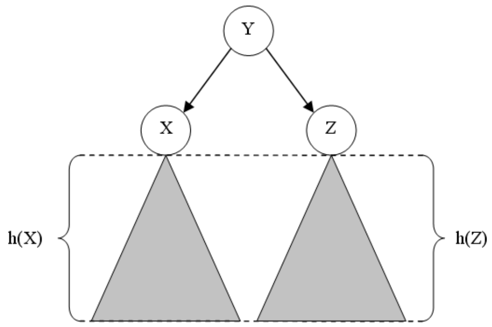
ASD - ćwiczenia VIII

Wyważone drzewa przeszukiwań binarnych – typu AVL

- własność wyważenia drzew AVL



  gdzie dla każdej trójki wierzchołków drzewa BST (X, Y, Z) różnica wysokości poddrzew wierzchołka Y jest następująca $h(X) - h(Z) \in \{-1, 0, 1\}$,

- definicja wskaźnikowa struktury typu węzeł drzewa AVL w pseudokodzie

```
typedef struct AVLTreeNode AVLTree;

struct AVLTreeNode {
  element elem;
  int h_left, h_right;
  struct AVLTreeNode left, right, parent;
};
```

- podstawowe operacje dla drzewa binarnego typu AVL:

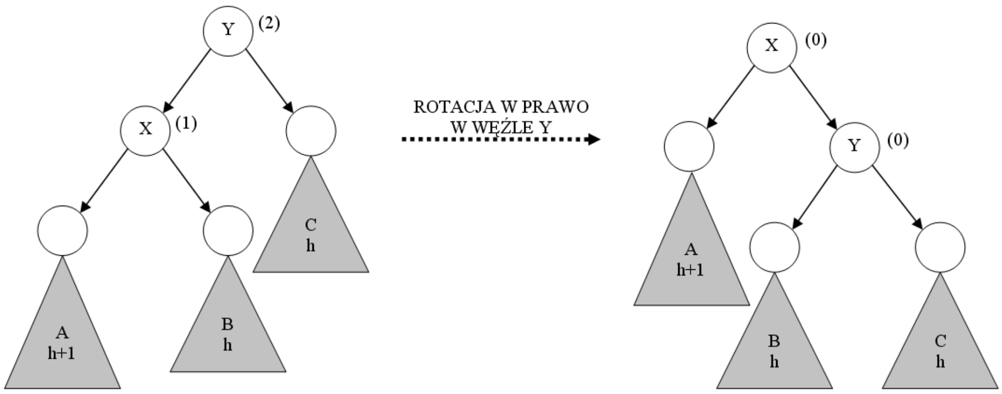
  - $EMPTY : \mathcal{T} \to \{TURE, FALSE\}$, sprawdzenie czy struktura jest pusta,
  - $INSERT : \mathcal{T} \times E \to \mathcal{T}$, wstawienie elementu do struktury,
  - $DELETE : \mathcal{T} \times E \to \mathcal{T}$, usunięcie elementu ze struktury,
  - $MEMBER : \mathcal{T} \times E \to \{TURE, FALSE\}$, sprawdzenie, czy dany element jest przechowywany w strukturze,

  gdzie $\mathcal{T}$ jest przestrzenią drzew typu AVL, E zbiorem etykiet wierzchołków drzewa typu AVL,

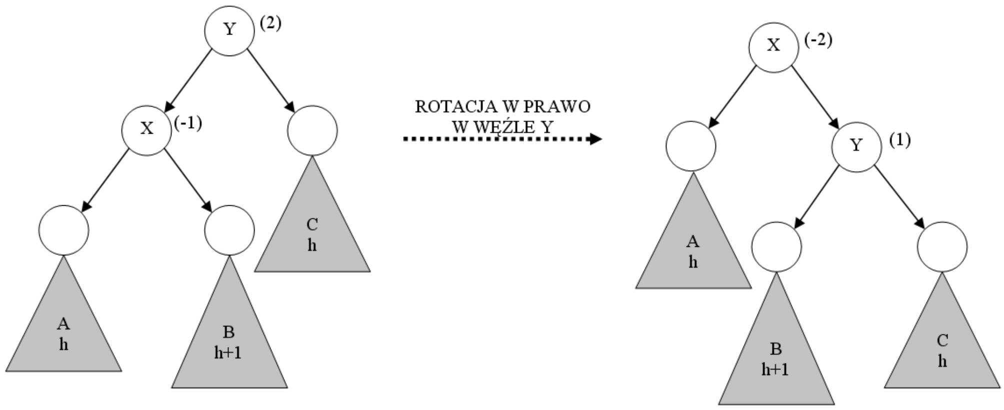
- złożoność czasowa podstawowych operacji n-elementowego drzewa typu BST:

