ASD, 2022.05.05

Drzewa AVL (Adelson, Velsky, Landis)

Zalety drzew BST

- mamy średni czas działania operacji stałych $O(\log_2 n)$, gdzie n – wielkość drzewa
- łatwa implementacja.

Wada

nie mamy gwarancji czasu $O(\log_2 n)$. (PRZY) niezrównoważonych drzewach możemy mieć koszt liniowy.

Remedium

drzewa AVL.

drzewa te gwarantują pesymistyczny czas $O(\log_2 n)$ operacji słownikowych

insert

find

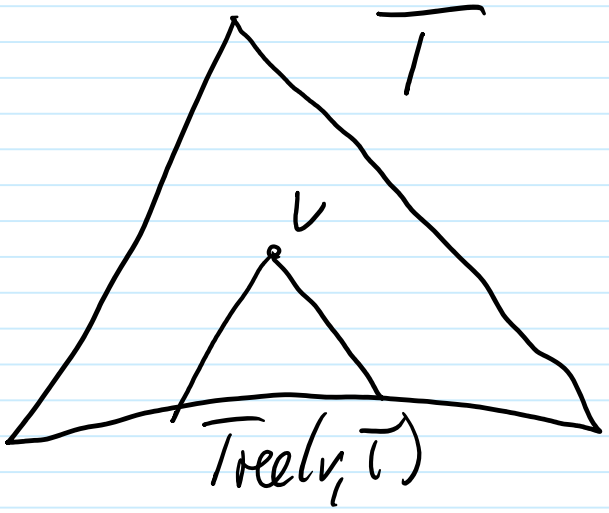
delete.

Definicje

Niech T - drzewo.

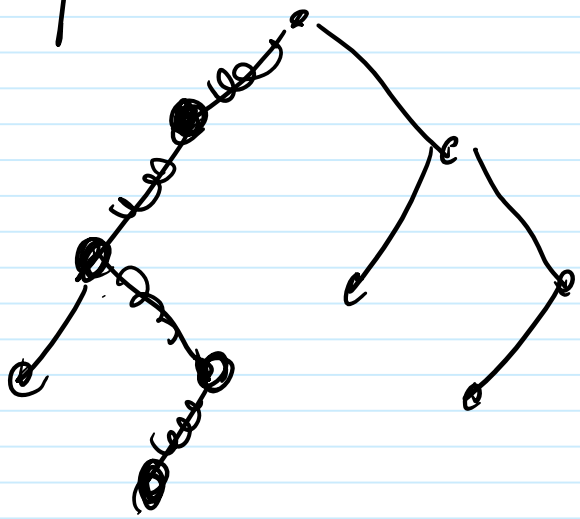
Niech $v \in T$.

$Tree(v, T)$ - poddrzewo T o korzeniu v.



Wysokość drzewa to długość najdłuższej ścieżki od korzenia do liścia.

T



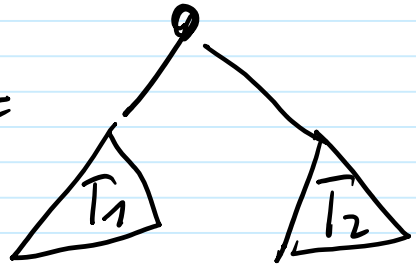
height(T) = 4

height( ) = 1

height ( ) = 0

height (∅) = -1 (przyjmujemy, żeby dobrze współgrało z rekurencją).

Fakt

Niech $T =$ 

, $height(T) = \max(height(T_1), height(T_2)) + 1$.

Wysokość wierzchołka $v \in T$ w drzewie T, to wysokość $Tree(v, T)$.

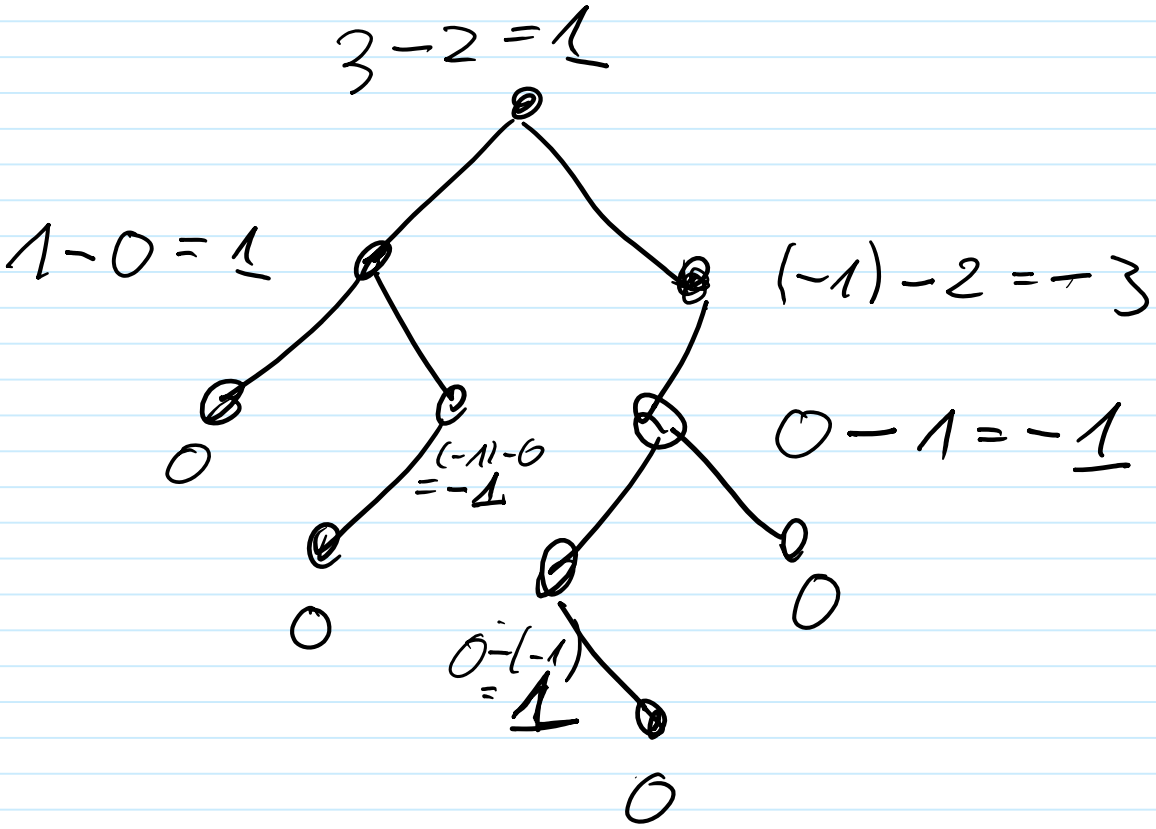
Fakt

Wysokość v w T to długość najdłuższej ścieżki od v do liścia w T.

Def Wyważenie wierzchołka v w drzewie T to różnica wysokości prawego i lewego podrzewa v

$balance(v) = height(v\text{->}right, T) - height(v\text{->}left, T)$

balance(v, T) = height(v->right, T) − height(v->left, T).



Def (drzewo AVL)

Niech T - binarnym drzewem BST.

T jest drzewem AVL jeśli dla każdego

$v \in T$

$balance(v) \in \{-1, 0, 1\}$.

Co zyskujemy pracując z drzewami AVL?

Niech $\{F_n\}_{n \in \mathbb{N}}$ ciąg Fibonacciego.

$F_0 = 1$

$F_1 = 1$

$F_{n+2} = F_{n+1} + F_n$, F_n rośnie wykładniczo wzgl. n

Fakt

Niech T będzie drzewem AVL o n wierzchołkach.
Wysokość T jest mniejsza bądź równa $1{,}45 \cdot (\log_2 n)$.

