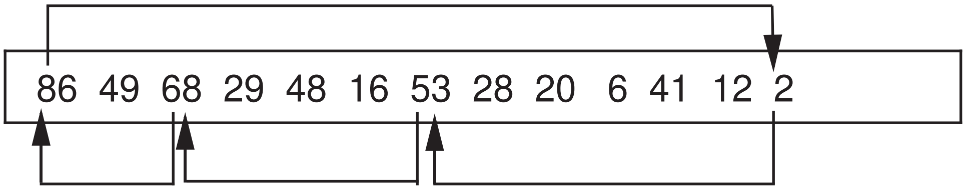
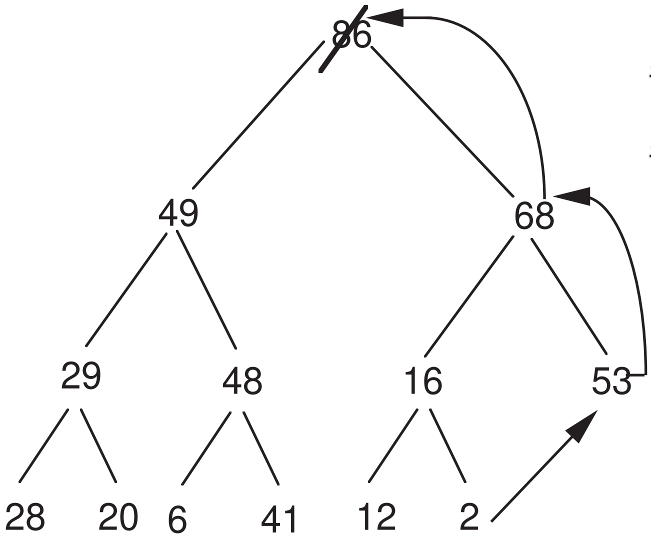
Examples

Deletemax in a Heap





Comparisons: 2 log n

move: log n

Deletemax $\Longrightarrow$ delete 86

Resulting array

68	49	53	29	48	16	2	28	20	6	41	12	86

Examples

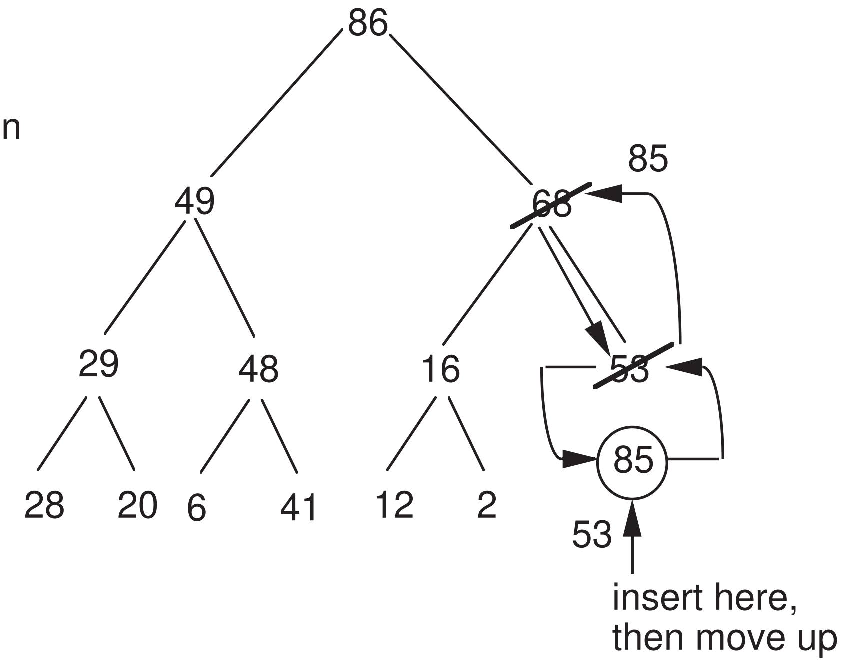
Insert 85 in Heap

86 49 68 29 48 16 53 28 20 6 41 12 2

85

comparisons: log n

moves: log n



Resulting Array

86 49 85 29 48 16 68 28 20 6 41 12 2 53

Heapsort

```
HEAPSORT.
   Create Heap:
   for i:= n step -1 to 2 do
       Deletemax and store max in A[i];
```

Cost: $2n + 2(n \log n - n) \pm O(n) \approx 2n \log n$.

Comments:

- Good: $2n \log n$ comparisons
- Good: in palce (no extra space used).
- Bad: indexing is bad (doubling)
- Bad: makes about $n \log n$ moves

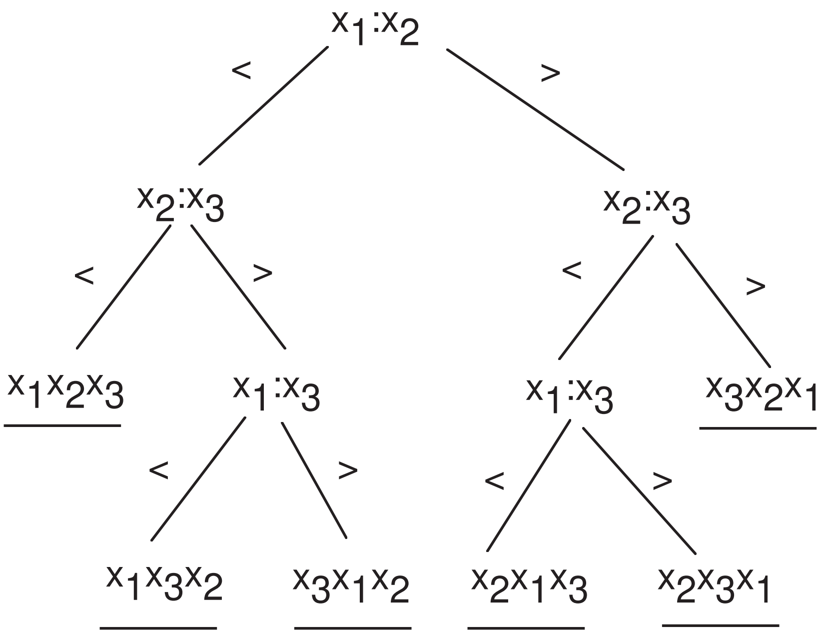
Lower Bound

- We will use the "decision-tree" model to prove a lower bound for sorting.

<u>Algorithm</u>	$\Rightarrow$	<u>decision tree</u>
a comparison		a node
an outcome		an edge
output		leaf (n!)
worst case		height of decision-tree
		$\geq \log(n!)$
		$\approx n \log n$

* If k is the height of the tree, $2^k \geq \#$ leaves $\geq n! \Rightarrow k \geq \log(n!) \approx n \log n$.

Lower bound for sorting algorithms based on comparisons:



- We must determine which of the $n!$ possible orders the file was in.

- Each comparison cuts # of possible orders in half (at best).

- Tree has $n!$ leaves (this many possible orders). So # comparisons $\geq \log n! \approx O(n \log n)$. (Also holds for average case.)