ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Δομές Δεδομένων - Εργασία 3

Τμήμα Πληροφορικής

Φθινοπωρινό Εξάμηνο 2017-2018

Διδάσκων: Ε. Μαρκάκης

Δέντρα Δυαδικής Αναζήτησης

Σκοπός της εργασίας αυτής είναι η εξοικείωση με τα δέντρα δυαδικής αναζήτησης και η υλοποίηση πίνακα συμβόλων με τυχαιοποιημένα δέντρα (randomized binary search trees).

Ένας μεγάλος εκδοτικός οίκος προμηθεύει βιβλία σε βιβλιοπωλεία και διάφορα άλλα καταστήματα σε πόλεις και χωριά της Ελλάδας. Για την καλύτερη προμήθεια και εξυπηρέτηση των καταστημάτων, η εταιρεία διατηρεί μεγάλες αποθήκες με τα βιβλία που εκδίδει σε κάποια κομβικά σημεία της χώρας. Η εταιρεία ανά τακτά διαστήματα αποφασίζει αν χρειάζεται να δημιουργήσει νέες αποθήκες (είτε σε νέες πόλεις ή σε πόλεις όπου ήδη υπάρχει κάποια αποθήκη) καθώς και αν χρειάζεται να κλείσει κάποια αποθήκη. Επομένως είναι αναγκαίο να υπάρχει μία δομή δεδομένων που να διατηρεί όλες τις απαραίτητες πληροφορίες για τα βιβλία που υπάρχουν σε κάθε αποθήκη. Μετά από συζητήσεις με τους αναλυτές λογισμικού, η εταιρεία αποφασίζει ότι η υλοποίηση του πίνακα συμβόλων πρέπει να γίνει με δυαδικά δέντρα αναζήτησης και πιο συγκεκριμένα με τυχαιοποιημένα δέντρα.

Ο στόχος είναι να υλοποιήσετε τον πίνακα συμβόλων με την μορφή τυχαιοποιημένου δυαδικού δέντρου. Η δομή σας θα πρέπει να έχει τα πεδία και τις μεθόδους που φαίνονται παρακάτω (ακολουθούν επεξηγήσεις):

```
class ST    {
private class TreeNode {
Κάθε αποθήκη θα αντιστοιχεί σε ένα αντικείμενο TreeNode
// τα πεδία που πρέπει να έχει η TreeNode αναλύονται παρακάτω
...
};
private TreeNode head; //ρίζα στο δέντρο των αποθηκών

void insertWarehouse(int nodeid, String name);
```

```
void insertBookAtWarehouse(int nodeid, int isbn, int copies);
void removeWarehouse(int nodeid);
void removeBook(int nodeid, int isbn);
void searchByWarehouse(int nodeid);
void searchBookInWarehouse(int nodeid, int isbn);
void searchBook(int isbn);
void printTree(PrintStream stream);
}
```

Μέρος Α: Προτού υλοποιήσετε τον πίνακα συμβόλων θα πρέπει να υλοποιήσετε τους εξής 2 τύπους δεδομένων.

Η κλάση TreeNode

Κάθε αποθήκη αντιστοιχεί σε έναν κόμβο του δέντρου, δηλαδή σε ένα αντικείμενο τύπου TreeNode. Κάθε κόμβος έχει ένα μοναδικό κωδικό (int id) και επίσης το όνομα της πόλης/περιοχής που βρίσκεται (είναι επιτρεπτό να υπάρχουν πολλές αποθήκες στην ίδια πόλη αλλά πρέπει να έχουν διαφορετικό κωδικό). Το δέντρο θα δημιουργείται **με κλειδί** το `nodeid.` Επομένως η κλάση Treenode πρέπει να περιέχει τουλάχιστον τα εξής πεδία (και ενδεχομένως ό,τι άλλο θέλετε εσείς να προσθέσετε):

```
private class Treenode {
   int id  // unique id of the node
   String city //city where the node is located
   Treenode l // pointer to left subtree
   Treenode r // pointer to right subtree
   int N  //number of nodes in the subtree starting at this TreeNode
   List booklist // sorted linked list of the books
                 // stored  in this TreeNode
}
```

Η κλάση BookInfo και το πεδίο booklist της Treenode

Το πεδίο booklist της TreeNode είναι μία ταξινομημένη λίστα μονής σύνδεσης που θα περιέχει αντικείμενα τύπου **BookInfo**. Η λίστα αυτή περιέχει όλα τα βιβλία που υπάρχουν στη συγκεκριμένη αποθήκη. Η κλάση BookInfo αναπαριστά τις πληροφορίες για ένα βιβλίο. Συγκεκριμένα πρέπει να περιέχει το ISBN του βιβλίου (International Standard Book Number) και τον αριθμό των αντιτύπων που είναι διαθέσιμα στη συγκεκριμένη αποθήκη. Για ευκολία θεωρήστε ότι το ISBN είναι ακέραιος αριθμός στο διάστημα [0, 9999] (στην πράξη το ISBN έχει 10 ή 13 ψηφία). Η λίστα θα έχει ένα αντικείμενο BookInfo για κάθε διαθέσιμο βιβλίο στην αποθήκη (όχι για κάθε αντίτυπο, ο αριθμός αντιτύπων είναι πεδίο της κλάσης). Η λίστα είναι ταξινομημένη σε αύξουσα σειρά ως προς το πεδίο ISBN και δεν επιτρέπεται να υπάρχουν παραπάνω από ένα βιβλία με το ίδιο ISBN.

