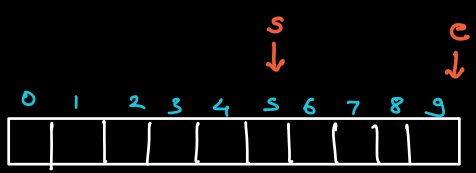
Google
Facebook
Amazon
Uber
MS
⋮

Q Given an array & a no. K.
Find sum of all sub-arrays of length K.

A: $\overset{0}{-3}, \overset{1}{4}, \overset{2}{-2}, \overset{3}{5}, \overset{4}{3}, \overset{5}{-2}, \overset{6}{8}, \overset{7}{2}, \overset{8}{-1}, \overset{9}{4}$ $\quad K = 5$

s	e	sum
0	4	7
1	5	8
2	6	12
3	7	16
4	8	10
5	9	11



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

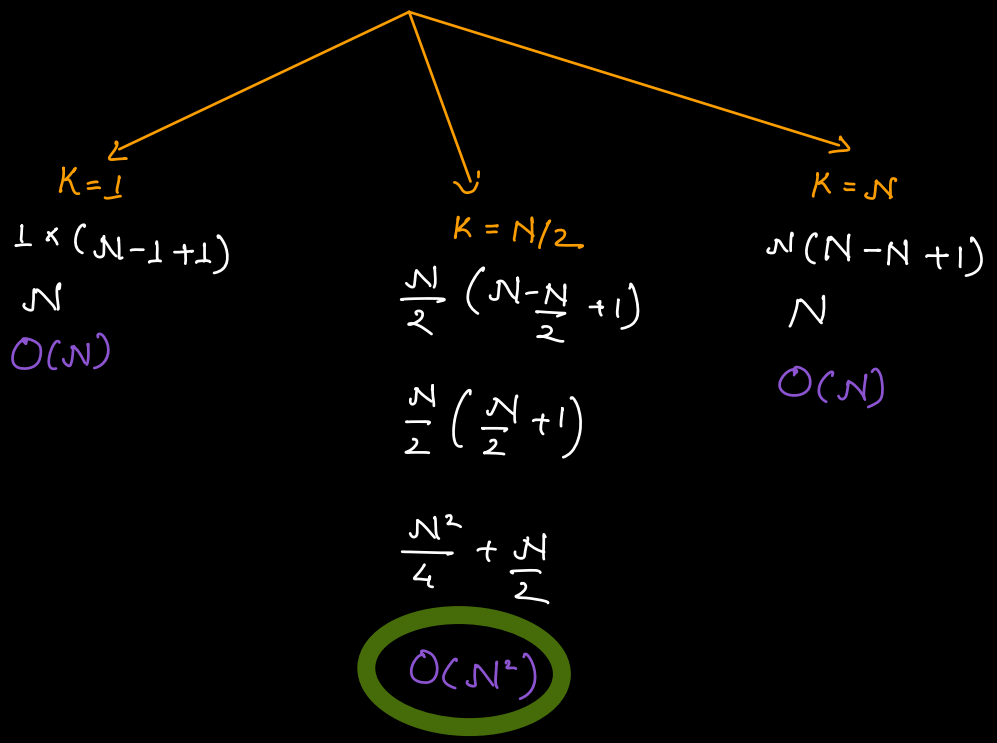
```
s = 0;
e = K-1;  // e : [K-1, N-1]
while (e < N) {          ---> # iterations : N-K+1
// iterate from s to e & calculate the sum
    Sum = 0;
    for (i = s; i <= e; i++) {
        Sum = sum + A[i];
    }                    # iterations : K
    Print (Sum);
    s++;
    e++;
}
```

Range sum s to e O(1) using PSI]

Total iterations = $K(N-K+1)$

$K\ (N-K+1)$

$K : [1, N]$



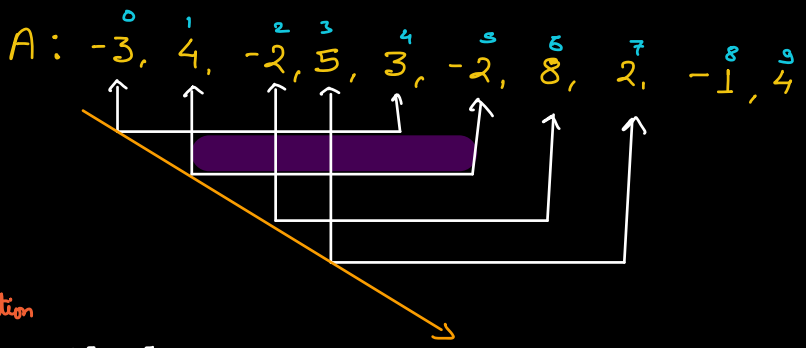
Using PS

iterations = N (Build PS) + $N-K+1$ (iterate over sub arrays of size K) $\longrightarrow 2N-K+1$

TC : $O(N)$

SC : $O(N)$

Sliding Window

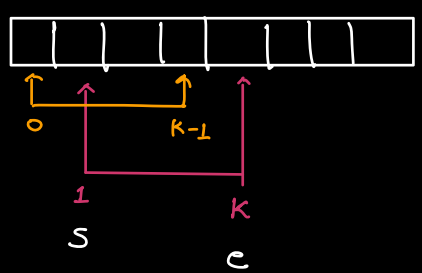


s	e	sum	
0	4	7	// iteration
1	5	8	$\Sigma[1,5] = \Sigma[0,4] - A[0] + A[5] \rightarrow 8$
2	6	12	$\Sigma[2,6] = \Sigma[1,5] - A[1] + A[6] \rightarrow 12$
3	7	16	$\Sigma[3,7] = \Sigma[2,6] - A[2] + A[7] \rightarrow 16$
4	8	10	$\Sigma[4,8] = \Sigma[3,7] - A[3] + A[8] \rightarrow 10$
5	9	11	$\Sigma[5,9] = \Sigma[4,8] - A[4] + A[9] \rightarrow 11$

$$\Sigma[i,j] = \Sigma[i-1, j-1] - A[i-1] + A[j]$$

```
// first window of length K
sum = 0;
for (i = 0; i < K; i++) {
    sum = sum + A[i];
}
Print(sum);
```

