Εργαστήριο Δομών Δεδομένων - 2η Άσκηση
Ημερομηνία Παράδοσης : 16/11/2016 (πριν την έναρξη του εργαστηρίου - 11.00 -)

Εργαστηριακός Διδάσκων Μαθήματος: Δούμα Αναστασία (sia@aegean.gr)

Στη δεύτερη εργαστηριακή άσκηση καλείστε να μελετήσετε τη δομή των συνδεδεμένων λιστών (απλή, διπλή και κυκλική) και να αναπτύξετε σχετική εφαρμογή. Στα πλαίσια της 2ης άσκησης θα ασχοληθείτε και με τις δομές δεδομένων στοίβας και ουράς.

Ζήτημα 1 (Βαρύτητα 40%):

Υλοποιήστε την ακόλουθη σειρά συναρτήσεων για τη διαχείριση απλής, διπλής και κυκλικής συνδεδεμένης λίστας:

1. Συνάρτηση με όνομα ***addNodeSingle*** η οποία θα προσθέτει έναν νέο κόμβο σε μια απλή συνδεδεμένη λίστα με ακέραιους αριθμούς. Η συνάρτηση θα δέχεται ως παραμέτρους την λίστα (μόνο το *head*), τα δεδομένα που θα αποθηκευτούν στον κόμβο (ένας ακέραιος αριθμός), και την θέση στην οποία θα γίνει η αποθήκευση (0 στην αρχή της λίστας και 1 στο τέλος της).

2. Συνάρτηση με όνομα ***addAscending*** η οποία θα προσθέτει έναν νέο κόμβο σε μια κατά αύξουσα σειρά ταξινομημένη λίστα (απλή συνδεδεμένη). Η συνάρτηση θα δέχεται δύο παραμέτρους: έναν δείκτη στην κεφαλή της λίστας (*head*) και έναν ακέραιο για τα δεδομένα του νέου κόμβου.

3. Συνάρτηση με όνομα ***countList*** η οποία θα δέχεται ως παράμετρο μία απλή συνδεδεμένη λίστα (μόνο το *head*) και θα επιστρέφει το πλήθος των στοιχείων της.

4. Συνάρτηση με όνομα ***displayList*** η οποία θα δέχεται ως παράμετρο μία απλή συνδεδεμένη λίστα (μόνο το *head*) και θα εμφανίζει τα δεδομένα (ακέραιοι αριθμοί) που είναι αποθηκευμένα στους κόμβους της λίστας.

5. Συνάρτηση με όνομα ***deleteFirst*** η οποία θα δέχεται ως παράμετρο μία απλή συνδεδεμένη λίστα (μόνο το *head*) και θα διαγράφει τον πρώτο κόμβο της λίστας.

6. Συνάρτηση με όνομα ***deleteNode*** η οποία θα δέχεται ως παράμετρο μία απλή συνδεδεμένη λίστα (μόνο το *head*) και έναν ακέραιο αριθμό, θα αναζητά αν ο αριθμός αυτός είναι αποθηκευμένος σε κάποιον κόμβο και αν είναι θα τον διαγράφει.

7. Συνάρτηση με όνομα ***appendLists*** η οποία θα δέχεται ως παράμετρο δύο απλές συνδεδεμένες λίστες (μόνο τα *heads*) οι οποίες είναι ταξινομημένες κατά αύξουσα σειρά και θα της συνενώνει σε μία νέα τρίτη λίστα διατηρώντας τα στοιχεία ταξινομημένα. Η συνάρτηση θα επιστρέφει την νέα λίστα. Οι αρχικές λίστες παραμένουν ως έχουν.

8. Συνάρτηση με όνομα ***addNodeDouble*** η οποία θα προσθέτει έναν νέο κόμβο σε μια διπλή συνδεδεμένη λίστα με ακέραιους αριθμούς. Η συνάρτηση θα δέχεται ως παραμέτρους την λίστα (το *head* και το *tail*), τα δεδομένα που θα αποθηκευτούν στον κόμβο (ένας ακέραιος αριθμός), και την θέση στην οποία θα γίνει η αποθήκευση (0 στην αρχή της λίστας και 1 στο τέλος της).

9. Συνάρτηση με όνομα ***ReversedisplayList*** η οποία θα δέχεται ως παράμετρο μία διπλά διασυνδεδεμένη λίστα (το *head* και το *tail*) και θα εμφανίζει τα δεδομένα (ακέραιοι αριθμοί) που είναι αποθηκευμένα στους κόμβους της λίστας με ανάποδη σειρά (πρώτα το τελευταίο μετά το προτελευταίο, κ.ο.κ).

