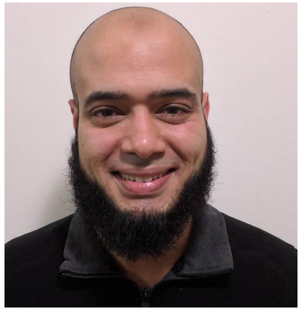
Data Structures

BST Homework 2

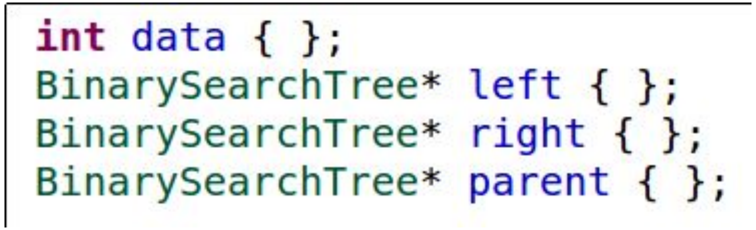
Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

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



Problem #1: Parent Link

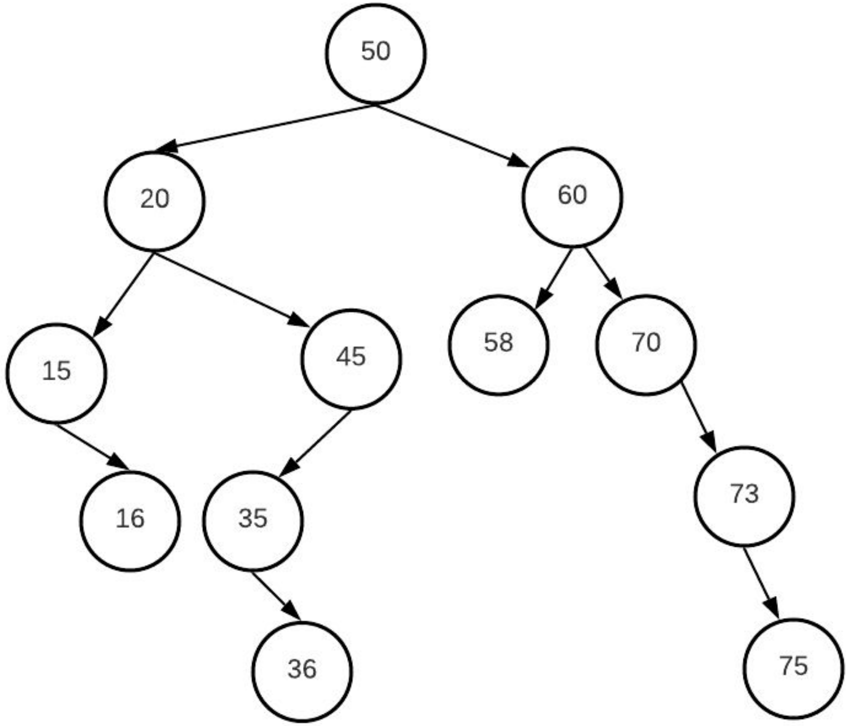
- In our implementation, we followed the no-parent link approach
- Rewrite the BST code for both insert and successor where your structure now have a parent link



```
int data { };
BinarySearchTree* left { };
BinarySearchTree* right { };
BinarySearchTree* parent { };
```