K iteration




```
s = 1;
e = K;  // e : [K, N-1]
while (e < N) {
    sum = sum - A[s-1] + A[e];
    Print(sum);
    e++;
    s++;
}
```

(N-K) iterations

Total iterations = K + N - K → N

TC: O(N)
SC: O(1)

Q Given an array & a no. K.
Find max sum of all sub-arrays of length K.

Q Given an array & a no. K.
Find min sum of all sub-arrays of length K.

Q Given an array & a no. K.
Find average of every window of size K. (Moving average)

Q Amazon Media.net

Given an array of size N & a number B.
Return the minimum no. of swaps required to bring all elements less than or equals to B together.

A : 1, 12, 10, 3, 14, 10, 5 B : 8
14 12, 10, [3, 1, 5,] 10 ⟶ 2

A : 5, 17, 100, 11 B : 20
5, 17, 11, 100

A : 7, 12, 19, 11, 6, 5, 9, 14 B : 8

A : 1, 12, 10, 3, 14, 10, 5, 12, 9, 14 B = 8

At the end all the numbers <= B will end up in a window of size same as count of these numbers (assume K)

In how min swaps can we bring all K number in a given window ? ⟹ No of elements greater than B in that window.

A : 1, 12, 10, 3, 14, 10, 5, 12, 9, 14 B = 8

2 2 2 2 2 2 2

1, 2, 10, 3, 14, 10 B = 8

1 1 2 2

Count = 1
min = 1

4, 6, 5, 9, 18, 10, 11, 7, 1 B = 8

3 3
5 4

13, 14, 15, 16, 17, 7, 6, 9 ..

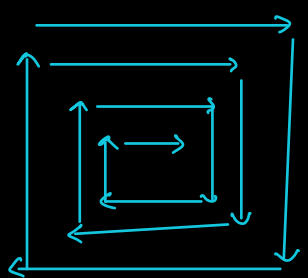
B = 8

Q. Given a matrix (N×N). Print it in spiral form.

Amazon
MS
DP world

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1, 2, 3, 4, 5, 10, 15, 20, 25, 24, 23, 22, 21, 16, 11, 6, 7, 8, 9, 14, 19, 18, 17, 12, 13



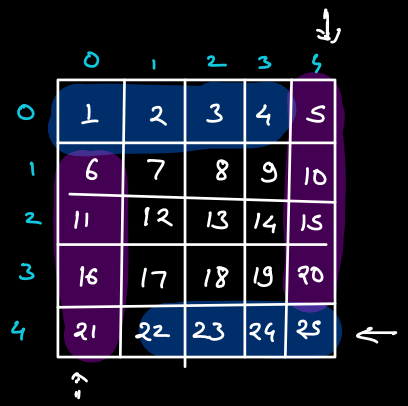
B = 6

3, 10, 15, 7, 12, 16, 9, 6, 5

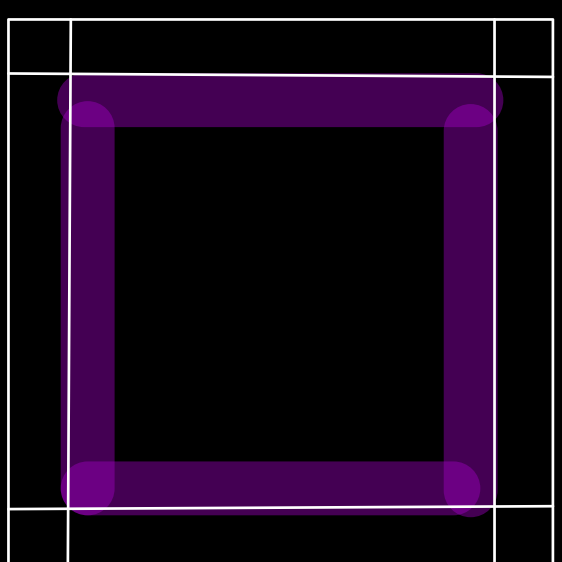
~~3~~, ~~10~~, 15, 7, 12, 16 C → 3 → 4 → ~~8~~ → 4

Print Boundry Elements

```
row = 0;
col = 0;

for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        col ++;
}

for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        row ++;
}

for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        col --;
}

for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        row --;
}
// row → 0,  col → 0
```



K	A[row][col]	col
1	A[0][0]	1
2	A[0][1]	2
3	A[0][2]	3
4	A[0][3]	4
5	→ Breaks	



N > 0
N > 1
N > 2
row <= N/2

```
row = 0;
col = 0;
While (N _ _ ) {
    for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        col ++;
    }

    for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        row ++;
    }

    for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        col --;
    }

    for (K=1 ; K<N;  K++) {
        Print ( A[row][col]);
        row --;
    }
    row ++;
    Col ++
```

TC : O(N²)
SC : O(1)
Extra

Edge case

```
    }   jy = jy - 2;
```

H.W : → ① While condition

② Edge case (The test case for which current algo will not work)

③ Solve same problem for N x M matrix