10. Συνάρτηση με όνομα ***swapElements1*** η οποία θα δέχεται ως παράμετρο μία απλά διασυνδεδεμένη λίστα (το *head*) και θα αντιστρέφει το πρώτο στοιχείο της λίστας με το

τελευταίο της. Σημειώστε στα σχόλια του κώδικα σας ποια είναι η χρονική πολυπλοκότητα της συνάρτησης αυτής ως συνάρτηση του αριθμού *n* στοιχείων της λίστας.

11. Συνάρτηση με όνομα ***swapElements2*** η οποία θα δέχεται ως παράμετρο μία διπλά διασυνδεδεμένη λίστα (το *head* και το *tail*) και θα αντιστρέφει το πρώτο στοιχείο της λίστας με το τελευταίο της. Σημειώστε στα σχόλια του κώδικα σας ποια είναι η χρονική πολυπλοκότητα της συνάρτησης αυτής ως συνάρτηση του αριθμού *n* στοιχείων της λίστας.

12. Συνάρτηση με όνομα ***swapElements3*** η οποία θα δέχεται ως παράμετρο μία απλά διασυνδεδεμένη κυκλική λίστα (το *head*) και θα αντιστρέφει το πρώτο στοιχείο της λίστας με το τελευταίο της. Σημειώστε στα σχόλια του κώδικα σας ποια είναι η χρονική πολυπλοκότητα της συνάρτησης αυτής ως συνάρτηση του αριθμού *n* στοιχείων της λίστας.

Σε όλες τις συναρτήσεις που θα υλοποιήσετε θα πρέπει να εμφανίζονται κατάλληλα μηνύματα σε περίπτωση που μία λειτουργία δεν μπορεί να εκτελεστεί αλλά και μηνύματα ενημέρωσης του χρήστη κατά τη διεκπεραίωση μιας λειτουργίας.

Υλοποιήστε κυρίως πρόγραμμα στο οποίο θα καλείτε τις συναρτήσεις που υλοποιήσατε με σκοπό να επιδείξετε την ορθή λειτουργίας τους. Για αυτό το σκοπό δεν απαιτείται η εισαγωγή στοιχείων από το χρήστη. Δημιουργήστε όσες λίστες απαιτούνται με όποια μορφή απαιτείται για να ελέγξετε τις συναρτήσεις (απλή, διπλή, ταξινομημένη, κυκλική).

Ζήτημα 2 (Βαρύτητα 45%):
Να υλοποιήσετε ένα πρόγραμμα το οποίο θα διαχειρίζεται μια διπλά διασυνδεδεμένη λίστα η οποία χρησιμοποιείται για την αποθήκευση των στοιχείων μιας συλλογής από βιβλία. Για κάθε βιβλίο καταχωρούμε τον τίτλο του, τον συγγραφέα του (θεωρούμε ότι είναι μόνο ένας), το είδος του και μια απλή περιγραφή. Για κάθε βιβλίο επίσης θα καταχωρείται και μια μεταβλητή η οποία ορίζει το πόσο δημοφιλές είναι ένα βιβλίο. Μόλις εισάγεται ένα βιβλίο στην λίστα η μεταβλητή αυτή θα αρχικοποιείται με 0.

Η ιδιαιτερότητα της συγκεκριμένης λίστας είναι ότι τα βιβλία θα τοποθετούνται στην κατάλληλη θέση στην λίστα ανάλογα με το πόσο δημοφιλή είναι αλλά και σε $2^{ο}$ επίπεδο με βάση τον τίτλο τους. Δηλαδή στην αρχή της λίστας βρίσκεται πάντα αποθηκευμένο το βιβλίο που είναι πιο δημοφιλές από όλα και ο τίτλος του αρχίζει με το γράμμα Α. Στο τέλος της λίστας αποθηκεύονται όλα τα βιβλία που καταχωρηθήκανε για πρώτη φορά στην συλλογή (η μεταβλητή που ορίζει το πόσο δημοφιλές είναι 0) αλλά πάντα ταξινομημένα με βάση τον τίτλο.

Το πόσο δημοφιλές είναι ένα βιβλίο διαμορφώνεται από το πόσες φορές εκτελείται για αυτό το βιβλίο η λειτουργία της αναζήτησης και προβολής των στοιχείων του. Όσο δηλαδή οι χρήστες διαβάζουν τα στοιχεία του τόσο πιο δημοφιλές θεωρείται. Κάθε φορά που εκτελείται η λειτουργία αυτή αυξάνεται η βαθμολογία του κατά 0.5. ***Προσοχή***: Όταν θα αλλάζει η δημοτικότητα του βιβλίου πρέπει να επανατοποθετείται το βιβλίο στην σωστή θέση στην λίστα με βάση τα κριτήρια που προαναφέραμε.

