**AD Skizze Aufgabe 4**

**Team**: <3>, <Patrick Steinhauer, Jan Dennis Bartels>

**Aufgabenaufteilung**:

Die Aufgaben Wurden zusammen bearbeitet und erstellt. Jeder hat sich was für die Skizze Überlegt, und dann wurde es zusammengefügt.

**Quellenangaben**: VorlesungsFolien Teilweise Wikipedia

**Begründung für Codeübernahme**:

**Bearbeitungszeitraum**: Ca 5 Stunde Bisher

**Aktueller Stand**: Skizze + Grundgerüst der AVL Bäume

**Änderungen in der Skizze**:

**Skizze**:

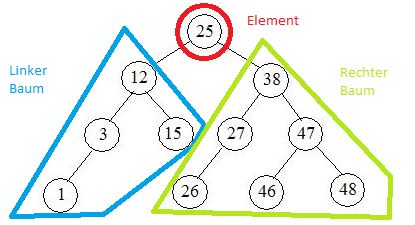
**Generelles:**

**Balance :**

* Die Balance muss nach jedem einfügen, löschen, rotieren neu berechnet werden.
* Die Balance errechnet sich aus :
  + Knotenanzahl rechter Teilbaum – Knotenanzahl linker Teilbaum
  + Bei einem Blatt zieht man die Höhe von sich selbst ab, also 1-1
  + Die Wurzel immer mitzählen

**Pseuso Codes:**

Aufbau eines AVL Baumes: {LinkerUnterBaum, {Element, Höhe}, RechterUnterbaum}



**einfügen / add**

add(Baum, Value)

{

Wenn (Baum == empty){

Baum = Value

}

Wenn (Baum.Elem < Value && Baum.Rechts == empty) {

Baum.Rechts = Value, rotate(Baum)

}

Wenn(Baum.Elem <Value && Baum.Rechts != empty) {

add(Baum.Rechts, Value)

}

Wenn (Baum.Elem > Value && Baum.Links == empty) {

Baum.Links = Value, rotate(Baum)

}

Wenn(Baum.Elem >Value && Baum.Links != empty) {

add(Baum.Links, Value)

}

}

**Löschen :**

**Generelles:**

* Nach dem eine Element gelöscht wurde muss die höhe des Baumes neu berechnet werden.

delete(Tree,Elem) {

if(Tree.Value == Elem) {

if (Tree.Right == empty) {

Tree.Value = Tree.Left, Tree.Left = NULL;

}

else if (Tree.left == empty) {

Tree.Value = Tree.right, Tree.right = NULL;

} else {

newElem = findSmallest(Tree.right);

Tree.Value = newElem;

Tree.right = delete(Tree.right,newElem);

rotate(Tree);

}

} else if (Tree.Value > ELem) {

delete(Tree.Right,Elem);

rotate(Tree);

} else {

delete(Tree.Left,Elem);

rotate(Tree);

}

}

Für alle rotate Funktionen eine Funktion machen die eine Unterscheidung für alle Fälle macht.

**RechtsRotation :**

**Generelles :**

* Eine Rechtsrotation wird durchgeführt, wenn
  + Die Höhe des oberen Knotens -2 ist
  + Die Höhe des unteren Knotens -1 ist

Wenn also Baum.Element == -2 und unterbaum.Element == -1dann

rotateRight(Baum) {

}

**LinksRotation :**

**Generelles :**

* Eine Linksrotation wird durchgeführt, wenn
  + Die Höhe des oberen Knotens +2 ist
  + Die Höhe des unteren Knotens + 1 ist

rotateLeft(Baum) {

}

**LinksRechtsRotation (Doppelrotation rechts) :**

**Generelles:**

* Eine Linksrechtsrotation wird durchgeführt, wenn
  + Die Höhe des oberen Knotens -2 ist
  + Die Höhe des unteren Knotens +1 ist

rotateLeftRight(Baum) {

}

**RechtsLinksRotation ( Doppelrotation Links)**

**Generelles :**

* Eine RechtsLinksRotation wird durchgeführt, wenn
  + Die Höhe des oberen Knotens +2 ist
  + Die Höhe des unteren Knotens -1 ist

rotateRightLeft(Baum) {

}