# *CS202 - Algorithm Analysis*

# Tree Algorithms - Module 2

Aravind Mohan

Allegheny College

March 21, 2023

# Discussion Based On ...

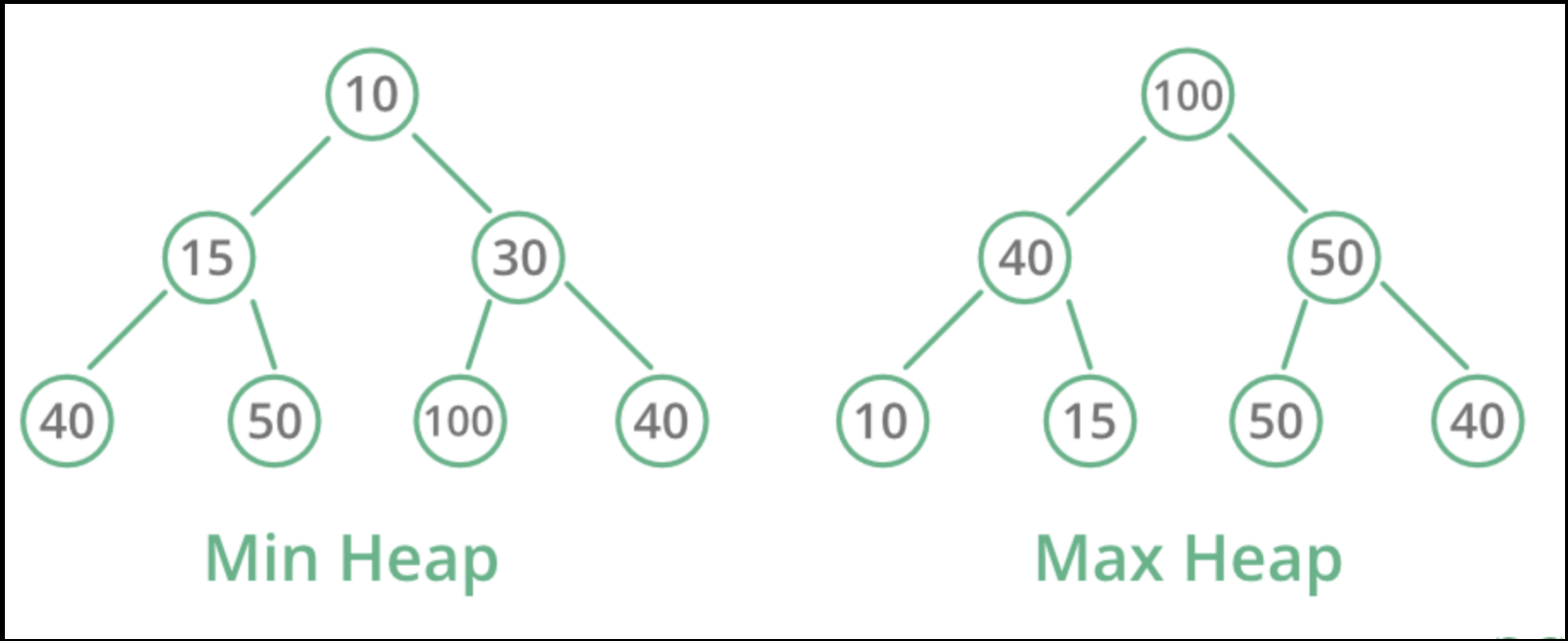
**Sedgewick 2.4 Heap Sort**

# What is a Binary Heap?

- Each node has atmost two children.
- Complete binary tree or atmost complete binary tree qualified as binary heap.
- Node with no children is also qualified as heap.
- Left skewed or right skewed tree is not a heap.
- There are two types of heap, namely:

  Max heap and Min heap.

# What is a Binary Heap?



# Binary Heap Properties

- Binary Heap has two main properties:
  1. Order property
  2. Shape property

# Binary Heap Properties

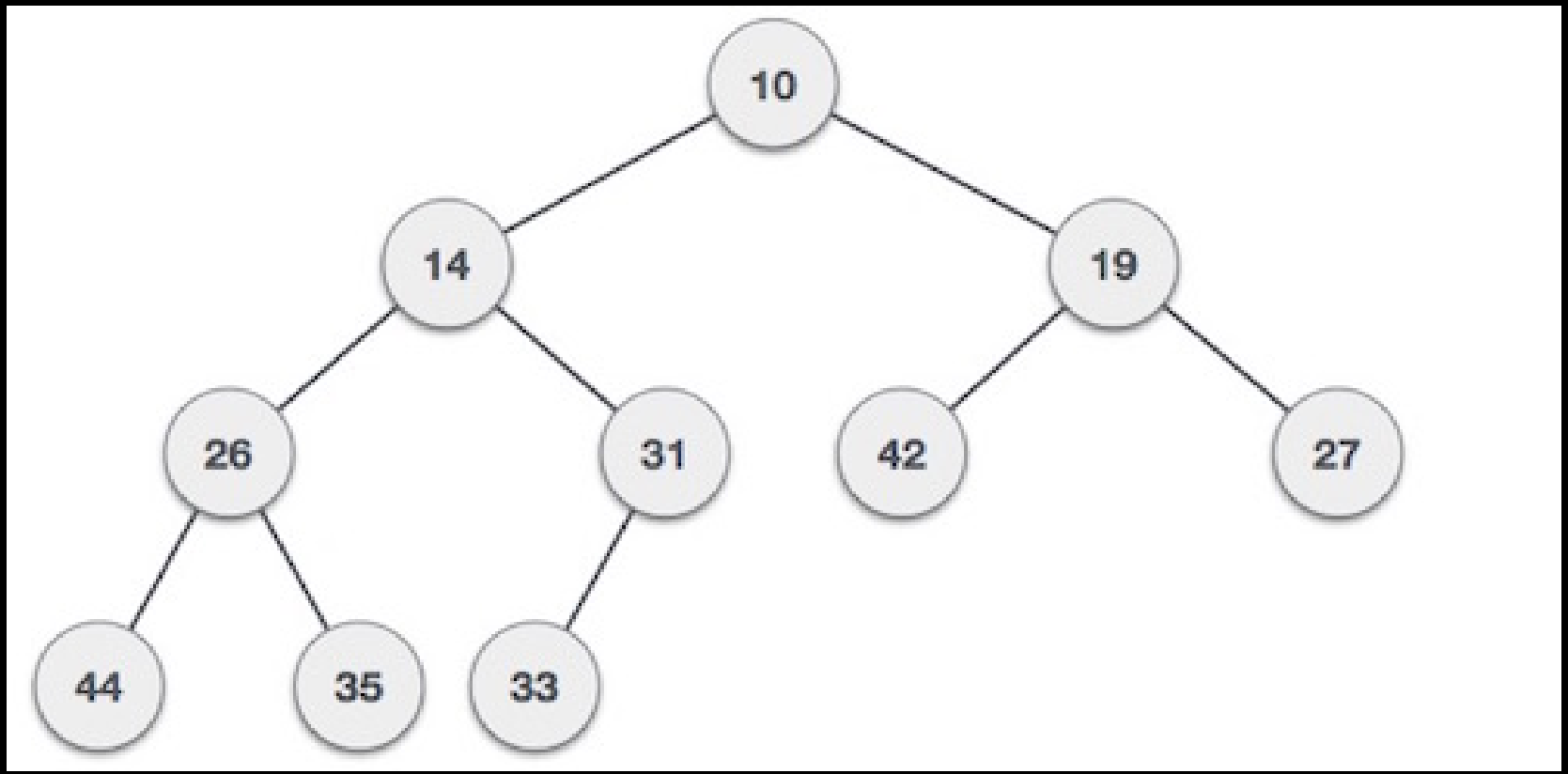
- **Order** property: The value in node $n$ is $\geq$ the values in its children, for every node $n$ (MAX heap).
- How about MIN heap?

# Binary Heap Properties

**Shape Property:**

- All leaves are either at depth $d$ or $d-1$ for some $d$
- All of the leaves at depth $d-1$ are to the right of the leaves at depth $d$
- And the following:
  1. There is at most 1 node with just 1 child $v$.
  2. $v$ is the left child of its parent.
  3. $v$ is the rightmost leaf at depth $d$.

# Binary Heap Example



# Heap Sort

- Phase 1: convert the array into an $n$-element heap
- Phase 2: repeatedly remove maximum element from the heap, and place that element in its proper position in the array
  - swap element at $0$th position with element at $(n-1)$th position and then "reheapify" considering only the first $n-1$ elements
  - repeat this process until heap size is reduced to $1$ (minimum element remains, at $0$th position)

# Heap Sort: Phase 1 - build the heap

```
for i = 1 to n − 1 do
  insert element s[i] into the heap consisting
   of the elements s[0]...s[i − 1]   [heapify]
```

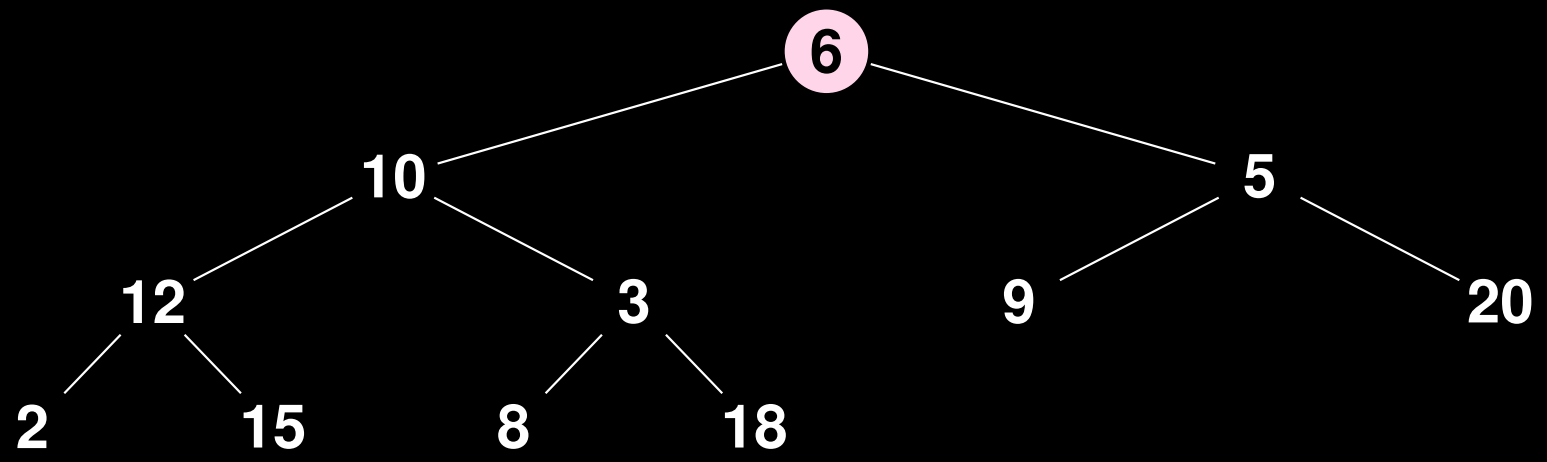
Once the heap is built, $s[0]$ will contain the maximum element

# Heap Sort Example

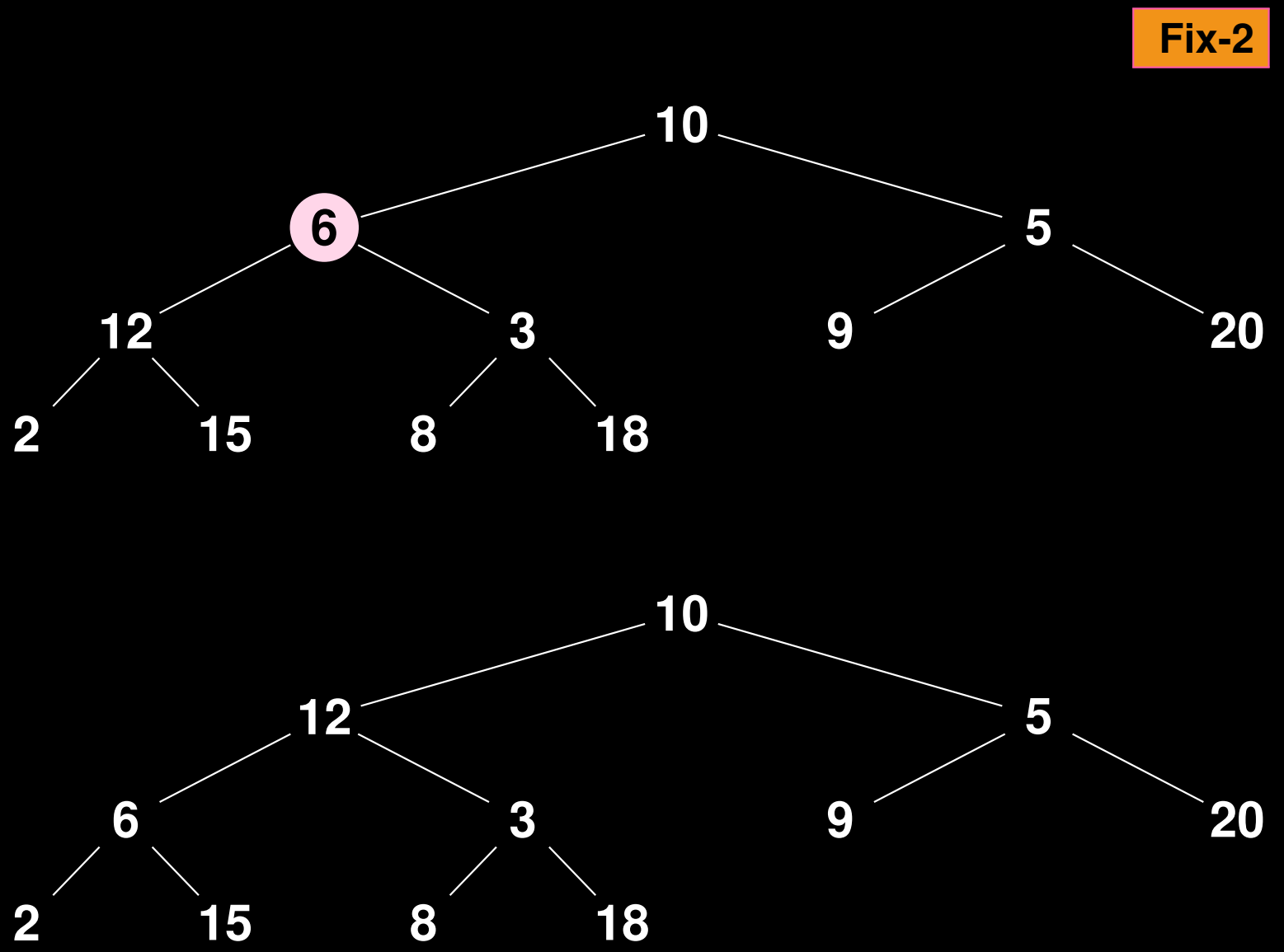
| 6 | 10 | 5 | 12 | 3 | 9 | 20 | 2 | 15 | 8 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|