Μέρος Β: Οι μέθοδοι του πίνακα συμβόλων:

Μέθοδοι εισαγωγής/διαγραφής:

```
void insertWarehouse(int nodeid, String name);
```

Η μέθοδος αυτή εισάγει μια νέα αποθήκη με κωδικό `nodeid` και όνομα πόλης `name` (ισοδυναμεί με την προσθήκη κόμβου στο δέντρο). Εάν υπάρχει ήδη αποθήκη με τον ίδιο κωδικό θα πρέπει να τυπώνει μήνυμα λάθους και να τερματίζει. Είναι επιτρεπτό να υπάρχουν πολλές αποθήκες στην ίδια πόλη (πολλοί κόμβοι στο δέντρο με ίδιο name, διαφορετικό όμως id). Η μέθοδος θα πρέπει να δουλεύει όπως ακριβώς η εισαγωγή σε τυχαιοποιημένο δέντρο που είδαμε στο μάθημα: *με πιθανότητα 1/N+1 θα πρέπει να γίνεται εισαγωγή στη ρίζα (μέσω περιστροφών), διαφορετικά θα πρέπει να γίνεται αναδρομικά εισαγωγή στο κατάλληλο υποδέντρο*. Μην ξεχνάτε την ενημέρωση του πεδίου N σε κάθε κόμβο.

```
void insertBookAtWarehouse(int nodeid, int isbn, int copies);
```

Η μέθοδος αυτή εισάγει στη λίστα της αποθήκης με κωδικό `nodeid`, `copies` αριθμό αντιτύπων του βίβλιου με κωδικό `isbn`. Αν δεν υπάρχει αποθήκη με αυτό τον κωδικό, η μέθοδος πρέπει να τυπώνει μήνυμα λάθους. Αν το βιβλίο υπάρχει ήδη στην αποθήκη, δεν προσθέτετε νέο αντικείμενο BookInfo στη λίστα της αποθήκης απλά προσθέτετε στον υπάρχοντα αριθμό αντιτύπων τα επιπλέον `copies`. Αν το βιβλίο δεν υπάρχει, τότε θα πρέπει να δημιουργήσετε νέο κόμβο στη λίστα `booklist`.

```
void removeWarehouse(int nodeid);
```

Η μέθοδος αυτή αφαιρεί την αποθήκη με κωδικό `nodeid`. Θα πρέπει να χρησιμοποιήσετε την μέθοδο αφαίρεσης που είδαμε για τυχαιοποιημένα δέντρα (η οποία είναι μια απλή παραλλαγή του τρόπου αφαίρεσης που είδαμε σε ΔΔΑ).

```
void removeBook(int nodeid, int isbn);
```

Αφαιρεί ένα αντίτυπο του βιβλίου με κωδικό `isbn` από την αποθήκη με κωδικό `nodeid`. Αν η αποθήκη δεν υπάρχει ή το συγκεκριμένο βιβλίο δεν υπάρχει στην αποθήκη τότε εκτυπώνει σχετικό μήνυμα. Διαφορετικά, μειώνει τα διαθέσιμα αντίτυπα κατά 1. Αν ο αριθμός αντιτύπων γίνει 0, τότε πρέπει να αφαιρέσετε το βιβλίο από τη λίστα.

Μέθοδοι αναζήτησης:

```
void searchByWarehouse(int nodeid);
```

Με τη μέθοδο αυτή γίνεται αναζήτηση στο δέντρο για την αποθήκη με κωδικό ίσο με `nodeid` (με βάση τον αλγόριθμο αναζήτησης που έχουμε δει, χρησιμοποιώντας ως κλειδί το `nodeid`). Αν υπάρχει, τυπώνωνται όλα τα διαθέσιμα βιβλία της, και ο αριθμός αντιτύπων κάθε βιβλίου.

```
void searchBookInWarehouse(int nodeid, int isbn);
```

Με τη μέθοδο αυτή ψάχνουμε αν υπάρχει το βιβλίο με το δοσμένο `isbn` στην αποθήκη με κωδικό `nodeid`. Αν το βιβλίο υπάρχει θα πρέπει να εμφανίζεται αντίστοιχο μήνυμα που να λέει και τον αριθμό αντιτύπων.

```
void searchBook(int isbn);
```