Dow

Lemat

Drzewo AVL o wysokości h ma przynajmniej F_h wierzchołków.

Dow

Niech T_h – ~~najmniejsze~~ (minimalne) (w sensie liczby wierzchołków) drzewo AVL o wysokości h.

"Pokażemy", że $\forall h$ $card(T_h) \geq F_{h+1}$

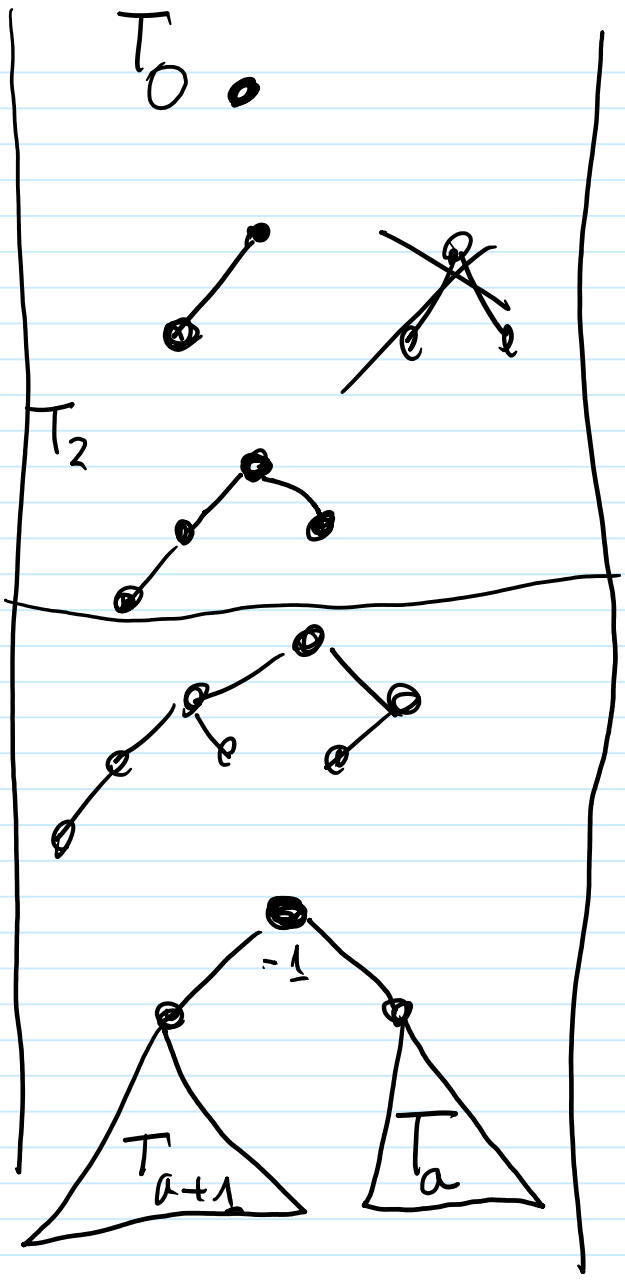
Pomagamy sobie

$h = 0$ T_0 • $card(T_h) = 1$

$= 1$ 2

$= 2$ 4

$= 3$ 7



$h = a + 2$

$card(T_{a+2}) = card(T_{a+1}) + card(T_a) + 1.$

Tatwo pokazać, że

$$\forall n \quad card(T_n) \geq F_n.$$

dla $n = 0, 1$ – na palcach

Krok indukcyjny z postaci minimalnego drzewa o wysokości $a+2$. ∎ Lemat.

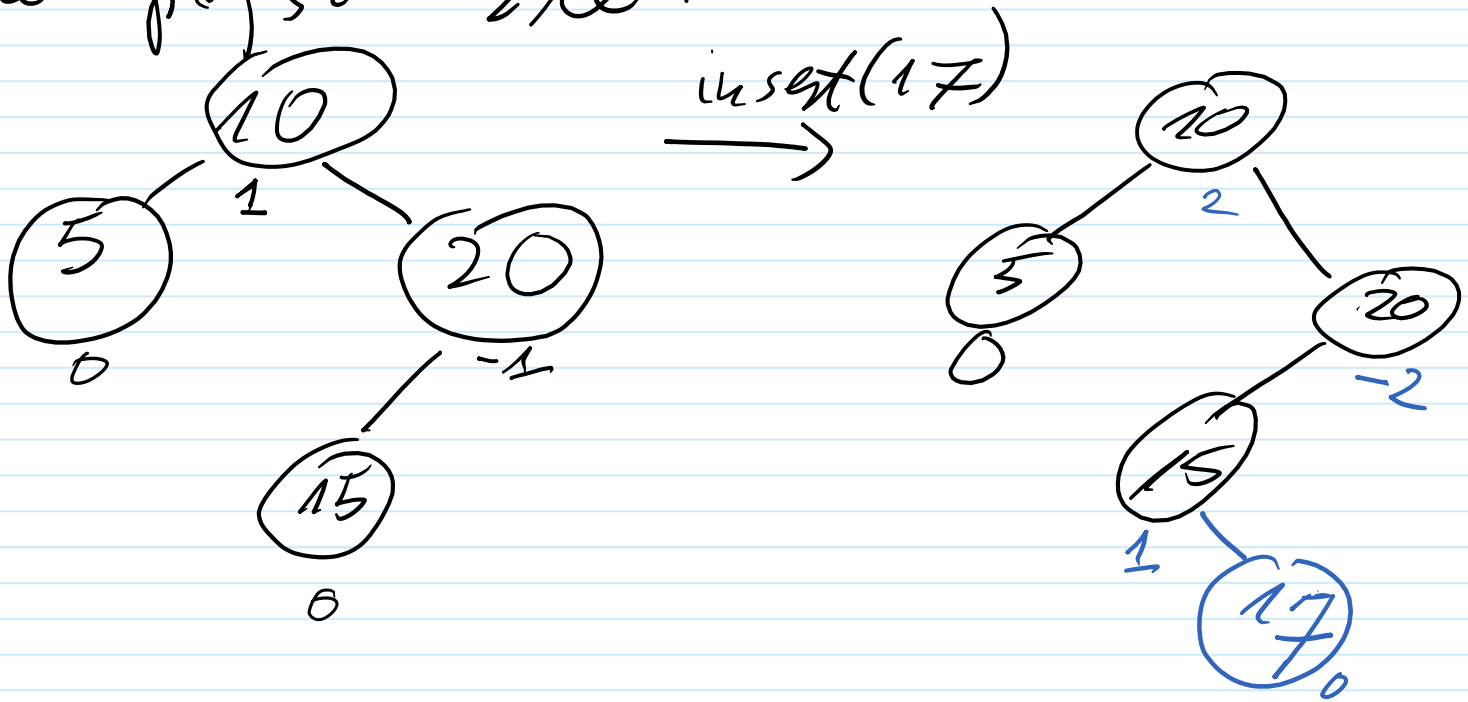
Mając lemat musimy umieć oszacować wielkość F_n.

$\varphi = \frac{1+\sqrt{5}}{2}$ $\quad \psi = 1 - \varphi = \frac{1-\sqrt{5}}{2}$, $\quad F_n = \frac{\varphi^n - \psi^n}{\sqrt{5}}$

$= \lfloor \frac{\varphi^n}{\sqrt{5}} + \frac{1}{2} \rfloor$, dla dużych n.

Problem: jak zaimplementować insert, delete tak, aby nie zaburzyły nam struktury drzewa AVL.

Co może pójść źle?