**Step 1: Initial Heapify**

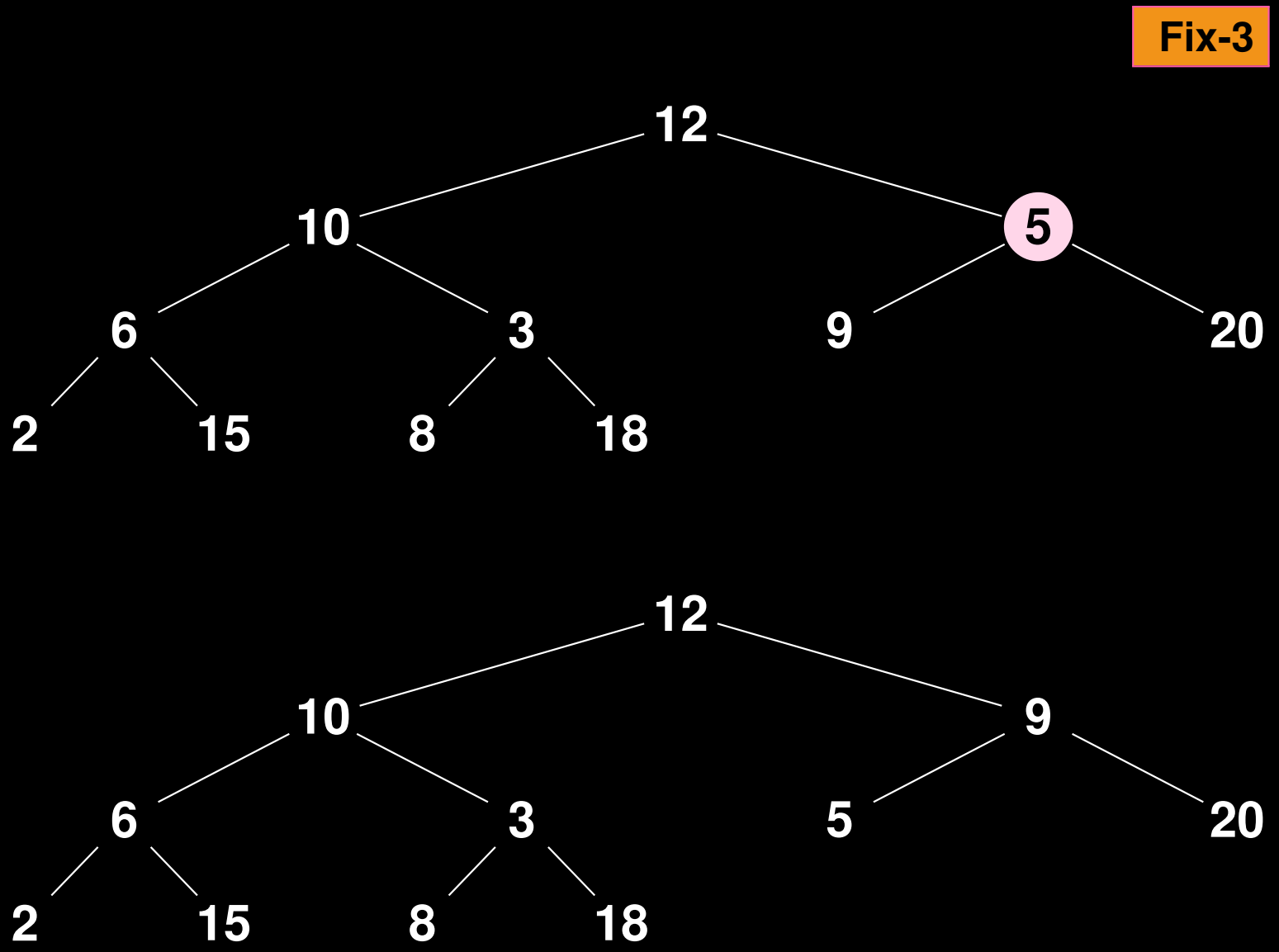
**Fix-1**



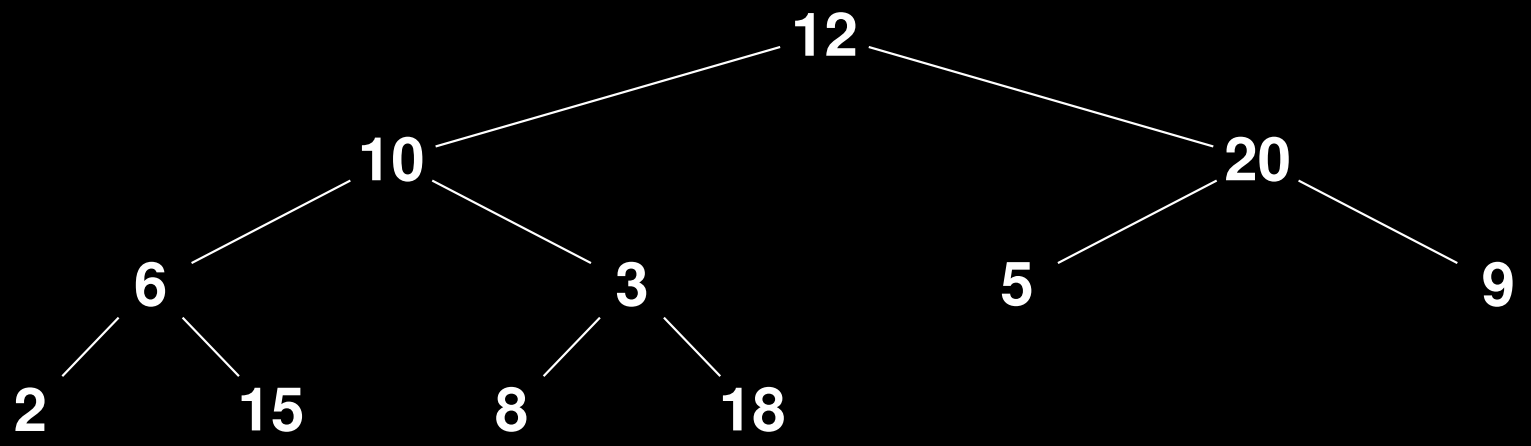
# Heap Sort Example

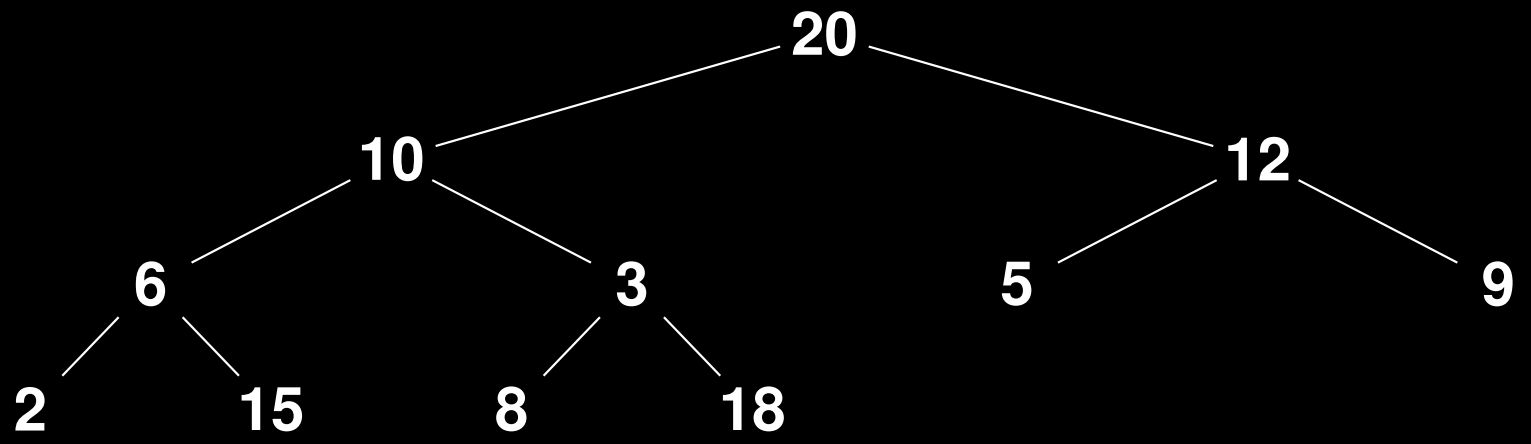


# Heap Sort Example



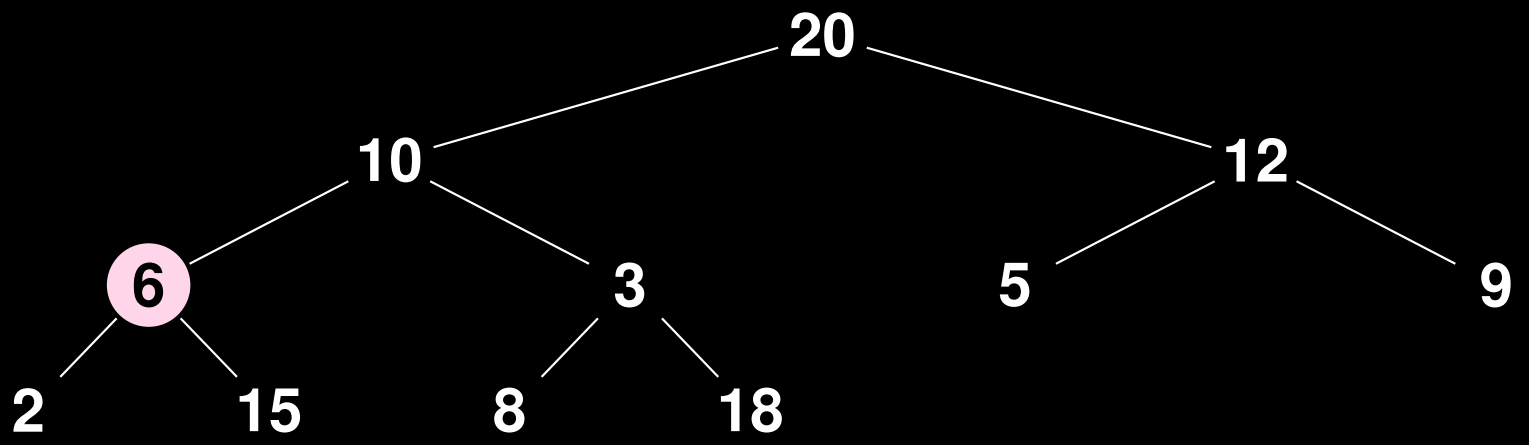
# Heap Sort Example

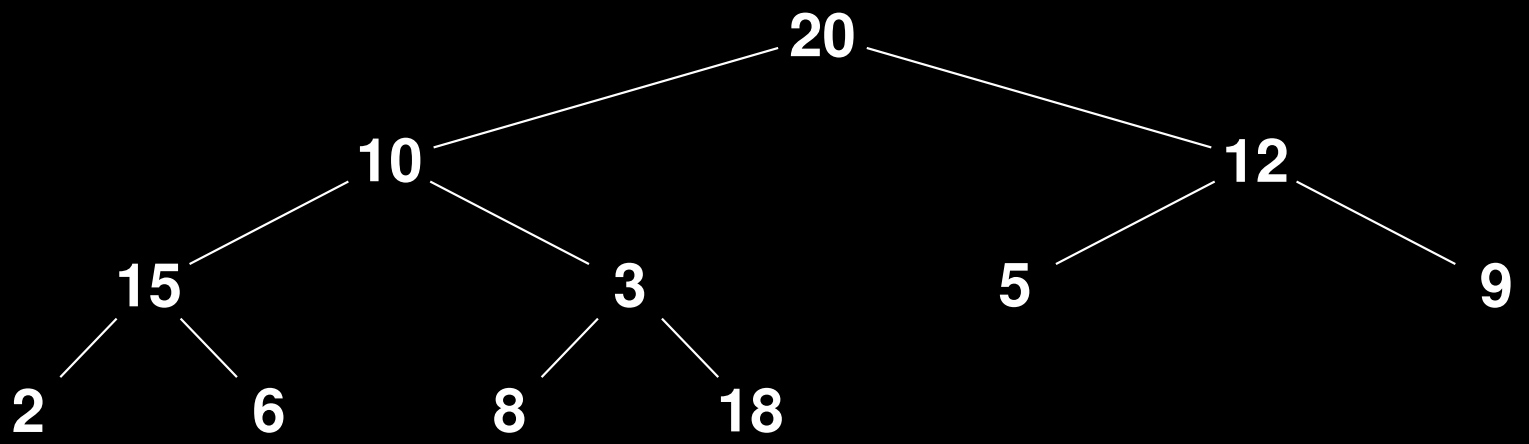


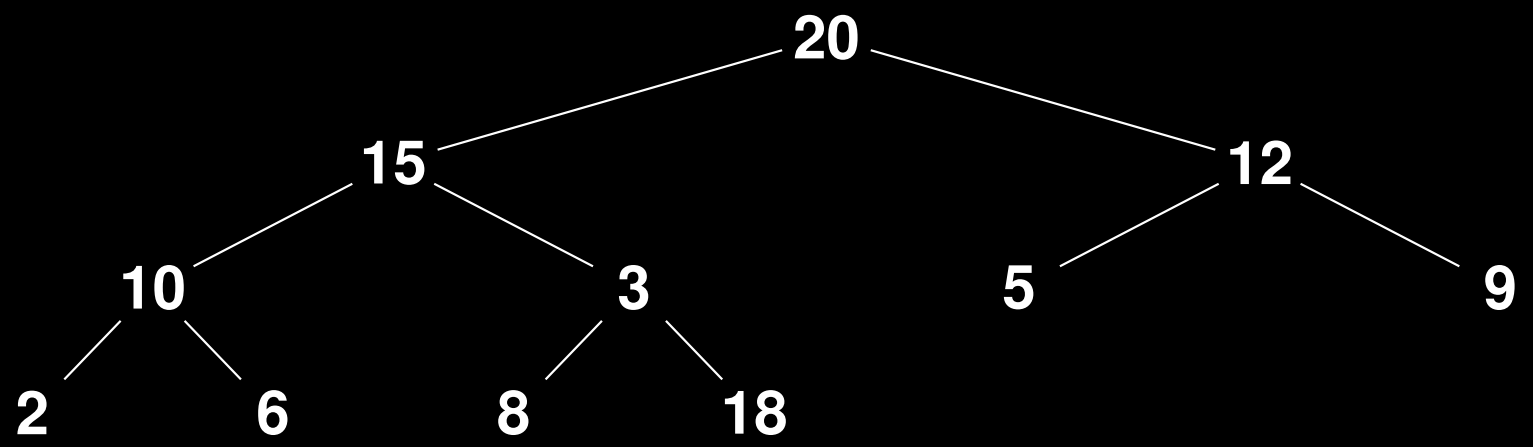
# Heap Sort Example

Fix-4



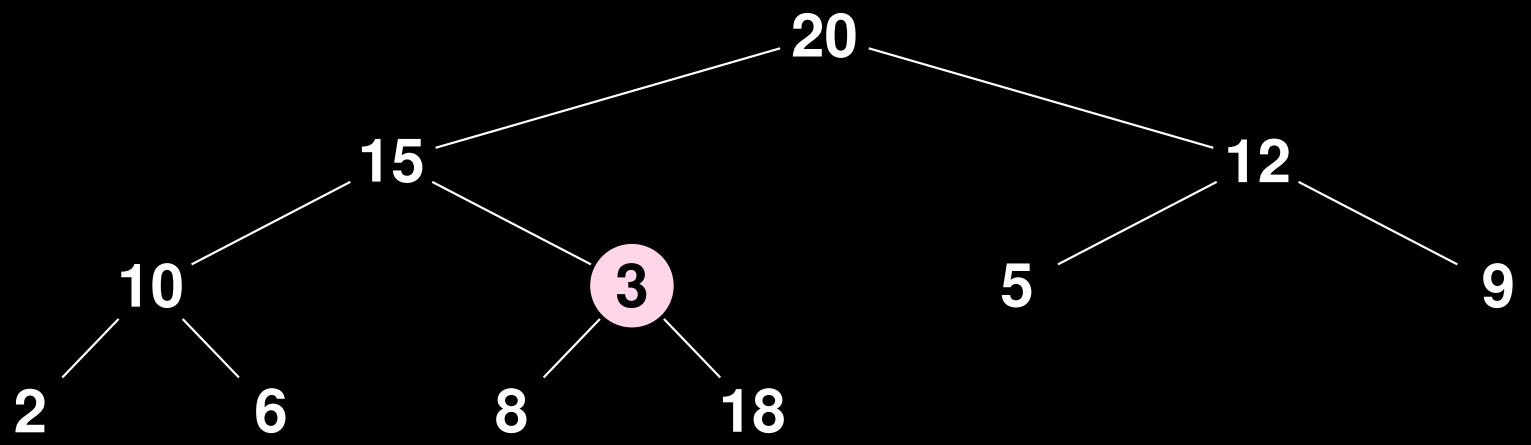



