Wiederholung BST Traversal

Fuer BSTs bietet sich eine rekursive Implementierung an

Es gibt mehrere Arten einen Baum zu Traversieren:

Depth-First-Search:

1) Inorder-/Reverse-Inorder-Traversal
links - knoten - rechts

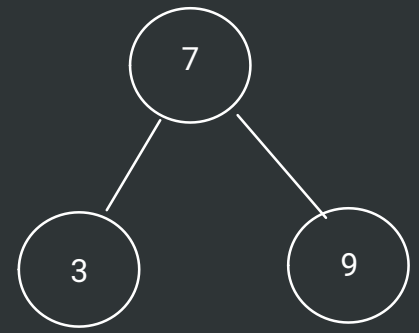
Ausgabe: 3, 7, 9 bzw. 9, 7, 3

2) Postorder-Traversal
links - rechts - knoten

Ausgabe: 3, 9, 7

3) Preorder-Traversal
knoten - links - rechts

Ausgabe: 7, 3, 9



Breadth-First Search traversiert stattdessen zuerst die Ebenen in horizontaler Reihenfolge

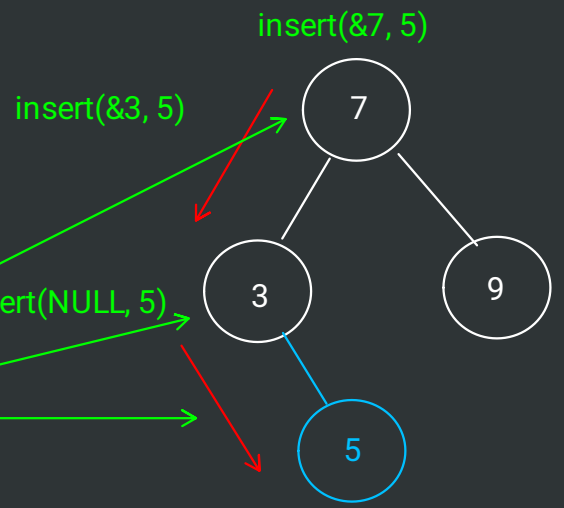
Post/Preorder haengen von der Einfuegungsreihenfolge ab, Inorder dagegen von der Ordnung der Werte

Einfuegen von einzigartigen Knoten im BST (keine Kollision)

Prototyp zum node alloziieren: struct node* new_node(int val);

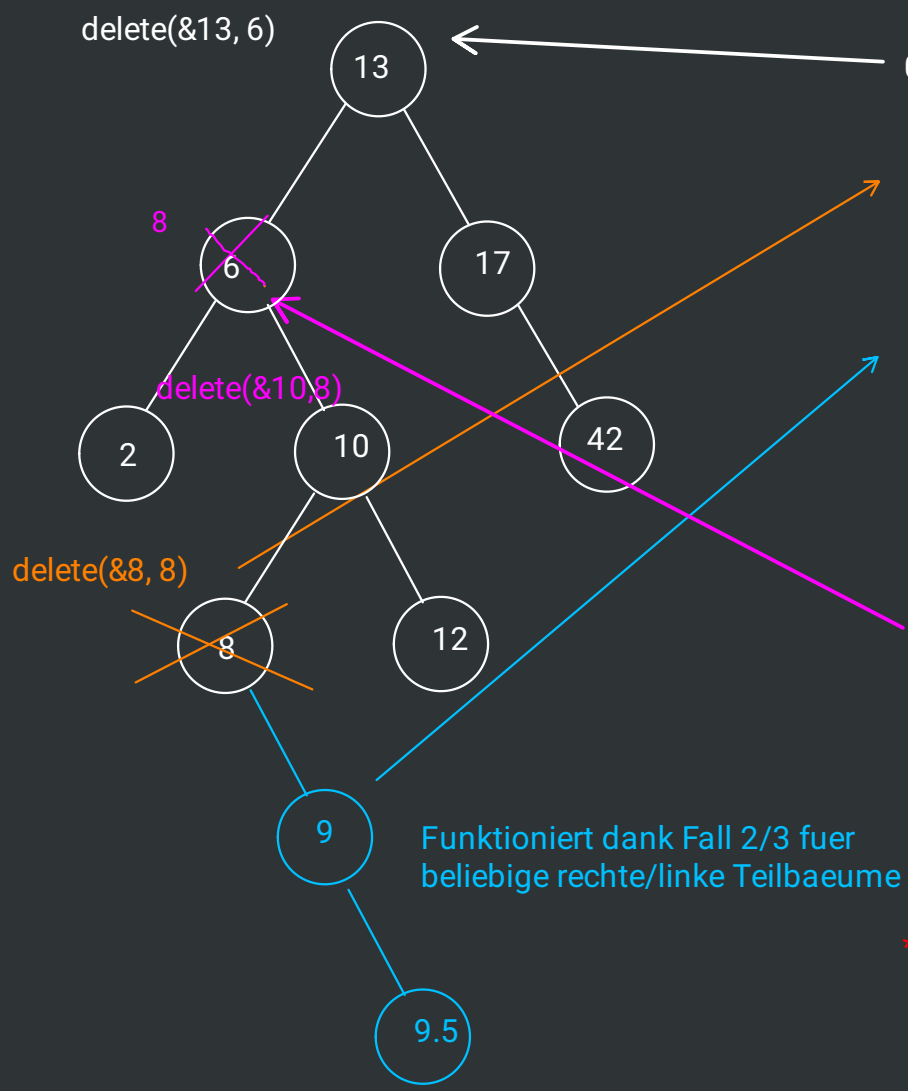
API:
Parameter:
root: Zeiger auf Wurzelknoten
val: Beliebiges Datum (in unserem Fall int)
Rueckgabewert:
Der Zeiger auf den (ggf. aktualisierten) Wurzelknoten

```
struct node* insert(struct node* root, int val) {
 if (root == NULL) {
  return new_node(val);
 } else if ( val > root->val  ) {
  root->right = insert(root->right, val);
 } else if ( val < root->val ) {
  root->left = insert(root->left, val);
 }
 return root;
}
```



Hausuebung: Schreiben Sie eine iterative Implementierung von insert()

Loeschen von Elementen im BST



0. Rekursiv absteigen zum gesuchten Wert, bis val == node->val

1. Fall (z.l. Knoten hat keine Kinder)
1.1 Speicher freigeben free(node)
1.2 Die Referenz invalidieren (node = NULL)

2. Fall (z.l Knoten hat nur ein rechtes Kind)
2.1 Merken der rechten Teilbaum Referenz
2.2 Freigeben des z.l. Knotens (z.B. 17)
2.3 Anhaengen des rechten Teilbaums an den Elternknoten

3. Fall (z.l. Knoten hat nur ein linkes Kind)
siehe analog zu (2) nur fuer links

4. Fall (z.l. Knoten hat zwei Kindknoten)
4.1 finde den "inorder successor" im rechten Teilbaum***
4.2 Ueberschreibe des z.l. Knoten mit dem Wert des inorder successors
4.3 Rufe delete_node erneut auf mit dem rechten Teilbaum und dem gefundenen inorder successor

Funktioniert dank Fall 2/3 fuer beliebige rechte/linke Teilbaeume

*** Statt des inorder successors kann ebenfalls der inorder precursor aus dem linken Teilbaum verwendet werden mit anschl. delete im linken Teilbaum