Q1) Given a N x N matrix, print the boundry elements in clockwise direction ?

A

N x N

5

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12
2	13	14	15	16	17	18
3	19	20	21	22	23	24
4	25	26	27	28	29	30
5	31	32	33	34	35	36

5 5 5

1 2 3 4 5 6 12 18 24 30 36 35 34 33 32 31 25 19 13 7

iter ≤ 6 - 1

itr :	1	2	3	4	5	6
row :	0	0	0	0	0	0
col :	0	1	2	3	4	5
	A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0 4]	

```
row=0,   col=0
for(iter=1; iter <= N-1; iter++){
    print (A[row] [col])
    col++
}
```

```
for(iter=1; iter <= N-1; iter++){
      print (A[row] [col])
        row ++
}
```

iter:	1	2	3	4	5	6
row:	0	1	2	3	4	5
col:	5	5	5	5	5	5
	↓	↓	↓	↓	↓	
	6	12	18	24	30	

```
for(iter=1; iter <= N-1; iter++){
      print (A[row] [col]
        col--
}
```

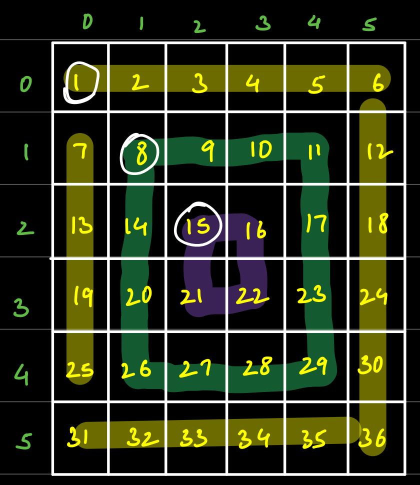
TC: O(N)

SC: O(1)

```
for(iter=1; iter <= N-1; iter++){
      print (A[row] [col]

        row - -
}
```

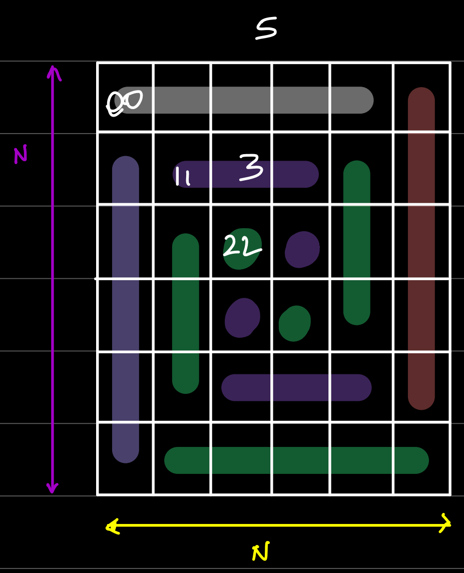
Q2) Print a matrix in spiral order?



	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12
2	13	14	15	16	17	18
3	19	20	21	22	23	24
4	25	26	27	28	29	30
5	31	32	33	34	35	36

Meta, Amazon, Google

Myntra



TC: $O(N^2)$

SC: $O(1)$

N = 8

starting point	N. of iteration	
0 0	N-1	→ 7
1 1	N-3	→ 5
2 2	N-5	→ 3
		→ 1
		-1

N > 0

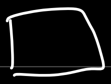
12 →

11 → 9 → 7 → 5 → 3 → 1 → -1

```
        row = 0    ,    col = 0
    while (N > 1) {
        for(iter = 1;  iter <= N-1;  iter++) {
            print (A[row] [col])
            col++
        }
        for(iter = 1;  iter <= N-1;  iter++) {
            print (A[row] [col])
            row ++
        }
        for(iter = 1;  iter <= N-1;  iter++) {
            print (A[row] [col]
            col--
        }
        for(iter = 1;  iter <= N-1;  iter++) {
            print (A[row] [col]
            row - -
        }
        row ++  , col++
        N - = 2
    }
    if (N == 1)  { print A[row] [col] }
```

$$\boxed{N \times M \text{ matrix}}$$

Todo



$$\begin{tikzcd}
5 \times 5 \arrow[r] \arrow[d] & \text{iter} \arrow[r] & 4 \arrow[d] \\
3 \times 3 \arrow[rr] \arrow[d] & & 2 \arrow[d] \\
1 \times 1 \arrow[rr] & & 0
\end{tikzcd}$$

