HY240: Δομές Δεδομένων
Χειμερινό Εξάμηνο – Ακαδημαϊκό Έτος 2011-12
Παναγιώτα Φατούρου

2ο Σετ Ασκήσεων

Ημερομηνία Παράδοσης: Δευτέρα, 7 Νοεμβρίου 2011
Τρόπος Παράδοσης: Οι ασκήσεις μπορούν να παραδοθούν είτε σε ηλεκτρονική μορφή χρησιμοποιώντας το πρόγραμμα submit (δείτε οδηγίες στην ιστοσελίδα του μαθήματος), είτε στους βοηθούς του μαθήματος στο εργαστήριο των μεταπτυχιακών στο υπόγειο των λευκών κτιρίων, τη Δευτέρα, 7 Νοεμβρίου 2011, ώρα 10:00-11:00. Ασκήσεις που παραδίδονται μετά τις 11:00 της Δευτέρας, 7/11/2011 θεωρούνται εκπρόθεσμες. Εκπρόθεσμες ασκήσεις γίνονται δεκτές σε ηλεκτρονική μορφή και η αποστολή τους πρέπει να γίνει χρησιμοποιώντας το πρόγραμμα submit. Εναλλακτικά, εκπρόθεσμες ασκήσεις μπορούν να παραδοθούν στη διδάσκουσα του μαθήματος ή να σταλούν στο hy240b@csd.uoc.gr μέσω ηλεκτρονικού ταχυδρομείου.

ΠΡΟΣΟΧΗ: Παραδίδουμε τις ασκήσεις στο βοηθό του μαθήματος ή στη διδάσκουσα ή χρησιμοποιούμε το πρόγραμμα submit. Δεν τοποθετούμε τις ασκήσεις σε κουτιά που βρίσκονται στο υπόγειο των λευκών κτιρίων (ακόμη και αν σε αυτά αναγράφονται πληροφορίες που φαίνονται σχετικές με το μάθημα και την τρέχουσα σειρά ασκήσεων), αφού αυτό δεν εξασφαλίζει την ασφάλεια τους. Ασκήσεις που θα βρεθούν σε κουτιά δεν θα γίνουν δεκτές.

Άσκηση 1 [30 μονάδες]

Οι αριθμητικές εκφράσεις αποτελούνται από τελεστές και μεταβλητές ή τελεστέους. Στην άσκηση αυτή εστιάζουμε σε τελεστές που είναι δυαδικοί (δηλαδή εφαρμόζονται σε δύο μεταβλητές). Ενδοθεματική (infix) ονομάζεται η μορφή παράστασης αριθμητικών εκφράσεων στην οποία ο τελεστής εμφανίζεται μεταξύ των δύο μεταβλητών. Στην μεταθεματική (postfix) μορφή παράστασης κάθε τελεστής εμφανίζεται μετά το ζεύγος των αντίστοιχων μεταβλητών, ενώ στην προθεματική ο τελεστής εμφανίζεται πριν το ζεύγος των δύο μεταβλητών.

Για παράδειγμα, η αριθμητική έκφραση (((a*(b+c))+(((d*e)+f)*(g+h)))*i) είναι σε ενδοθεματική μορφή. Η μεταθεματική ισοδύναμη έκφραση αυτής είναι abc+*de*f+gh+*+i*.

Ζητείται αλγόριθμος που να διαβάζει την ενδοθεματική (infix) μορφή μιας αριθμητικής έκφρασης και να τυπώνει τη μεταθεματική (postfix) μορφή της. Για παράδειγμα, όταν ο αλγόριθμος παίρνει ως είσοδο την αριθμητική έκφραση (((a*(b+c))+(((d*e)+f)*(g+h)))*i) θα πρέπει να τυπώνει abc+*de*f+gh+*+i*.

Ο αλγόριθμος μπορεί να χρησιμοποιεί στοίβες και ουρές.

Παρατήρηση: Θεωρήστε ότι σας παρέχεται μια βιβλιοθήκη στοιβών και ουρών. 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, Type x). Αντίστοιχα, για την ουρά υποστηρίζονται οι λειτουργίες: (1) void MakeEmptyQueue(Queue Q), (2) boolean IsEmptyQueue(Queue Q), (3) Type Front(Queue Q), (4) Type Dequeue(Queue Q), (5) void Enqueue(Queue Q, Type x).

