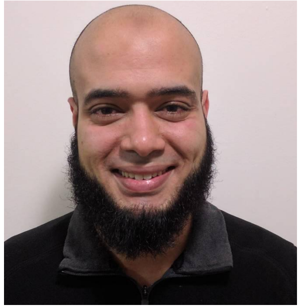
Data Structures

Heap Deletion

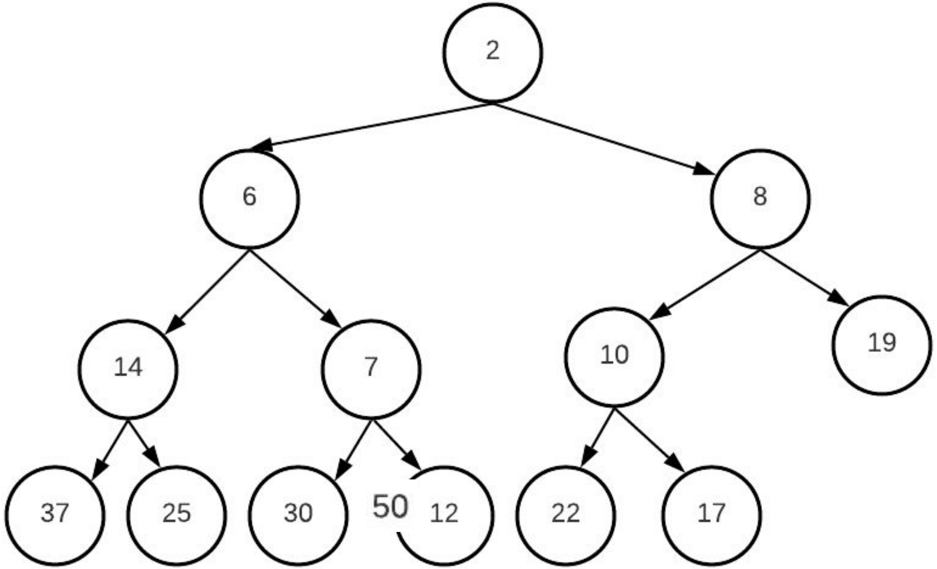
Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

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



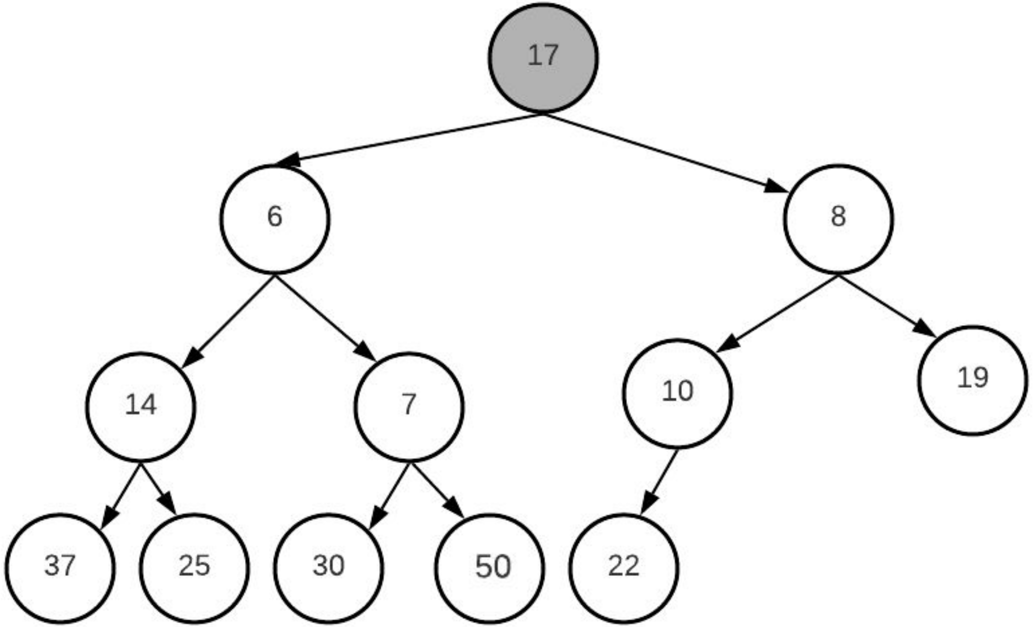
Let’s delete the minimum

- We want to remove the minimum element, update the tree, AND maintain our heap
- Root = min, which is easy to get
- Once again, let's follow an update and fix approach
- We will take the final node in our heap/array - and (temporarily) make it our root
  - Then fix the tree
  - How can we place 17 in its correct location?
  - Think for 10 minutes