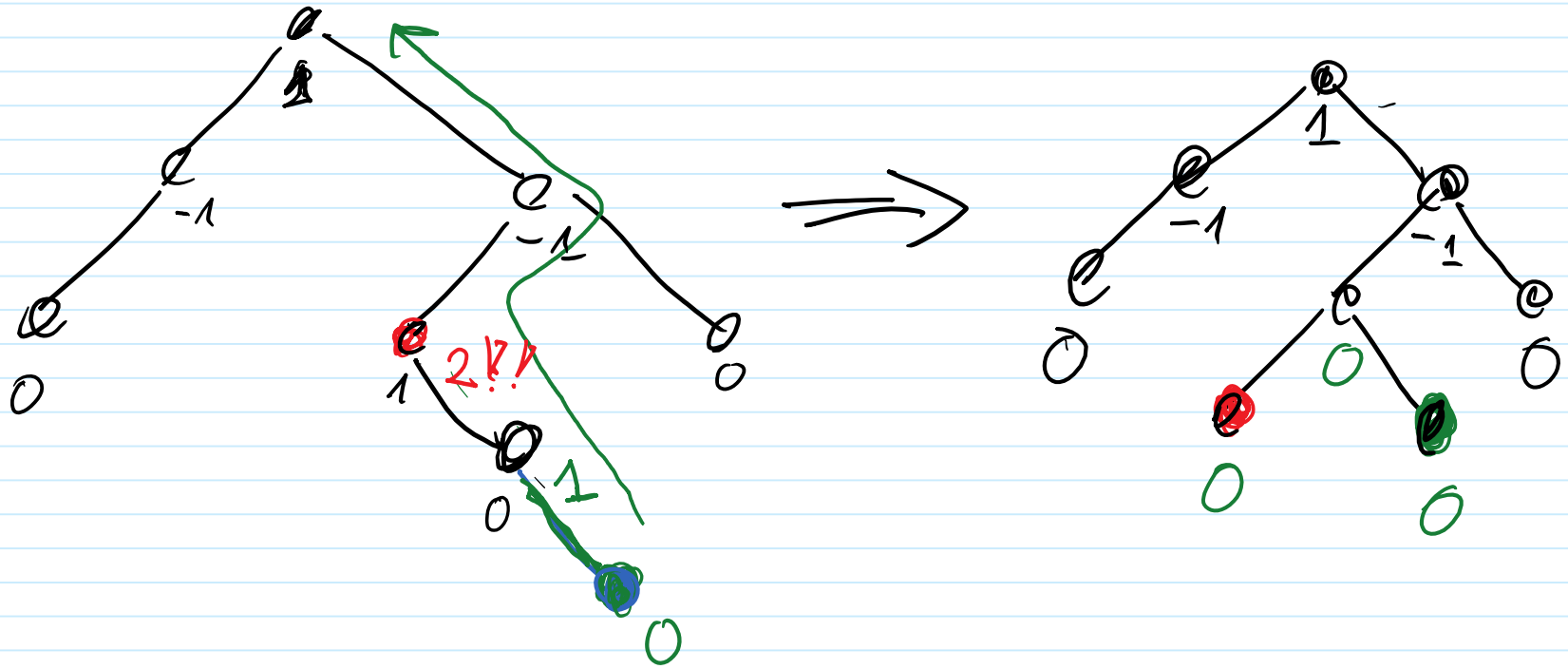
Analogiczny problem pojawia się przy usuwaniu elementu.

Rozwiązanie

Funkcje insert, delete działają dwuetapowo.

W pierwszej fazie działają jak ich odpowiedniki dla drzew BST.

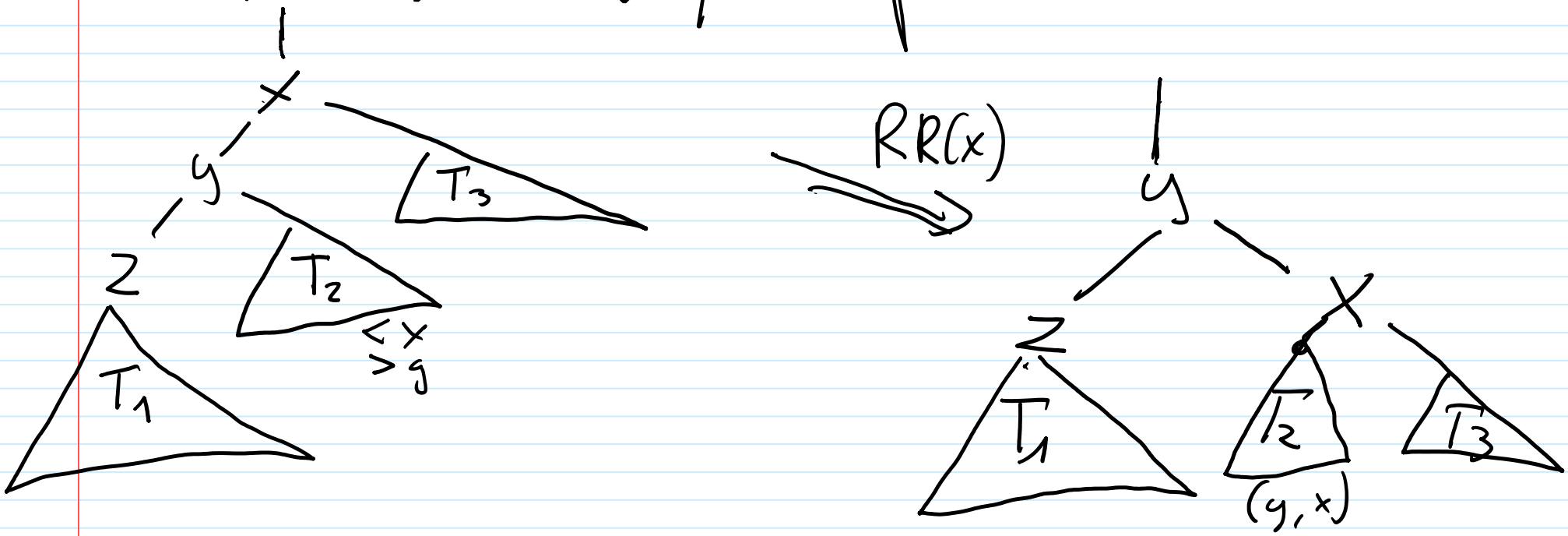
W drugiej fazie, "zwijając rekursję" po ścieżce do miejsca gdzie nastąpiła zmiana struktury drzewa wyliczamy nowe wysokości i jeśli trzeba poprawiamy drzewo.

