Εργασια στο μαθημα Δομες Δεδομενων 2020

Ονοματεπωνυμα

Βελισαριος Φαφουτης Αεμ 3117

Γεωργιος Πετκακης Αεμ 3023

Για την δημιουργία των απαιτούμενων δομών δεδομένων υλοποιήθηκαν οι εξης κλάσσεις:

1. AvlTree
2. BinarySearchTree
3. HashTable

1.AvlTree

Για την δημιουργια του Δυαδικου Δενδρου Αναζητησης τυπου AVL υλοποιηθηκαν δυο κλασσεις.

1. AvlTree:Αντιπροσωπευει το ιδιο το δενδρο
2. avl_node:Αντιπροσωπευει έναν κομβο του δενδρου

Η κλαση avl_node περιλαμβανει:

- avl_node():Κενος κατασκευαστης της κλασης
- avl_node(avl_node* l, avl_node* r, string aWord, int aps, int h, int bf):Κατασκευαστης με ορισματα για αρχικοποιγησει των περιεχομενων της κλασης.
- word:Αυτή η μεταβλητη αντιπροσωπευει την λεξη που θα αποθηκευεται μεσα στον κομβο του δενδρου
- appearances: Αυτή η μεταβλητη αντιπροσωπευει το πληθος των εμφανισεων της συγκεκριμενης λεξης μεσα στο δενδρο.
- height:το υψος του κομβου
- balanceFactor:Αυτή η μεταβλητη χρησιμοποιειται για να γνωριζουμε αν είναι απαραιτητη η περιστροφη του συγκεκριμενου κομβου

- δεικτες *left, *right οι οποιοι δειχνουν στο αριστερο και δεξι παιδι αντιστοιχα, του συγκεκριμενου κομβου

Η κλαση AvlTree περιλαμβανει:

- έναν δεικτη *root ο οποιος δειχνει στην ριζα του δενδρου
- avl_node* insertion(avl_node* root, string aWord):μεθοδος η οποια χρησιμοποιειται για την εισαγωγη μιας λεξης στο δενδρο
- avl_node* deletion(avl_node* root, string aWord):μεθοδος η οποια χησιμοποιειται για την διαγραφη μιας λεξης από το δενδρο
- avl_node* search(avl_node* root, string aWord)):μεθοδος η οποια χησιμοποιειται για την αναζητηση μιας λεξης στο δενδρο
- int getHeight(avl_node* root): μεθοδος η οποια επιστρεφει το υψος του δενδρου
- int getBfactor(avl_node* root):μεθοδος η οποια επιστρεφει τον συντελεστη εξισορροπησης του δενδρου
- *avl_node rightRotation(avl_node* root):μεθοδος για την περιπτωση της δεξιας περιστροφης του κομβου
- avl_node* leftRotation(avl_node* root)):μεθοδος για την περιπτωση της αριστερης περιστροφης του κομβου
- avl_node* leftLeftCase(avl_node* root)):μεθοδος για την αντιμετωπιση της αριστερης αριστερης περιπτωσης
- avl_node* rightRightCase(avl_node* root) : μεθοδος για την αντιμετωπιση της δεξιας δεξιας περιπτωσης
- avl_node* leftRightCase(avl_node* root) : μεθοδος για την αντιμετωπιση της αριστερης δεξιας περιπτωσης
- avl_node* rightLeftCase(avl_node* root)) : μεθοδος για την αντιμετωπιση της δεξιας αριστερης περιπτωσης

