Pseudocodes der RB-Tree-Funktionen nach Cormen, Algorithms, MIT Press 2nd ed, 2001

RB-Insert(T, z) und *RB-Insert-Fixup(T, z)*

RB-Delete(T, z) und *RB-Delte-Fixup(T, x)*

```
RB-INSERT(T, z)
 1  y ← nil[T]
 2  x ← root[T]
 3  while x ≠ nil[T]
 4        do y ← x
 5           if key[z] < key[x]
 6             then x ← left[x]
 7             else x ← right[x]
 8  p[z] ← y
 9  if y = nil[T]
10     then root[T] ← z
11     else if key[z] < key[y]
12             then left[y] ← z
13             else right[y] ← z
14  left[z] ← nil[T]
15  right[z] ← nil[T]
16  color[z] ← RED
17  RB-INSERT-FIXUP(T, z)
```

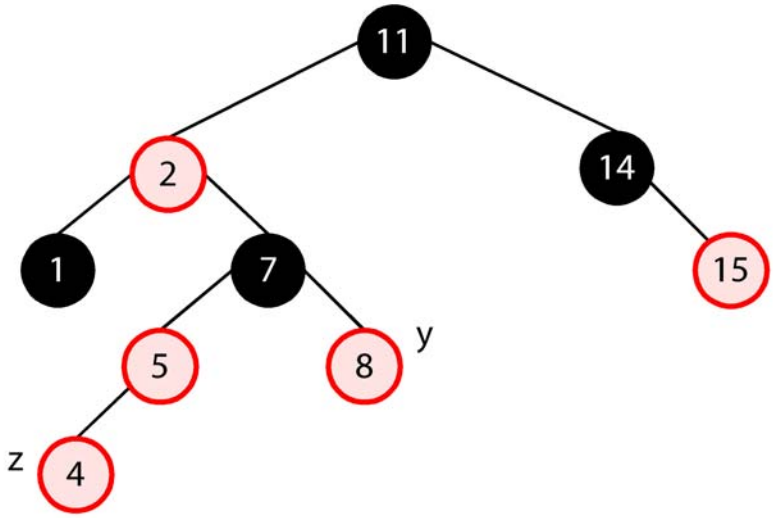
```
RB-INSERT-FIXUP(T, z)
 1  while color[p[z]] = RED
 2     do if p[z] = left[p[p[z]]]
 3           then y ← right[p[p[z]]]
 4                if color[y] = RED
 5                  then color[p[z]] ← BLACK                ▷ Case 1
 6                       color[y] ← BLACK                   ▷ Case 1
 7                       color[p[p[z]]] ← RED               ▷ Case 1
 8                       z ← p[p[z]]                        ▷ Case 1
 9                  else if z = right[p[z]]
10                         then z ← p[z]                    ▷ Case 2
11                              LEFT-ROTATE(T, z)           ▷ Case 2
12                       color[p[z]] ← BLACK                ▷ Case 3
13                       color[p[p[z]]] ← RED               ▷ Case 3
14                       RIGHT-ROTATE(T, p[p[z]])           ▷ Case 3
15           else (same as then clause
                       with "right" and "left" exchanged)
16  color[root[T]] ← BLACK
```

```
RB-DELETE(T, z)
 1  if left[z] = nil[T] or right[z] = nil[T]
 2     then y ← z
 3     else y ← TREE-SUCCESSOR(z)
 4  if left[y] ≠ nil[T]
 5     then x ← left[y]
 6     else x ← right[y]
 7  p[x] ← p[y]
 8  if p[y] = nil[T]
 9     then root[T] ← x
10     else if y = left[p[y]]
11             then left[p[y]] ← x
12             else right[p[y]] ← x
13  if y ≠ z
14     then key[z] ← key[y]
15          copy y's satellite data into z
16  if color[y] = BLACK
17     then RB-DELETE-FIXUP(T, x)
18  return y
```

```
RB-DELETE-FIXUP(T, x)
 1  while x ≠ root[T] and color[x] = BLACK
 2        do if x = left[p[x]]
 3              then w ← right[p[x]]
 4                   if color[w] = RED
 5                     then color[w] ← BLACK                          ▷ Case 1
 6                          color[p[x]] ← RED                         ▷ Case 1
 7                          LEFT-ROTATE(T, p[x])                      ▷ Case 1
 8                          w ← right[p[x]]                           ▷ Case 1
 9                   if color[left[w]] = BLACK and color[right[w]] = BLACK
10                     then color[w] ← RED                            ▷ Case 2
11                          x ← p[x]                                  ▷ Case 2
12                     else if color[right[w]] = BLACK
13                             then color[left[w]] ← BLACK            ▷ Case 3
14                                  color[w] ← RED                    ▷ Case 3
15                                  RIGHT-ROTATE(T, w)                ▷ Case 3
16                                  w ← right[p[x]]                   ▷ Case 3
17                          color[w] ← color[p[x]]                    ▷ Case 4
18                          color[p[x]] ← BLACK                       ▷ Case 4
19                          color[right[w]] ← BLACK                   ▷ Case 4
20                          LEFT-ROTATE(T, p[x])                      ▷ Case 4
21                          x ← root[T]                               ▷ Case 4
22              else (same as then clause with "right" and "left" exchanged)
23  color[x] ← BLACK
```