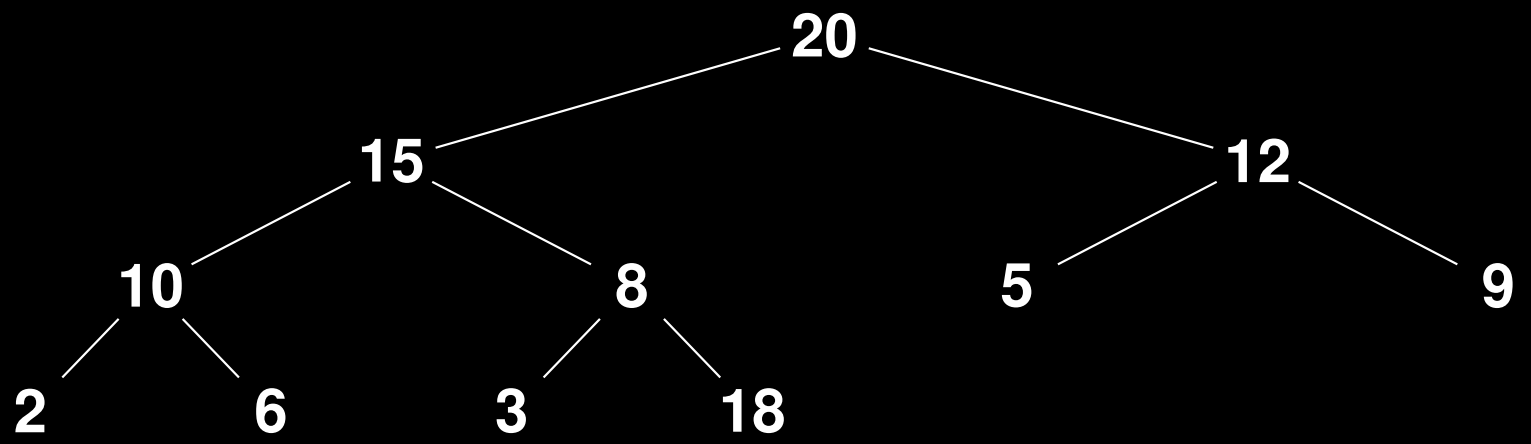
# Heap Sort Example



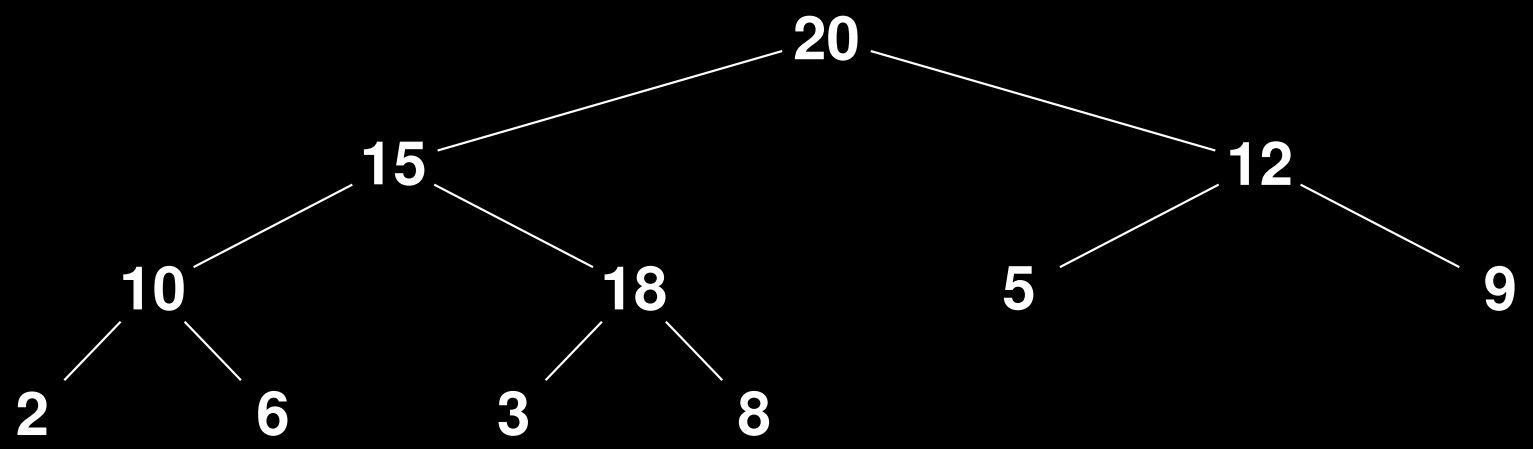
# Heap Sort Example

Fix-5

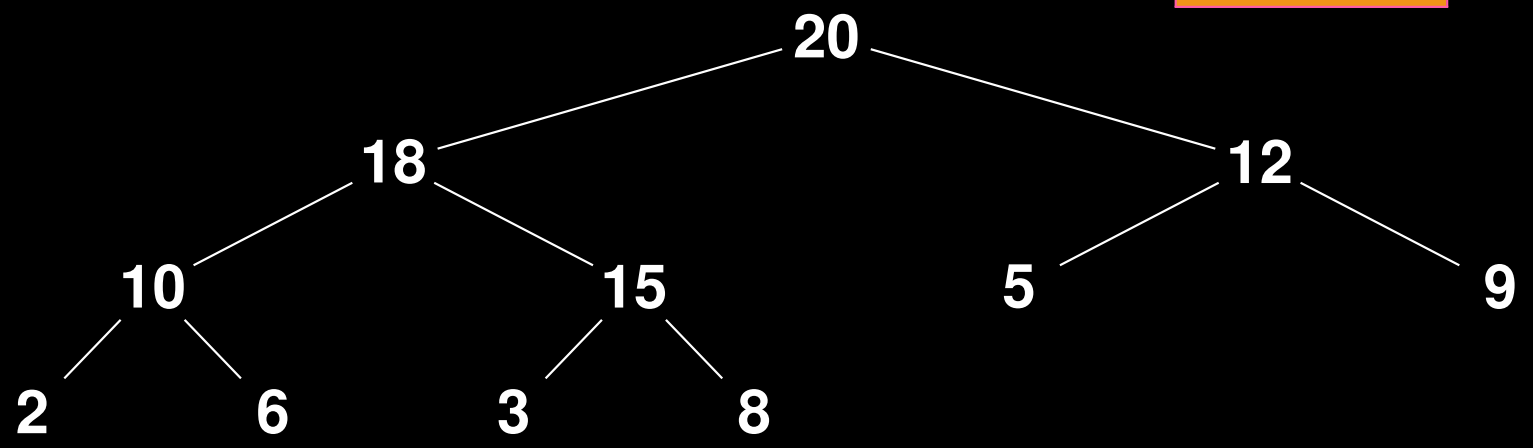


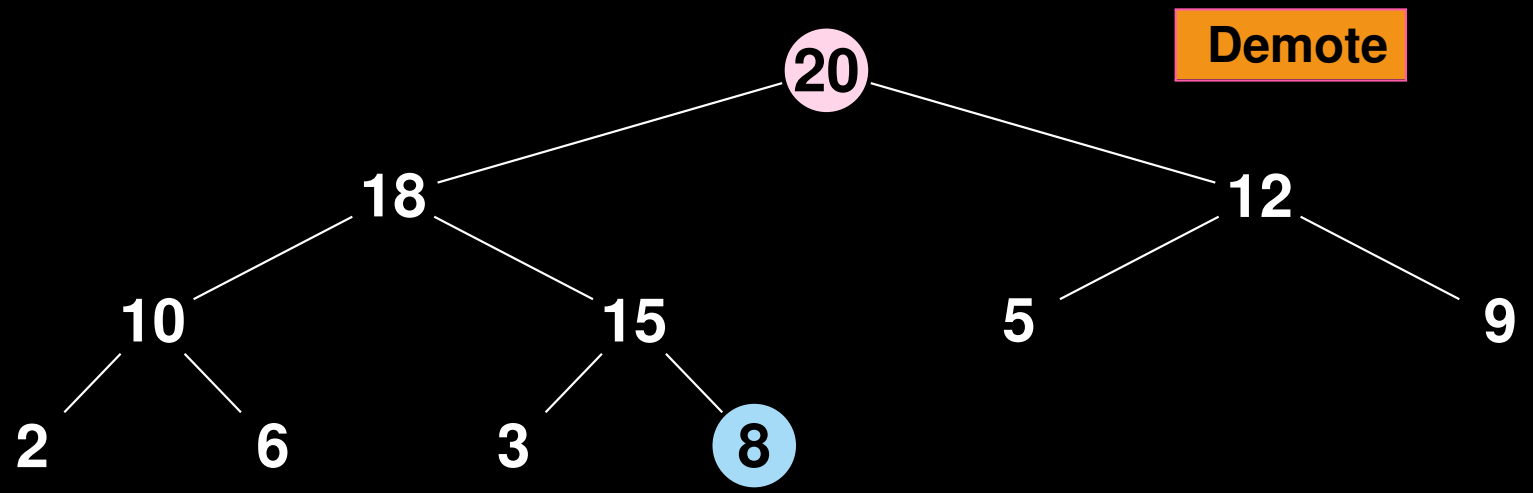
# Heap Sort Example

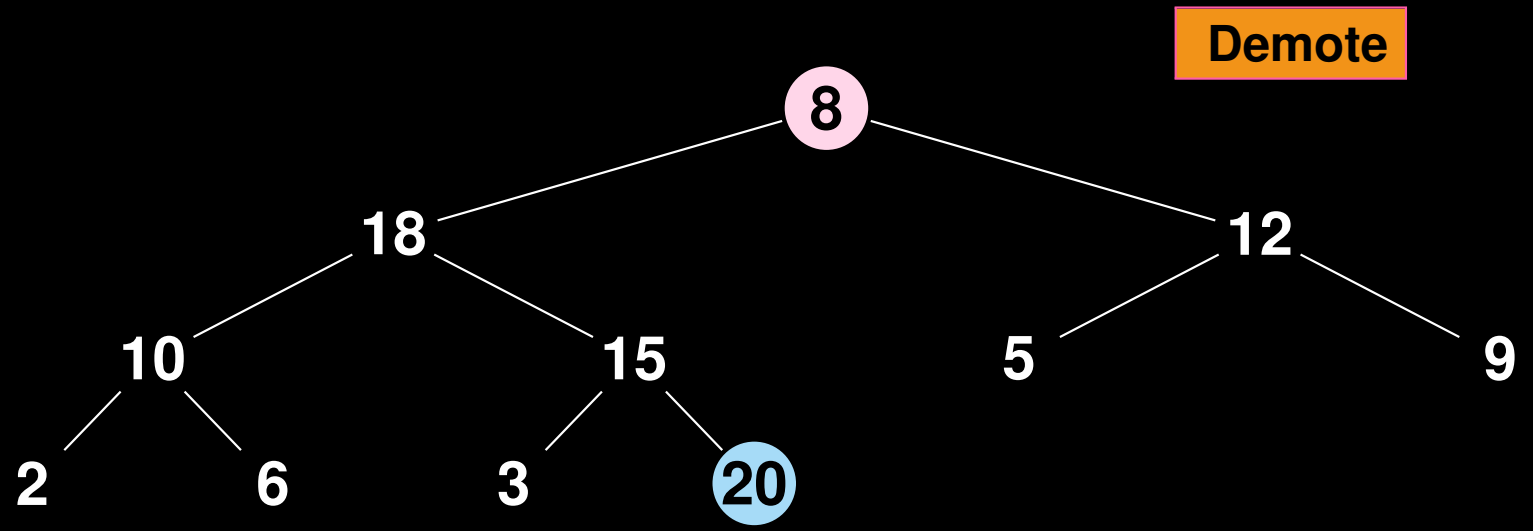



**Heapified**



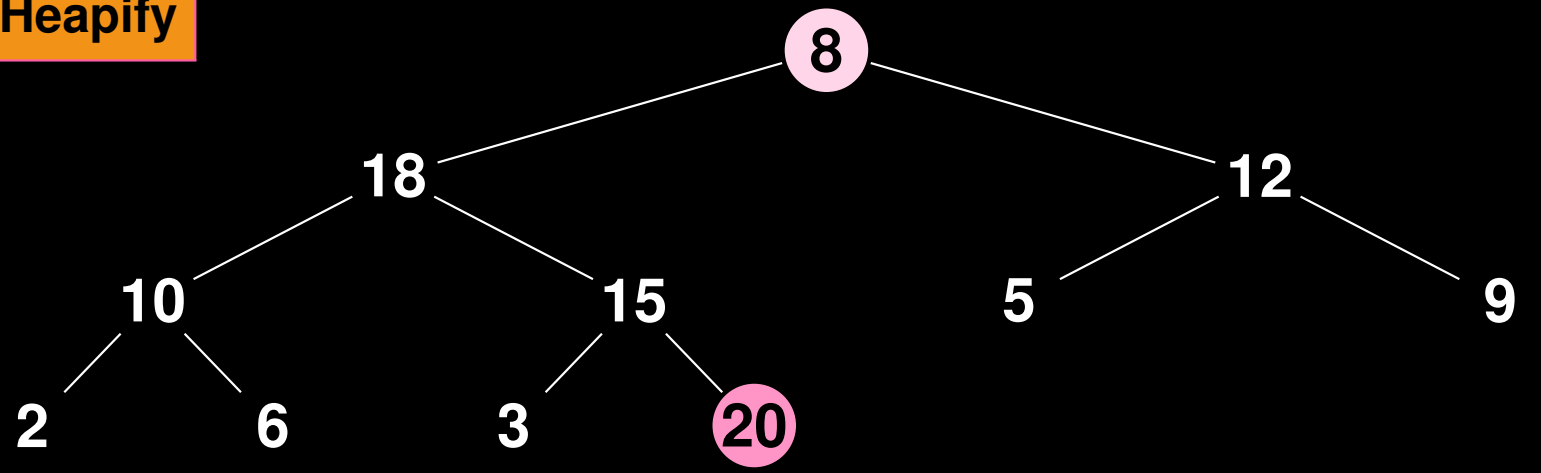
# Heap Sort Example

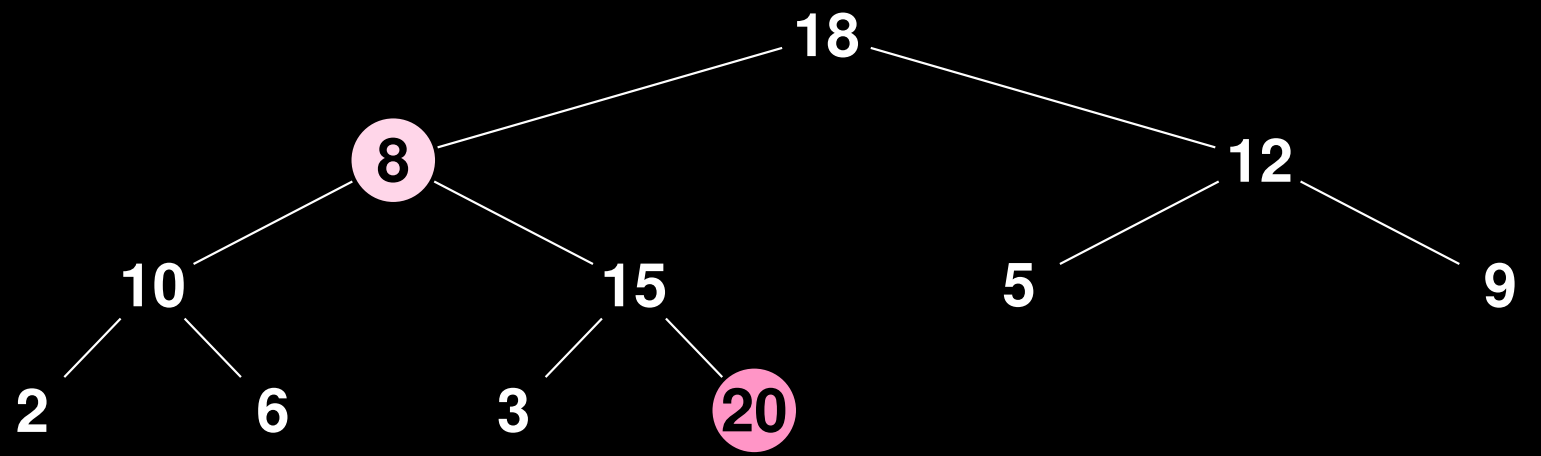


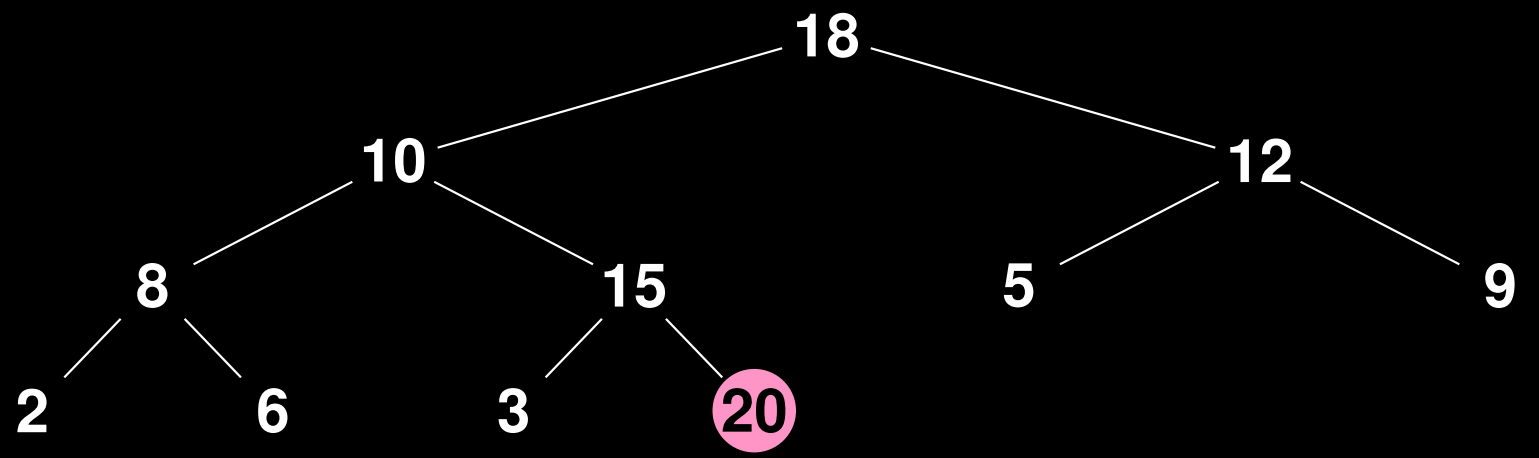
