Data Structures

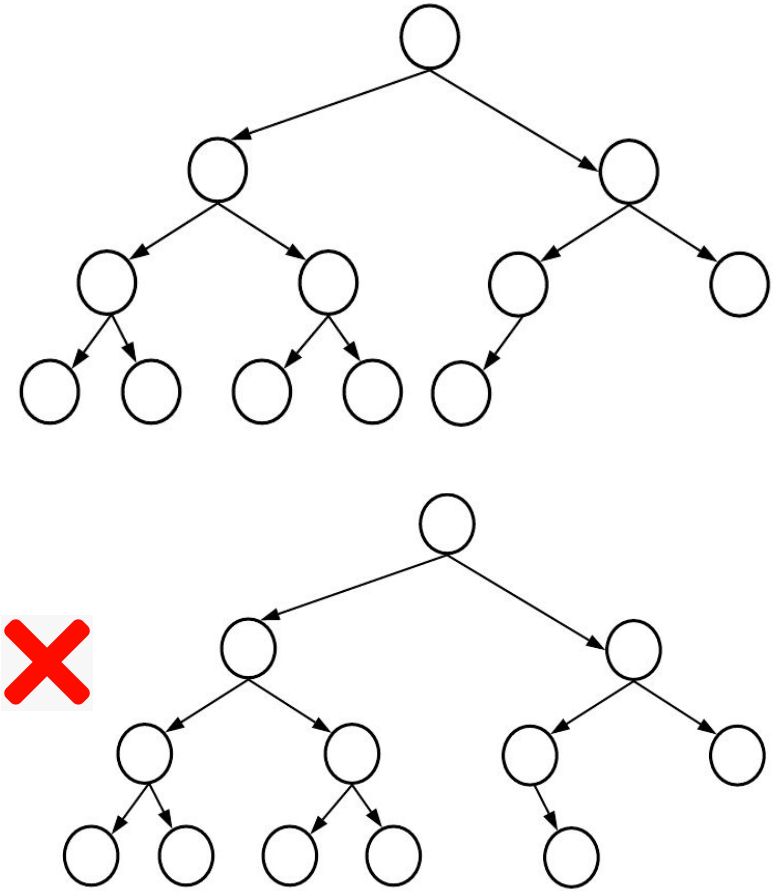
Binary Heap

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Recall: Complete Binary Tree

- All levels are **complete**
  - **except possibly** the last one, which is filled from the **left**.
- Top tree
  - 4 levels
  - First 3 are complete
  - Last one has left nodes
- Bottom tree: NOT complete
  - a right node before a left one
- Height = ceil($\log_2(n+1)$) - 1
  - Each complete level is 2^i nodes