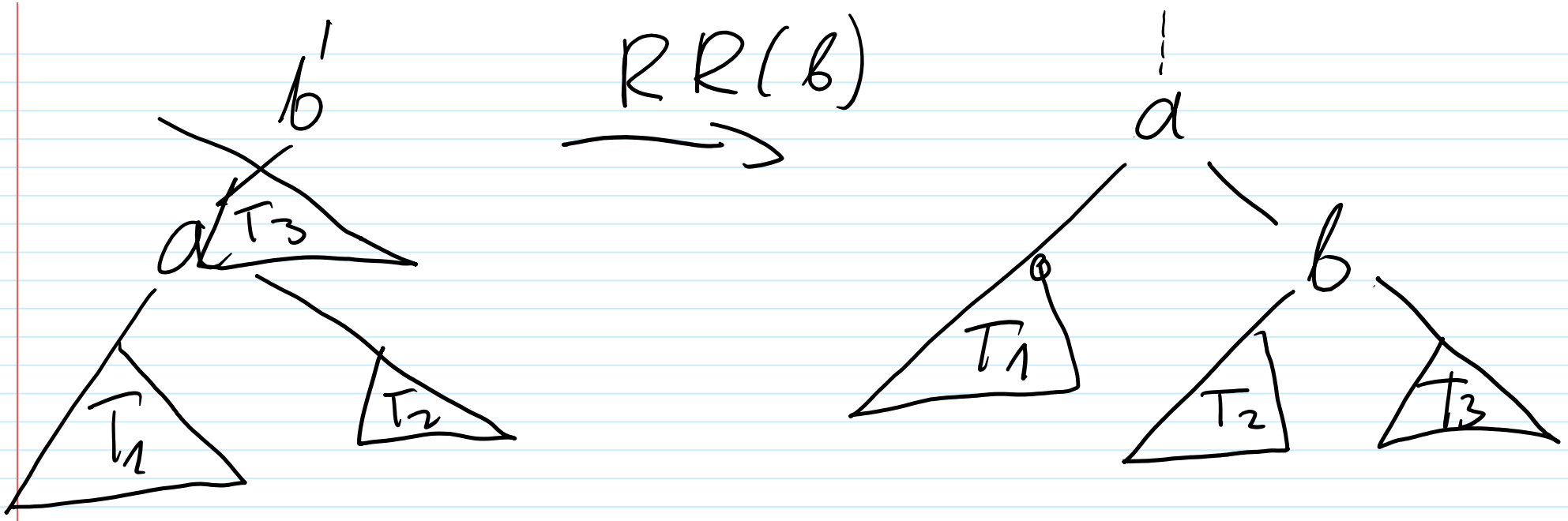

2!?!

Jakich operacji używamy poprawiając strukturę drzewa?

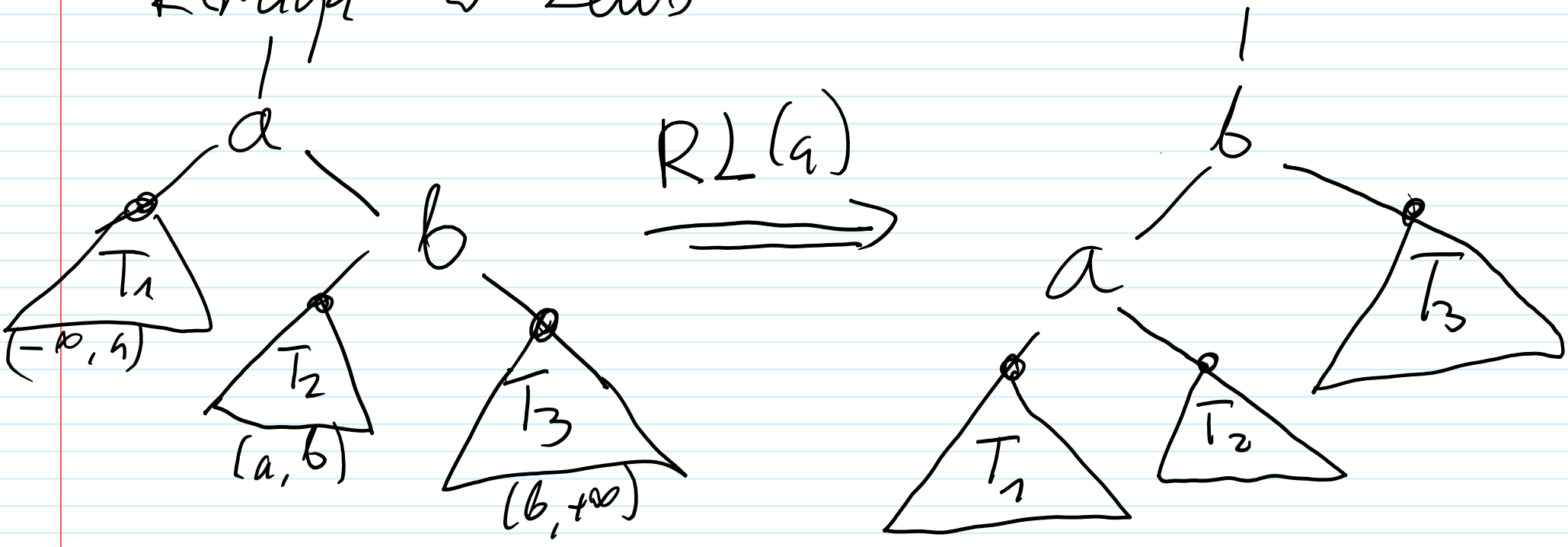
Rotacje w lewo i w prawo wierzchołków.

