



### Einfügen:

- Blatt suchen, in das eingefügt werden soll.
  - \* Wenn beim Durchlauf nach unten ein Knoten mit 4 Söhnen (3 Schlüssel) gefunden wird, diesen splitten (mittleren Schlüssel nach oben reichen, linker und rechter Schlüssel in eigenen Knoten).
- Schlüssel einfügen (Blatt)
  - \* Wenn 1 oder 2 Schlüssel => einfach einfügen
  - \* Wenn bereits 3 Schlüssel => erst nach dem Splitten einfügen.

```
private int treeToVine(Node tail) {
    Node rest = tail.right;
    int n = 0;
    while(rest != null)
        if (rest.left == null) { // move tail down
            tail = rest;
            rest = rest.right;
            n ++;
        }
        else { // rotate
            Node p = rest.left;
            rest.left = p.right;
            p.right = rest;
            rest = p;
            tail.right = p;
        }
    return n;
}
```

```
private void vineToTree(Node pseudoRoot, int n){
    int k = 1;
    while (k <= n + 1)
        k *= 2; // k = 2 * 2^floor(log(n+1))
    int i = k / 2 - 1; // i = 2^floor(log(n+1)) - 1
    compress(pseudoRoot, n - i);
    while (i > 1) {
        i /= 2;
        compress(pseudoRoot, i);
    }
}
```

```
private void compress (Node p, int n){
    for (int i = 1; i <= n; i++) {
        Node son = p.right;
        p.right = son.right;
        p = p.right;
        son.right = p.left;
        p.left = son;
    }
}
```

```
public void balance() {
    Node pseudoRoot = new Node(); // Dummy node
    pseudoRoot.right = root;
    int n = treeToVine(pseudoRoot);
    vineToTree(pseudoRoot, n);
    root = pseudoRoot.right;
}
```