- avl_node* getBalanced(avl_node* root):μεθοδος που εξισορροπει έναν κομβο σε περιπτωση που είναι απαραιτητο
- void updateAvl_node(avl_node* root):ενημερωνει τα δεδομενα ενός κομβου
- avl_node* findMin(avl_node* root):επιστρεφει τον κομβο με το ελαχιστο στοιχειο που βρισκεται στο υποδενδρο με ριζα root
- void inOrder(avl_node* root) :μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε ενδοδιατεταγμενη σειρα
- void preOrder(avl_node* root) :μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε προδιατεταγμενη σειρα
- void postOrder(avl_node* root): μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε μεταδιατεταγμενη σειρα
- void print(avl_node* root, string aWord):αυτή η μεθοδος χρησιμοποιειται για την εμαφανιση των λεξεων που αναζητηθηκαν από τις 3 δομες δεδομενων
- AvlTree():κενος κατασκευαστης που αρχικοποιει την ριζα του δενδρου
- bool insertion(string aWord):εισαγει μια λεξη στο δενδρο
- bool deletion(string aWord): διαγραφει μια λεξη από το δενδρο
- bool search(string word):αναζητει μια λεξη στο δενδρο
- void inOrder():μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε ενδοδιατεταγμενη σειρα
- void preOrder():μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε προδιατεταγμενη σειρα
- void postOrder()): μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε μεταδιατεταγμενη σειρα
- void print(string aWord)):αυτή η μεθοδος χρησιμοποιειται για την εμαφανιση των λεξεων που αναζητηθηκαν από τις 3 δομες δεδομενων

2.BinarySearchTree

Για την δημιουργια του απλου Δυαδικου Δενδρου Αναζητησης υλοποιηθηκαν δυο κλασσεις.

1. BinarySearchTree:Αντιπροσωπευει το ιδιο το δενδρο

2.bst_node:Αντιπροσωπευει έναν κομβο του δενδρου

Η κλαση bst_node περιλαμβανει:

- bst_node ():Κενος κατασκευαστης της κλασης
- bst_node(bst_node* l, bst_node* r, string aWord, int aps):Κατασκευαστης με ορισματα για αρχικοποιγησει των περιεχομενων της κλασης.
- word:Αυτή η μεταβλητη αντιπροσωπευει την λεξη που θα αποθηκευεται μεσα στον κομβο του δενδρου
- appearances: Αυτή η μεταβλητη αντιπροσωπευει το πληθος των εμφανισεων της συγκεκριμενης λεξης μεσα στο δενδρο.
- δεικτες *left, *right οι οποιοι δειχνουν στο αριστερο και δεξι παιδι αντιστοιχα, του συγκεκριμενου κομβου

Η κλαση BinarySearchTree περιλαμβανει:

- έναν δεικτη *root ο οποιος δειχνει στην ριζα του δενδρου
- bst_node* insertion(bst_node* root, string aWord):μεθοδος η οποια χρησιμοποιειται για την εισαγωγη μιας λεξης στο δενδρο
- bst_node* deletion(bst_node* root, string aWord):μεθοδος η οποια χησιμοποιειται για την διαγραφη μιας λεξης από το δενδρο

- bool search(bst_node* root, string aWord):μεθοδος η οποια χησιμοποιειται για την αναζητηση μιας λεξης στο δενδρο
- bst_node* findMin(bst_node* root):επιστρεφει τον κομβο με το ελαχιστο στοιχειο που βρισκεται στο υποδενδρο με ριζα root
- void inOrder(bst_node* root):μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε ενδοδιατεταγμενη σειρα
- void preOrder(bst_node* root):μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε προδιατεταγμενη σειρα
- void postOrder(bst_node* root): μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε μεταδιατεταγμενη σειρα
- BinarySearchTree():κενος κατασκευαστης που αρχικοποιει την ριζα του δενδρου
- bool insertion(string aWord):εισαγει μια λεξη στο δενδρο
- bool deletion(string aWord): διαγραφει μια λεξη από το δενδρο
- bool search(string word):αναζητει μια λεξη στο δενδρο
- void inOrder():μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε ενδοδιατεταγμενη σειρα
- void preOrder():μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε προδιατεταγμενη σειρα
- void postOrder()): μεθοδος που εκτυπωνει τα δεδομενα του δενδρου σε μεταδιατεταγμενη σειρα

3. HashTable

Για την δημιουργια του πινακα κατακερματισμου ανοιχτης διευθυνσης υλοποιηθηκαν δυο κλασσεις.