RR(x) - rotacja w prawo

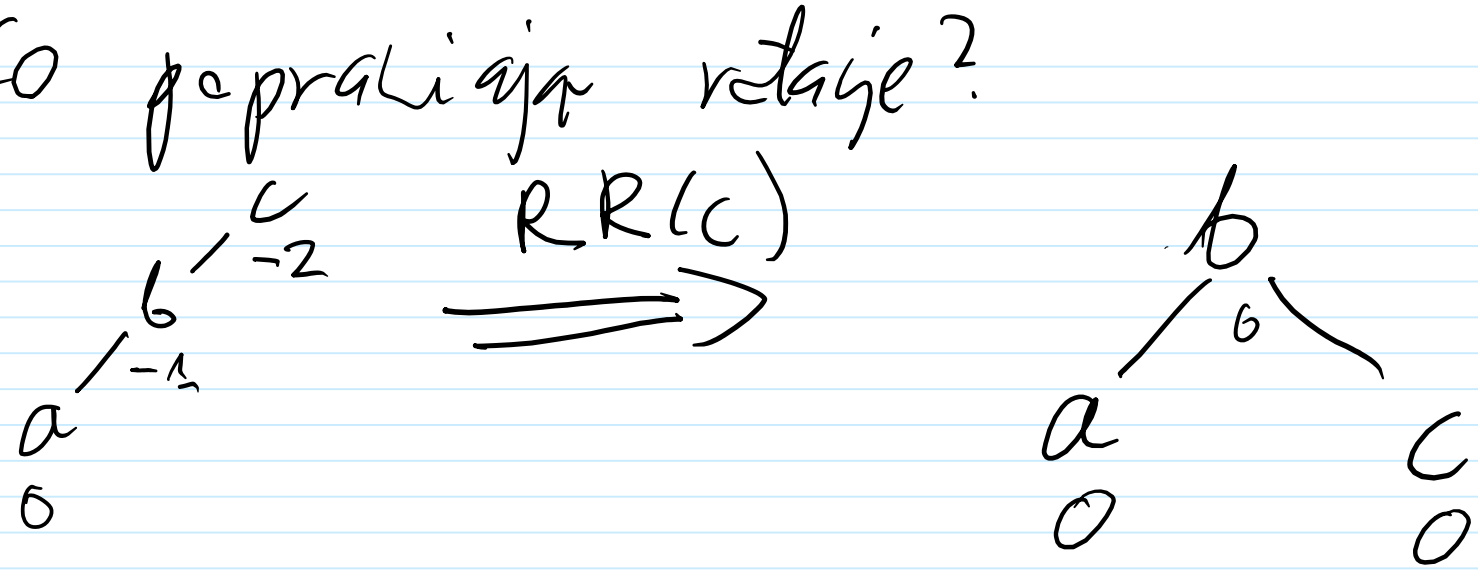


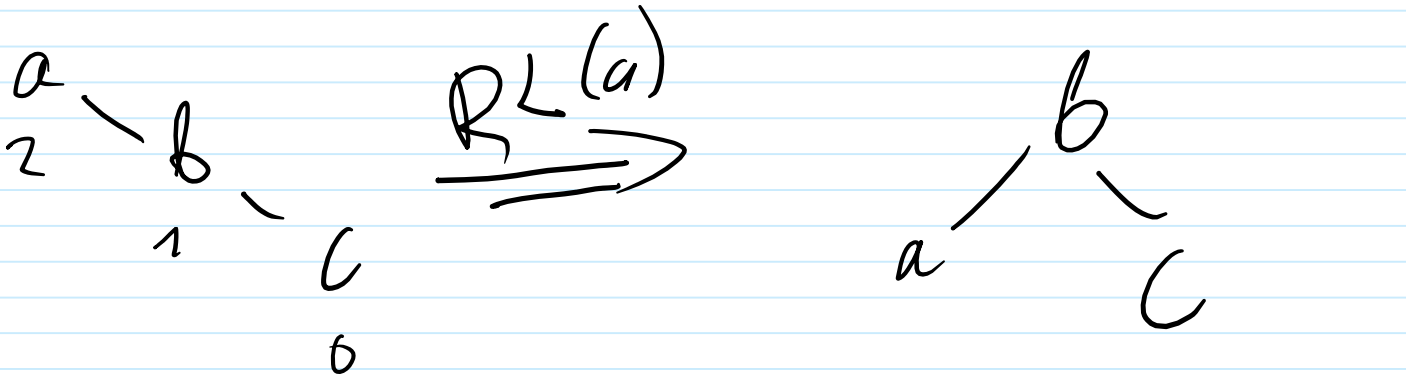

b'
a
T3
T1
T2
RR(b)
a
T1
b
T2
T3

Rotacja w Lewo

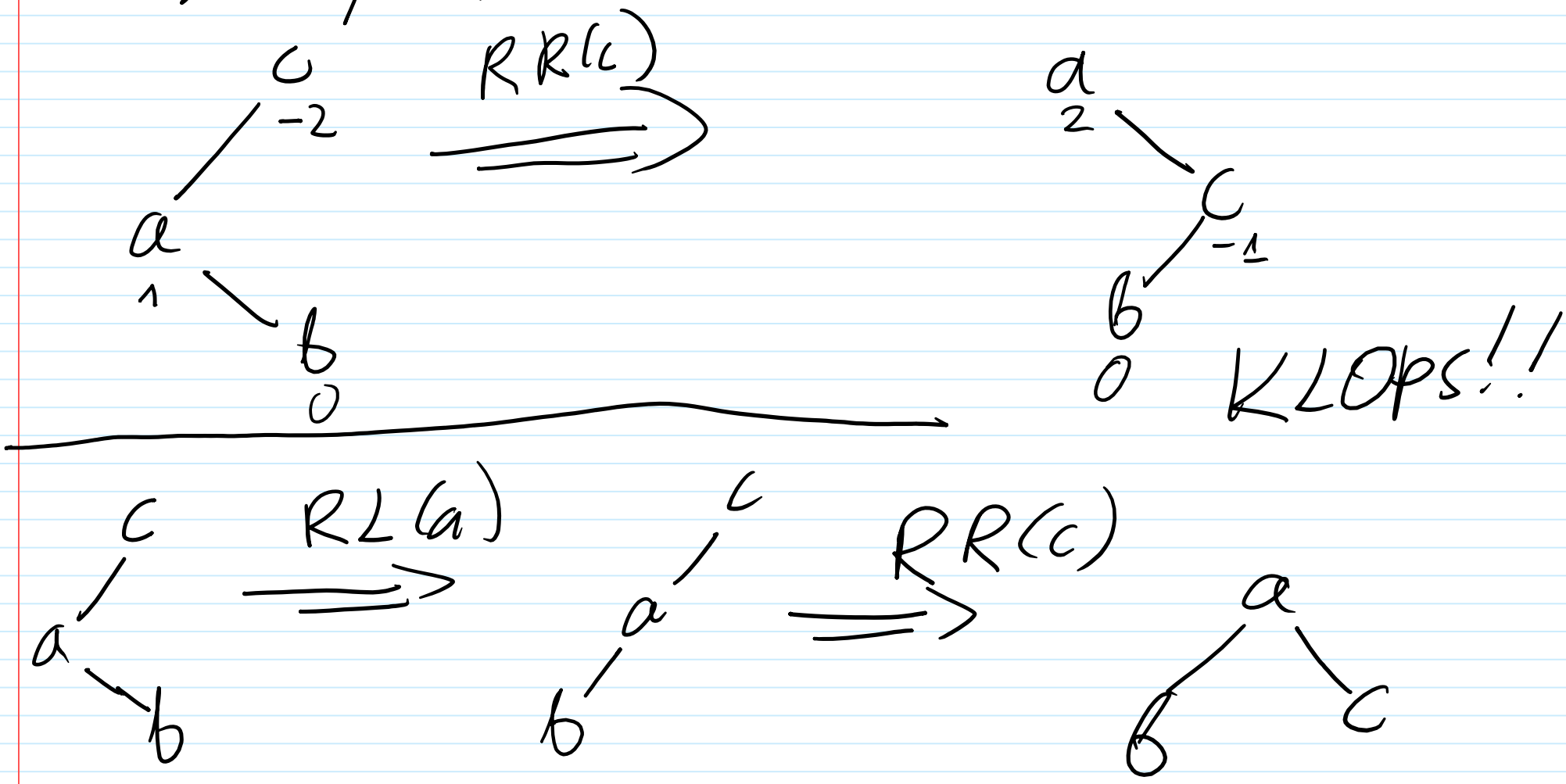


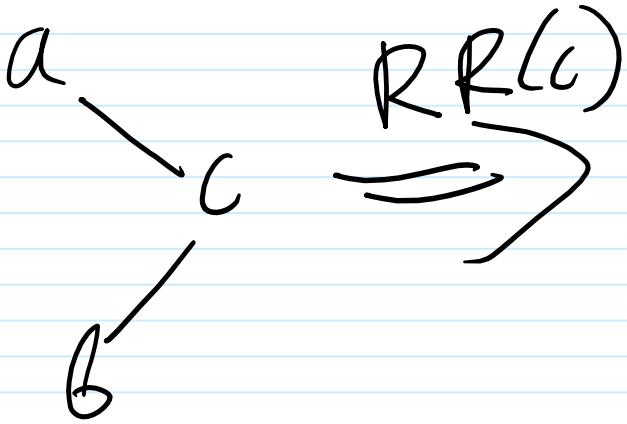
Co poprawiają relacje?

