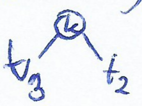
Lista 6, zadanie 8

Do operacji split będziemy potrzebować operacji join. $Join(t_1, k, t_2)$ łączy dwa drzewa t_1 i t_2 oraz wartość k takie, że wszystkie elementy t_1 są mniejsze lub równe k które jest mniejsze lub równe wszystkim elementom t_2. Czyli wynikowe drzewo zwrócone przez operację join będzie takie, że wartości będą w porządku t_1, k, t_2.

Jeśli drzewa t_1 i t_2 różnią się co najwyżej o 1 wysokością, to wystarczy stworzyć nowy wierzchołek 

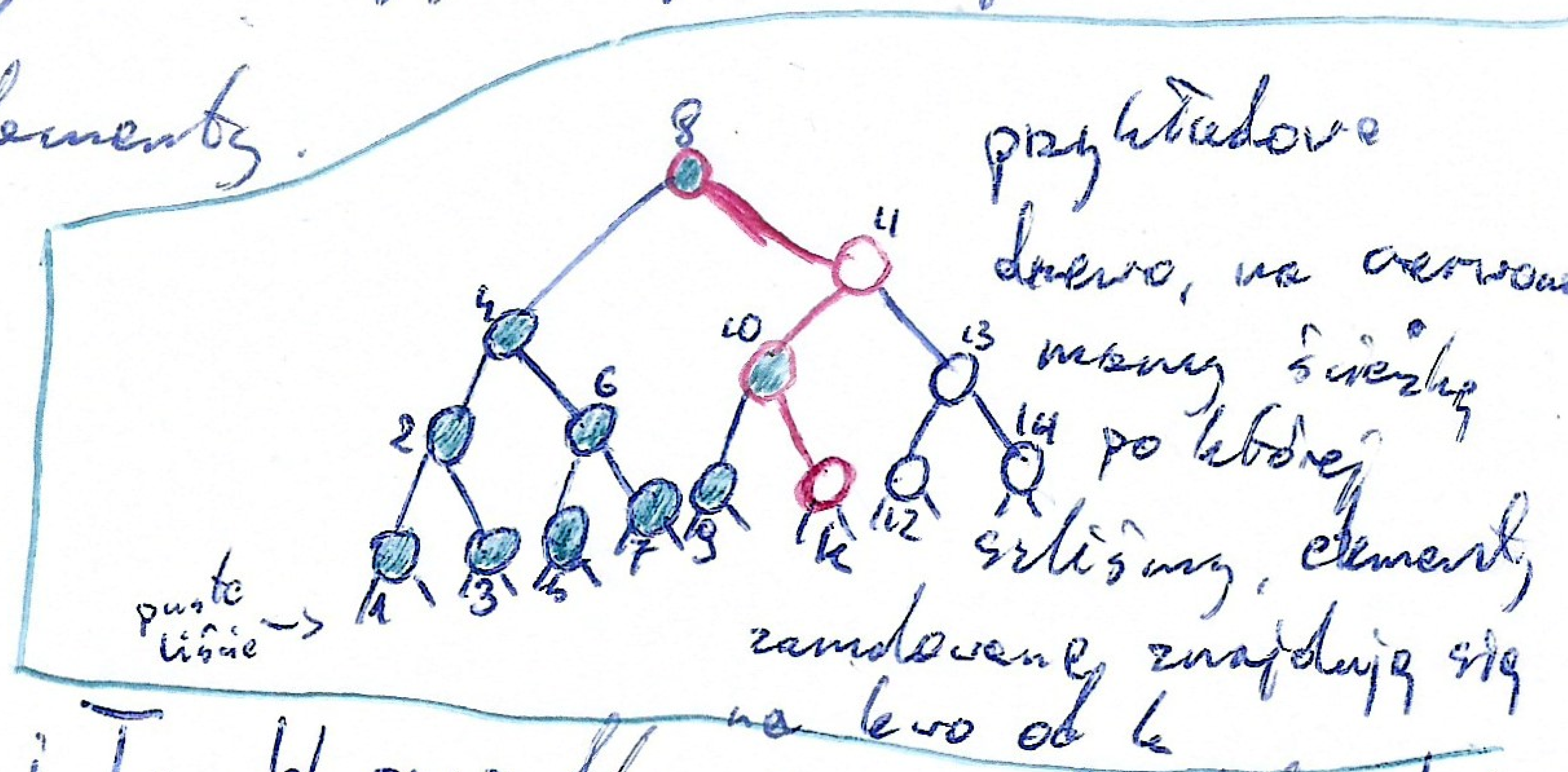
. Natomiast jeśli różnica wysokości jest większa niż 1, to potrzebujemy zbalansować drzewo.

