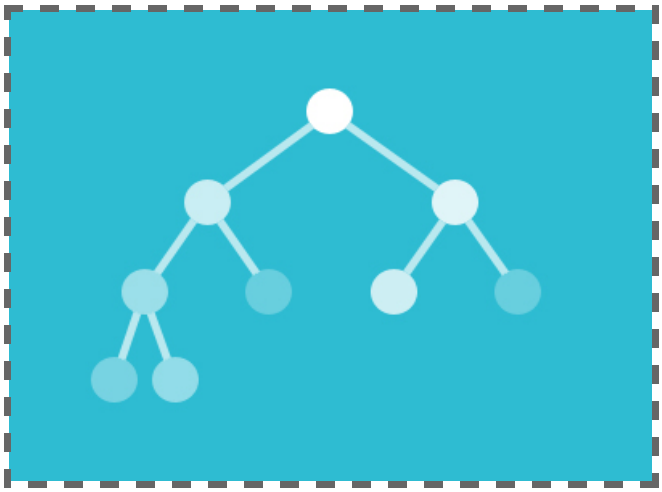
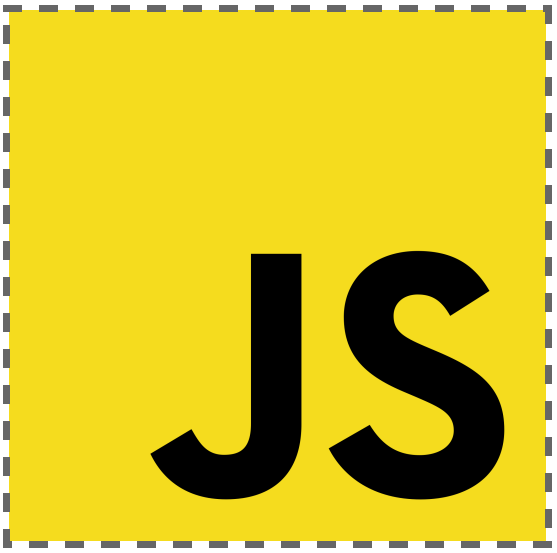
Heap in JavaScript

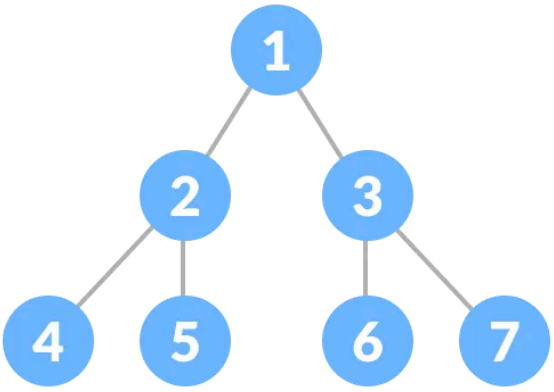
Did you know CPU scheduling uses a **priority queue**, which is usually implemented using a **heap** data structure?





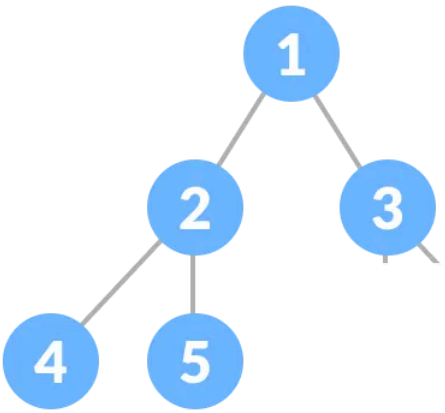
Binary Tree

- A **binary tree** is a tree where each node can have **at most two children** — called **left** and **right** child.
- The **level numbering** starts from **0**.
    - Example: Root = level 0, its children = level 1, and so on.
- The **maximum number of nodes** at level **n** = 2^n.



Almost Binary Tree

- A **tree** is called **almost binary tree** (or **complete binary tree**) when **all levels are completely filled** except possibly the **last level**,
  and the **last level's nodes are as far left as possible**.



Heap

- A **heap** is a **special kind of binary tree** that follows a specific **ordering rule**.
- There are **two types of heaps**:
    1. **Min-Heap**
    2. **Max-Heap**

Min-Heap

- Each **parent node** is **smaller than or equal to** its **children**.
- The structure must be an **almost binary tree**.

Max-Heap

- Each **parent node** is **greater than or equal to** its **children**.
- The structure must also be an **almost binary tree**.

Bubble Up & Sink Down

- **Bubble Up (Heapify Up):**
  When a new element is inserted, it moves **upward** until the heap property is restored.

- **Sink Down (Heapify Down):**
  When the root element is removed, the last element replaces it and **moves downward** to restore the heap property.

Data Structure Used to Create Heap

- A **heap** is implemented using an **array**.

- The elements are stored **level-wise** from **top to bottom, left to right**.

Index Calculations in Heap (Array Representation)

Let the index of a node be **i**:

Relation	Formula	Example (for i = 4)	Result
Parent Index	`(i - 1) / 2` *(take floor value)*	`(4 - 1)/2`	`1`
Left Child Index	`(2 * i) + 1`	`(2 * 4) + 1`	`9`
Right Child Index	`(2 * i) + 2`	`(2 * 4) + 2`	`10`