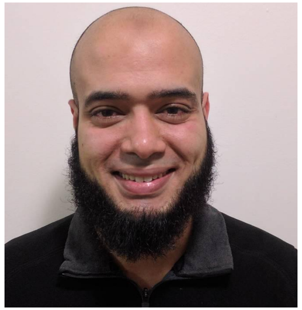
Data Structures

BST Homework 4

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)

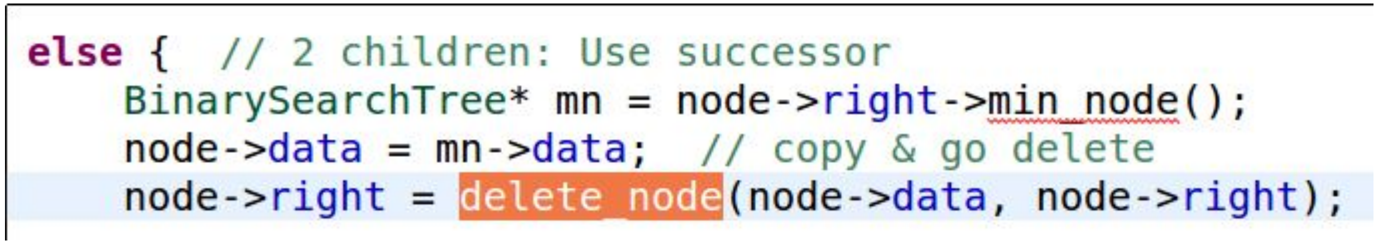


Problem #1: Node Deletion using predecessor

- In the lecture’s code, we used the successor to handle the 2 children case
- Use instead the predecessor
- BinarySearchTree* delete_node_p(int target, BinarySearchTree* node)

Problem #2: Node Deletion without recursion

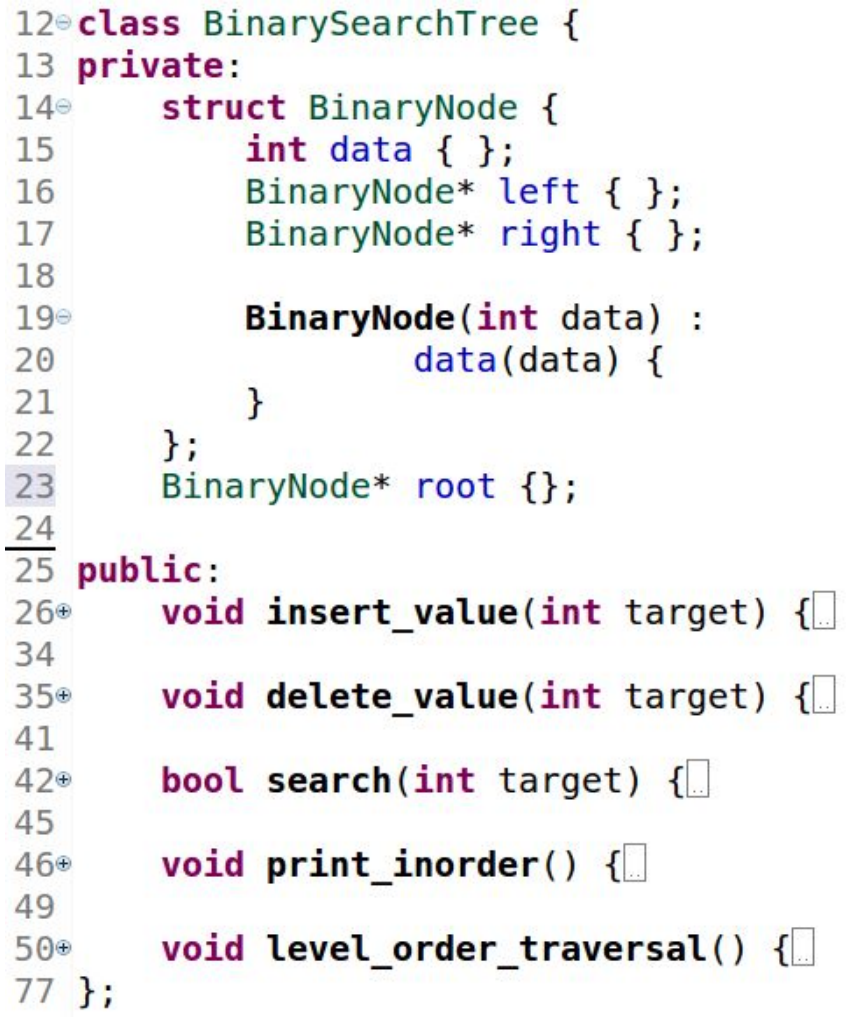
- In lecture’s code, we utilized the recursion to easily delete the successor node
- Change the code to do this deletion locally without another call
- Sketch carefully the different cases.
- Don’t complicate things. Keep it simple



```
else {  // 2 children: Use successor
    BinarySearchTree* mn = node->right->min_node();
    node->data = mn->data;  // copy & go delete
    node->right = delete_node(node->data, node->right);
```

Problem #3: Rewriting Binary Search Tree

- Let’s rewrite the BST to have an internal nodes
  - Don’t change the BinaryNode code
  - Solve the 2 issued in deletion by using ***root**
- Be careful with the root when whole tree can be deleted or the root itself could be deleted
- Test well



```cpp
class BinarySearchTree {
private:
	struct BinaryNode {
		int data { };
		BinaryNode* left { };
		BinaryNode* right { };

		BinaryNode(int data) :
				data(data) {
		}
	};
	BinaryNode* root {};

public:
	void insert_value(int target) {

	void delete_value(int target) {

	bool search(int target) {

	void print_inorder() {

	void level_order_traversal() {
};
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."