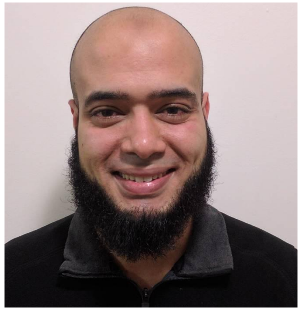
Data Structures

Minimum & Successor 3

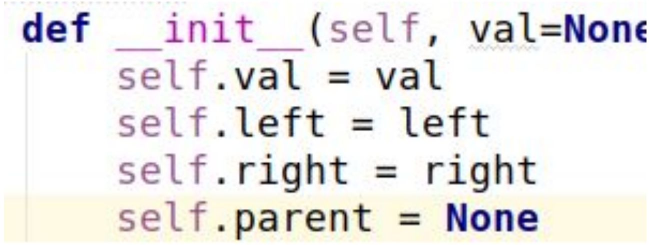
Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

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



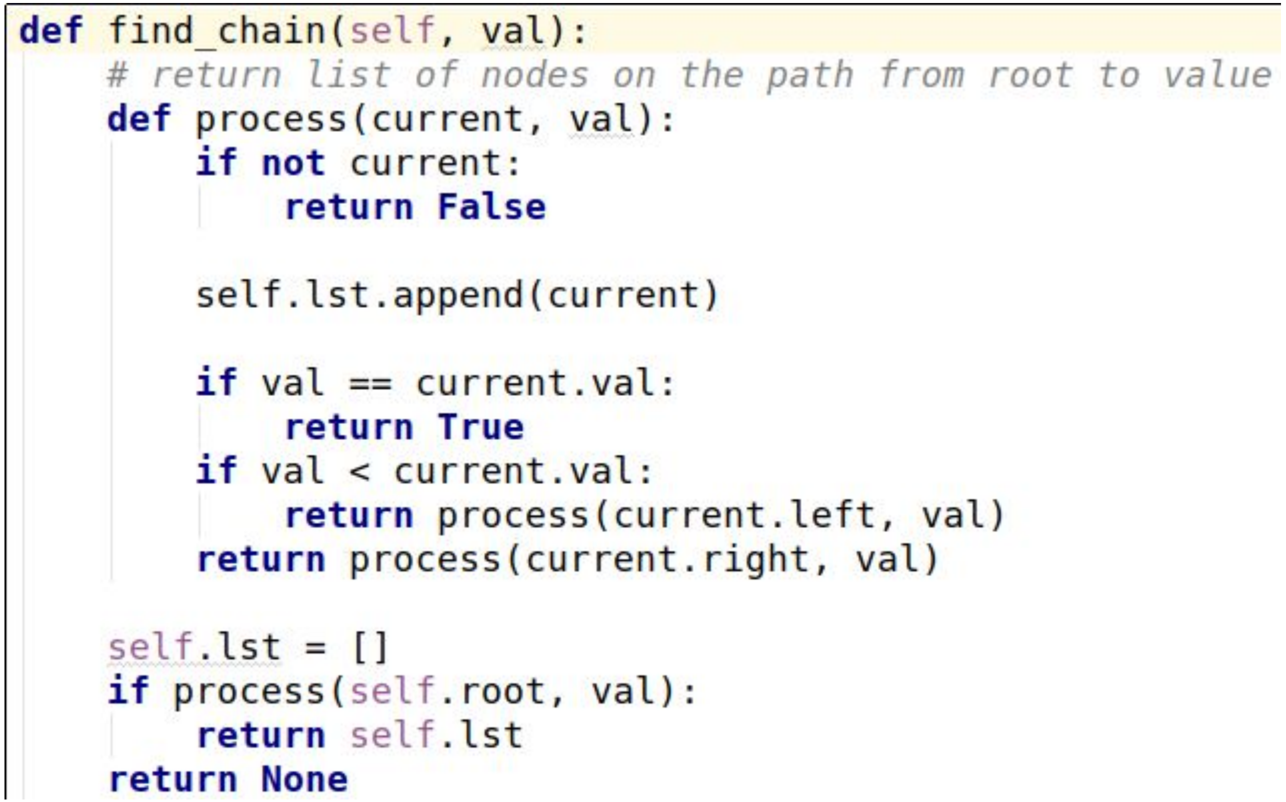
Implementation

- Now we need to keep going up in the tree!
- However, we don't have an 'up' or 'parent' node!
- 2 approaches
  - 1) Add parent node
    - You have to maintain it in insertion & deletion of nodes
  - 2) Get the ancestors nodes from root to target



```
def __init__(self, val=None
    self.val = val
    self.left = left
    self.right = right
