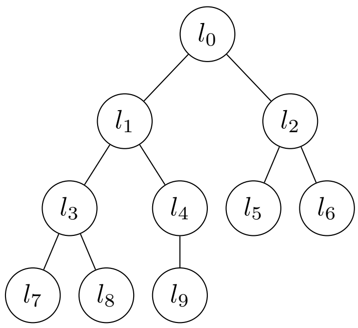
Sorting

$$l = (l_0, l_1, l_2, \dots, l_{n-1})$$

$$l_0 \leq l_1 \leq \cdots \leq l_{n-1}$$

On a tree



Heaps

Heap: complete binary tree.

Max heap: each parent bigger than children.

Min heap: each parent smaller than children.

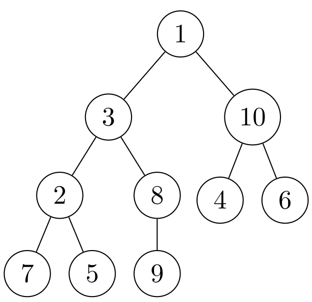
Check for min/max heap in $(n-1)$ comparisons.

To min or max heap

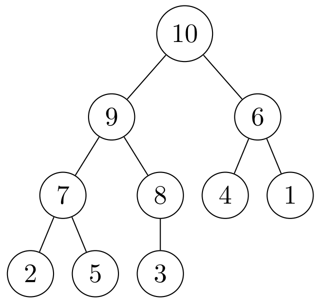
1. Start with last node, moving backwards.
2. Compare node to children, swap if needed.
3. Swap parent down tree until we have a heap.

Example heap

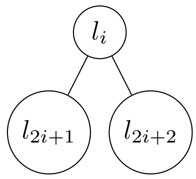
Sort ascending $\rightarrow$ use max heap.



Last five nodes have no children. Sixth-last has one child, is smaller so swap. Now have a heap from sixth-last. Same for seventh-last: swap 2 for 5. Third node is a heap. Second node swaps 9 for 3, and filters down swapping 3 for 8. Finally, the root is swapped with 1 and then 6.



As an array



l_0	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
1	3	10	2	8	4	6	7	5	9
1	3	10	2	9	4	6	7	5	8
1	3	10	7	9	4	6	2	5	8
1	9	10	7	3	4	6	2	5	8
1	9	10	7	8	4	6	2	5	3
10	9	1	7	8	4	6	2	5	3
10	9	6	7	8	4	1	2	5	3

Heap Sort

1. Convert complete binary tree to heap.
2. Swap root for last child, ignore last child.
3. Repeat $n-1$ times.

l_0	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
10	9	6	7	8	4	1	2	5	3
3	9	6	7	8	4	1	2	5	10
9	8	6	7	3	4	1	2	5	10
5	8	6	7	3	4	1	2	9	10
8	7	6	5	3	4	1	2	9	10
2	7	6	5	3	4	1	8	9	10

Comparisons

To heap: $O(n \log n)$

Replace root: $O(\log n)$ but $O(n)$ times.

Check heap: $O(n)$