Εργασία εξαμήνου στις Δομές Δεδομένων

Στην εργασία παρουσιάζονται πέντε δομές, οι βασικές λειτουργίες τους και ο χρόνος που χρειάζεται η κάθε δομή να εισάγει τις λέξεις ενός μεγάλου αρχείου κειμένου, να αναζητήσει και να βρει πόσες φορές η κάθε λέξη ενός τυχαίου υποσυνόλου (Q) των λέξεων του αρχείου εμφανίζεται.

Κόστος big O (worst case):

- Αταξινόμητος πίνακας: αναζήτηση O(n), εισαγωγή και διαγραφή O(1)
- Ταξινομημένος πίνακας: αναζήτηση O(log n), εισαγωγή και διαγραφή O(n)
- Δυαδικό δέντρο: αναζήτηση, εισαγωγή και διαγραφή O(h), όπου h το ύψος του
- Δυαδικό δέντρο AVL: αναζήτηση, εισαγωγή και διαγραφή $O(\log_2 h)$, όπου h το ύψος
- Πίνακας κατακερματισμού: αναζήτηση, εισαγωγή και διαγραφή O(n)

1) Αταξινόμητος Πίνακας

Στην πρώτη δομή (αταξινόμητος πίνακας) έχουμε δημιουργήσει έναν κατασκευαστή που θέτει τον πίνακα των λέξεων (wordsUn) και τον πίνακα μετρητή (countUn) σε nullptr. Στην συνέχεια έχουμε κατασκευάσει getter και setter για κάθε μεταβλητή, επιστρέφοντας το μέγεθος του πίνακα, την λέξη στην αντίστοιχη θέση και πόσες φορές εμφανίζεται κάθε λέξη. Έχουμε υλοποιήσει τις συναρτήσεις της σειριακή αναζήτησης των στοιχείων (επιστρέφει -1 αν το στοιχείο δεν υπάρχει) και της διαγράφης. Να σημειωθεί ότι οι πίνακες είναι δυναμικοί μεταβλητού μεγέθους και εισάγουμε στοιχεία μέσω της συνάρτησης Pushbackstring.

2) Ταξινομημένος Πίνακας

Στην δεύτερη δομή (ταξινομημένος πίνακας) έχουμε δημιουργήσει όπως και στην πρώτη όλα τα βασικά στοιχεία που χρειάζεται μια κλάση με την διαφορά ότι τα ονόματα είναι wordsSor και countSor αντίστοιχα. Οι συναρτήσεις που έχουν υλοποιηθεί για την κλάση του αταξινόμητου πίνακα είναι η insertSorted (εισάγει ένα string στην κατάλληλη θέση στον ταξινομημένο πίνακα και αλλάζει και την θέση του μετρητή), sortArray (ταξινομεί τον πίνακα), push_back_string (αυξάνει το μέγεθος των πινάκων κατά ένα), IncreaseSorted (αυξάνει το μέγεθος μια συγκεκριμένης θέσης του πίνακα countSor κατά 1), deleteElements (διαγράφει ένα συγκεκριμένο στοιχείο του πίνακα), binarySearch (που υλοποιεί την λειτουργία της δυαδικής αναζήτησης) και τις συναρτήσεις που εμφανίζουν ένα μεμονωμένο ή όλα τα στοιχεία της δομής και πόσες φορές εμφανίζονται.

3) Δυαδικό Δένδρο Αναζήτησης

Στην τρίτη δομή (δυαδικό δέντρο αναζήτησης) έχουμε υλοποιήσει ένα struct που αναπαριστά έναν κόμβο του δέντρου και περιέχει όλα όσα χρειάζονται για να δημιουργηθεί ένα δυαδικό δέντρο αναζήτησης (ριζά, γονέας, αριστερός δείκτης, δεξιός δείκτης, μετρητής). Στην κλάση btree έχουμε υλοποιήσει της λειτουργίες ενός δυαδικού δέντρου αναζήτησης. Οι λειτουργίες αυτές είναι: insert (εισαγωγή στοιχείου στο δέντρο), search (αναζήτηση στοιχείου του δέντρου), destroy_tree (καταστροφή του δέντρου), PreOrder (εμφάνιση του δέντρου με προδιατεταγμένη διαπέραση), InOrder (εμφάνιση του δέντρου με ενδοδιατεταγμένη διαπέραση), PostOrder (εμφανιση του δέντρου με μεταδιατεταγμένη διαπέραση), findMin (εύρεση ελάχιστου), remove (αφαίρεση κόμβου). Οι συναρτήσεις στο public μέρος της κλάσης καλούν τις συναρτήσεις στο protected για να είναι πιο “κομψό” το πρόγραμμά μας στην main και να αποφευχθεί το πέρασμα της ρίζας στην κλήση των συναρτήσεων από την main (για να μην έχει ο χρήστης πρόσβαση στην ριζά). Επίσης, υπάρχει ο κατασκευαστής που θέτει σε NULL όλους τους δείκτες της ρίζας (ριζά, γονέας, αριστερός δείκτης, δεξιός δείκτης κλπ).

