Red indicates the unsorted segment of the array.

SelectionSort Average and Worst O(n*n/2)

4 5 2 3 1

1 5 2 3 4

1 2 5 3 4

1 2 3 5 4

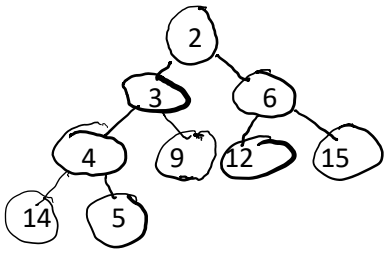
1 2 3 4 5

InsertionSort Average O(n*n/4) and Worst O(n*n/2)

4 5 2 3 1
4 5 2 3 1
2 4 5 3 1
2 3 4 5 1
1 2 3 4 5

Heap-sort

9 6 5 14 4 12 15 3 2



9 6 5 14 4 12 15 3 2

parent = i/2; left = i*2; right = left+1

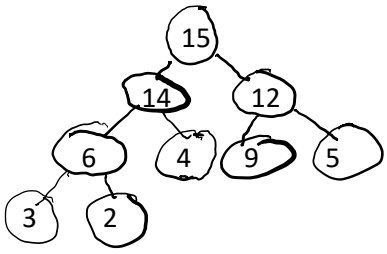
After inserting into the Priority Queue

	2	3	6	4	9	12	15	14	5	
0	1	2	3	4	5	6	7	8	9	10

	6	9	12	14	15					
0	1	2	3	4	5	6	7	8	9	10

2 3 4 5

We need a max-heap to sort into ascending order in the Heap-Sort.



9 6 5 14 4 12 15 3 2

parent = i/2 - 1; left = i*2+1; right = left+1

9	6	5	14	4	12	15	3	2		
0	1	2	3	4	5	6	7	8	9	10

After buildHeap

15	14	12	6	4	9	5	3	2		
0	1	2	3	4	5	6	7	8	9	10

Now remove and swap to end of the array to Sort

9	6	5	3	4	2	12	14	15		
0	1	2	3	4	5	6	7	8	9	10

The max-heap is in indices 0-5 and the sorted elements are in indices 6-8 after first 3 times through the loop. Note that the sorted segment increases and the heap (unsorted) segment decreases until the whole array is sorted.

ceiling(log (n+1)) $\leq$ h $\leq$ n

{} => { {} }
{1} => { {} {1} }
{1,2} => { {} {1} {2} {1,2} }
{1,2,3} => { {} {1} {2} {1,2} {3} {1,3} {2,3} {1,2,3} }