d. rix ^{N}

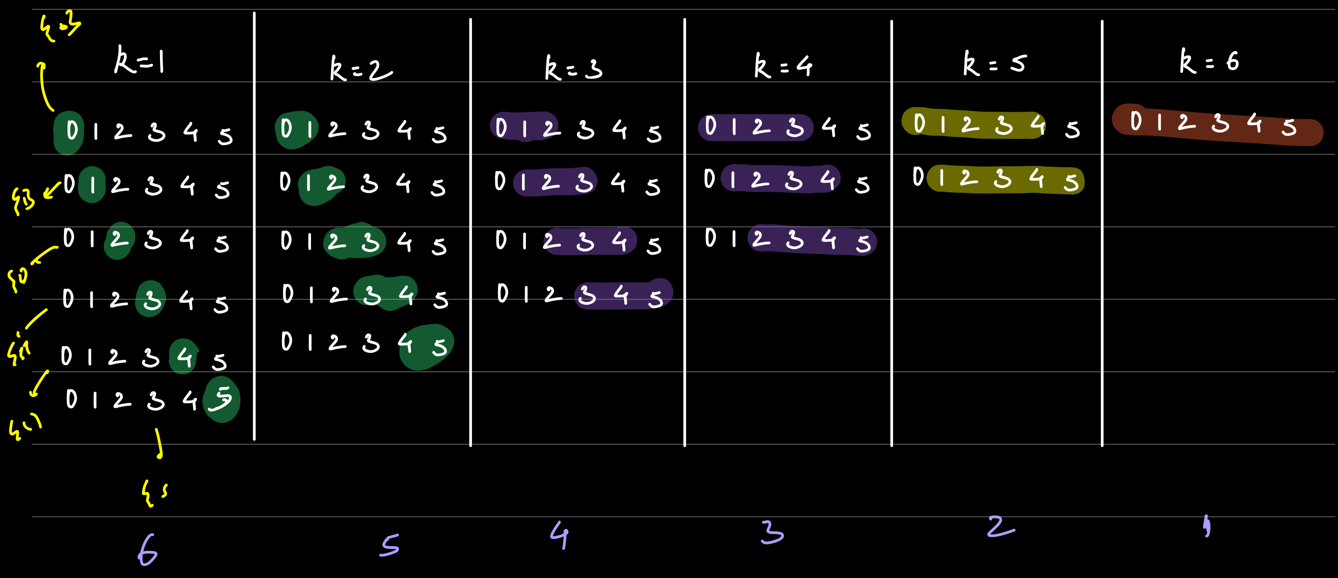
Q3) Given an array find number of subarrays of length k.

$\boxed{N-k+1}$

$N=6$ $\leftarrow$ $k=4$ $6-4+1=3$

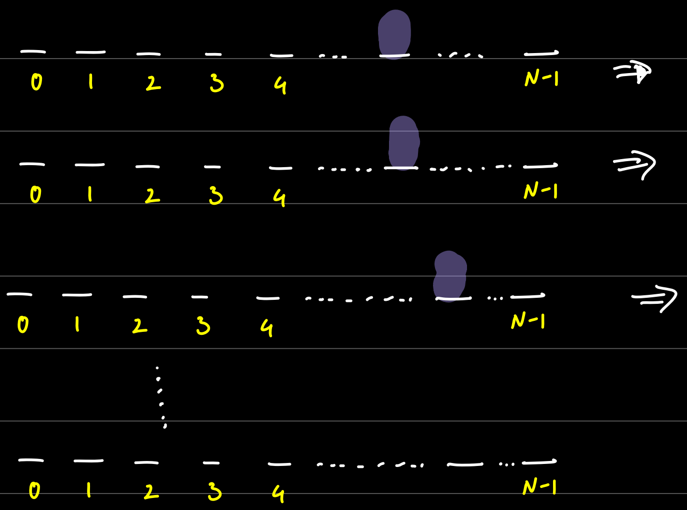
$N=6$ $k=2$ 5

Ex) $N=6$



N, k

s e



$[s \quad e] \Rightarrow k$

$e-s+1=k$

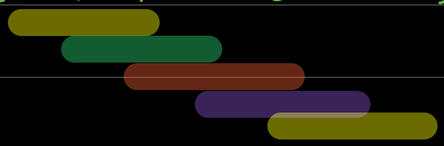
$e=k+s-1$

k

Q3) Given an array and subarray size, print the starting and ending index of all subarrays of size = k

$$\begin{array}{cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ A = \{ & 5 & 7 & 9 & 6 & 3 & 2 & 1 \} \end{array}$$

$k = 3$



$N - k + 1$

$= 7 - 3 + 1$

$= 5$

$N = 7$

$k = 3$

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

$N - k + 1$

```
for (s=0; s<=N-k; s++){
        e = k+s-1
        print(s, e)
}
```

