HY240: Δομές Δεδομένων
Χειμερινό Εξάμηνο – Ακαδημαϊκό Έτος 2016-17
Παναγιώτα Φατούρου

2ο Σετ Ασκήσεων

Ημερομηνία Παράδοσης: Δευτέρα, 24 Οκτωβρίου 2016
Τρόπος Παράδοσης: Οι ασκήσεις μπορούν να παραδοθούν είτε σε ηλεκτρονική μορφή χρησιμοποιώντας το πρόγραμμα turnin (δείτε οδηγίες στην ιστοσελίδα του μαθήματος), είτε στους βοηθούς του μαθήματος στο εργαστήριο των μεταπτυχιακών, τη Δευτέρα, 24 Οκτωβρίου 2016, ώρα 15:00-16:00. Ασκήσεις που παραδίδονται μετά τις 16:00 της Δευτέρας, 24/10/2016 θεωρούνται εκπρόθεσμες. Εκπρόθεσμες ασκήσεις γίνονται δεκτές σε ηλεκτρονική μορφή και η αποστολή τους πρέπει να γίνει χρησιμοποιώντας το πρόγραμμα turnin. Εναλλακτικά, εκπρόθεσμες ασκήσεις μπορούν να παραδοθούν στη διδάσκουσα.

Άσκηση 1 [30 μονάδες]

Έστω ότι μια βιβλιοθήκη σας παρέχει πρόσβαση σε στοίβες και ουρές χαρακτήρων. Η βιβλιοθήκη σας επιτρέπει να ορίσετε μια στοίβα (ή μια ουρά) και να καλέσετε τις 5 βασικές λειτουργίες σε αυτή. Για παράδειγμα, ο ορισμός μιας στοίβας (ή μιας ουράς) γίνεται γράφοντας: Stack S; (αντίστοιχα, Queue Q;). Για τη στοίβα υποστηρίζονται οι εξής λειτουργίες: (1) void MakeEmptyStack(Stack S), (2) boolean IsEmptyStack(Stack S), (3) Type Top(Stack S), (4) Type Pop(Stack S), (5) void Push(Stack S, char x). Αντίστοιχα, για την ουρά υποστηρίζονται οι λειτουργίες: (1) void MakeEmptyQueue(Queue Q), (2) boolean IsEmptyQueue(Queue Q), (3) char Front(Queue Q), (4) char Dequeue(Queue Q), (5) void Enqueue(Queue Q, char x).

Επιλύστε τα προβλήματα που παρουσιάζονται στη συνέχεια:

α. Στην προδιατεταγμένη παράσταση, κάθε πράξη εφαρμόζεται στους δύο τελεστές που την ακολουθούν. Για παράδειγμα, '+ 1 2' αναπαριστά το άθροισμα του αριθμού 1 με τον αριθμό 2. Ο ορισμός είναι αναδρομικός και επομένως κάθε πράξη μπορεί να αναφέρεται σε τελεστές που είναι το αποτέλεσμα άλλων πράξεων. Π.χ., '* + 1 2 – 3 4' είναι η προδιατεταγμένη αναπαράσταση της '(1+2)*(3-4)'. Γράψτε μια συνάρτηση που θα διαβάζει μια έκφραση σε προδιατεταγμένη μορφή και θα επιστρέφει το αποτέλεσμα του υπολογισμού της. [15M]

β. Παρουσιάστε ψευδοκώδικα για μια συνάρτηση η οποία θα ταξινομεί τα στοιχεία μιας ουράς χρησιμοποιώντας το πολύ δύο επιπρόσθετες ουρές. Χρησιμοποιήστε την τεχνική της BubbleSort() για την ταξινόμηση. Θεωρήστε πως τα στοιχεία της αρχικής ουράς είναι ακέραιοι. Ο αλγόριθμός σας μπορεί να χρησιμοποιεί ένα σταθερό πλήθος από ακέραιες μεταβλητές. Ποια είναι η χρονική πολυπλοκότητα του αλγορίθμου σας; [15M]

Σημειώσεις: Ι. Οι αλγόριθμοι και για τα δύο ερωτήματα θα πρέπει να χρησιμοποιούν δομές δεδομένων από τη βιβλιοθήκη που σας παρέχεται, καθώς και ενδεχόμενα κάποιες βοηθητικές μεταβλητές. Δεν επιτρέπεται η χρήση πινάκων ή άλλων δομών δεδομένων που δεν ανήκουν στη βιβλιοθήκη για την αποθήκευση ή την επεξεργασία των αλφαριθμητικών ή των ακολουθιών χαρακτήρων. Δεν γνωρίζετε αν οι στοίβες και οι ουρές που παρέχει η βιβλιοθήκη υλοποιούνται με στατικό ή με δυναμικό τρόπο. ΙΙ. Θεωρήστε πως σας παρέχεται μια βοηθητική ρουτίνα char getchar() για την ανάγνωση αριθμών ή χαρακτήρων και ότι κάθε προδιατεταγμένη παράσταση τελειώνει με το χαρακτήρα '\n'.