Άσκηση 2 [40 μονάδες]

1. Παρουσιάστε αποτελεσματική υλοποίηση τεσσάρων στοιβών χρησιμοποιώντας έναν μόνο πίνακα Ν στοιχείων. Πιο συγκεκριμένα, θα πρέπει να παρέχετε ψευδοκώδικα για τις λειτουργίες MakeEmptyStack(), IsEmptyStack(), Top(), Pop() και Push() για τις στοίβες αυτές. Οι πρώτες δύο στοίβες θα πρέπει να υλοποιηθούν στο πρώτο μισό του πίνακα ενώ οι άλλες δύο στοίβες στο δεύτερο μισό του πίνακα. Μια στοίβα δεν θα πρέπει να προκαλεί υπερχείλιση αν το μέγεθος των συνολικών στοιχείων που υπάρχουν σε αυτήν και στην στοίβα που υλοποιείται στο ίδιο μισό του πίνακα με αυτήν είναι μικρότερο του N/2.

   Υπόδειξη: Κάθε μια από τις λειτουργίες που ζητείται να υλοποιηθούν θα πρέπει να παίρνει ως παράμετρο έναν αριθμό που να υποδηλώνει σε ποια στοίβα θα πρέπει να εκτελεστεί η εν λόγω λειτουργία.

2. Παρουσιάστε αλγόριθμο ο οποίος δεδομένων δύο λιστών L_1 με n_1 στοιχεία και L_2 με n_2 στοιχεία θα δημιουργεί μια άλλη λίστα L_3 με $(n_1+n_2)/2$ στοιχεία για την οποία θα πρέπει να ισχύουν τα εξής. Τα στοιχεία στις περιττές θέσεις (1^o, 3^o, 5^o, κλπ.) της L_3 είναι εκείνα που βρίσκονται στις άρτιες θέσεις της L_1, ενώ τα στοιχεία που βρίσκονται στις άρτιες θέσεις της L_3 είναι τα τελευταία $n_1/2$ στοιχεία της L_2. Τα n_1 και n_2 μπορούν να είναι οποιοιδήποτε άρτιοι αριθμοί (π.χ., 0, 2, 4, κλπ.). Ο αλγόριθμός σας θα πρέπει να εκτελείται σε χρόνο $O(n_1+n_2)$.

   Παράδειγμα: Έστω πως $L_1 = 1 \rightarrow 5 \rightarrow 12 \rightarrow 7 \rightarrow 3 \rightarrow 9$ και $L_2 = 2 \rightarrow 6 \rightarrow 10 \rightarrow 4$. Τότε θα πρέπει μετά το πέρας της εκτέλεσης του αλγορίθμου $L_3 = 5 \rightarrow 10 \rightarrow 7 \rightarrow 4 \rightarrow 9$.

   Υπόδειξη: Θεωρήστε πως $n_2 > n_1$**.** Η άσκηση θα θεωρηθεί σωστή και αν θεωρήσετε πως $n_1 = n_2$.

Άσκηση 3 [30 μονάδες]

Θεωρήστε τη λειτουργία Subset(S_1, S_2) (υποσύνολο), όπου S_1 και S_2 είναι δύο σύνολα. Παρουσιάστε αλγόριθμο που να υλοποιεί την Subset δεδομένου ότι τα σύνολα υλοποιούνται με:

1. **Ταξινομημένες λίστες.** Ο αλγόριθμος σας θα πρέπει να έχει πολυπλοκότητα $O(n_1+ n_2)$, όπου n_1 και n_2 είναι το πλήθος στοιχείων των συνόλων S_1 και S_2. Εξηγήστε γιατί ο αλγόριθμός σας επιτυγχάνει την πολυπλοκότητα αυτή.
2. **Μη-ταξινομημένες λίστες.** Τι πολυπλοκότητα έχει ο αλγόριθμός σας και γιατί;

Η Subset πρέπει να παίρνει σαν όρισμα έναν δείκτη στο πρώτο στοιχείο της λίστας που υλοποιεί το S_1 και έναν δείκτη στο πρώτο στοιχείο της λίστας που υλοποιεί το S_2. Θα πρέπει να επιστρέφει TRUE αν το ένα σύνολο είναι υποσύνολο του άλλου και FALSE διαφορετικά.