<n; e++) {

            if (s==0) {

                sopln (ps[e]);

            }
            else {

                sopln (ps[e] - ps[s-1]);

            }
        }
    }
}
```

A =	3	2	-4	6
	0	1	2	3
ps =	3	5	1	7
	0	1	2	3

s=0, e=0, ps[0] = 3

s=0, e=1, ps[1] = 5

s=0, e=2, ps[2] = 1

s=0, e=3, ps[3] = 7

⋮

s=1, e=3, ps[3]-ps[0]

= 7-3=4

TC: O(N^2)

SC: O(N)

further Improvisation (carry forward)

$$A = \begin{array}{cccccc} [3 & 4 & -2 & 5 & 1] \\ 0 & 1 & 2 & 3 & 4 \end{array}$$

s=1 , e: 1 2 3 4

(1,1) sum= A[1]

(1,2) $sum = \underbrace{A[1]}_{sum} + A[2]$

(1,3) $sum = \underbrace{A[1] + A[2]}_{sum} + A[3]$

(1,4) $sum = \underbrace{A[1] + A[2] + A[3]}_{sum} + A[4]$

```
void solve (int[] A) {
    int n= A.length;
    for (int s=0; s<n; s++) {
        int sum=0;
        for (int e=s; e<n; e++) {
                sum+= A[e];
                sopln(sum);
        }
    }
}
```

TC: O(n²)

SC: O(1)

$$A = \begin{array}{ccc} [3 & 2 & 4] \\ 0 & 1 & 2 \end{array}$$

s	e	sum	o/p
0	0	3	3
	1	3+2	5
	2	3+2+4	9
1	1	2	2
	2	2+4	6
2	2	4	4

Q.4 find total sum of all subarrays sum. { hoogle, FB }

A = [3 -1 4]
0 1 2

s	e	sa	sum
0	0	[3]	3
0	1	[3 -1]	2
0	2	[3 -1 4]	6
1	1	[-1]	-1
1	2	[-1 4]	3
2	2	[4]	4
			17

```
int   solve (int [ ] A ) {
      int n= A.length;
      int ts = 0;
      for (int s=0; s < n; s++ ) {
          int sum = 0;
          for (int e=s; e< n; e++) {
                  sum += A[e];
                  ts += sum;
          }
      }
      return ts;
}
```

TC: $O(n^2)$

SC: $O(1)$

Expected TC : O(n)

A = [3 -1 4]
 0 1 2

s	e	sa	sum
0	0	[3]	3
0	1	[3 -1]	2
0	2	[3 -1 4]	6
1	1	[-1]	-1
1	2	[-1 4]	3
2	2	[4]	4
			17

A[0]

A[0] + A[1]

A[0] + A[1] + A[2]

A[1]

A[1] + A[2]

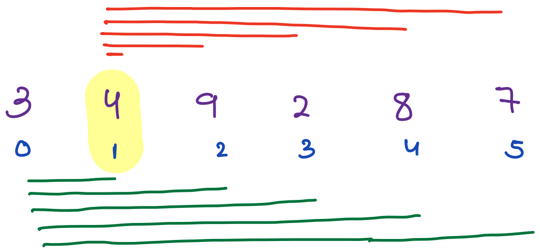
A[2]

3*A[0] + 4*A[1] + 3*A[2]

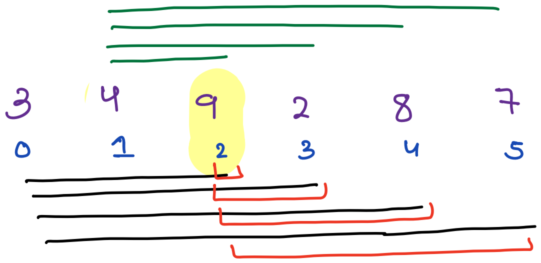
3*3 + 4*(-1) + 3*4 = 17

ith ele is coming in how many subarrays.

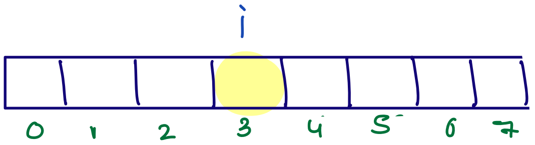
=) ith element is coming in how many subarrays.



count = 10



count = 12



a, b
=> b-a+1

sp	ep
0	3
1	4
2	5
3	6
	7

valid start points : 0 to i => i+1 sp

valid end points : i to n-1 => n-i ep

count of valid subarray = $(i+1) * (n-i)$