    self.parent = None
```

Find Chain of nodes from root to value

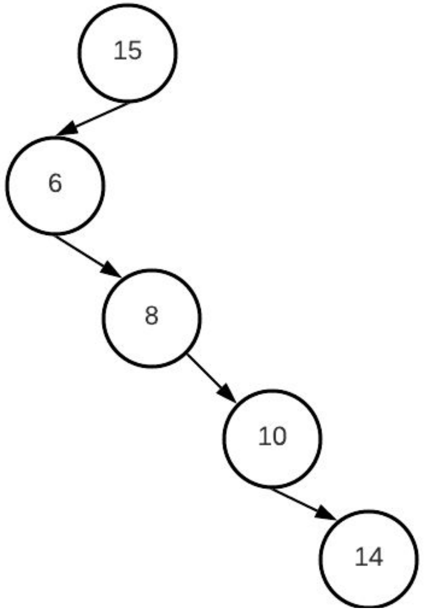


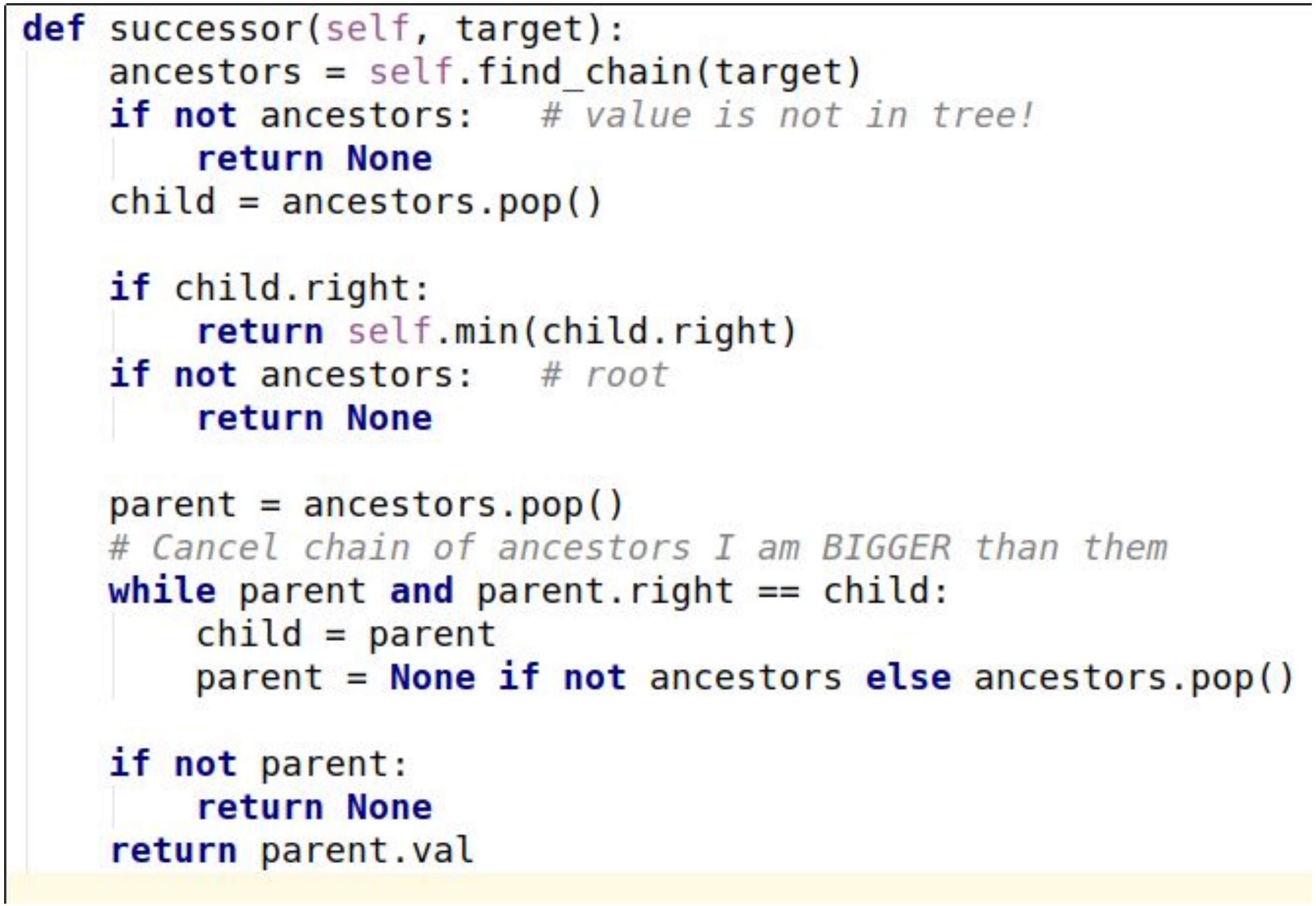
```python
def find_chain(self, val):
    # return list of nodes on the path from root to value
    def process(current, val):
        if not current:
            return False

        self.lst.append(current)

        if val == current.val:
            return True
        if val < current.val:
            return process(current.left, val)
        return process(current.right, val)

    self.lst = []
    if process(self.root, val):
        return self.lst
    return None
```