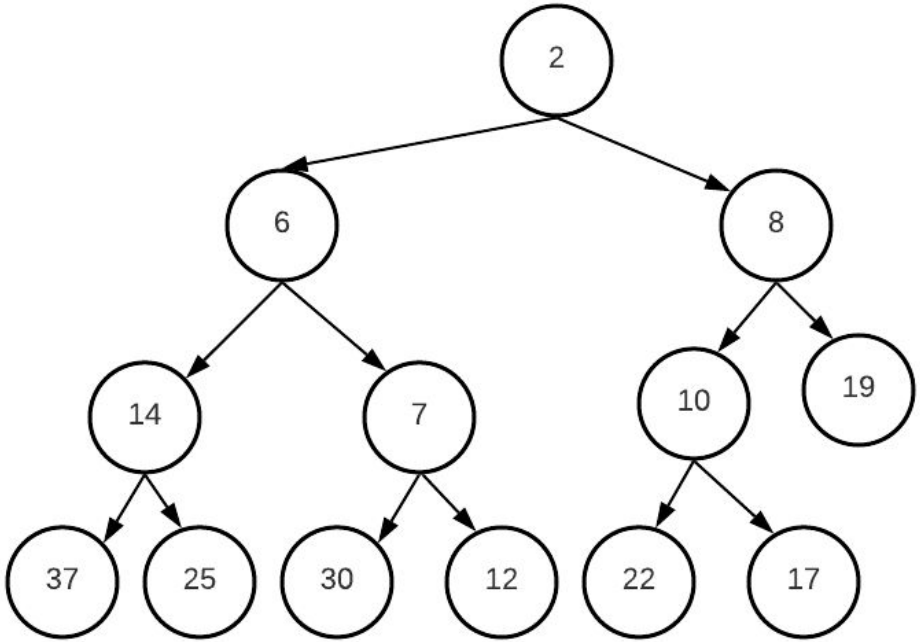


Binary Tree, Binary Search Tree and Binary Heap

- Binary Tree: max 2 children per node - simple structure
  - But search is O(N)
- We need to find a faster search structure!
- BST root > left and < right
  - We can search in O(H), which is great IFF tree is **balanced**
- In many cases we need to find **min**(or max) node and **delete** it fast!
  - This is where binary heap comes!

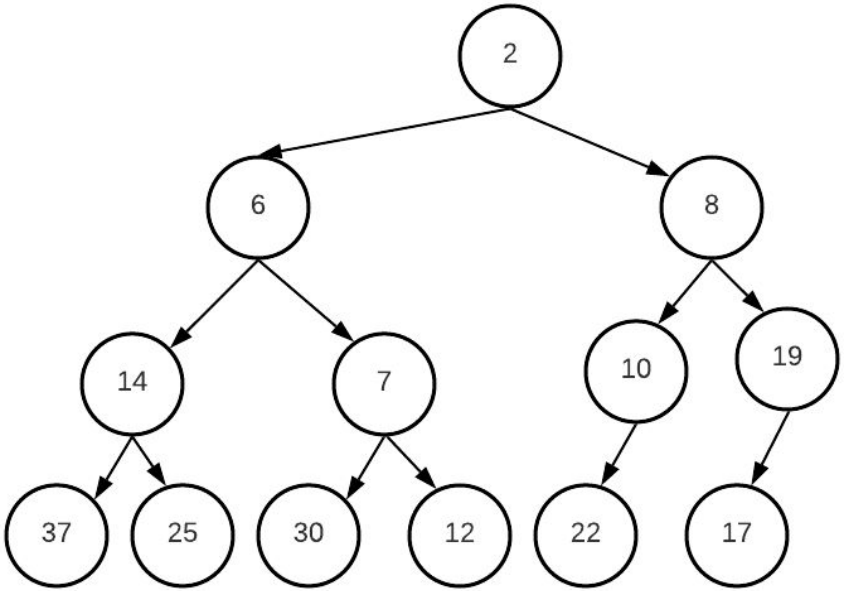
What is (min) (binary) heap

- A **complete** binary tree where any node <= **ALL** its children.
- Hence: Root has the minimum value in a tree!
- In max heap, we just has the opposite definition
  - >= all its children
- Note: Any time I say/write heap I mean **min binary heap**