```
int  solve (int [] A) {

    int ans= 0;
    int n= A.length;

    for (int i=0; i< n; i++) {

        ans+ = A[i] * ((i+1) * (n-i));
    }

    return ans;
}
```

$$A = \begin{array}{ccc} [\,3 & -1 & 4\,] \\ 0 & 1 & 2 \end{array}$$

i	
0	3 * (1 * 3) = 9
	+
1	-1 * (2 * 2) = -4
	+
2	4 * (3 * 1) = 12
	17

Contribution technique

contest :

tuesday → 9pm – 11pm (contest)

11pm onwards contest discussions.

3 coding questions

(specially : 3 Arrays class)

DSA
- carry forward
- prefix sum
- subarray

Doubts

Product Array

$$A = \begin{array}{ccccc} [\,2 & 4 & 1 & 3 & 5\,] \\ 0 & 1 & 2 & 3 & 4 \end{array}$$

$$ans = \begin{array}{ccccc} [\,60 & 30 & 120 & 40 & 24\,] \\ 0 & 1 & 2 & 3 & 4 \end{array}$$

ans[i] = product of all ele except ith ele

$$A = \begin{array}{ccccc} [\,2 & 4 & 1 & 3 & 5\,] \\ 0 & 1 & 2 & 3 & 4 \end{array}$$

$prod = 2 * 4 * 1 * 3 * 5$

$$ans[i] = \left.\frac{prod}{A[i]}\right\}$$ manage for val = 0.

Note: don't use division.

$$A = \begin{array}{ccccc} [\,2 & 4 & 1 & 3 & 5\,] \\ 0 & 1 & 2 & 3 & 4 \end{array}$$

$$pm = [\,2 \quad 8 \quad 8 \quad 24 \quad 120\,]$$

$$sm = [\,120 \quad 60 \quad 15 \quad 15 \quad 5\,]$$

$$ans[i] = \underbrace{prod(0, i-1)}_{pm[i-1]} \times \underbrace{prod(i+1, n-1)}_{sm[i+1]}$$

$pm[i]$ => mult of all elements from 0 to i

$pm[i] = pm[i-1] * A[i]$

$sm[i]$ => mult of all elements from i to end

$sm[i] = sm[i+1] * A[i]$

Leader if it is > all elements on right side.

↙

A =	{	2	9	4	5	1	3	}
		0	1	2	3	4	5	

rmax = ~~3~~ ~~5~~ 9

count = ~~1~~ ~~2~~ 3

```
int rmax = A[n-1];
 int count = 1;

  for (int i = n-2; i>=0; i--) {
             if( A[i] > rmax) {
                    count++;
                    rmax = A[i];
             }
  }

  return count;
```

Bulb problem

```
int  solve ( int [] A )  {
    int  c= 0 ;
    for (int  i=0; i<n; i++)  {
        if ( c%2 == 1)  {
            a[i] = 1-a[i];
        }
        if (a[i] == 0)  {
            c++;
        }
    }

    return c;
}
```

0 ~~0~~ ~~1~~ 0 ~~0~~

1 0 1

c = ~~0~~ ~~1~~ ~~2~~ 3

i	
0	
1	
2	
3	
4	

7 5 9 16 2

maintain leftmin and try to calculate best profit.