Η διαχείριση της συλλογής των βιβλίων θα πρέπει να γίνεται μέσω του παρακάτω μενού επιλογών:

- Εισαγωγή βιβλίου στην λίστα (Δεν πρέπει να υπάρχει ήδη ο τίτλος του βιβλίου στην λίστα)
- Αναζήτηση και εμφάνιση στοιχείων βιβλίου με συγκεκριμένο τίτλο (αύξηση της δημοτικότητας του)
- Διαγραφή βιβλίου με συγκεκριμένο τίτλο

- Εμφάνιση πλήθους βιβλίων στην λίστα
- Εμφάνιση βιβλίων με την μεγαλύτερη βαθμολογία/πιο δημοφιλή (μπορεί να υπάρχουν πολλά βιβλία που έχουν την μέγιστη βαθμολογία)
- Εμφάνιση των βιβλίων με την ελάχιστη δημοτικότητα
- Εμφάνιση ολόκληρης της συλλογής των βιβλίων (με τη σειρά που είναι τοποθετημένα στη λίστα)

Είναι επιθυμητό οι περισσότερες λειτουργίες της εφαρμογής να υλοποιηθούν σε σχετικές συναρτήσεις.

Ζήτημα 3 (Βαρύτητα 15%):
Υλοποιήστε πρόγραμμα το οποίο θα διαβάζει έναν δεκαδικό αριθμό που ανήκει στο διάστημα από [1-255] και θα τον μετατρέπει σε δυαδικό χρησιμοποιώντας ως δομή δεδομένων για αποθήκευση των δυαδικών ψηφίων μία στοίβα. Πιο αναλυτικά:

- Θα διαβάζει τον αριθμό και θα ελέγχει την εγκυρότητα του.
- Θα δημιουργεί δυναμικά την δομή της στοίβας (για τη διαχείριση της στοίβας χρησιμοποιήστε ως δομή δεδομένων έναν πίνακα).
- Θα μετατρέπει τον δεκαδικό αριθμό ως ένα σύνολο από δυαδικά ψηφία και θα τα τοποθετεί στη στοίβα εκτελώντας την λειτουργία της ώθησης.
- Χρησιμοποιώντας τη δομή δεδομένων της στοίβας και εκτελώντας την λειτουργία της απώθησης θα εμφανίζει τον δυαδικό αριθμό στον οποίο αντιστοιχεί ο αρχικός αριθμός.

Υλοποιήστε σε ξεχωριστές συναρτήσεις τις λειτουργίες ώθησης και απώθησης.

Παραδοτέα:
α) Τον κώδικα που θα έχετε υλοποιήσει μέσα στο εργαστηρίου θα πρέπει να τον παραδώσετε με την ολοκλήρωση του εργαστηρίου (ισχύει μόνο για όσους παρακολουθούν το εργαστήριο). Υπάρχει σχετικός σύνδεσμος στο eclass. Το αρχείο που θα ανεβάσετε θα πρέπει να έχει όνομα ***Exercise2_lab.cpp (ή .c - θα πρέπει να το κάνετε .zip για να το ανεβάσετε). Στην 1η γραμμή του κώδικα σας θα πρέπει υποχρεωτικά να αναγράφεται το όνομα και ο αριθμός μητρώου των μελών της ομάδας που συμμετείχε στην υλοποίηση που έγινε στο εργαστήριο.***

(β) Τελική παράδοση εργασίας :
Για την τελική παράδοση της εργασίας θα πρέπει να δημιουργήσετε **τρία αρχεία** με τον πηγαίο κώδικα για κάθε ένα από τα ζητήματα της άσκησης και όνομα ***Exercise21.cpp (ή .c), Exercise22.cpp (ή .c)*** και ***Exercise23.cpp (ή .c)***.

Τέλος θα δημιουργήσετε ένα zip αρχείο με όνομα αρχείου «το email σας_όνομα άσκησης» (πχ. icsd000666_Exercise2.zip) (μόνο ένα μέλος της ομάδας θα πρέπει να καταγράψει το email του). Το zip αρχείο θα πρέπει να περιέχει τα παραπάνω τρία αρχεία. ΠΡΟΣΟΧΗ: ΚΑΜΙΑ ΕΡΓΑΣΙΑ ΔΕΝ ΘΑ ΔΙΟΡΘΩΘΕΙ ΑΝ ΔΕΝ ΑΠΟΣΤΑΛΕΙ ΜΕ ΤΟΝ ΠΑΡΑΠΑΝΩ ΤΡΟΠΟ!

Η παράδοση του αρχείου θα γίνει με τη χρήση της πλατφόρμας ηλεκτρονικής μάθησης του τμήματος (http://www.icsd.aegean.gr/eclass).

Σε όλα τα αρχεία που θα δημιουργήσετε (πηγαίος κώδικας και pdf) θα πρέπει να αναφέρετε στην αρχή του κειμένου τα ονόματα των μελών της ομάδας εργασίας.

Υποδείξεις. Η εργασία είναι ***2 ατόμων.*** Τα προγράμματα πρέπει να υλοποιηθούν σε γλώσσα C ή C++.