Załóżmy, że $h(t_1) > h(t_2)+1$ (w przeciwnym przypadku będzie analogicznie). Idziemy w prawo aż znajdziemy poddrzewo t_3, takie że $h(t_3) \leq h(t_2)+1$. Tworzymy nowy wierzchołek 

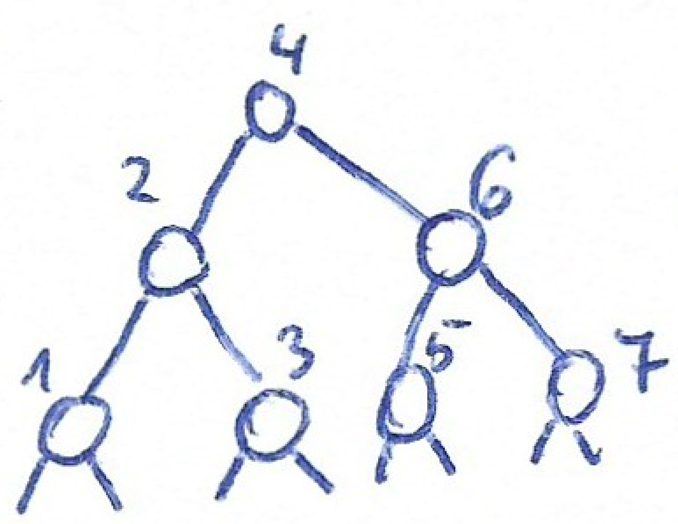
i wstawiamy go w miejsce t_3. Nowy wierzchołek spełnia wymogi AVL bo $h(t_3)=h(t_2)$ lub $h(t_3)=h(t_2)+1$, a jego wysokość jest o 1 większa niż wysokość t_3. Z tego powodu mogą się zepsuć przodkowie t_3. Możemy to naprawić na dwa sposoby: jeśli problem wystąpił w rodzicu t_3, to musimy wykonać dwie rotacje, jeśli wyżej to wystarczy nam rotacja w lewo. Ta jedna poprawa naprawi nam też problemy dla wszystkich innych przodków.

Złożoność operacji join to $O(|h(t_1)-h(t_2)|)$ a drzewo wynikowe t spełnia właściwość $\max(h(t_1), h(t_2)) \leq h(t) \leq \max(h(t_1), h(t_2))+1$.

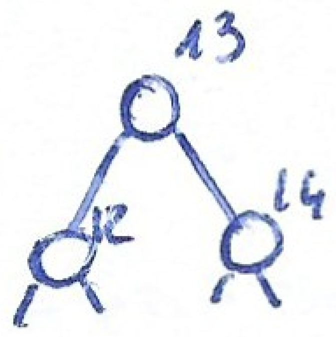
Aby wykonać split(T, k), musimy znaleźć miejsca podziału. Możemy to zrobić np. wstawiając k do T. Wtedy wszystkie elementy na lewo od ścieżki, po której szliśmy wstawiając są mniejsze od k a na prawo mamy pozostałe elementy. Koszt tego to $O(\log n)$.



Teraz dostajemy dużo poddrzew, które musimy połączyć w dwa drzewa - T_L i T_R. W przypadku z rysunku będzie tak:

T_{L1} = join( , 8, join(9, 10, ⊖))

pusty liść, syn k

T_{R1} = join(⊖, 11, ).

Rozważmy tylko złożoność dla T_L, dla T_R będzie analogicznie. Załóżmy, że do po podziale mamy l drzew - $T_1, T_2, \dots, T_l$ takich, że $h(T_1) \leq h(T_2) \leq \dots \leq h(T_l)$. Teraz łączymy po kolei: T_1 z T_2, i z tego otrzymamy T_2', T_2' łączymy z T_3 i otrzymamy T_3' itd aż połączymy wszystkie drzewa. Złożoność tego to koszt $l-1$ operacji join. Z operacji join wiemy, że

$h(T_i') \le h(T_i)+1$, czyli $h(T_i') \le h(T_i)+1 \le h(T_{i+1})+1$. Koszt jednej operacji to $O(|h(T_{i+1}) - h(T_i')|)$. Suma wszystkich kosztów to $\sum_{i=2}^{l-1} |h(T_{i+1}) - h(T_i')| + |h(T_2) - h(T_1)|$ Dla uproszczenia zapisu niech $T_1' = T_1$, przy czym T_1' spełnia zależność wyżej. Wtedy mamy

$$\sum_{i=1}^{l-1} |h(T_{i+1}) - h(T_i')| \le \sum_{i=1}^{l-1} h(T_{i+1}) - h(T_i') + 2 = (l-1)\cdot 2 + \sum_{i=2}^{l-1}\left(h(T_i) - h(T_i')\right) +$$
$$+ h(T_l) - h(T_1') = 2l-2+h(T_l)-h(T_1)+\sum_{i=2}^{l-1}\left(h(T_i)-h(T_i')\right),$$

a ponieważ wiemy, że $\max(h(T_i), h(T'_{i-1})) \le h(T_i') \le \max(h(T_i, h(T'_{i-1})) + 1$, to

$$2l-2+h(T_l)-h(T_1)+\sum_{i=2}^{l-1}\left(h(T_i)-h(T_i')\right) \le 2l-2+h(T_l)-h(T_1)+l-1 =$$
$$= 3l-3+h(T_l)-h(T_1).$$

Teraz zauważmy, że $l \le \cancel{\log}\ h(T)$. Wynika to z tego, że „przecinając" drzewo ustawieniem do niego k, w każdym zejściu poziom niżej tworzymy tylko jedno nowe poddrzewo do połączenia. Zauważmy też, że o ostatnim kroku tworzymy dwa drzewa, z których większe jest poddrzewem oryginalnego T. Czyli $h(T_l) \le h(T)-1$. Stąd mamy złożoność $3l-3+h(T_l)-h(T_1) \le 3h(T)+h(T)-4-h(T_1) = O(h(T))$, czyli $O(\log n)$ □