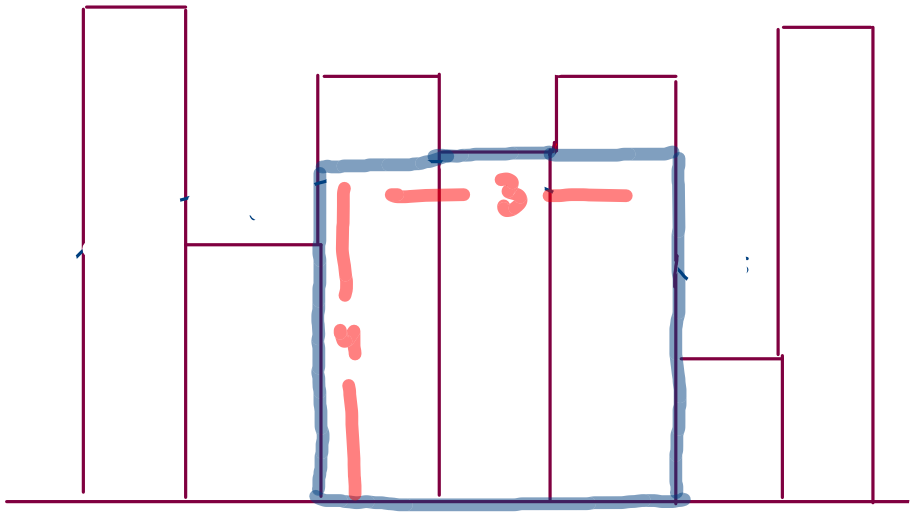
gap = 6

ele = gap + 1

w = ele - 2

w = gap - 1



idx - based

nsel or + nsel (idx based)

nser (doesn't exist)

↳ 7

nsel (doesn't exist)

↳ -1

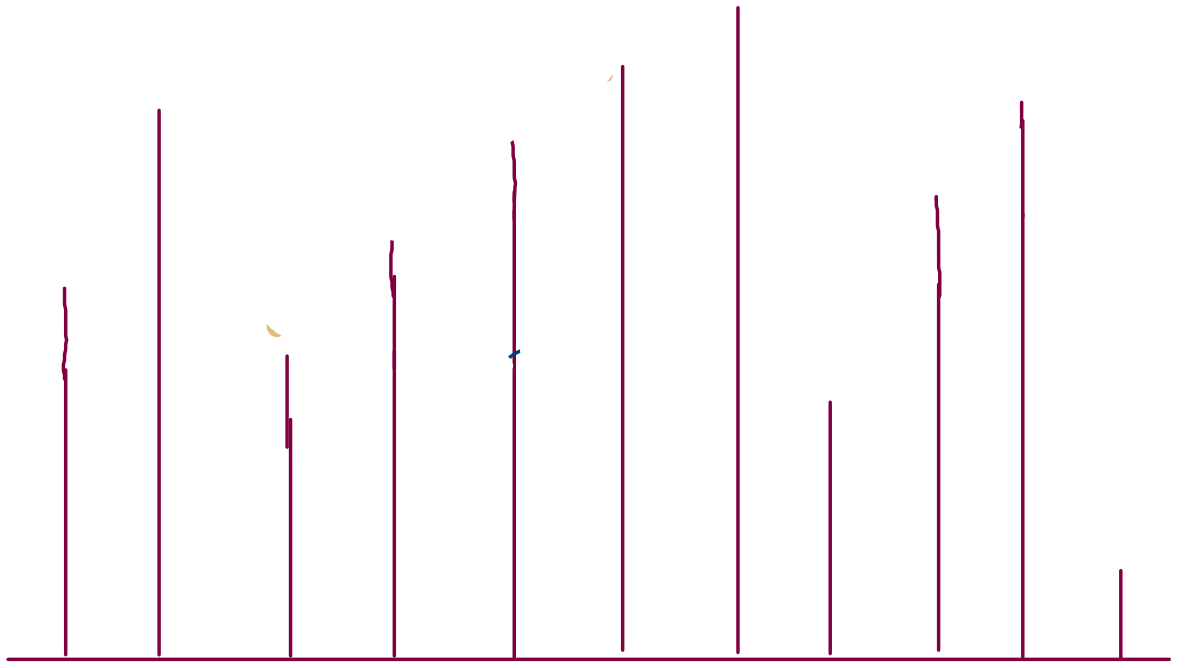
	6_0	2_2	5_2	4_3	5_4	1_5	6_6
nsel	-1	-1	1	1	3	-1	5
nser	1	5	3	5	5	7	7
width	1	5	2	3	1	7	1
Area	6	10	5	12	5	7	6