# Heap Sort Example

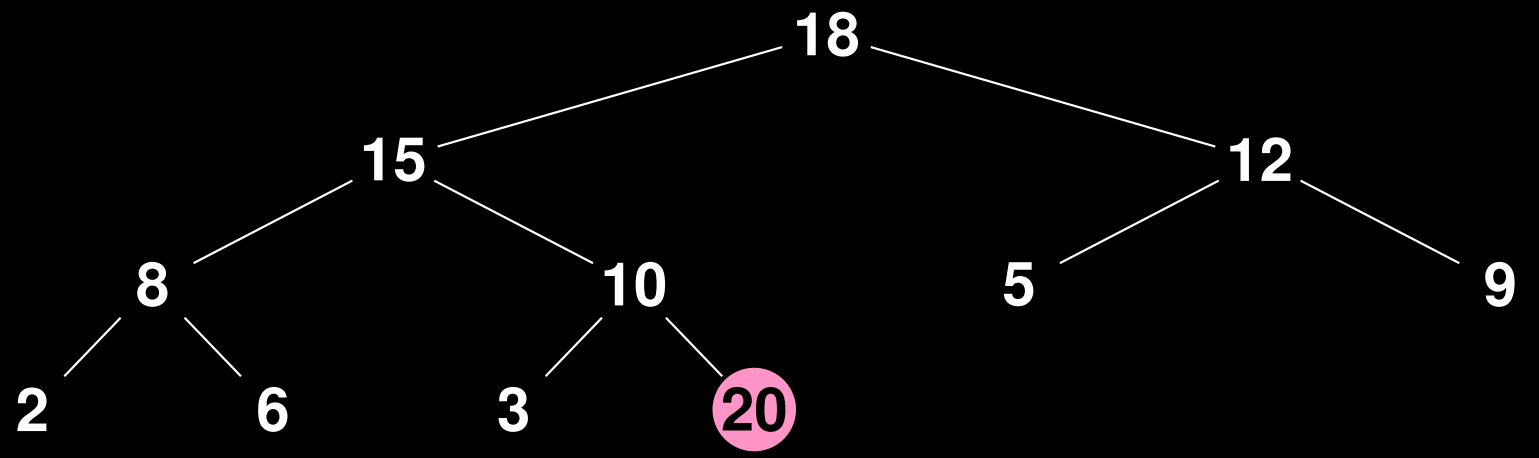
**Step 2: Heapify**



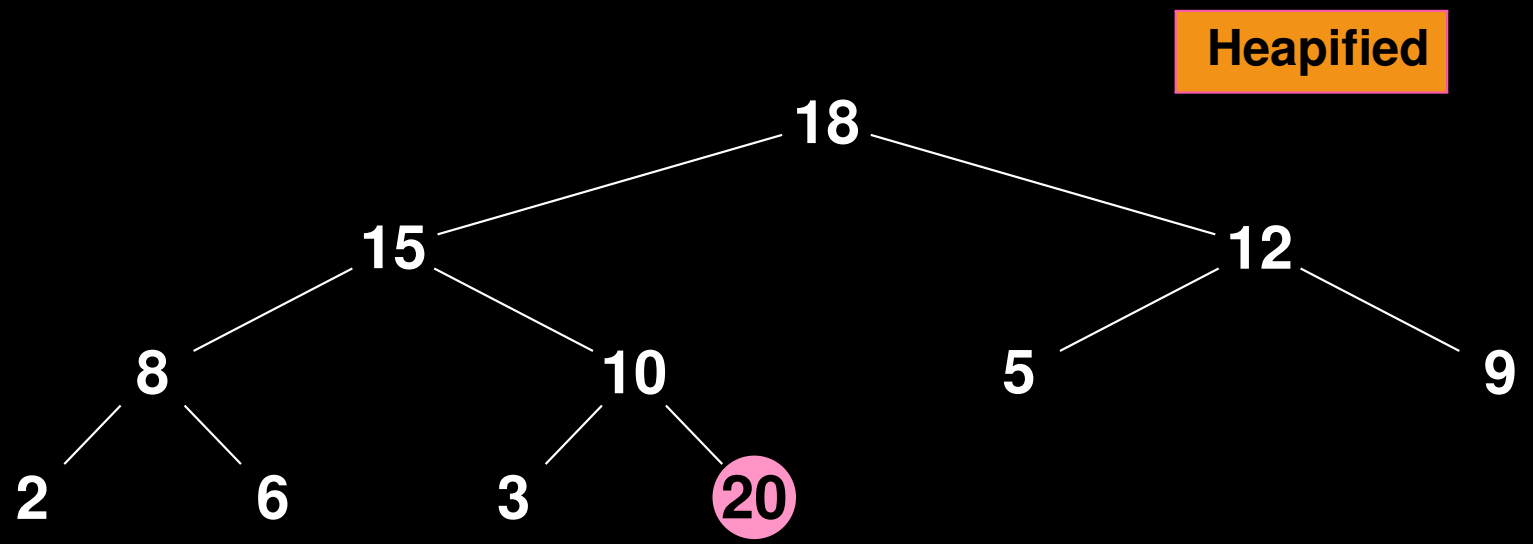



# Heap Sort Example

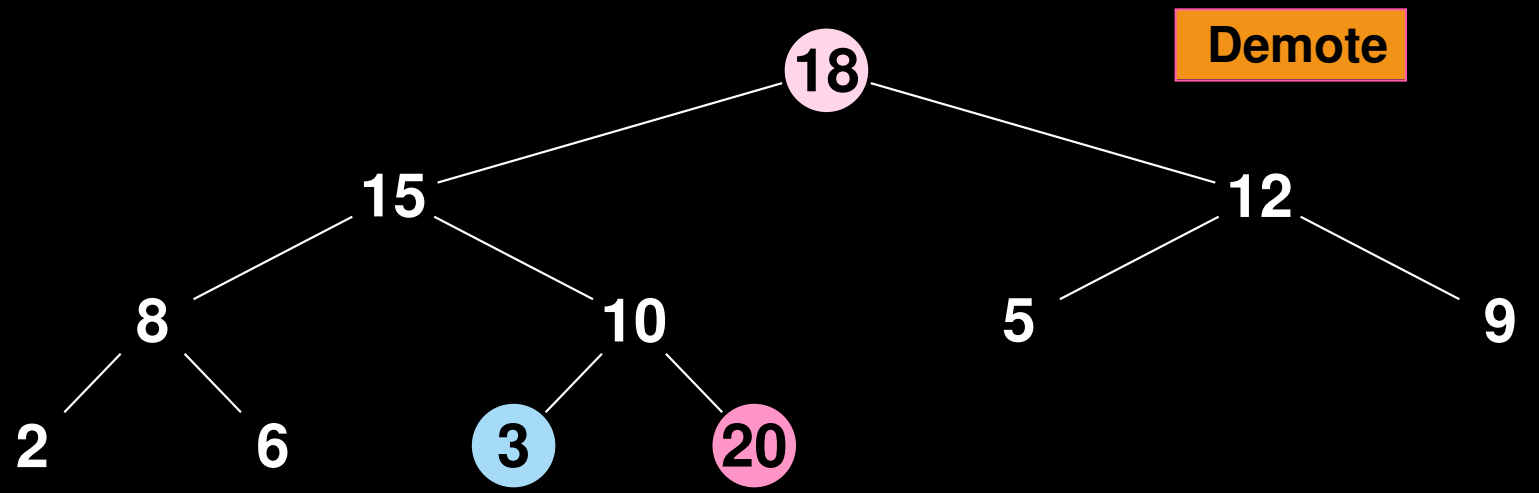


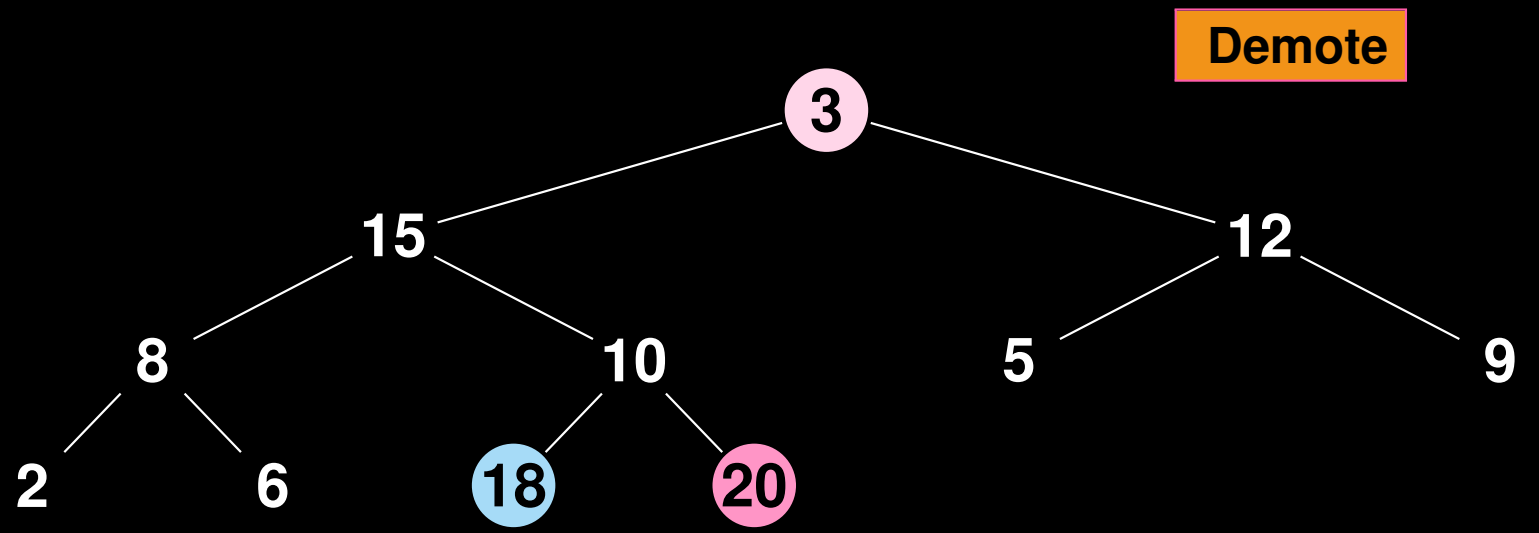



# Heap Sort Example



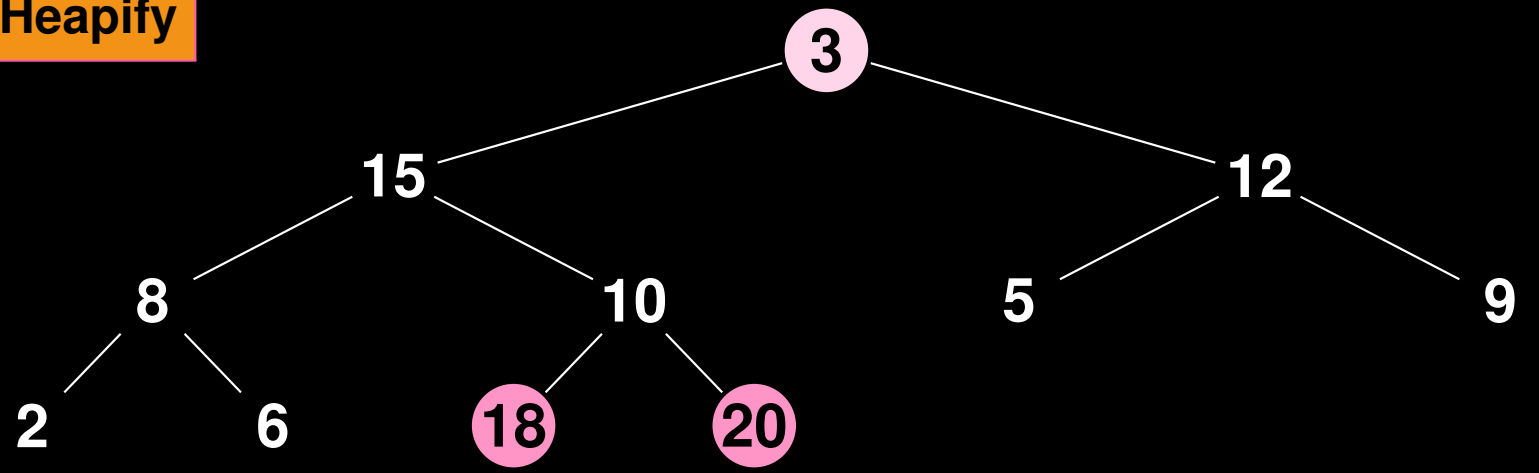
# Heap Sort Example

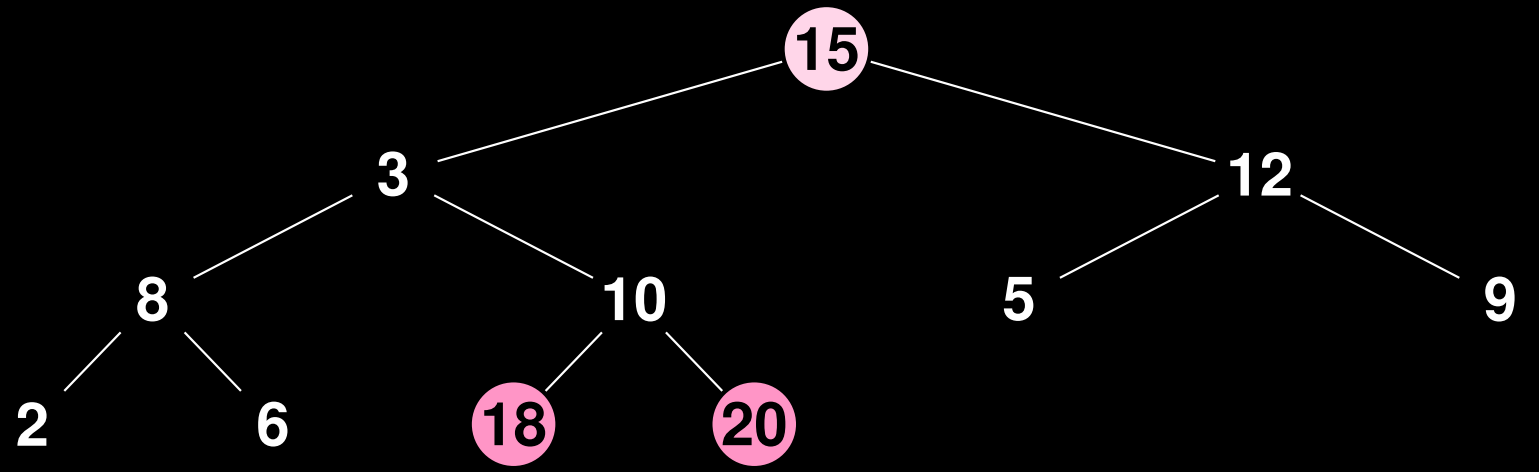


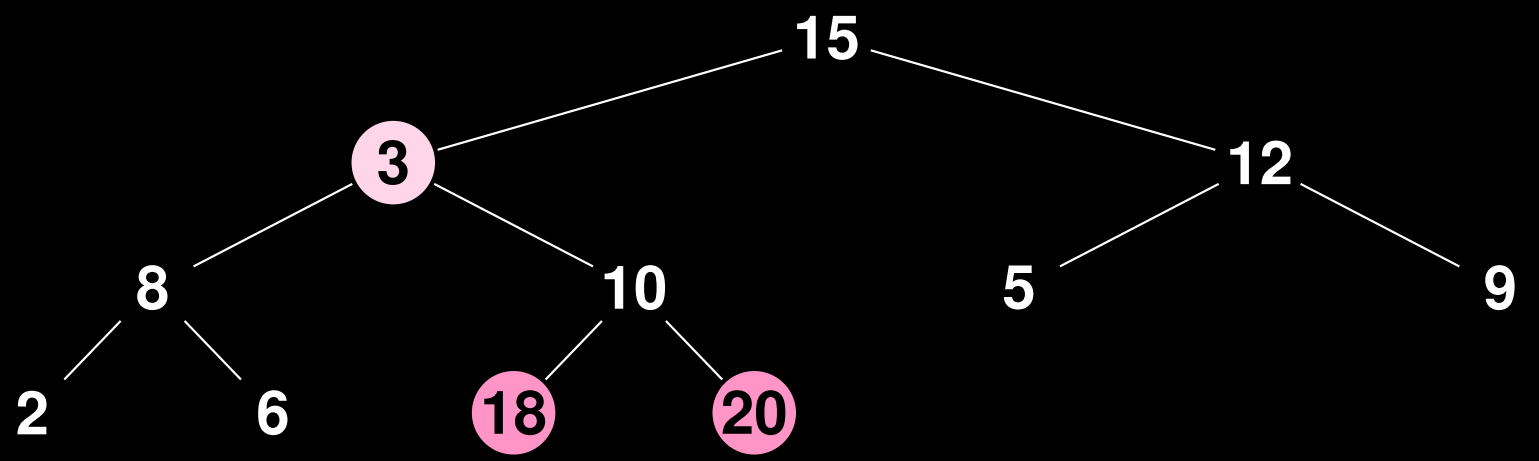
# Heap Sort Example

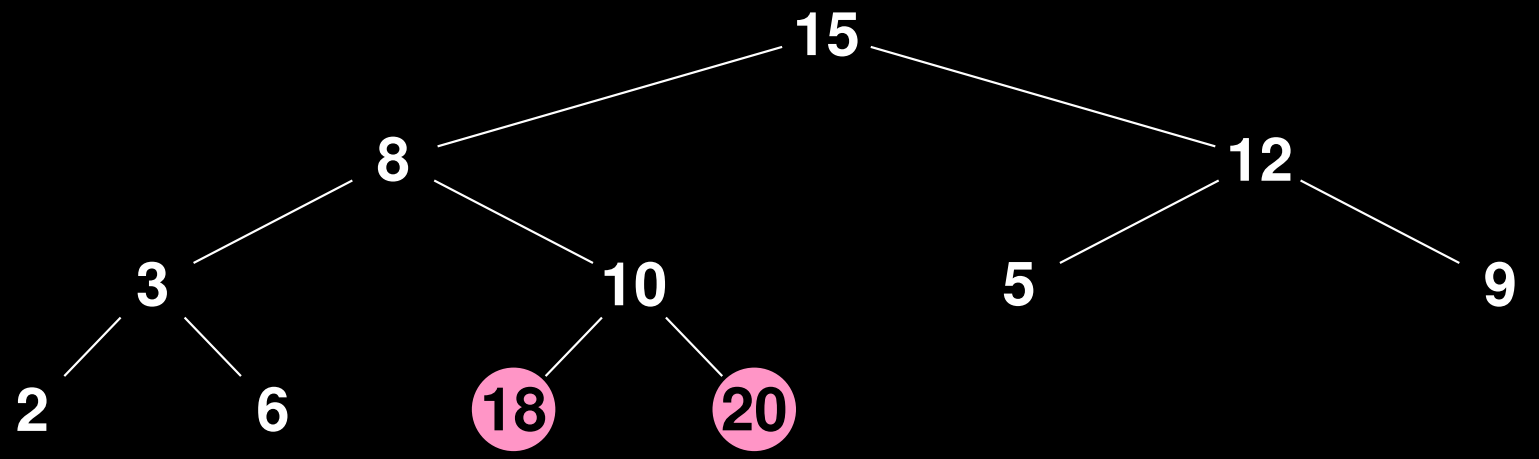