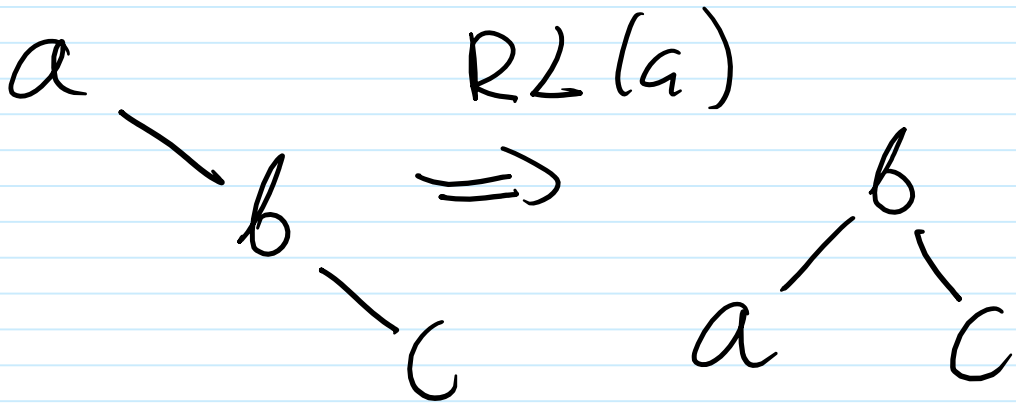




Drugs sytuacja

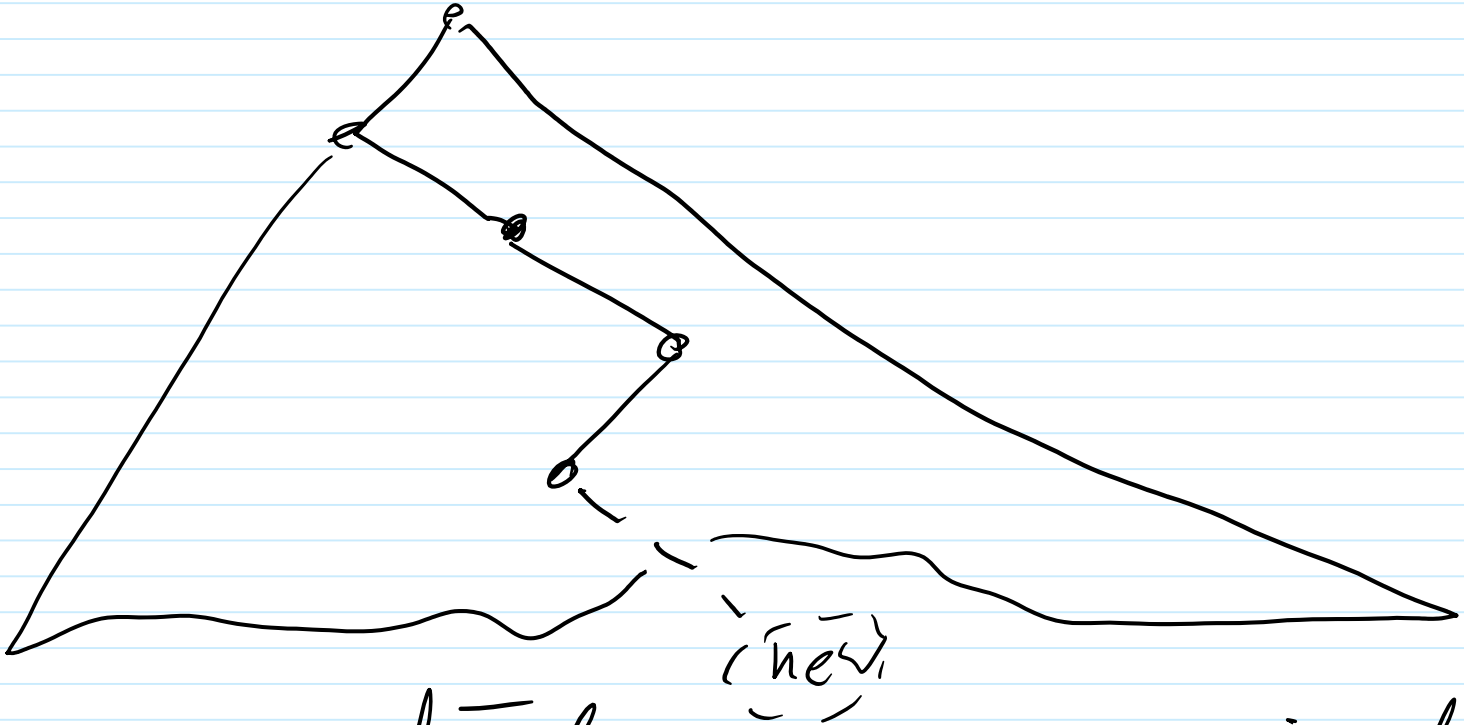



a
c
b
RR(c)


a
b
c
RL(a)
b
a
c

Wstawianie do drzew AVL

- implementacja: musimy dołożyć jedno pole w drzewie
  - int balance - to pole przechowuje wyważenie wierzchołka.