Άσκηση 2 [40 Μονάδες]

α. Θεωρήστε λίστα της οποίας το κάθε στοιχείο είναι ένα struct με 3 πεδία: (1) έναν ακέραιο id, (2) έναν ακέραιο distance, και (3) ένα δείκτη next στο επόμενο στοιχείο της λίστας. Παρουσιάστε ψευδοκώδικα που θα επιλύει το ακόλουθο πρόβλημα στη λίστα. Δεδομένου ενός id, ο αλγόριθμός σας θα πρέπει να εξετάζει αν υπάρχει στοιχείο με τέτοιο id στη λίστα (αν όχι, τυπώνετε ένα μήνυμα). Αν το στοιχείο υπάρχει στη λίστα (έστω ότι είναι το e και έστω ότι η τιμή του πεδίου distance του e είναι d), ο αλγόριθμός σας θα πρέπει να βρίσκει και να επιστρέφει το id και την τιμή του πεδίου distance του στοιχείου που προηγείται του e κατά d θέσεις στη λίστα. Παρουσιάστε ψευδοκώδικα που να επιλύει το παραπάνω πρόβλημα χρησιμοποιώντας:

i) δύο δείκτες που μπορούν να κινηθούν μόνο προς τα εμπρός στη λίστα [7M]

ii) μια ``zig-zag'' διάσχιση χρησιμοποιώντας μια στοίβα από δείκτες (προσοχή: θα πρέπει να παρουσιάστε την υλοποίηση της στοίβας καθώς και όλα τα structs που θα χρησιμοποιήσετε). [12M]

iii) τη μέθοδο αναστροφής δεικτών λίστας [11M]

β. Επιλύστε το ίδιο πρόβλημα θεωρώντας πως η λίστα είναι διπλά συνδεδεμένη, δηλαδή κάθε κόμβος της έχει ένα επιπρόσθετο πεδίο prev, που είναι δείκτης στο προηγούμενο στοιχείο στη λίστα. [10M]

Άσκηση 3 [30 μονάδες]

α. Θεωρήστε τον αφηρημένο τύπο δεδομένων ΔΛ με τις ακόλουθες λειτουργίες: [15M]

i. Insert(x,L,j): εισάγει το κλειδί x μετά το j-οστό στοιχείο της L.

ii. Delete(L,j): διαγράφει το j-οστό στοιχείο της L και επιστρέφει το κλειδί του.

Παρουσιάστε έναν **αναδρομικό αλγόριθμο** που θα υλοποιεί τη λειτουργία Insert() μιας ΔΛ και έναν ακόμη **αναδρομικό αλγόριθμο** που θα υλοποιεί τη Delete() σε μια τέτοια λίστα.

β. Παρουσιάστε αλγόριθμο ο οποίος δεδομένων δύο λιστών L1, L2 με τον ίδιο αριθμό στοιχείων (έστω ότι ο αριθμός αυτός είναι 2n, δηλαδή είναι άρτιος) θα δημιουργεί μια απλά-συνδεδεμένη λίστα L3 με επίσης 2n στοιχεία για την οποία θα πρέπει να ισχύουν τα εξής. Τα στοιχεία που βρίσκονται στις περιττές θέσεις της L3 είναι τα ίδια με τα στοιχεία που βρίσκονται στις περιττές θέσεις της L1, ενώ τα στοιχεία που βρίσκονται στις άρτιες θέσεις της L3 είναι τα ίδια με εκείνα που βρίσκονται στις τελευταίες n θέσεις της L2 (με σειρά από το τέλος προς την αρχή, δηλαδή το $2^{ο}$ στοιχείο της L3 είναι το τελευταίο της L2, το $4^{ο}$ στοιχείο της L3 είναι το προτελευταίο της L2, κ.ο.κ.). Θεωρήστε πως η L1 είναι απλά-συνδεδεμένη, ενώ η L2 είναι διπλά-συνδεδέμενη. **Ο αλγόριθμός σας θα πρέπει να εκτελείται σε χρόνο O(n).** [15M]