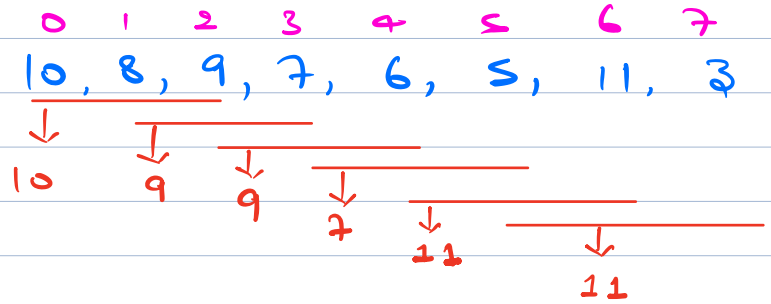
Sliding Window Maximum.

Google
FB(Meta)
Amazon
Walmart
Adobe

Given an array of size N.

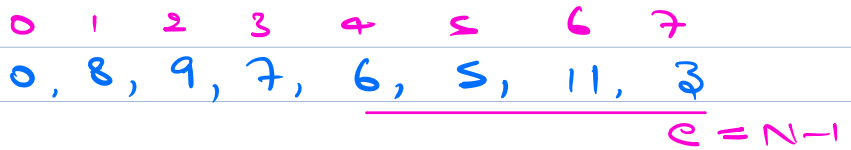
Return an array containing the max value of every sliding window of size K ← Input

Eg: A: 10, 8, 9, 7, 6, 5, 11, 3 K=3



O/P ⇒ [10, 9, 9, 7, 11, 11]

1) Brute force



1st sliding window s = 0, e = K-1

vector / arraylist ans;
s = 0, e = K-1

while (e <= N-1) {

window max = INT_MIN

```
for (i = s; i <= e; i++) {
    windowMax = max(windowMax, A[i]);
}
ans.push_back(windowMax);
s++;
e++;
}
return ans;
```

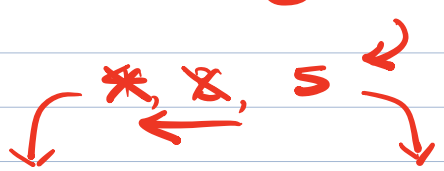
K = 4

Eg 1: 5 4 3 2 8 7 9 6 12 3

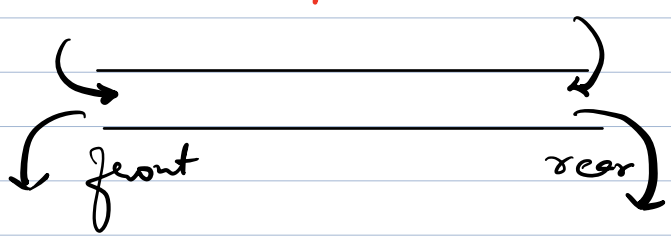
max

5, 4, 3, 2

4 3 5 2

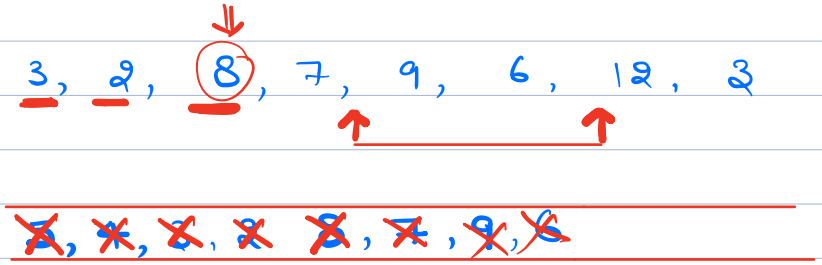


Deque



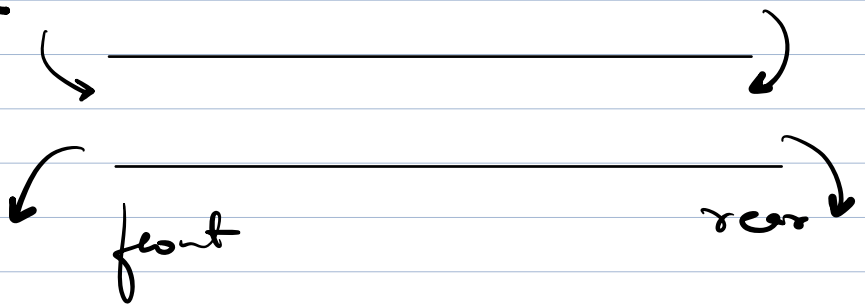
K = 4

Eg 1: 5 4 3 2 8 7 9 6 12 3



Ans: 5, 8, 8, 9, 9

Deque



1) push_front(x)
2) push_back(x)
3) remove_front()
4) remove_back()
5) front()
6) rear()
7) size()
8) isEmpty()

Code

```
Deque<int> dq = new Deque<int>();

for (i = 0; i < K; i++) {
    while (!dq.isEmpty() && dq.rear() < A[i]) {
        dq.remove_back();
    }
```