Die drei Fälle von RB-Insert-Fixup(T, z):

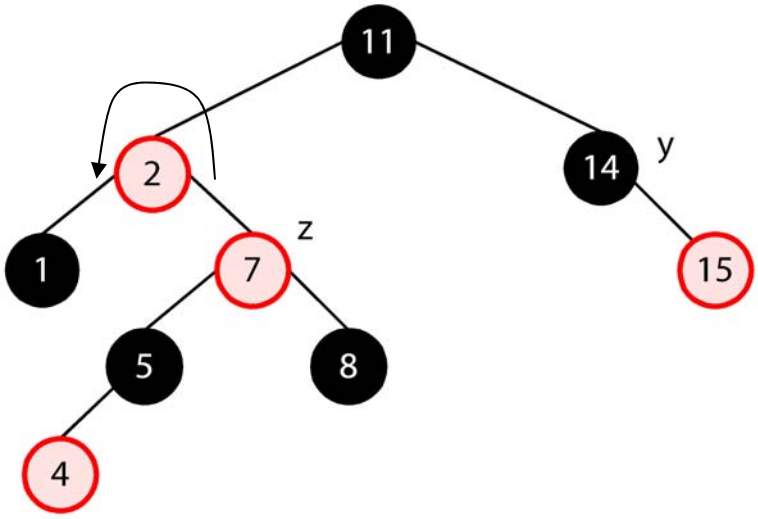
Fall 1: z‘s Onkel y ist rot



→ Umfärben und z’s Großvater wird neues z.

Fall 2: z‘s Onkel y ist schwarz und z ist rechtes Kind von p[z]

→ LINKS-ROTATION bei p[z]

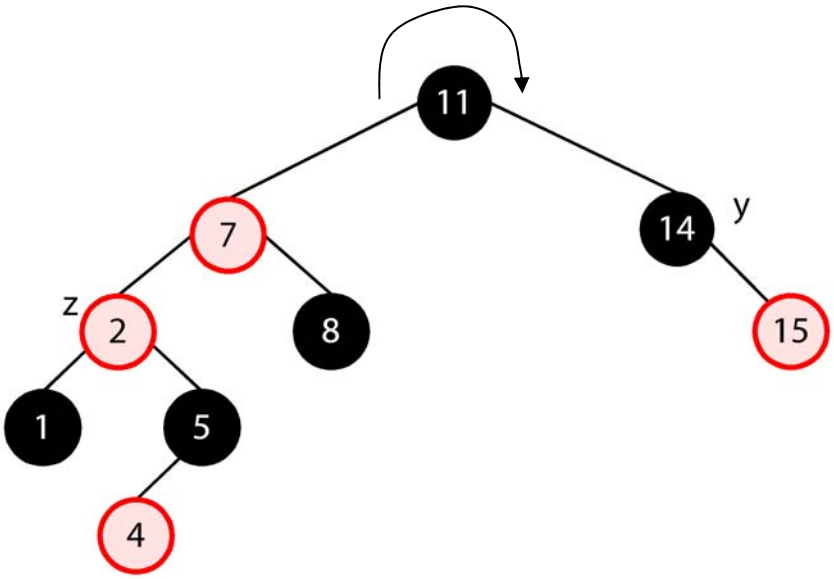


→ anschließende RECHTS-ROTATION

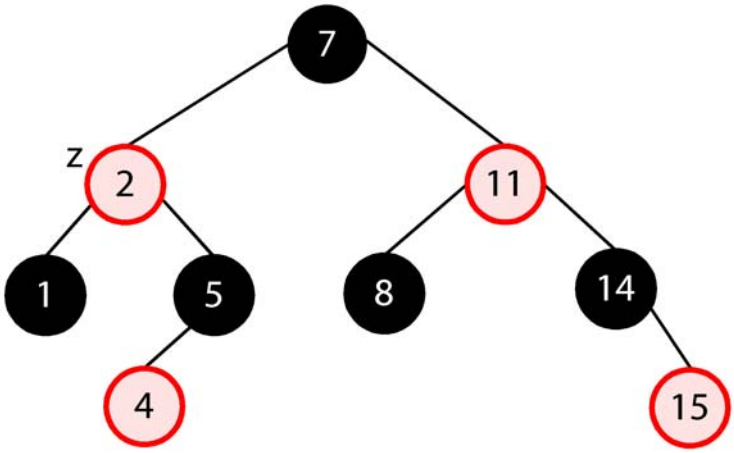
Beachte: Vor der LINKS-ROTATION wird z ← p[z]. Darum wird nach der LINKS-ROTATION z linkes Kind von p[z]. Automatischer Übergang zu Fall 3.

Fall 3: z‘s Onkel y ist schwarz und z ist linkes Kind von p[z]

→ RECHTS-ROTATION



→



Fertig!