Break 10:38 – 10:48

$[s \ e] \rightarrow k$

$e - s + 1 = k$

$\boxed{e = k + s - 1}$

Deque

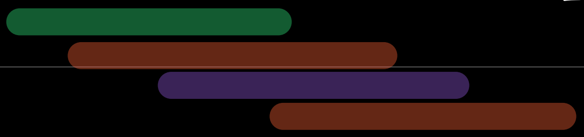
```
for (s=0; s<= N-k; s++) {
        e = k+s-1

        print(s, e)
}
```

s:	0	1	2	3	~~4~~
e:	3	4	5	6	

0 1 2 3 4 5 6

{1 3 5 9 7 8 2}   k = 4

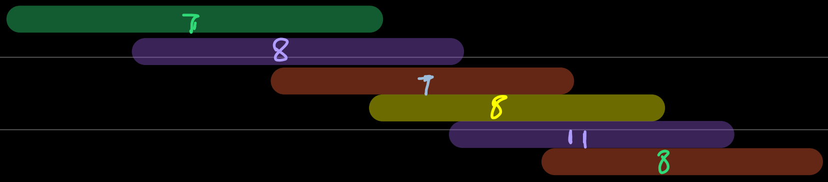


Q4) Find max sum subarray for all subarrays of size = k

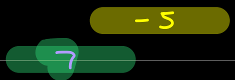
No. of subarrays : N-k+1 = 6

N = 9   k = 4

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



Ans : 11



[-3 -4 -1]

k = 2

```
maxsum = -∞

for (s=0; s<=N-k; s++) {
    e = k+s-1
    sum = 0
      # sum between [s e]
    for (i=s; i<=e; i++) {
            sum = sum + A[i]
    }
    maxsum = max(maxsum, sum)   }
}
```

$O((N-k) \times k)$

⇓

```
if (sum > maxsum) {
    max Sum = sum
}
```

k=1

k=2

$O(N^2)$

TC: $O((N-k) \times k)$

$k = N/2$

$$\left(N - \frac{N}{2}\right) \times \frac{N}{2} = \frac{N^2}{4}$$

$k=2$ $(N-2)\times 2 \rightarrow O(N)$

$O(N^2)$

TC: $O(N^2)$

TC: $O((N-k)\times k)$ $k=N/2$

$N/4 \rightarrow O(N^2)$

$$\left(N-\frac{N}{4}\right)\times\frac{N}{4}$$

$$\frac{3N}{4}\times\frac{N}{4} \Rightarrow \frac{3N^2}{16} \Rightarrow O(N^2)$$

```
// calculate pf array → O(N)
maxsum = -∞

for (s=0; s<= N-k; s++) {
    e= k+s-1
    sum = 0
      # sum between [s e]
    for (i=s; i<=e; i++) {
        sum = sum + A[i]
    }
    maxsum = max(maxsum, sum) }
}
```

∞

O(N) + O(N-k)

⇒ O(N-k)

↓

k ⇒ worst case

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

```
for (s=0; s<= N-k; s++) {
    e= k+s-1
    sum = 0

    pf[e] - pf[s-1]

    maxsum = max(maxsum, sum) }
}
```

$k = 4$

s	e	
0	3	$A[0] + A[1] + A[2] + A[3] = 7$
1	4	$+ A[4] - A[0] = 7-5+6 = 8$
2	5	$8 + A[5] - A[1] = 8+2-3$ $= 7$
3	6	$7 + A[6] - A[2] = 7-1-(-2)$ $= 8$
4	7	$8 + A[7] - A[3] = 8+4-1$ $= 11$
5	8	$11 + A[8] - A[4] = 11+3-6$ $= 8$

$$\boxed{s \quad e \qquad \text{perv} + A[e] - A[s-1]}$$

$[0 \quad \underline{e}] \Rightarrow k$

$e - 0 + 1 = k$

$e = k - 1$

$[1 \quad e] \Rightarrow k$

$e - 1 + 1 = k \qquad e = k$

k=4

```
    0    1    2    3    4    5    6    7    8
{   5    3   -2    1    6    2   -1    4    3   }
```

```
mexsum = -∞
sum = 0
for (i=0 ; i <= k-1 ; i++) {
|    sum = sum + A[i]
}
maxsum = max(maxsum, sum)
  s = 1
  e = k
 while ( TODO ) {
      sum = sum + A[e] - A[s-1]
      maxsum = max(maxsum, sum)
        s += 1
        e += 1
 }
 return   maxsum
```

Done!

Q5) Given an array of size N and a number B return min number of swaps required to bring all elements less than or equal to B together.

B=6

A: { 1 10 12 14 3 7 5 }