Η/Υ 432: Δομές Δεδομένων

Χειμερινό Εξάμηνο Ακαδημαϊκού Έτους 2007-2008

Παναγιώτα Φατούρου – Ευριπίδης Μάρκου

2ο Σετ Ασκήσεων

Ημερομηνία Παράδοσης: Παρασκευή, 23/11, 12:00-14:00.

Τρόπος Παράδοσης: Οι ασκήσεις παραδίδονται στους βοηθούς του μαθήματος το αργότερο μέχρι την ημερομηνία παράδοσης. Το γραφείο των βοηθών του μαθήματος είναι το Α25.

Άσκηση 1

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) char Top(Stack S) (4) char 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).

Θεωρήστε πως στη βιβλιοθήκη έχει δηλωθεί ένας πίνακας S από στοίβες: stack S[1..N], καθώς και ένας πίνακας Q από ουρές: queue Q[1..N], τις οποίες μπορείτε να χρησιμοποιείτε χωρίς να τις δηλώσετε. Θεωρήστε επίσης ότι το N είναι αρκετά μεγάλο ώστε να καλύπτει τις ανάγκες σε στοίβες και ουρές των αλγορίθμων που ζητούνται στη συνέχεια.

i. Παλίνδρομα Αλφαριθμητικά

Ένα αλφαριθμητικό a ονομάζεται παλίνδρομο αν έχει τη μορφή a = wcw', όπου w' είναι το αντίστροφο αλφαριθμητικό του w και c είναι ένας χαρακτήρας. Για παράδειγμα, το αντίστροφο αλφαριθμητικό του «yam» είναι το «may». Έτσι, π.χ., το αλφαριθμητικό «yamamay" είναι παλίνδρομο. Να παρουσιαστεί αλγόριθμος που θα διαβάζει ένα αλφαριθμητικό από το πληκτρολόγιο και θα ελέγχει αν αυτό είναι παλίνδρομο.

ii. Ταξινόμηση ακολουθίας χαρακτήρων

Σχεδιάστε αλγόριθμο ο οποίος θα διαβάζει μια ακολουθία από χαρακτήρες από το πληκτρολόγιο (που δεν είναι απαραίτητα αλφαβητικά ταξινομημένη) και θα χρησιμοποιεί ουρές για να τυπώσει τους χαρακτήρες αυτούς σε (αλφαβητικά) αύξουσα διάταξη. Ο αλγόριθμος θα πρέπει να χρησιμοποιεί το μικρότερο δυνατό πλήθος ουρών.

Υπόδειξη: Ο αλγόριθμος θα πρέπει να τοποθετεί προσεκτικά χαρακτήρες από την είσοδο στις διάφορες ουρές, έτσι ώστε η υπακολουθία χαρακτήρων που βρίσκεται σε κάθε ουρά να είναι ταξινομημένη. Ο αλγόριθμος θα πρέπει να χρησιμοποιεί μια καινούρια ουρά μόνο εάν αυτό είναι απαραίτητο.

Προσοχή: 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.

Άσκηση 2

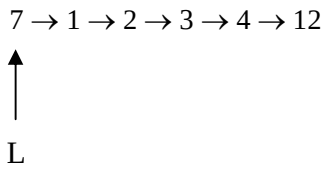
Δίνεται η ακόλουθη υλοποίηση για αραιούς δισδιάστατους πίνακες. Κάθε αραιός δισδιάστατος πίνακας A[1..n, 1..m] υλοποιείται με έναν boolean δισδιάστατο πίνακα B[1..n, 1..m] (κάθε στοιχείο του B έχει τιμή 0 ή 1) και μια δυναμική συνδεδεμένη λίστα που περιέχει τα μη-μηδενικά στοιχεία. Για τον boolean δισδιάστατο πίνακα ισχύουν τα ακόλουθα. Για κάθε i, j, με $1 \le i \le n$, $1 \le j \le m$, B[i][j] = 1 αν και μόνο αν A[i][j] ≠ 0, και B[i][j] = 0 αν και μόνο αν A[i][j] = 0. Τα μη-μηδενικά στοιχεία στη λίστα αποθηκεύονται ως εξής: «τα μη-μηδενικά στοιχεία οποιασδήποτε γραμμής i απαντώνται στη λίστα πριν από τα μη-μηδενικά στοιχεία οποιαδήποτε γραμμής j > i και τα μη-μηδενικά στοιχεία οποιασδήποτε στήλης k απαντώνται πριν από τα μη-μηδενικά στοιχεία οποιασδήποτε στήλης l > k». Για παράδειγμα, δίνεται ο αραιός πίνακας A:

0	7	0	0	0
1	2	0	0	3
0	0	4	0	0
12	0	0	0	0

Ο πίνακας B που αντιστοιχεί στον A είναι ο ακόλουθος:

0	1	0	0	0
1	1	0	0	1
0	0	1	0	0
1	0	0	0	0

και η λίστα έχει την εξής μορφή:



Παρουσιάστε αλγόριθμους που θα υλοποιούν τις ακόλουθες λειτουργίες στον αραιό πίνακα:

i. **Insert(***int x, int i, int j***):** εισάγει το στοιχείο x στη i-οστή γραμμή και τη j-οστή στήλη του αραιού πίνακα.
ii. **Delete(***int x***):** αναζητά το x στον αραιό πίνακα και αν υπάρχει το διαγράφει.
iii. **AddArrays(***sparse_table A1, A2, A3***):** προσθέτει τους αραιούς πίνακες A1 και A2, δηλαδή δημιουργεί έναν νέο αραιό πίνακα A3 για τον οποίο ισχύει ότι A3[i][j] = A1[i][j] + A2[i][j], ∀ i,j με $1 \le i \le n$ και $1 \le j \le m$. Θεωρούμε πως όλοι οι αραιοί πίνακες, A1, A2 και A3, υλοποιούνται με τον τρόπο που περιγράφτηκε πιο πάνω.