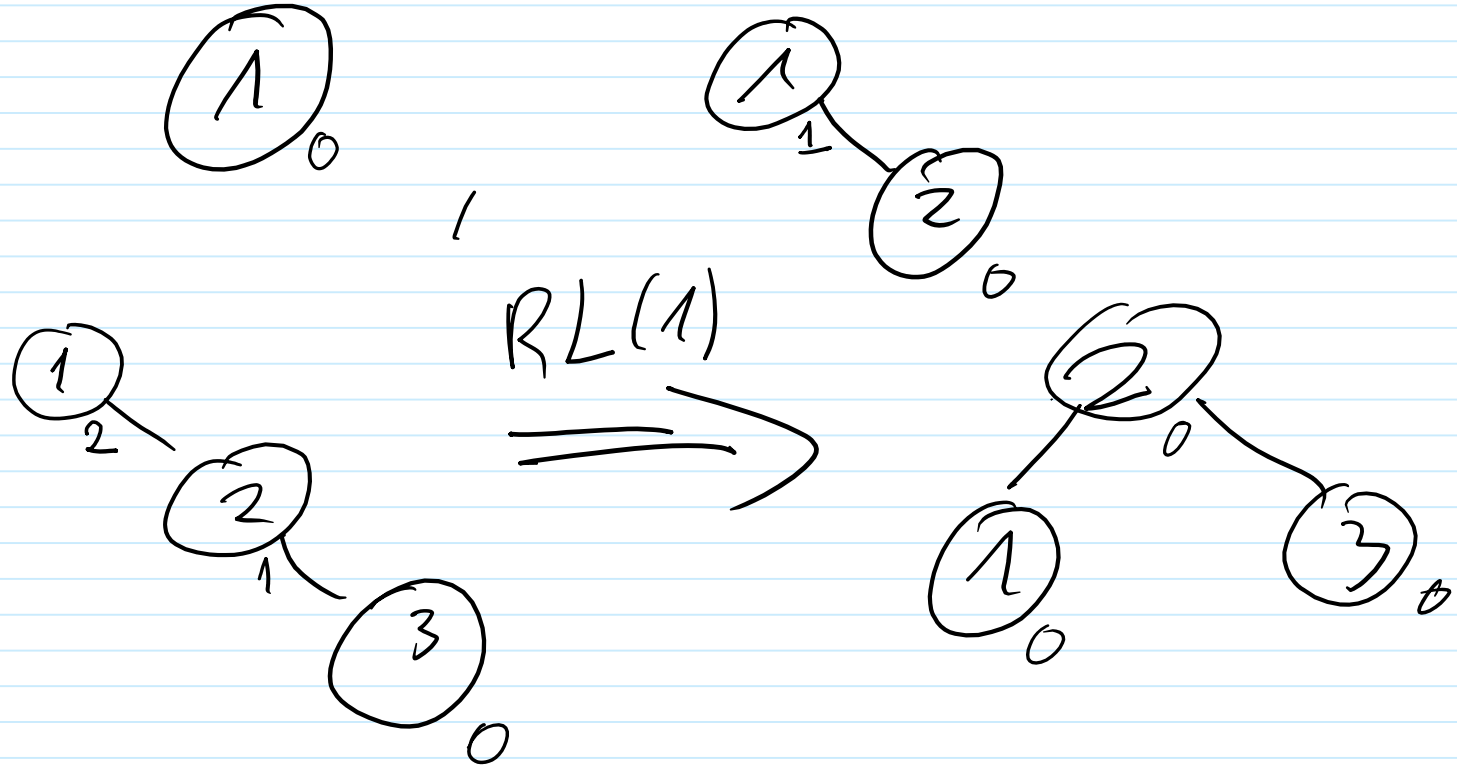


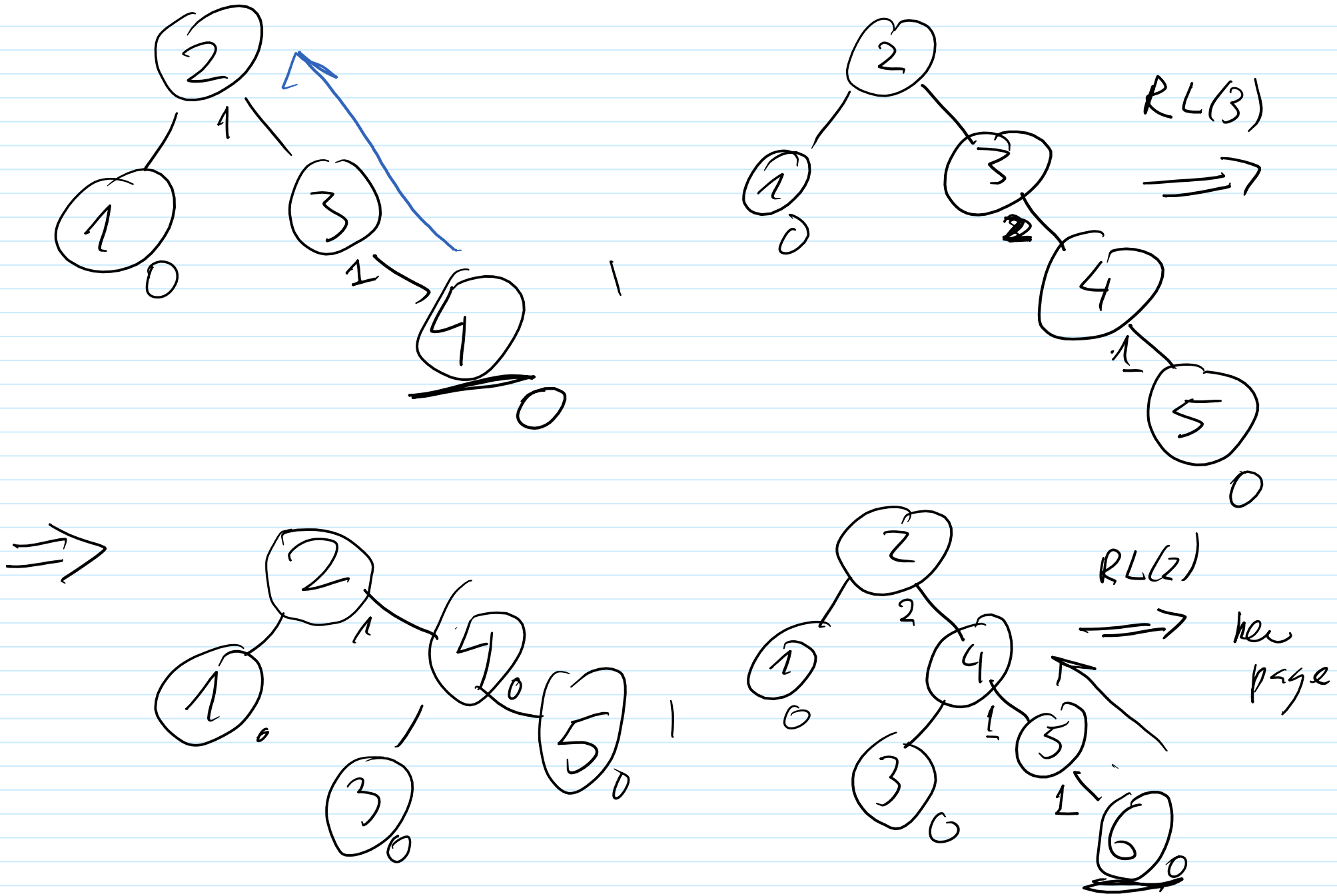
1 etap - dokładamy nowy liść

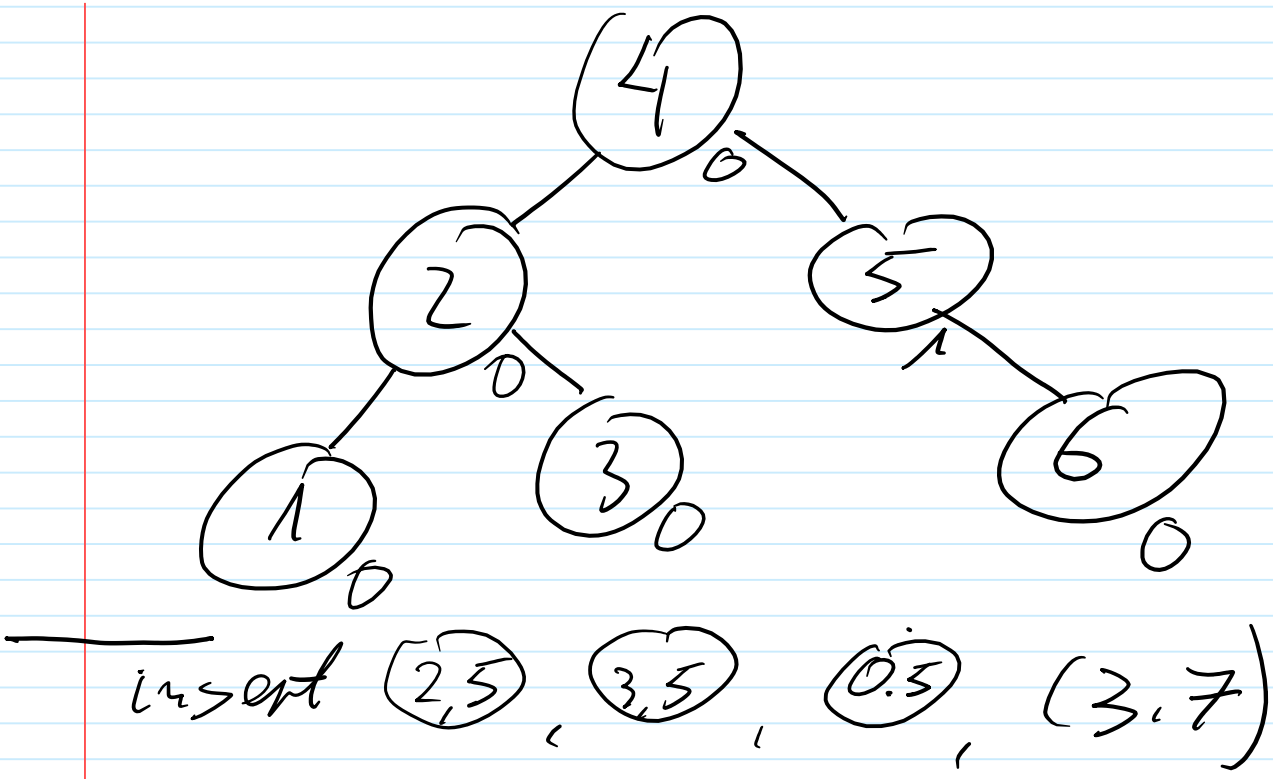
2 etap - wracamy ścieżką w górę drzewa
(np. podczas wyjścia rekurencji)
i aktualizujemy wyłącznie węzły,
jeśli trzeba to je poprawiamy.