w = nsr[i] - nsl[i] - 1

sliding window maximum

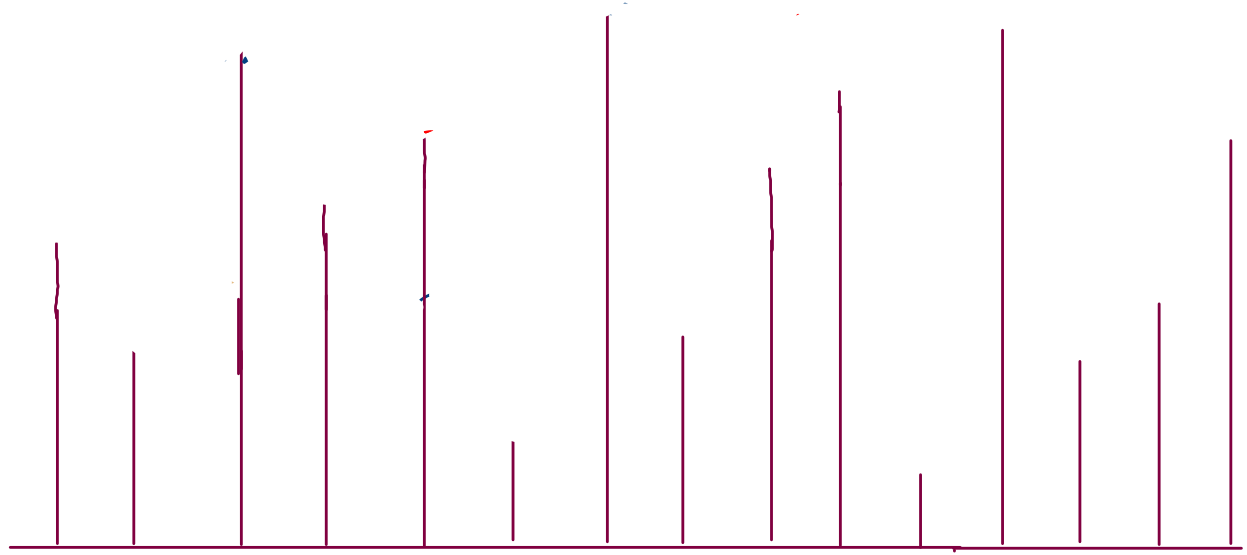
k = 4

8	14	6	9	12	15	17	5	11	13	2



14 14 15 17 17 17 17 13

```java
public static void slidingWindowMax(int[]arr,int k) {
    int[]ngr = nextGreaterRight(arr);
    int n = arr.length;

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

        //i+k-1 is i'th window's ending point
        while(ngr[j] < i+k) {
            //next greater lies within window
            j = ngr[j];
        }

        System.out.println(arr[j]);
    }
}
```

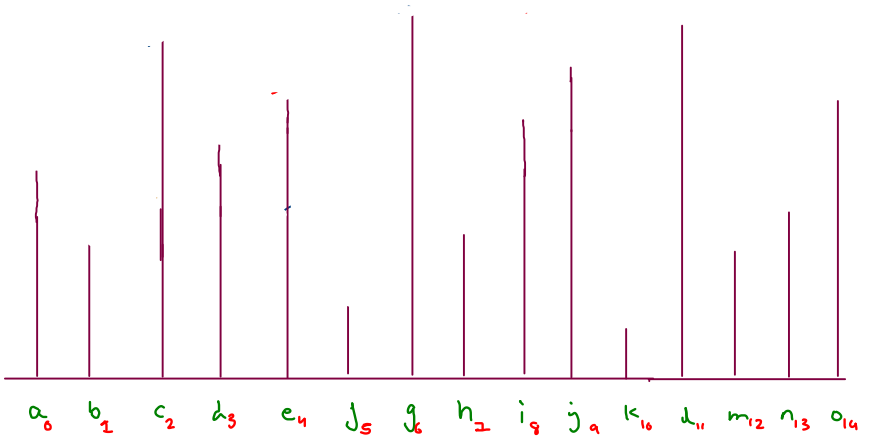


$k = 6$

	a_0	b_1	c_2	d_3	e_4	f_5	g_6	h_7	i_8	j_9	k_{10}	l_{11}	m_{12}	n_{13}	o_{14}
ngr	2	2	6	4	6	6	15	8	9	11	11	15	13	14	15
ans	c	g	g	g	g	g	g	l	l	l					