**Step 3: Heapify**



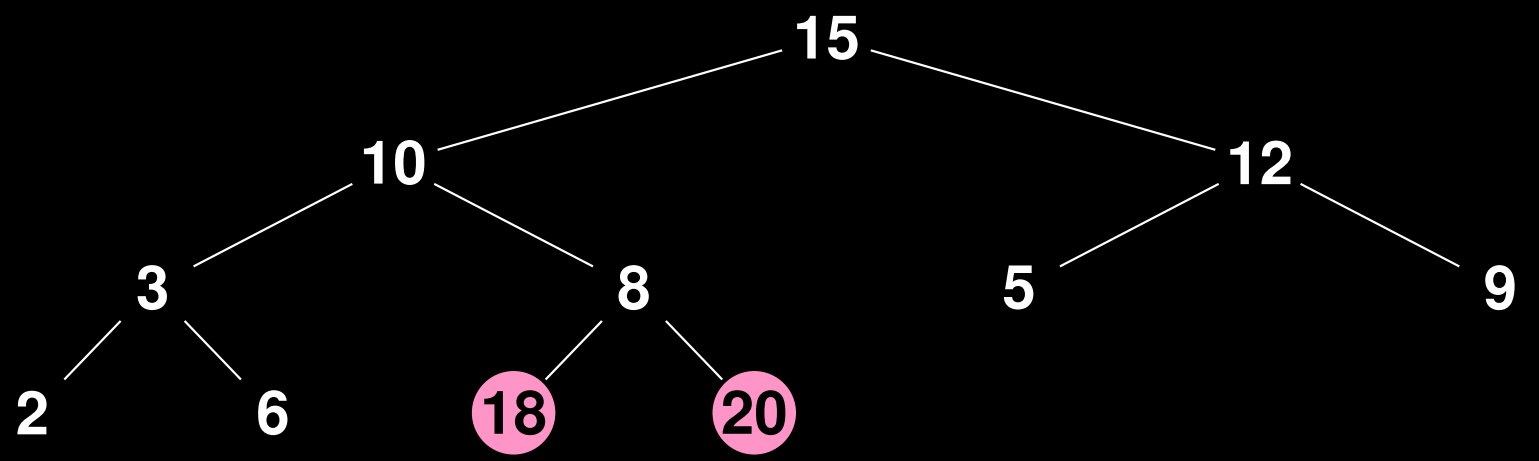



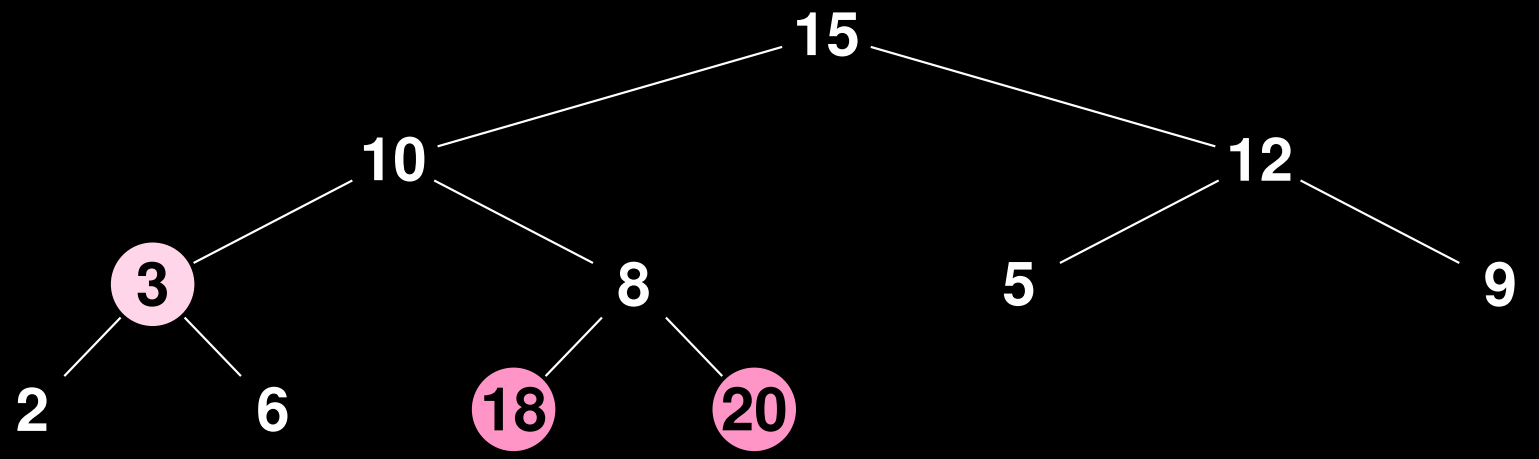
# Heap Sort Example



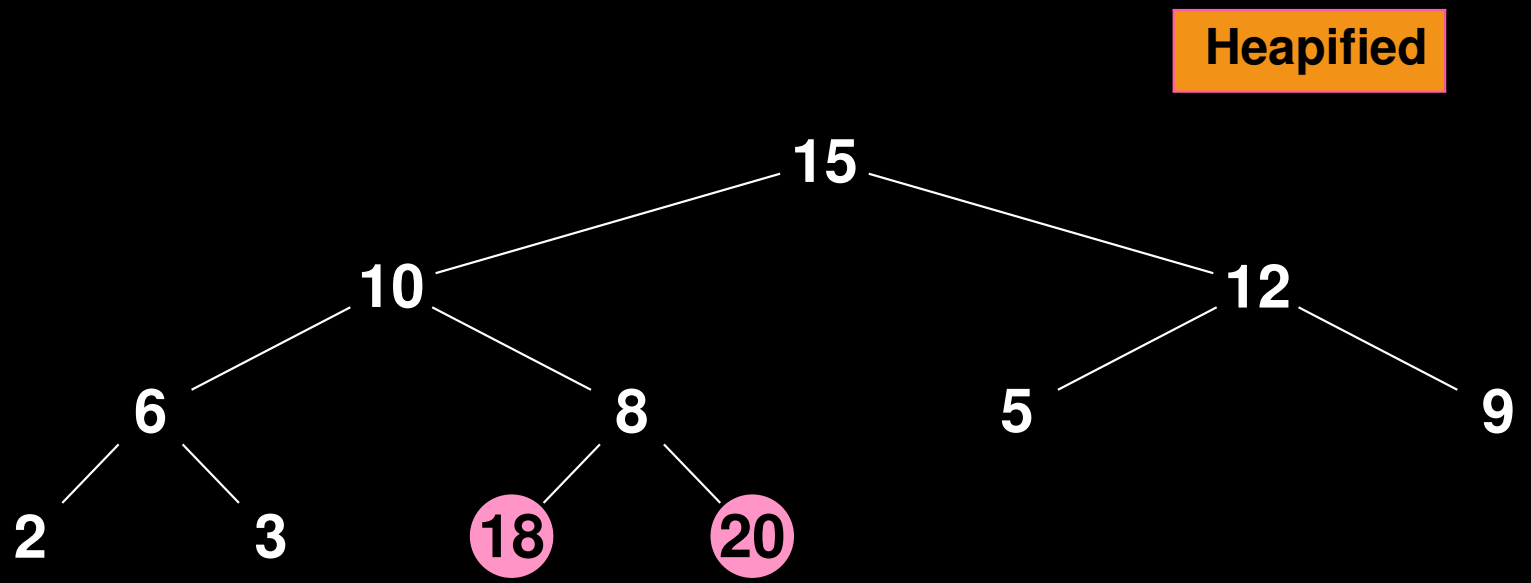



# Heap Sort Example

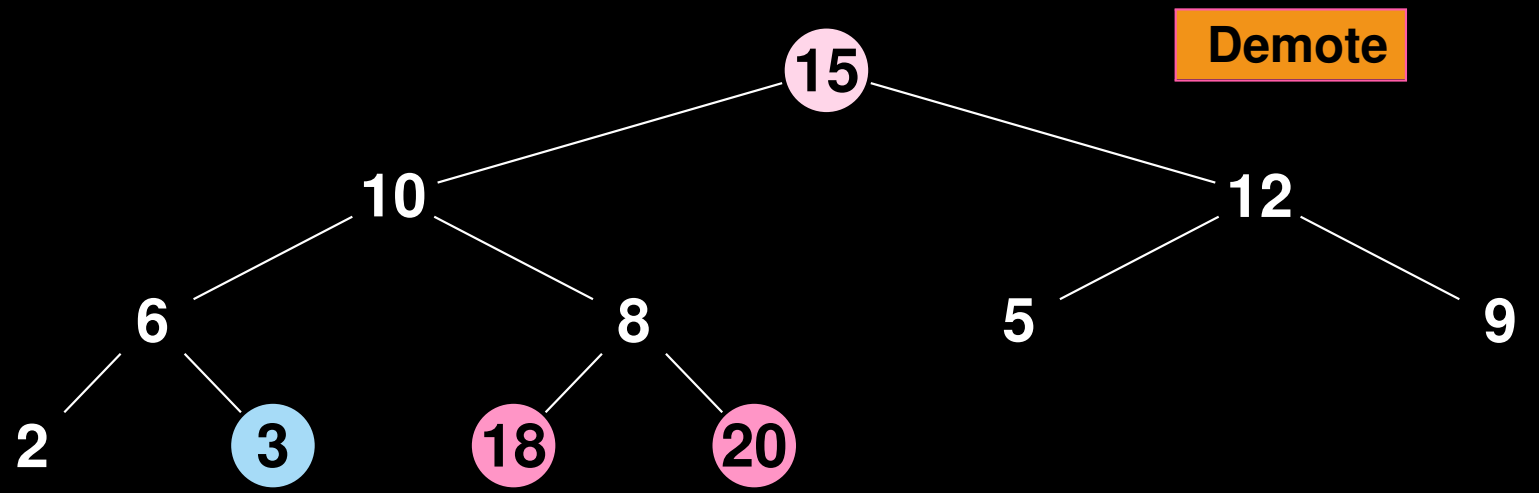


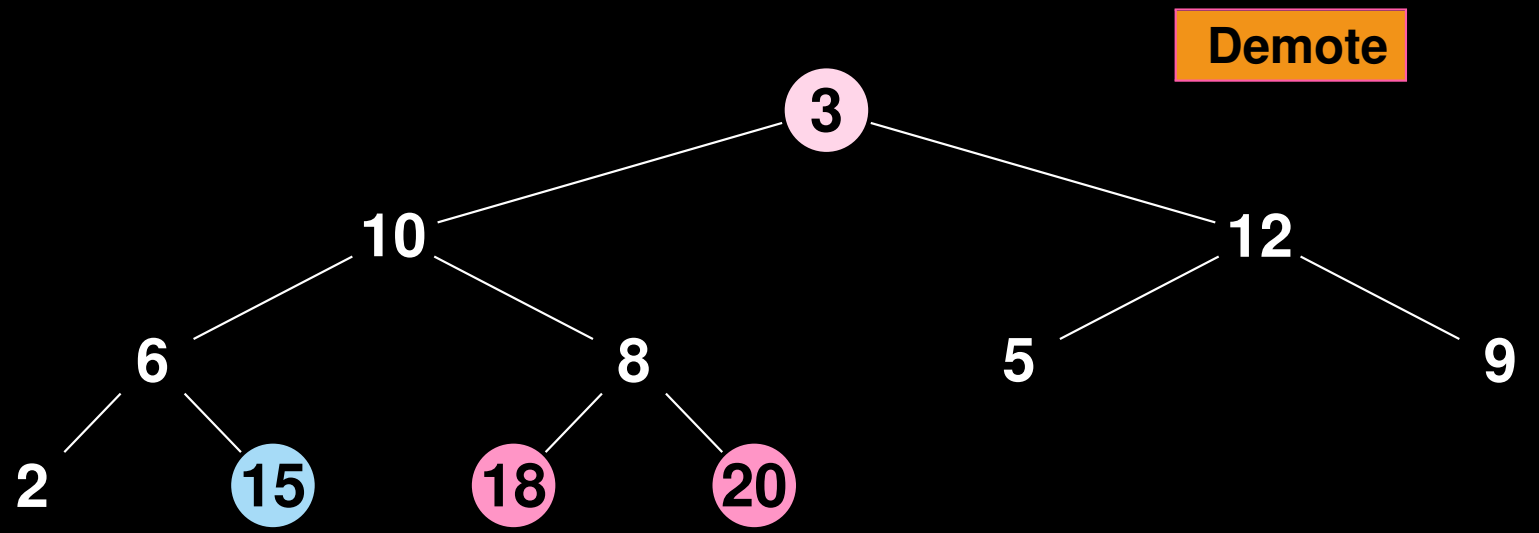



# Heap Sort Example



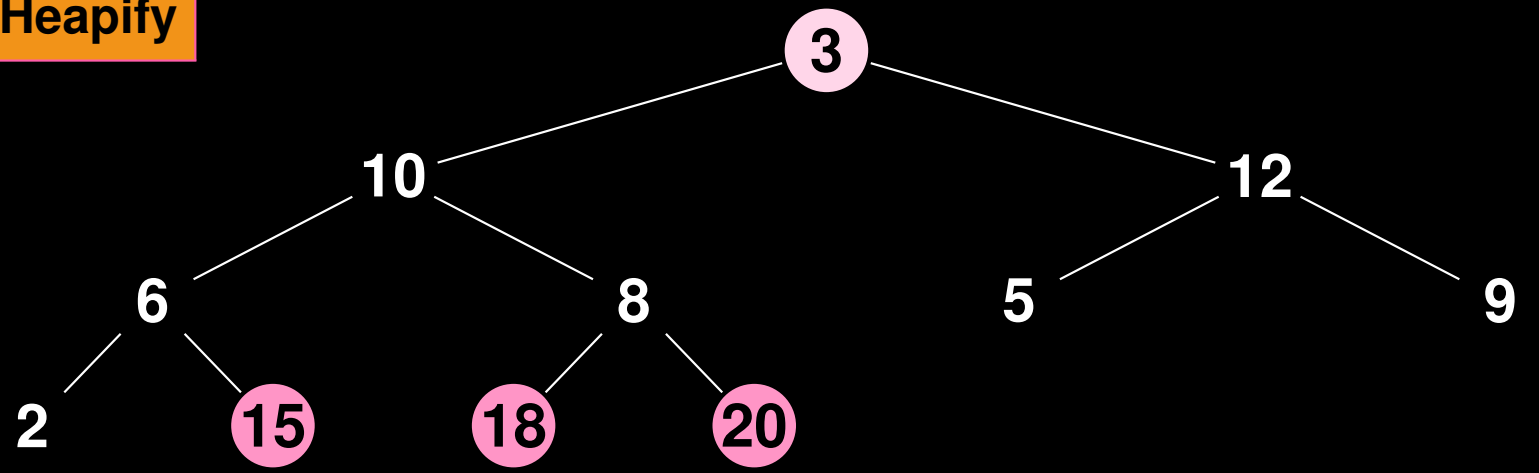
# Heap Sort Example

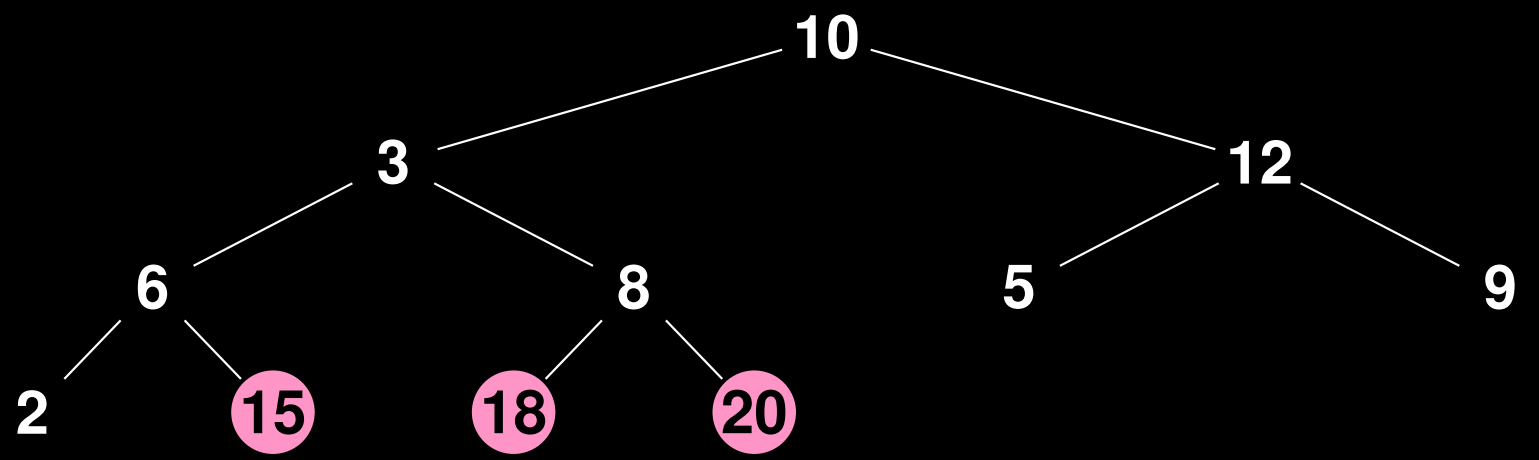


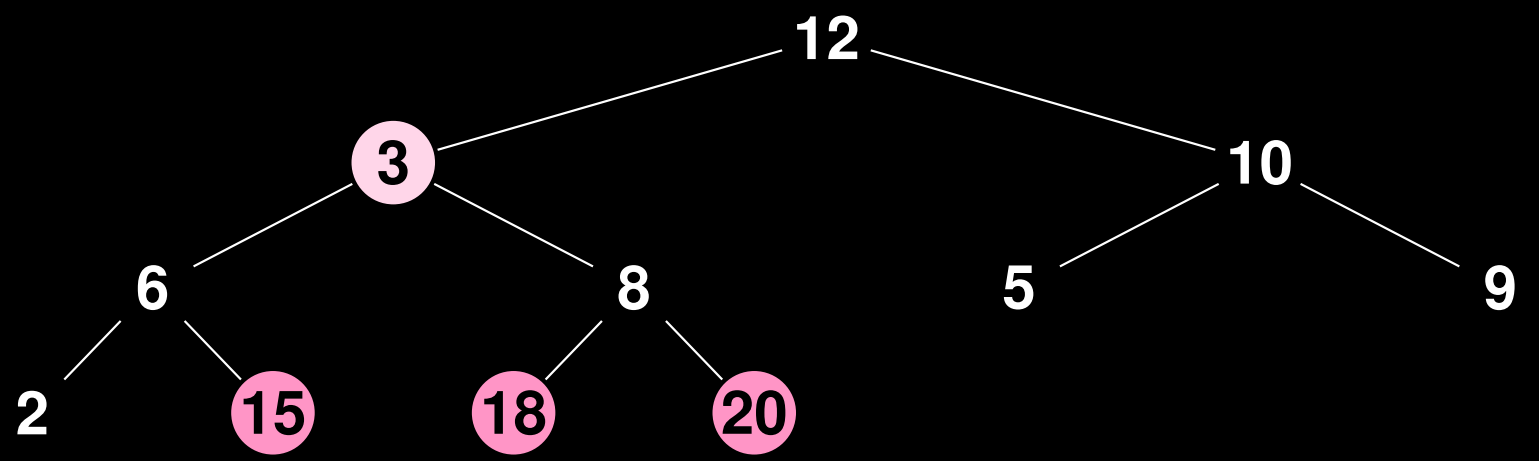
