

(A)

Rank data structure

1	2	3	4	5	6	7
5	1	6	2	3	4	7

rank[i]

(rank of element with index i)

1	2	3	4	5	6	7
2	4	5	6	1	3	7

index[i]

(index of element ranked i)

1	2	3	4	5	6	7
10	1	15	1	3	5	20

sequence[i]

(element numbered i)

(B)

sequence array

S=3

1	2	3	4	5	6	7
10	1	15	1	3	5	20

Marker data structure $mm_S(w)=2^{nd}$ not marked

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=1

1	2	3	4	5	6	7
10	1	15	1	3	5	20

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=2

1	2	3	4	5	6	7
10	1	15	1	3	5	20

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=3

1	2	3	4	5	6	7
10	1	15	1	3	5	20

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=4

1	2	3	4	5	6	7
10	1	15	1	3	5	20

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=5

S=5

1	2	3	4	5	6	7
10	1	15	1	3	5	20

 $mm_S(w)=3^{rd}$ not marked

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=1

1	2	3	4	5	6	7
10	1	15	1	3	5	20

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=2

1	2	3	4	5	6	7
10	1	15	1	3	5	20

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=3

S=7

1	2	3	4	5	6	7
10	1	15	1	3	5	20

 $mm_S(w)=4^{th}$ not marked

1	2	3	4	5	6	7
1	1	3	5	10	15	20

w=1

(C)

Moving medians algorithm at window size S

1. **Cross out** all elements that are not in sliding window $w=1$.
2. **Set median pointer** equal to the remaining element numbered $mm_S(w)=(S+1)/2$ (blue arrow).

3. **FOR** $w=2$ **TO** $N-S+1$

– **Update** the Marker structure:

☒ **insert** the element that has entered it

☒ **cross out** the element that has left it.

– **Move** the median pointer according to the following rules:

☞ **Jump up** to the next remaining element, **if** the rank of the element that has left the window is smaller or equal to the pointer value, and the rank of the element that has entered the window is larger than the pointer value.

☞ **Jump down** to the previous remaining element, **if** the rank of the element that has left the window is smaller than or equal to the pointer value, and the rank of the element that has entered the window is larger than the pointer value.

☞ **Remain** where placed, **otherwise**.