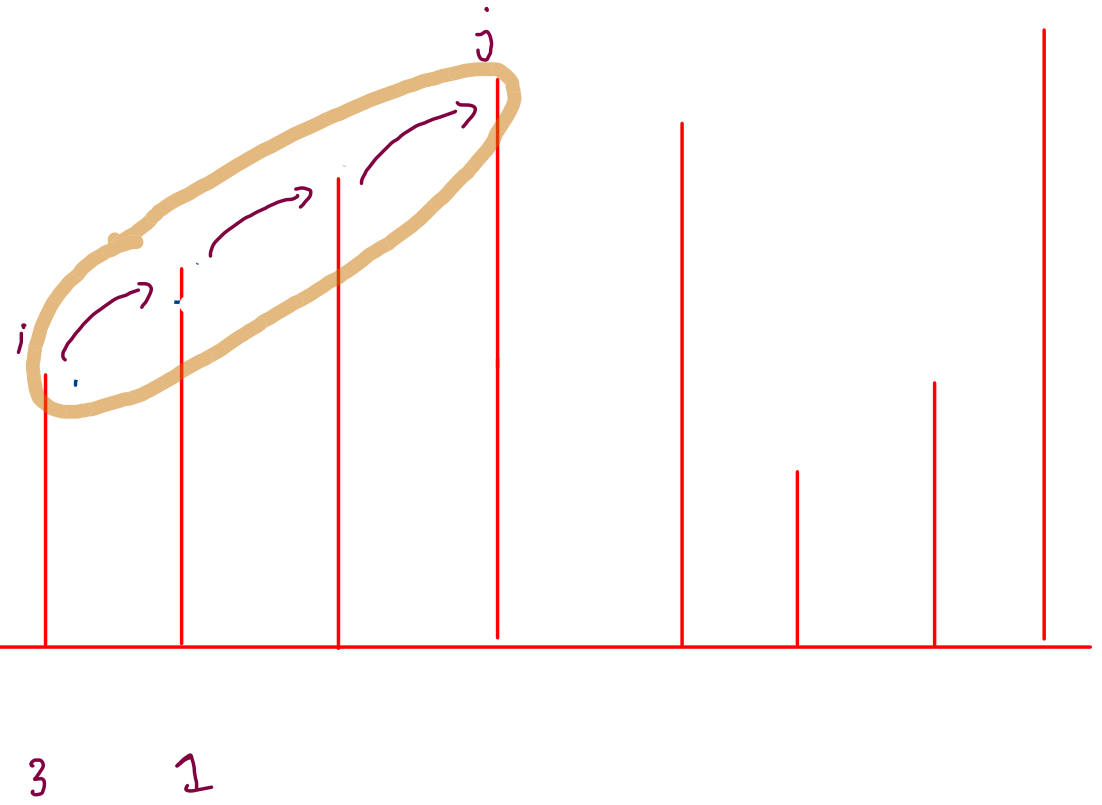
K = 6

```
for(int i=0; i <= n-k;i++) {
    if(j < i) {
        j = i;
    }

    //i+k-1 is i'th window's ending point
    while(ngr[j] < i+k) {
        //next greater lies within window
        j = ngr[j];
    }

    System.out.println(arr[j]);
}
```