4) Δυαδικό Δένδρο Αναζήτησης AVL

Η τέταρτη δομή (ισοζυγισμένο δέντρο) έχει υλοποιηθεί με τέτοιο τρόπο ώστε να κληρονομεί όλες τις ιδιότητες του δυαδικού δέντρου αναζήτησης. Επιπλέον έχουμε προσθέσει τις συναρτήσεις height (μετράει το ύψος του υποδέντρου, του κόμβου για τον οποίο καλείται), diff (επιστρέφει την διαφορά του ύψους του δεξιού υποδέντρου με εκείνο του αριστερού, του κόμβου για τον οποίο καλείται), balance (ισορροπεί το δέντρο, το φέρνει σε μορφή AVL), και τις απαραίτητες περιστροφές του δέντρου ώστε να έρθει σε ισορροπία (rr_rotation, rl_rotation, lr_rotation, ll_rotation).

5) Πίνακας Κατακερματισμού

Στην πέμπτη δομή (πίνακας κατακερματισμού) έχουμε υλοποιήσει δυο κλάσεις. Η πρώτη κλάση (myPair) χρησιμοποιείται για τα ζευγάρια στον πίνακα κατακερματισμού και περιέχει μόνο το κλειδί (λέξη) και το δεδομένο (αριθμός εμφανίσεων) του κάθε ζευγαριού. Στην δεύτερη κλάση στο private μέρος έχουμε το μέγεθος της δομής, έναν δισδιάστατο πίνακα τύπου myPair, μια συνάρτηση search (δέχεται μία λέξη και επιστρέφει την διεύθυνση του myPair στο οποίο ανήκει) και δύο συναρτήσεις κατακερματισμού που επιστρέφουν την θέση των στοιχείων. Στο public μέρος έχουμε τις συναρτήσεις insert (για την εισαγωγή στοιχείων στην δομή) και την search (αναζήτηση ενός στοιχείου της δομής και επιστροφή μέσω κλήσης με αναφορά του αριθμού των εμφανίσεων της λέξης).

➢ Αρχείο MainFunctions

Σε αυτό το αρχείο υπάρχουν τρείς συναρτήσεις που χρησιμοποιούνται στην main. Η RemoveCaps που μετατρέπει τα κεφαλαία γράμματα σε μικρά. Η RemovePunctuation που αφαιρεί όλα τα σημεία στίξης και η randomNum που επιστρεφει μέσω κλήσης με αναφορά έναν τυχαίο αριθμό μέσα στα όρια που της θέτουμε.

➢ Συνάρτηση main

Στο κύριο μέρος του πρότζεκτ (main) υλοποιούμε ξεχωριστά μία μία τις δομές με την εμφάνιση κατάλληλων μηνυμάτων και την μέτρηση του χρόνου που χρειάζεται κάθε δομή για να εισάγει και να αναζητήσει τα στοιχεία του συνόλου Q στους πίνακες της. Μετά την υλοποίηση του αταξινόμητου πίνακα έχουμε δημιουργήσει μια συνάρτηση που επιλεγεί τυχαίους αριθμούς από το 0 μέχρι το μέγεθος του αταξινόμητου πίνακα και οι αριθμοί αυτοί αντιστοιχούν σε μια λέξη της συγκεκριμένης δομής που την εισάγουμε στον πίνακα Q του οποίου το μέγεθος κυμαίνεται από 800 μέχρι 1200 (γίνεται και εκεί χρήση της παραπάνω συνάρτησης). Στην συνέχεια σε κάθε δομή αναζητούμε την κάθε λέξη του συνόλου Q εμφανίζοντας πόσες φορές υπάρχει. Πριν την εισαγωγή των λέξεων σε κάθε δομή χρησιμοποιούμε τις δυο συναρτήσεις που αφαιρούν τα κεφαλαία γράμματα και τα σημεία στίξης για την ευκολότερη εισαγωγή των στοιχείων. Στο αρχείο εξόδου γράφουμε τον χρόνο που χρειάστηκε κάθε δομή να εισάγει όλες τις λέξεις του αρχείου, γράφουμε τις λέξεις του συνόλου Q καθώς και το πόσες φορές εμφανίζονται και τον χρόνο που χρειάστηκε για να τις αναζητήσει.

Γεώργιος Παππάς

ΑΕΜ 3878

email: pappasgc@csd.auth.gr

Σπυρίδων Καρβούνης

ΑΕΜ 3928

email: skarvoun@csd.auth.gr