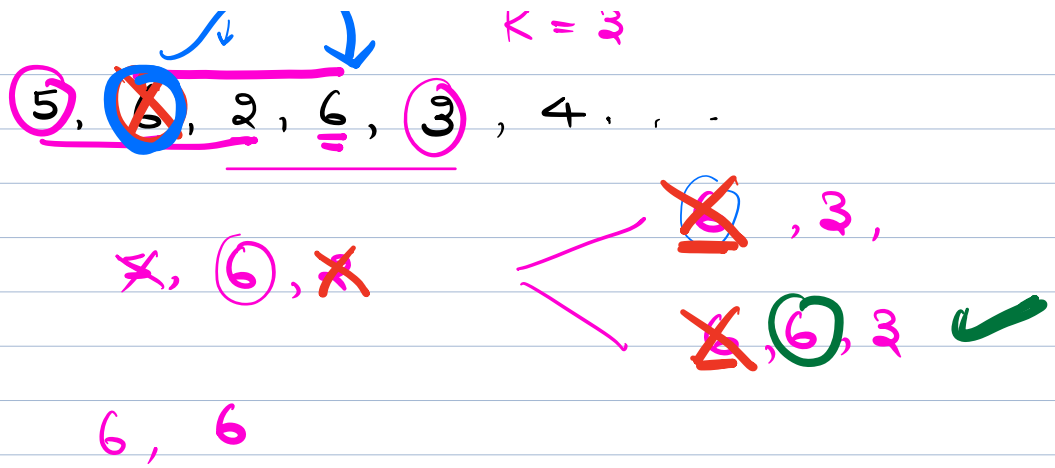
```
        dq.push_back(A[i]);
    }

vector<int> ans = new vector<int>;
Arraylist  "     "     "     "     "     "
ans.add(dq.front);
   s = 1;    e = K;
   while (e <= N-1) {
        if (A[s-1] == dq.front()) {
                dq.remove_front();
        }

        while (!dq.isEmpty() && dq.rear < A[e]) {
                dq.remove_back();
        }
        dq.push_back(A[e]);
        ans.add(dq.front());
        s++;
        e++;
   }
```

T.C. = O(N)

S.C. = O(K)



Q Google Meta Scaler

Given an encrypted string.
Find the Kth char after decrypty the string.

Eg: S: ab2c3 K = 5

ab ab c 3

ababcababcababc

Ans

2) x2y3

xxy3

xxyxxyxxy

xxxyxyxy

3) abc100de500e7004y1000

4) a b 2 c 3 → a b a b c 3

0 1 2 3 4 — a b a b c

5 6 7 8 9 — a b a b c

10 11 12 13 14 — a b a b c

$l = 5$

$K = 8$

5 → 0	10 → 0
6 → 1	11 → 1
7 → 2	12 → 2
8 → 3	13 → 4
9 → 4	14 → 5

$$ch[K] = ch[K \,\%\, l]$$

0 1 2 3 4

a b a b c

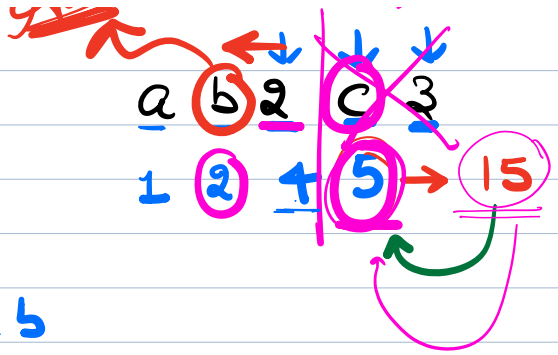
a b 2 c 3

1 2 4 5 → 15

$K = 9$

C1: Find 9th index in string of length 15.

C2: Find $9 \,\%\, 5$ index in string of length 5 → 4



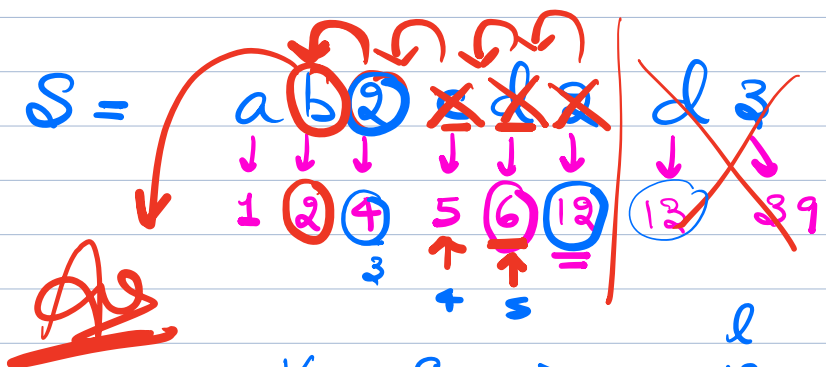
K = 8

a b a b

8 .\. 5 = 3

3 .\. 2 = 1

S = a b 2 c d 3



K = 9

K = 9 → l 12

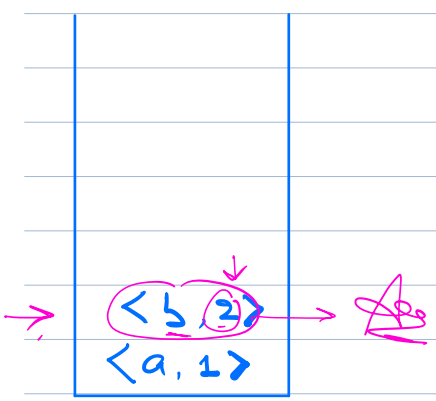
9 .\. 6 → 6

3

3 → 4

3 .\. 2 → 2

1



Stack

Try Code (H.W.)