Δένδρο AVL

Υποθέσεις:

- Οι λέξεις με κεφαλαία είναι άλλες από αυτές με πεζά.
- Οι αριθμοί δεν είναι λέξεις.
- Στην διαγραφή στοιχείου διαγράφετε ολόκληρη η θέση.

Μεταβλητές:

Το struct aNode (κόμβος) περιέχει:

- **key,** η λέξη που αποθηκεύεται στον κόμβο.
- **times,** οι φορές που εμφανίζεται η λέξη.
- **right, left,** δείκτες που δείχνουν προς το δεξί και αριστερό παιδί του κόμβου.
- **height,** το ύψος που έχει ο συγκεκριμένος κόμβος (δηλαδή το ύψος του υπό δένδρου με ρίζα τον κόμβο) Ουσιαστικά σε πιο επίπεδο βρίσκεται ο συγκεκριμένος κόμβος.

Μέθοδοι:

- Int **GetHeightOfNode**(aNode* **Ptr**)
  - Ορίσματα:
    Ένας δείκτης ***Ptr** που δείχνει σε έναν κόμβο.
  - Επιστρέφει:
    Επιστρέφει 0 αν ο δείκτης **Ptr** δείχνει σε NULL, διαφορετικά επιστρέφει την τιμή της μεταβλητής **height** του κόμβου.

- Int **FindHeight**(aNode* **Ptr**)
  - Ορίσματα:
    Ένας δείκτης ***Ptr** που δείχνει σε έναν κόμβο.
  - Επιστρέφει:
    Αν ο δείκτης ***Ptr** δείχνει σε NULL, τότε επιστρέφεται η τιμή 0. Σε κάθε άλλη περίπτωση, καλείται η αναδρομικά η μέθοδος για να βρεθει το δεξί και αριστερό ύψος (**leftHeight** = **FindHeight**(**Ptr**->**left**) και **rightHeight** = **FindHeight**(**Ptr**->**right**)) και επιστρέφεται το μεγαλύτερο από τα δύο ύψη + 1 (**h** = **leftHeight** + 1 ή **h = rightHeight** + 1).
- Int **FindDifference**(aNode* **Ptr**)
  - Ορίσματα:

Ένας δείκτης ***Ptr** που δείχνει σε έναν κόμβο.

- Επιστρέφει

  Καλεί την συνάρτηση **GetHeightOfNode** για να βρει το δεξί και αριστερό ύψος του κόμβου και επιστρέφει την διαφορά τους (**diff** = **leftHeight** - **rightHeight**). Αν η διαφορά είναι μεγαλύτερη από 1 (**diff** >= 1), τότε το αριστερό υποδένδρο είναι μεγαλύτερο, ενώ αν η διαφορά είναι μικρότερη από -1 (**diff** <= -1), τότε το δεξί υποδένδρο είναι μεγαλύτερο.
  Αν η διαφορά είναι μεταξύ -1 και 1, τότε το δένδρο είναι ισορροπημένο.

- aNode* **Balance**()
  - Ορίσματα:

    Ένας δείκτης ***Ptr** που δείχνει σε έναν κόμβο.
    Το Key οπού το Key του κόμβου.
  - Επιστρέφει

    Τον Ptr οπού καλέστηκε αλλά αφού έχει ισορρόπησή τα δυο υπό δέντρα

- nNode **CreateLeaf**(string **key**)
  - Ορίσματα :

    Η λέξη **key.**
  - Επιστρέφει:

    Έναν καινούργιο κόμβο **n** .

- void **AddLeaf**(string **key**)
  - Ορίσματα:

    Η λέξη **key.**
  - Επιστρέφει:

    (κενό) καλεί την συνάρτηση **AddLeafPrivate**(string **key**, nNode ***Ptr**).

- void **AddLeafPrivate**(string **key**, nNode ***Ptr**)
  - Ορίσματα:

    Η λέξη **key**
    Ένας δείκτης ***Ptr** που δείχνει σε έναν κόμβο.
  - Επιστρέφει:

    Αρχικά δημιουργείτε το Leaf και έπειτα επιστρέφετε ο δείκτης οπού προστέθηκε ο κόμβος
- aNode* leftRotate(aNode*)
  - Ορίσματα:

    Ένας δείκτης που δέχεται τον κόμβο για να κάνει το left Rotation
  - Επιστρέφει:

    Τον συγκεκριμένο κόμβο

- aNode* rightRotate(aNode*)
    - Ορίσματα:
        Ένας δείκτης που δέχεται τον κόμβο για να κάνει το right Rotation
    - Επιστρέφει:
        Τον συγκεκριμένο κόμβο

Το Search παραμένει ίδιο με το BST