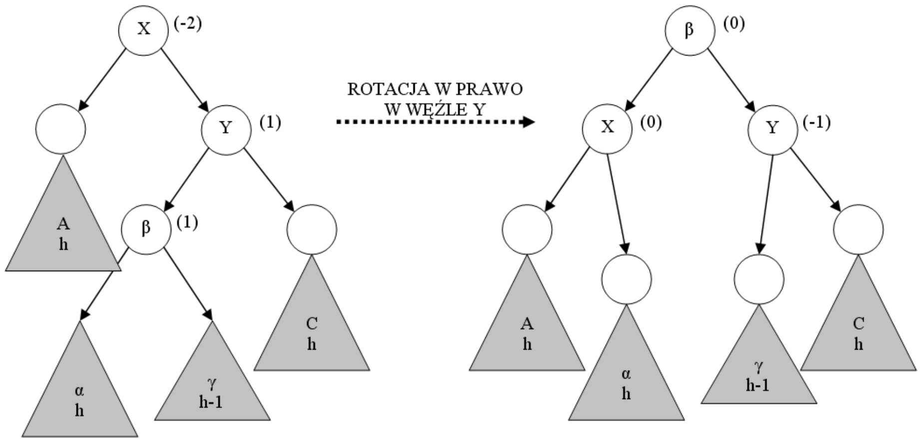
  - $A(\texttt{EMPTY}(), n) = O(1)$, $W(\texttt{EMPTY}(), n) = O(1)$,
  - $A(\texttt{INSERT}(), n) = O(\log(n))$, $W(\texttt{INSERT}(), n) = O(\log(n))$,
  - $A(\texttt{DELETE}(), n) = O(\log(n))$, $W(\texttt{DELETE}(), n) = O(\log(n))$,
  - $A(\texttt{MEMBER}(), n) = O(\log(n))$, $W(\texttt{MEMBER}(), n) = O(\log(n))$,

- operacji rotacji pojedynczej w prawo (rotacja w lewo jest przypadkiem symetrycznym)



- operacja rotacji podwójnej w prawo (rotacja w lewo jest przypadkiem symetrycznym)





Zadania

1. Przedstaw schemat rotacji pojedynczej w lewo z uwzględnieniem wag w węzłach drzewa AVL. Na przykładowym drzewie AVL przeprowadź operację wstawienia elementu, która wymusiłaby taką rotację (tj. drzewo wejściowe → wstawienie elementu → drzewo po rotacji).

2. Przedstaw schemat rotacji podwójnej w prawo z uwzględnieniem wag w węzłach drzewa AVL. Na przykładowym drzewie AVL przeprowadź operacje usunięcia elementu, która wymusiłaby taką rotację (tj. drzewo wejściowe → usunięcie elementu → drzewo po rotacji).

3. Ile maksymalnie operacji rotacji należy wykonać aby wstawić dany element do drzewa AVL o wysokości h i $2^h - 1$ wierzchołkach? Zakładamy, że wstawiany element nie występuje w rozważanej strukturze.

4. Które z podanych poniżej stwierdzeń jest prawdziwe, odpowiedź uzasadnij:

   (a) średni koszt wstawienia m elementów do losowego drzewa BST zawierającego już n elementów to $\Theta(\log^m(n))$, gdzie $m = \Theta(n)$,

   (b) pesymistyczny koszt wstawienia m elementów do losowego drzewa AVL zawierającego już n elementów to $\Theta(n^2)$, gdzie $m = \Theta(n^2)$,

   (c) niech $leaf_{BST}(h)$ oraz $leaf_{AVL}(h)$ oznaczają kolejno minimalną liczbę liści w drzewie BST i drzewie AVL wysokości h, wtedy $leaf_{AVL}(h) = O(leaf_{BST}(h))$,

   (d) istnieje liniowy algorytm konstruowania drzewa BST dla dowolnego wejściowego n-elementowego ciąg liczb naturalnych.

5. Załóżmy, że wierzchołki pewnego drzewa binarnego T są etykietowane liczbami całkowitymi. Napisz funkcję rekurencyjną

   ```
   int IS_AVL(Tree T),
   ```

   która:

   - jeżeli drzewo T jest drzewem typu AVL, wyznaczy jego wysokość,
   - jeżeli drzewo T nie jest drzewem typu AVL, zwróci wartość (-1).

6. Czy prawdziwe są poniższe stwierdzenia:

   (a) każde drzewo AVL jest idealnym drzewem BST[1],

   (b) każde idealne drzewo BST jest drzewem AVL.

[1] Drzewo binarne nazywamy idealnym, jeżeli dla każdego jego wierzchołka v prawdziwa jest zależność $l(v) - r(v) \in \{-1, 0, 1\}$, gdzie $l(v)$, $r(v)$ to odpowiednio liczba wierzchołków w lewym i prawym poddrzewie wierzchołka v. Jeżeli dodatkowo drzewo to jest drzewem BST to nazywamy je idealnym drzewem BST.