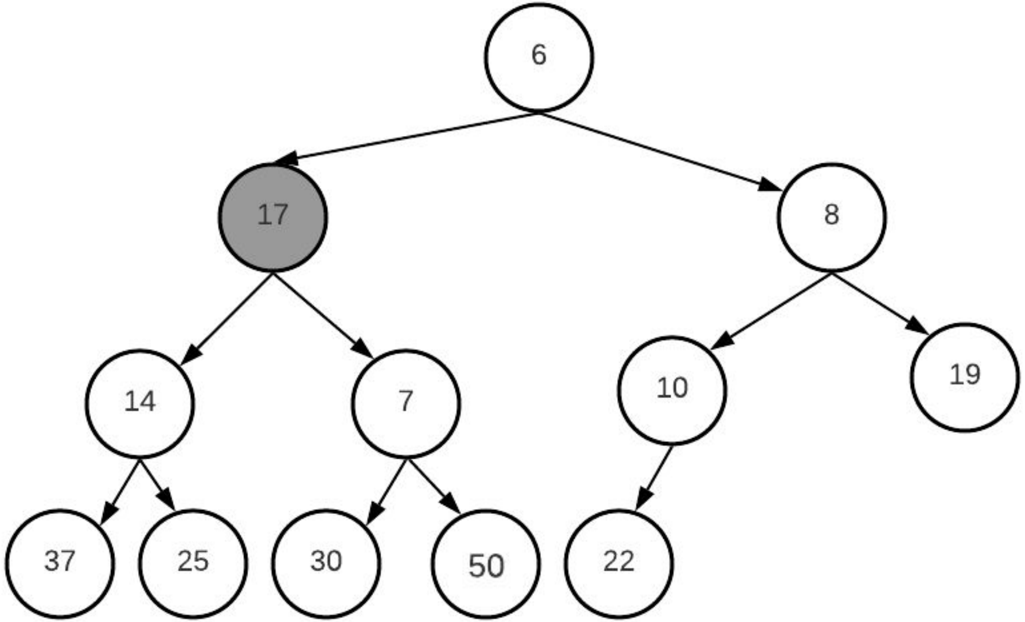
Fix the tree

- We want to move 17 to correct location
- Think level by level
- 17 is the parent to both 6 and 8
- How can we fix it?
- The new root MUST be the min among these child nodes
- In other words, swap 17 with the minimum child (6)



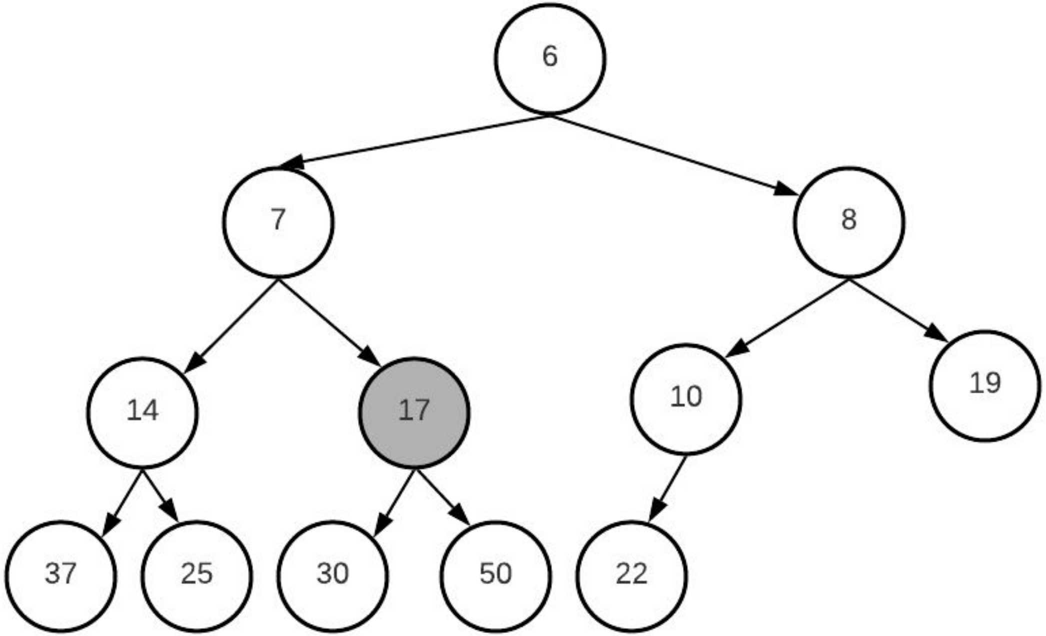
Fix the tree

- Now, having been adjusted once, 17 is the parent node to both 14 and 7
- 17 is a larger value! It can't be the parent to either node!
- min(7, 14) should be the new parent
- Swap 17 with 7



Fix the tree

- After a further adjustment, 17 is now the parent node to both 30 and 50
- This time 17 is in a perfect position: We can **stop**
- This is called **heapify-down**
- In this process, we take a value that is misplaced (and therefore, invalidates our heap), and move it down into the correct place!



“Acquire knowledge and impart it to the people.”

“Seek knowledge from the Cradle to the Grave.”