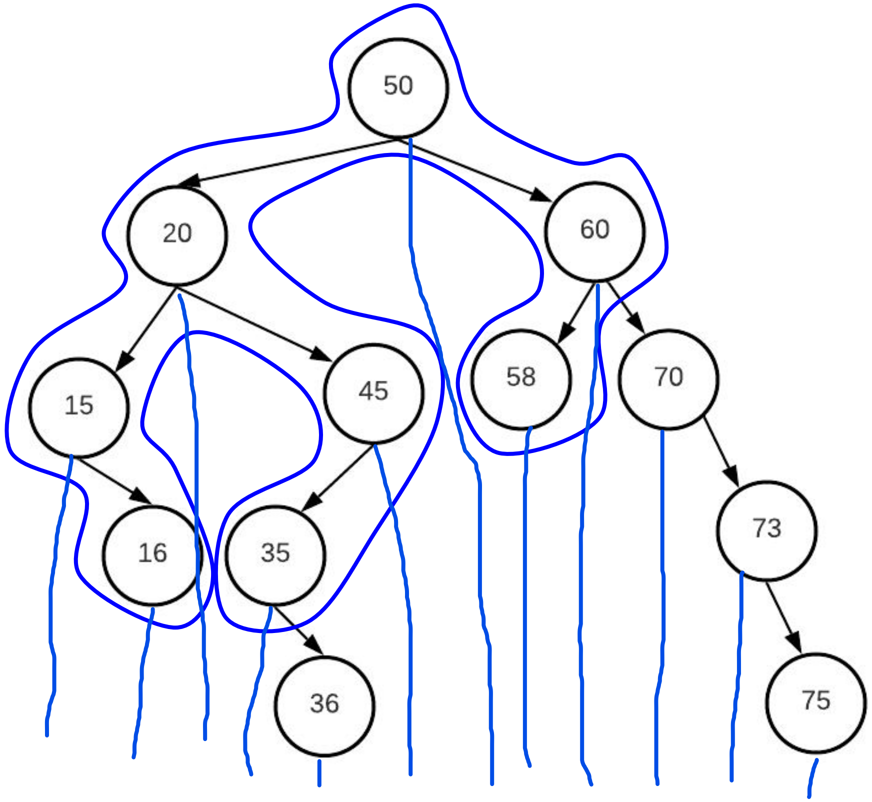
Problem #2: Queries of ancestors

- Assume we have deque<int> q that has **sorted** items to find their successors.
    - Input ⇒ {15, 20, 58}
    - Successors ⇒ {16, 35, 60}



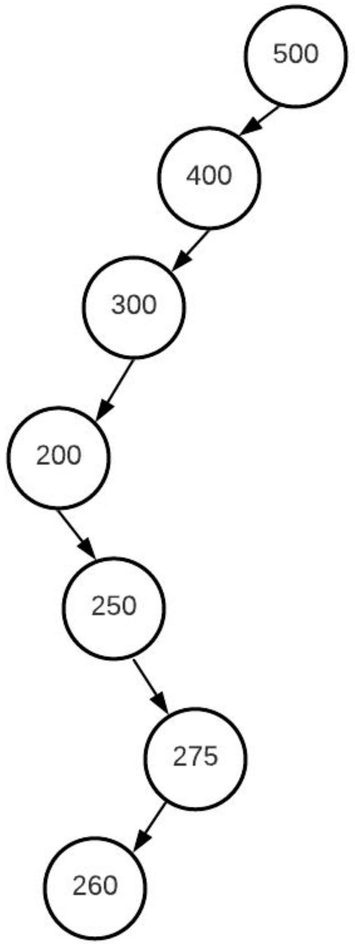
Problem #2: Queries of ancestors

- Develop a function that finds all of them such:
  - You don’t do complete traversal. **Stop as early as possible** similar to lecture
    - {15, 20, 58}
  - Don’t use parent pointer
  - Don’t get chain of ancestors like lecture
- Tip:
  - We know inorder traversal already moves toward our successor. Why don’t we just catch it once we find it
  - Code is a modified search function



Problem #3: Is degenerate tree

- bool is_degenerate(vector<int> &preorder)
- Given a preorder of BST of N nodes, return true of it is degenerate tree of height N-1.
    - All values are distinct and in range [1, 1000]
- Do it on O(n).
    - 25, 8, 11, 13, 12 ⇒ True
    - 100, 70, 101 ⇒ False
    - 100, 70, 60, 75 ⇒ False
    - 100, 70, 60, 65 ⇒ True
    - 9, 8, 7, 6, 5, 4, 3 ⇒ True
    - 500, 400, 300, 200 , 250 , 275, 260 ⇒ True
    - 500, 400, 300, 200 , 250 , 275, 260, 280 ⇒ False



"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."