1.HashTable:Αντιπροσωπευει τον ιδιο τον πινακα

2.Hash_Node:Αντιπροσωπευει ένα στοιχειο του πινακα

Η κλαση Hash_Node περιλαμβανει:

- Hash_Node():κενος κατασκευαστης
- Hash_Node(string aWord, int aps):Κατασκευαστης με ορισματα για την αρχικοπιηση των τιμων του συγκεκριμενου στοιχειου
- Hash_Node(const Hash_Node& hash_Node):Κατασκευαστης αντιγραφου
- stirng word :μεταβλητη που αντιπροσωπευει την αντιστοιχη λεξη που είναι αποθηκευμενη μεσα στο συγκεκριμενο στοιχειο του πινακα
- int appearances: μεταβλητη που αντιπροσωπευει το πληθος των εμφανισεων της λεξης που είναι αποθηκευμενη μεσα στο συγκεκριμενο στοιχειο του πινακα

Η κλαση HashTable περιλαμβανει:

- Hash_Node **array : μεταβλητη που περιλαμβανει τα στοιχεια του πινακα
- int capacity: μεταβλητη που δειχνει ποσες λεξεις χωρανε μεσα στον πινακα
- int size : δειχνει το πληθος των λεξεων που υπαρχουν μεσα στον πινακα τη δεδομενη χρονικη στιγμη
- void insert(Hash_Node* value, int hashIndex) :μεθοδος που εισαγει μια λεξη στον πινακα
- int search2(string value) :μεθοδος που αναζητει αν η συγκεκριμενη λεξη υπαρχει στον πινακα
- int hashFunction(string word) :μεθοδος που επιστρεφει την θεση στην οποια θα μπει η λεξη στον πινακα

- HashTable(int capacity):κατασκευαστης που αρχικοποιει το μεγεθος του πινακα
- void insert(string value):μεθοδος εισαγωγης μια λεξης
- bool search(string aWord):μεθοδος αναζητησης μιας λεξης

Επισης δημιουργηθηκε και η συναρτηση main μεσα από την οποια εκτελουνται οι απαραιτητες διεργασιες με σκοπο να φερουν αποτελεσμα οι παραπανω κλασεις

Η συναρτηση main περιλαμβανει τα εξης δεδομενα:

- ifstream f: αντιπροσωπευει το αρχειο από το οποιο θα εισαχθουν οι λεξεις στις δομες
- string word:αντιπροσωπευει την λεξη που διαβαστηκε από την συγκεκριμενη θεση του αρχειου
- int s: μεταβλητη που δειχνει ποσες λεξεις περιλαμβανει το αρχειο
- string setOfWords[setSize]:αντιπροσωπευει το συνολο που περιλαμβανει τις λεξεις που θα αναζητηθουν από τις 3 δομες
- int randomNumber: χρησιμοποιειται για την τυχαια εισαωγη λεξεων από το αρχειο
- int hashTableTime: αντιπροσωπευει τον συνολικο χρονο που χρειαστηκε για τον πινακα κατακερματισμου να αναζητηθουν οι λεξεις που περιεχονται στο συνολο
- int avlTreeTime: αντιπροσωπευει τον συνολικο χρονο που χρειαστηκε για το δυαδικο δενδρο αναζητησης να αναζητηθουν οι λεξεις που περιεχονται στο συνολο

- int bsTreeTime: αντιπροσωπευει τον συνολικο χρονο που χρειαστηκε για το απλο δυαδικο δενδρο αναζητησης να αναζητηθουν οι λεξεις που περιεχονται στο συνολο
- bstTeeBegin , bstTreeend : αντιπροσπωπευουν τη χρονικη στιγμη εναρξης και ληξης της αναζητησης των λεξεων από το απλο δυαδικο δενδρο αναζητησης
- avlbegin, avlend: αντιπροσπωπευουν τη χρονικη στιγμη εναρξης και ληξης της αναζητησης των λεξεων από το δυαδικο δενδρο αναζητησης τυπου AVL
- hashbegin, hashend: αντιπροσπωπευουν τη χρονικη στιγμη εναρξης και ληξης της αναζητησης των λεξεων από τον πινακα κατακερματισμου