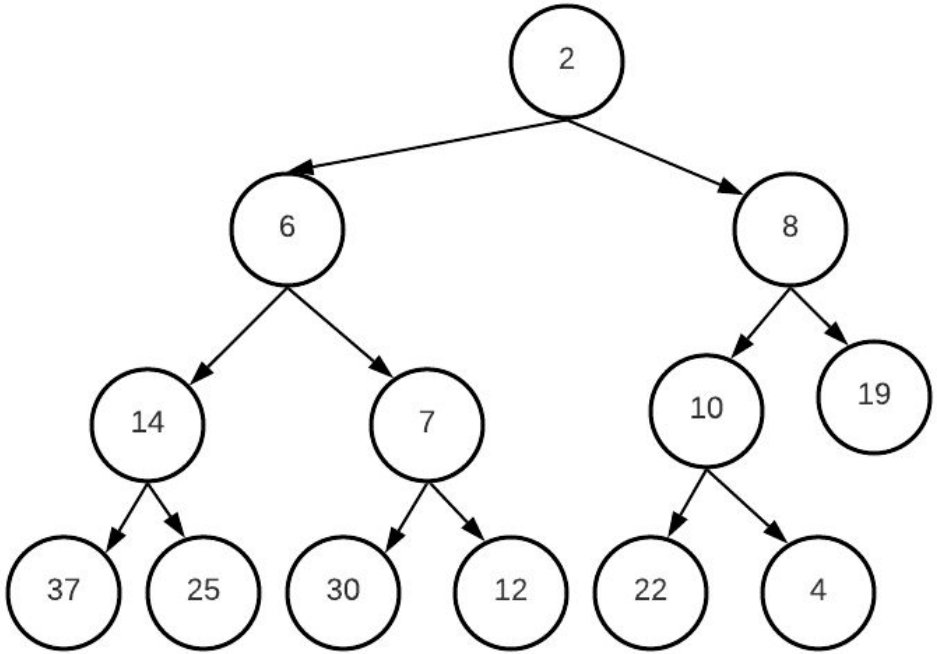
Not a min heap tree

- Not a complete tree
  - Node 10 doesn’t have right
  - Then no further node on its level or next one from this point



Not a min heap tree

- A complete tree, but node(10) is not smaller than its 2 children!
  - 10 > 4



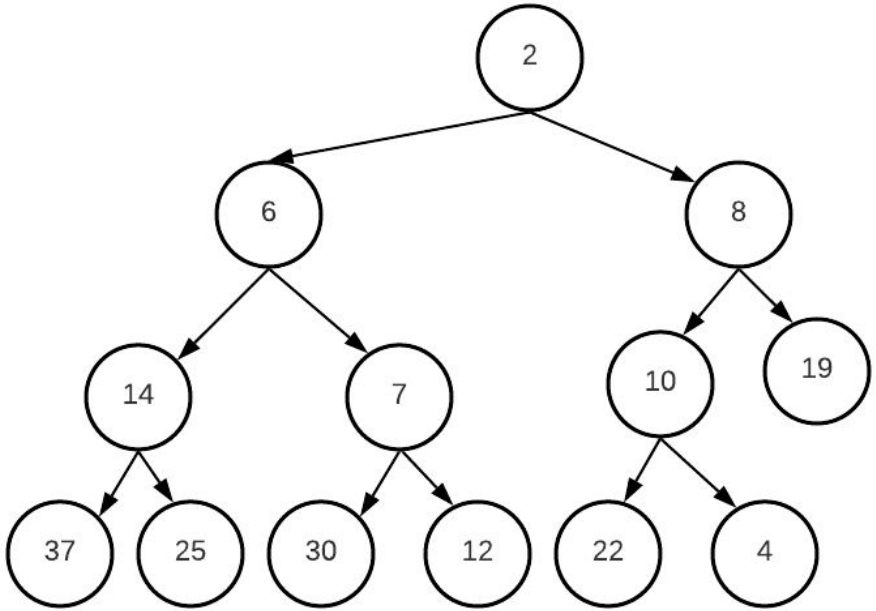
Heap ADT

- top() refers to the **min value** in the heap
- pop() will **remove the smallest** value from the heap
- Otherwise, normal push and pop like a queue
- Using isempty/top/pop we can print the content

```
 6  class MinHeap {
 7  private:
 8  public:
 9      int top() {
13      void push(int key) {
17      void pop() {
24      bool isempty() {
27  };
28
29  int main() {
30      MinHeap heap;
31
32      heap.push(7);
33      heap.push(1);
34      heap.push(5);
35
36      while (!heap.isempty()) {
37          cout << heap.top() << " ";
38          heap.pop();
39      }
```

Your turn: Think for 10 min (for each)

- We learned to code a binary tree based on pointers
- Complete binary trees can be represented using arrays in an interesting. How?
  - Recall how many nodes per level
- Assume we have this min heap: how to insert value (5)?
  - Tip: Add it in the next available node (left of 19), then **fix** the tree branch!



"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."