# Heap Sort Example

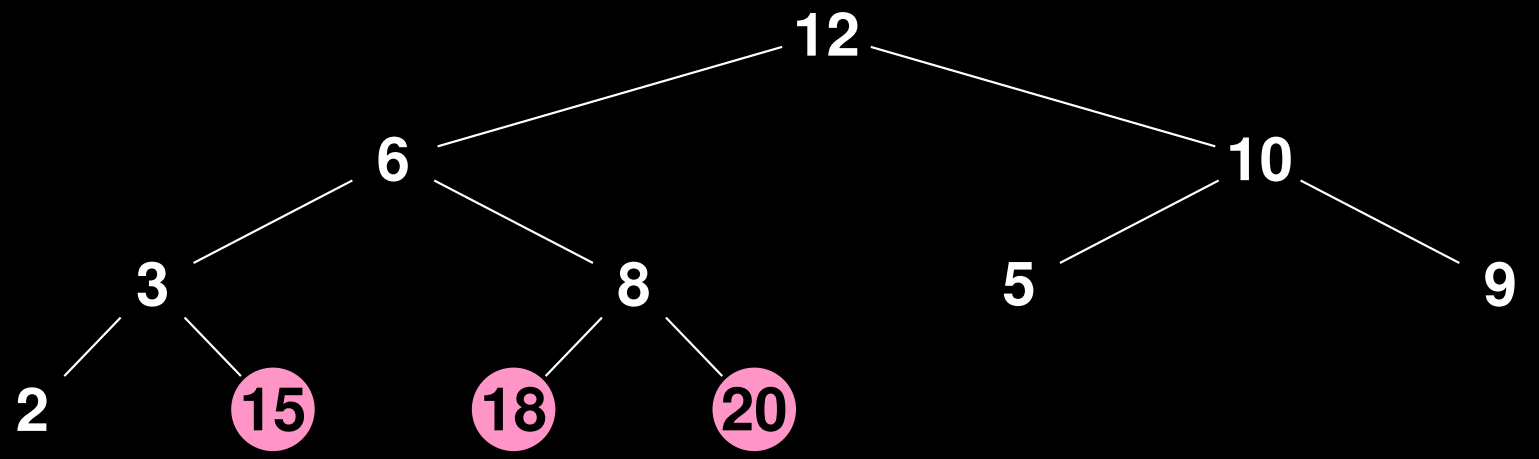
**Step 4: Heapify**



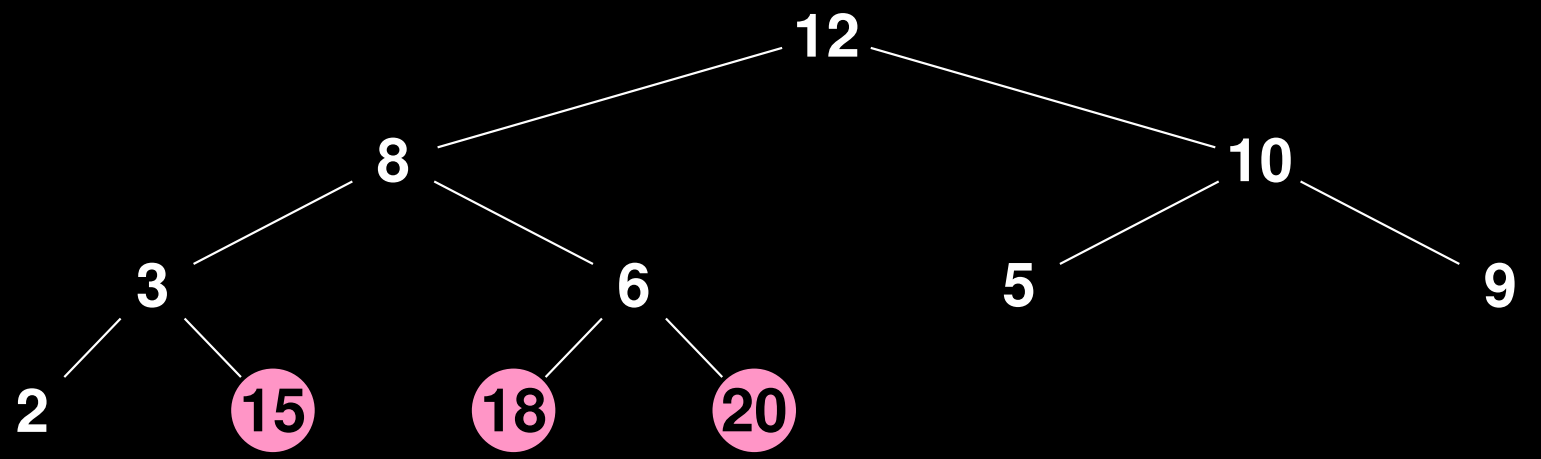



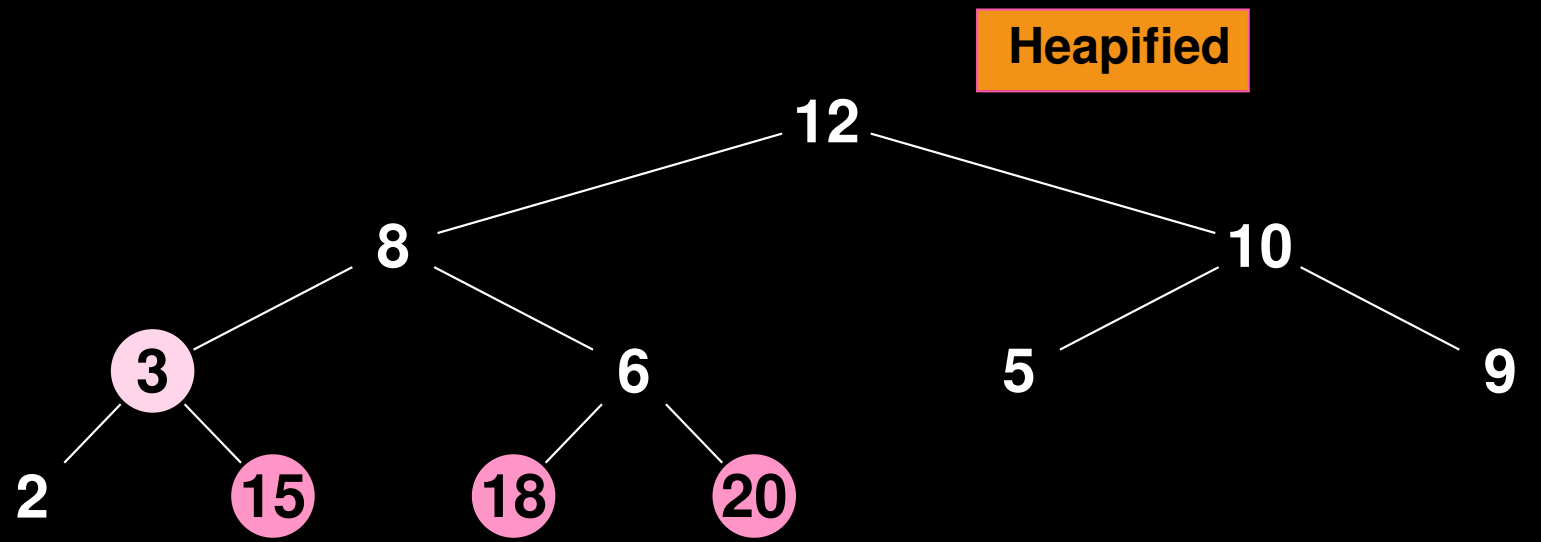
# Heap Sort Example



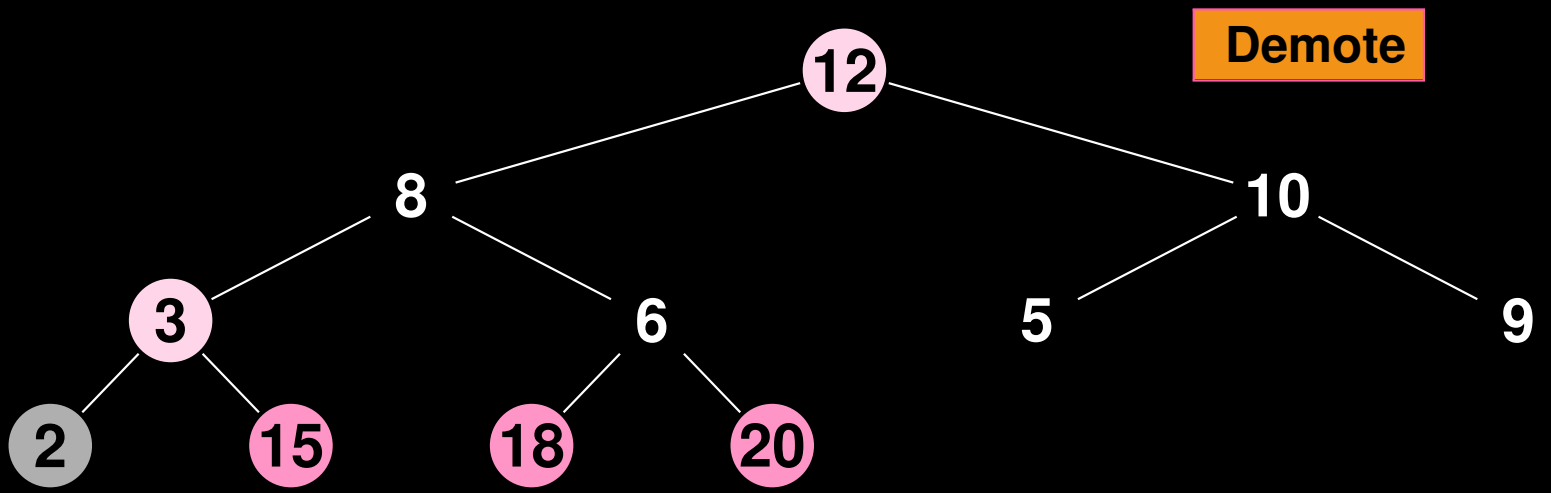



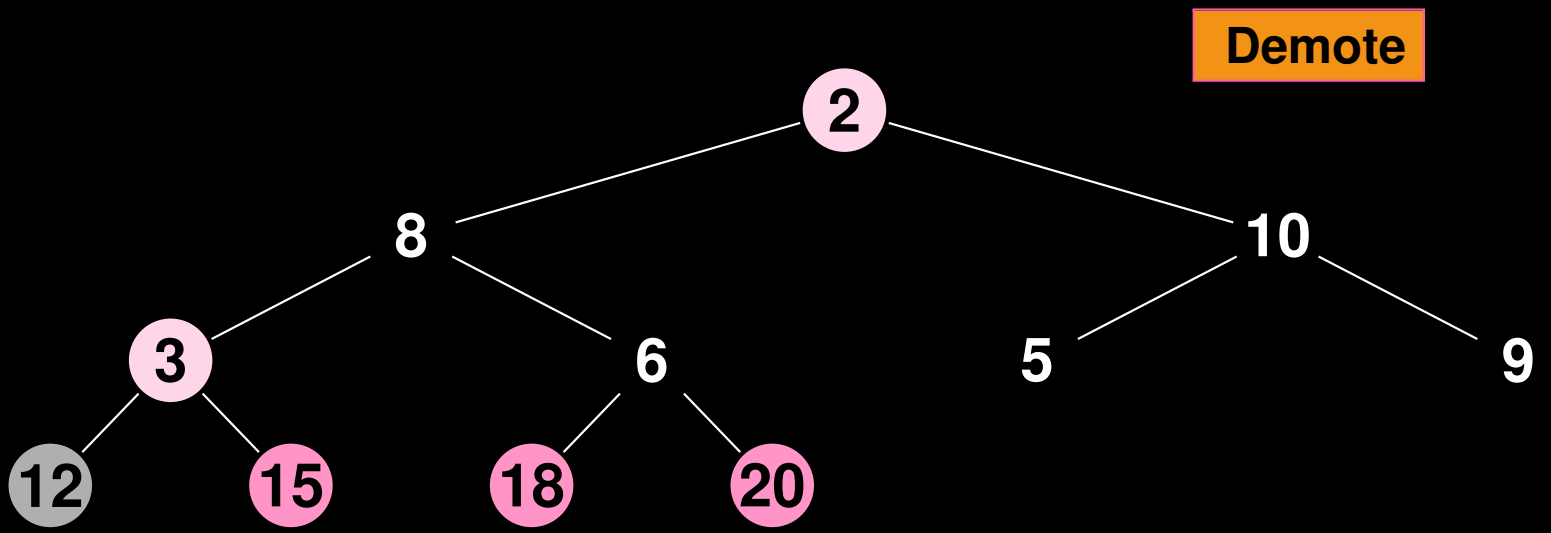
# Heap Sort Example



# Heap Sort Example

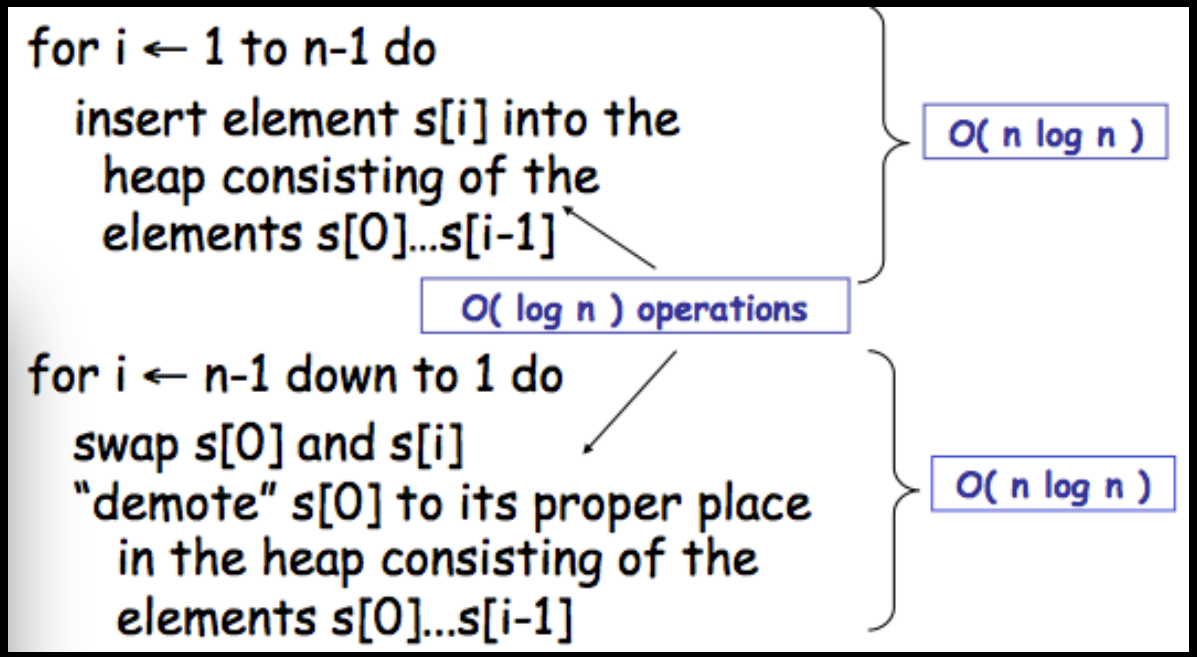


# Heap Sort

Complete other steps.
Repeat steps till all nodes becomes **Red**
At the end of all steps, given array is **Sorted**.

# Heap Sort Complexity



# Heap Sort

Note that heap sort is just a more clever version of selection sort since a maximum is repeatedly selected and placed in its proper position.

**Selection Sort in next Lab.**

# Heap Sort

Best, Average, and Worst case is **O(n logn)**

**Unstable**, and Inefficient for complex data.

# Sorting Algorithms - Comparison

| Algorithm | Time | Notes |
|---|---|---|
| selection-sort | $O(n^2)$ | slow<br>in-place<br>for small data sets (< 1K) |
| insertion-sort | $O(n^2)$ | slow<br>in-place<br>for small data sets (< 1K) |
| heap-sort | $O(n \log n)$ | fast<br>in-place<br>for large data sets (1K — 1M) |
| merge-sort | $O(n \log n)$ | fast<br>sequential data access<br>for huge data sets (> 1M) |

# Reading Assignment

**Sedgewick 2.4 Heap Sort**

# Questions?

**Please ask if there are any Questions!**