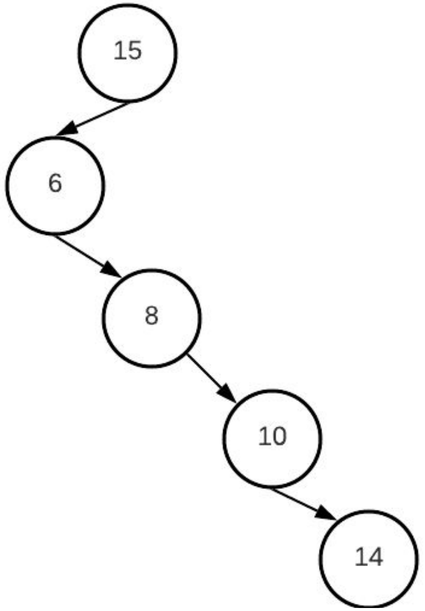


```python
def successor(self, target):
    ancestors = self.find_chain(target)
    if not ancestors:   # value is not in tree!
        return None
    child = ancestors.pop()

    if child.right:
        return self.min(child.right)
    if not ancestors:   # root
        return None

    parent = ancestors.pop()
    # Cancel chain of ancestors I am BIGGER than them
    while parent and parent.right == child:
        child = parent
        parent = None if not ancestors else ancestors.pop()

    if not parent:
        return None
    return parent.val
```



Max and Predecessor

- To get the Max node, we just keep going right!
- The predecessor is simply the opposite of the successor!
    - Find node y that is the largest y < x
- If the inorder traversal is 10 20 30 40 50
    - Node 30's successor is 40 (immediately after)
    - Node 30's predecessor is 20 (directly before)
- Note: to find successor/predecessor of a node, the query must be for a value in the tree

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."