Με τη μέθοδο αυτή ψάχνουμε σε ποιες αποθήκες υπάρχει το βιβλίο. Αν το βιβλίο υπάρχει, θα πρέπει να εμφανίζονται όλες οι αποθήκες στις οποίες βρέθηκε, το όνομα της πόλης όπου βρίσκεται η κάθε αποθήκη, καθώς και ο αριθμός αντιτύπων σε κάθε αποθήκη. Η υλοποίηση της μεθόδου μπορεί να γίνει με κάποια διάσχιση του δέντρου.

```
void printTree(PrintStream stream);
```
Η μέθοδος printTree είναι χρήσιμη για debugging. Θα πρέπει να κάνει μία διάσχιση του δέντρου (διαλέξτε μία από τις inorder, postorder, preorder) και να τυπώνει για κάθε κόμβο το περιεχόμενο της λίστας των βιβλίων.

Μπορείτε να προσθέσετε επίπλεον βοηθητικές συναρτήσεις, όπου κρίνετε εσείς ότι είναι απαραίτητο.

Μέρος Γ: Μενού διαχείρισης

Κατασκευάστε μια απλή εφαρμογή διαχείρισης αποθήκης (δεν χρειάζεται γραφικό περιβάλλον) στην οποία θα φαίνονται όλες οι λειτουργίες της δομής σας. Το πρόγραμμα θα πρέπει να αλληλεπιδρά με το χρήστη ώστε ο χρήστης να μπορεί να εισάγει και να διαγράφει αποθήκες, να εισάγει/αναζητά/διαγράφει βιβλία (το πρόγραμμα αυτό είναι ούτως ή άλλως χρήσιμο να το φτιάξετε για debugging).

Παράδειγμα

Έστω ότι έχετε εισάγει 3 αποθήκες. Η αποθήκη με κωδικό 3 είναι στη Θεσσαλονίκη και περιέχει 2 αντίτυπα από τα βιβλία με ISBN 134 και 256. Η αποθήκη με κωδικό 13 είναι στο Ηράκλειο και έχει 1 αντίτυπο του 134 και 3 αντίτυπα του βιβλίου 349. Η αποθήκη με κωδικό 33 είναι στην Ιθάκη και έχει 1 αντίτυπο των βιβλίων με ISBN 17, 31, 349.

Η κλήση της μεθόδου `searchByWarehouse(13)` θα τυπώσει:

```
Warehouse 13 located in Iraklio:
Book 134, copies:1
Book 349, copies:3
```

Η `searchBookInWarehouse(3, 349)` θα πρέπει να τυπώνει:

```
Warehouse 3 does not have this book.
```

Η `searchBook(349)` θα πρέπει να εμφανίζει:

```
The book is available at
Warehouse 13 located in Iraklio, copies: 3
Warehouse 33 located in Ithaki, copies: 1
```

Οδηγίες και παρατηρήσεις:

- **Δεν επιτρέπεται** χρήση των έτοιμων δομών δεδομένων της Java (Vector, ArrayList κλπ).
- Μπορείτε να επαναχρησιμοποιήσετε κώδικα του εργαστηρίου ή/και προηγουμένων εργασιών σας. Μην ξεχνάτε κατά την εισαγωγή και διαγραφή να ενημερώνετε το πεδίο N κάθε εμπλεκόμενου κόμβου.

Παραδοτέα: Για την εργασία θα υποβάλετε τα εξής αρχεία:

1) ST.java, ο ΑΤΔ για πίνακα συμβόλων.
2) BookInfo.java, η κλάση για την αναπαράσταση πληροφοριών ενός βιβλίου.
3) SystemMenu.java, ο κώδικας με το μενού διαχείρισης σας, το οποίο θα το υλοποιείτε εντός της main.

4) Ό,τι άλλο αρχείο κώδικα χρησιμοποιήσετε ενδεχομένως και από το υλικό του εργαστηρίου ή/και από προηγούμενες εργασίες, προκειμένου να μεταγλωττίζεται ο κώδικάς σας.
5) Μία σύντομη αναφορά σε pdf αρχείο με όνομα project3-report.pdf, στην οποία θα αναφέρετε τα μέλη της ομάδας σας και θα περιγράψετε συνοπτικά την υλοποίησή σας (π.χ. ποια μέθοδο διάσχισης υλοποιήσατε και πώς, πώς εκτελείτε τις μεθόδους διαγραφής, και τις μεθόδους εισαγωγής).

Όλα τα παραπάνω αρχεία θα πρέπει να μπουν σε ένα αρχείο zip. Το όνομα που θα δώσετε στο αρχείο αυτό θα είναι ο αριθμός μητρώου σας πχ. 3030056_3030066.zip ή 3030056.zip (αν δεν είστε σε ομάδα). Στη συνέχεια, θα υποβάλετε το zip αρχείο σας στην περιοχή του μαθήματος «Εργασίες» στο e-class.

Η προθεσμία παράδοσης της εργασίας είναι Κυριακή, 21 Ιανουαρίου 2017 και ώρα 23:59.