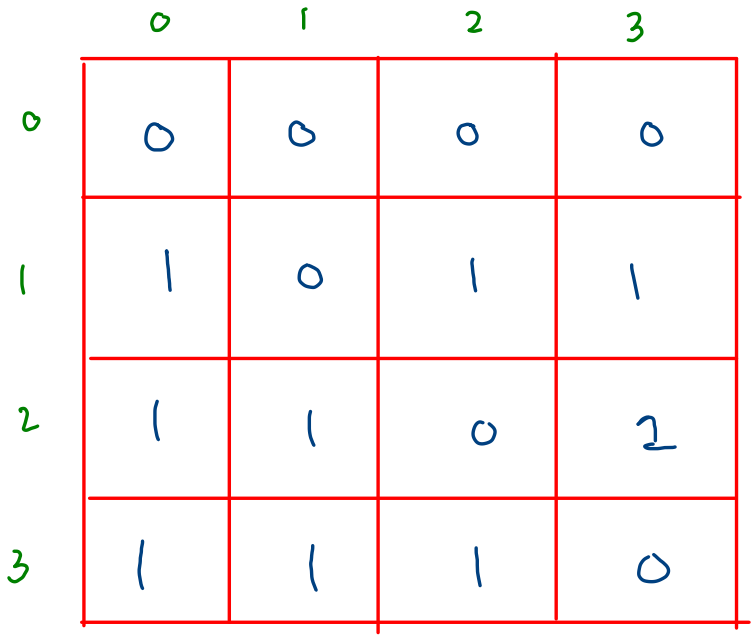


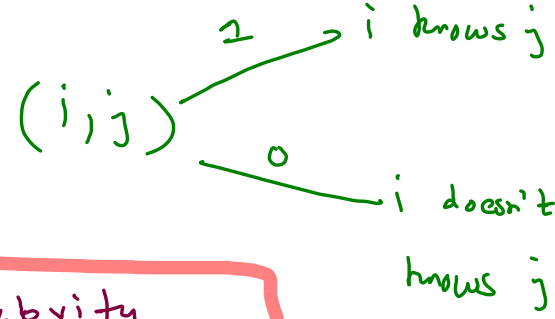
ngr	2	2	6	4	6	6	15	8	9	11	11	15	13	14	15

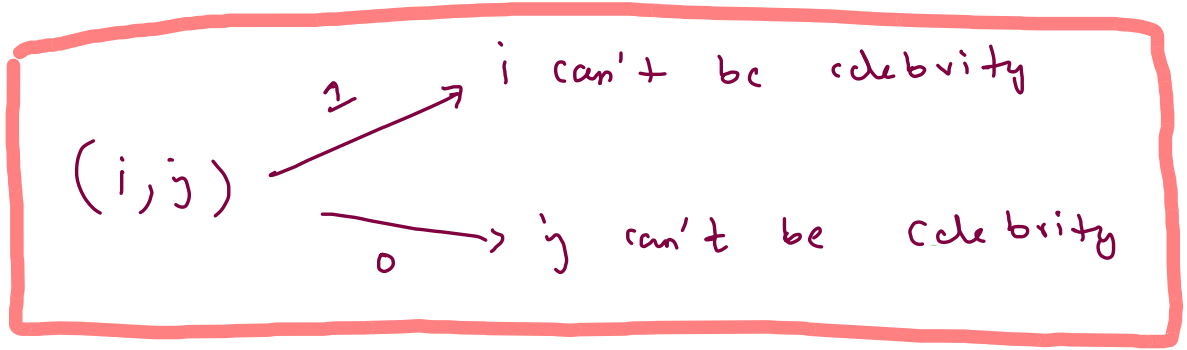
```java
for(int i=0; i <= n-k;i++) {
    if(j < i) {
        j = i;
    }

    //i+k-1 is i'th window's ending point
    while(ngr[j] < i+k) {
        //next greater lies within window
        j = ngr[j];
    }

    System.out.println(arr[j]);
}
```



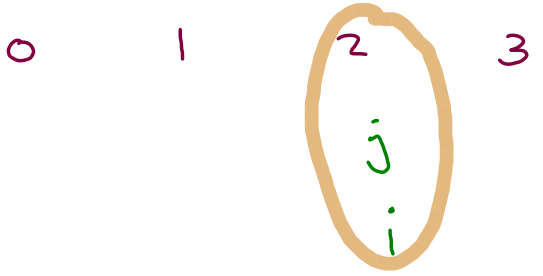
```
4
0000
1011
1101
1110
```







	0	1	2	3
0	0	1	1	0
1	1	0	1	1
2	0	0	0	0
3	1	1	1	0



(0,3) = 0 3 X

(0,2) = 1 0 X

(1,2) = 1 1 X