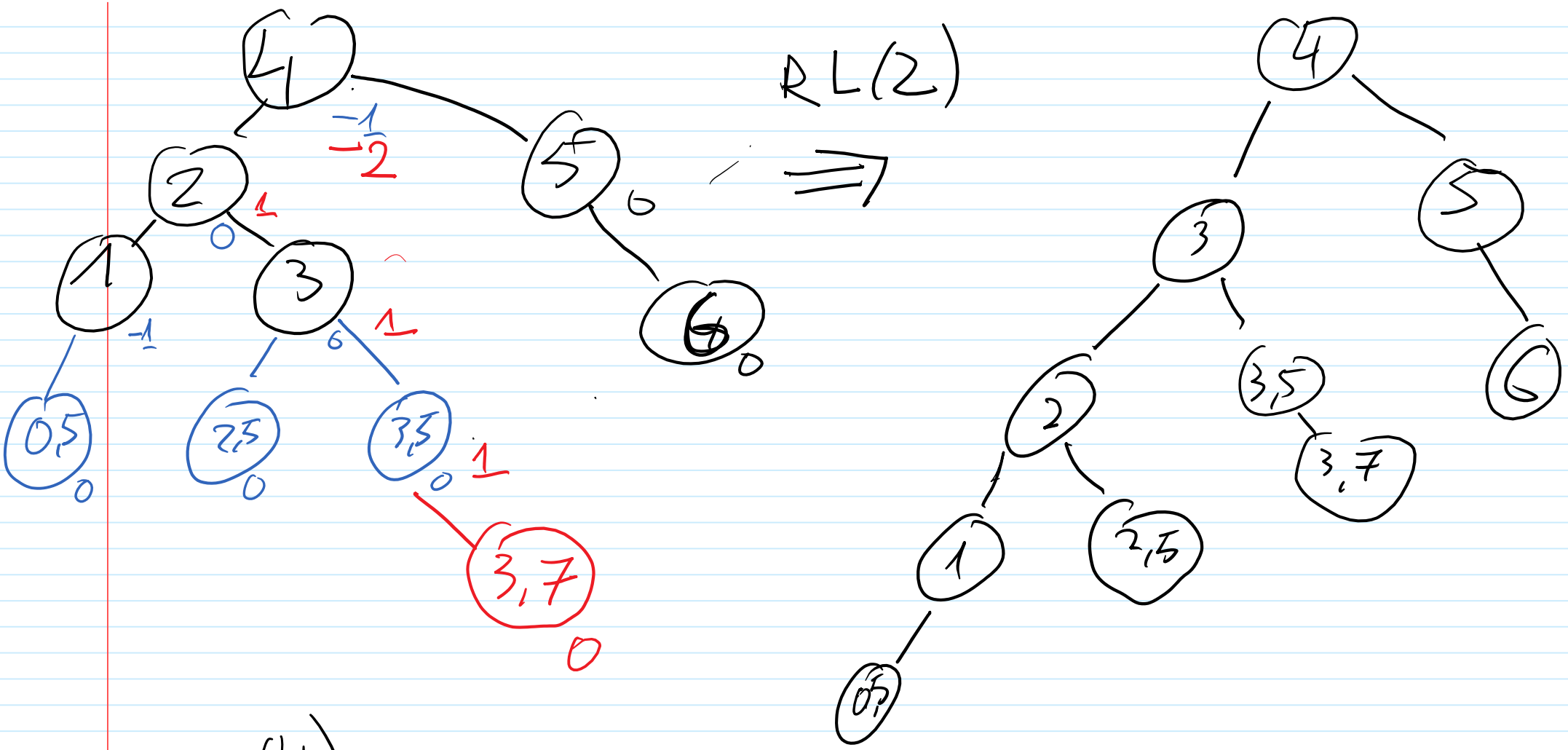
Przykład

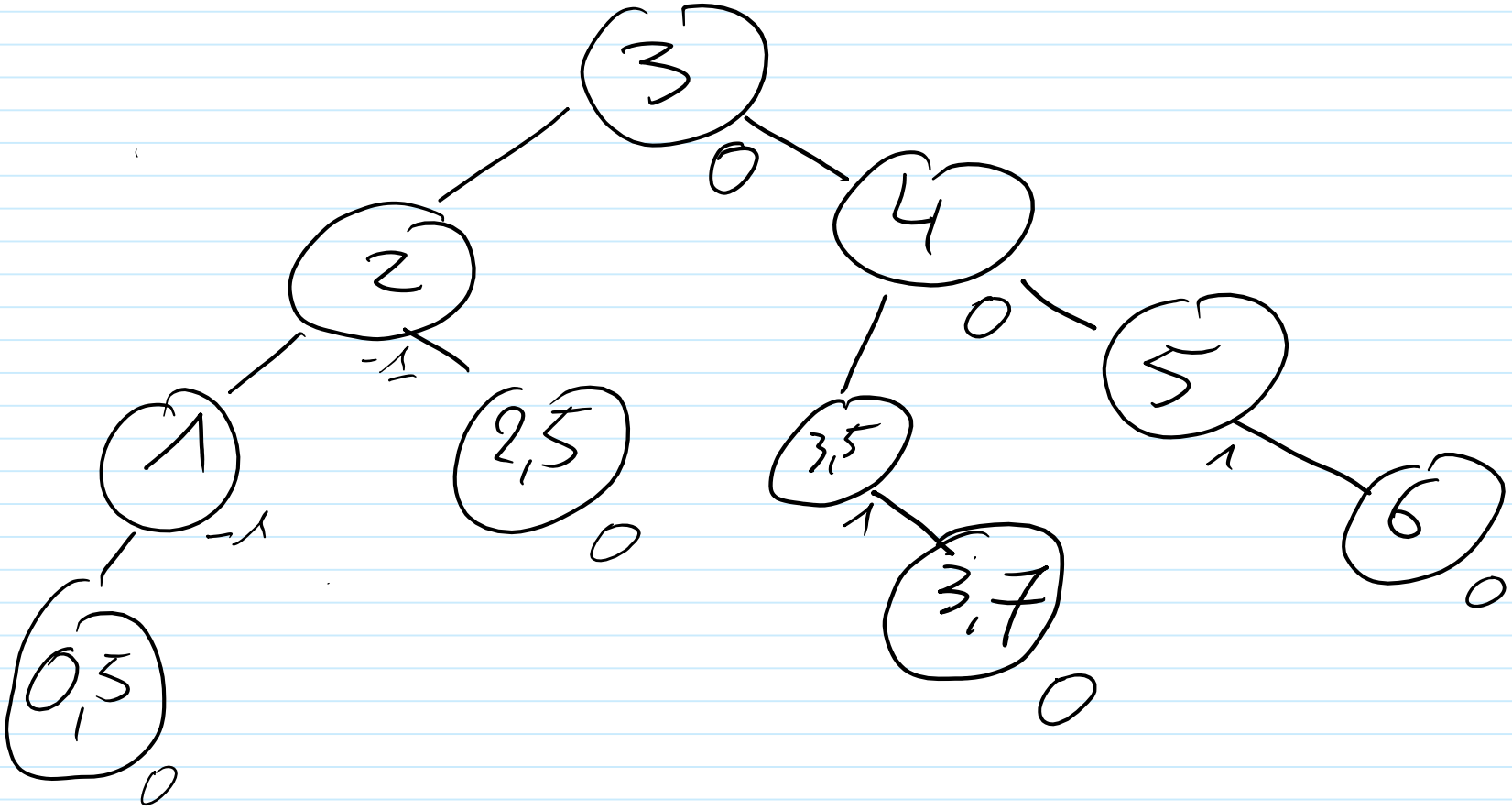
Wstawiamy do drzewa AVL elementy w kolejności: 1, 2, 3, 4, 5, 6, 7, 8.




RL(3)
RL(2)
new
page


4
0
2
0
5
1
1
0
3
0
6
0
insert (2,5), (3,5), (0.5), (3.7)


RL(2)
RR(4)


3
0
2
-1
4
0
1
-1
2,5
0
3,5
1
5
1
0,5
0
3,7
0
6
0