ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Δομές Δεδομένων - Εργασία 3

Τμήμα Πληροφορικής

Φθινοπωρινό Εξάμηνο 2018-2019

Διδάσκων: Ε. Μαρκάκης

Δέντρα Δυαδικής Αναζήτησης

Σκοπός της εργασίας 3 είναι η εξοικείωση με τις δομές που χρησιμοποιούνται για την υλοποίηση πίνακα συμβόλων, όπως τα δέντρα δυαδικής αναζήτησης. Το ζητούμενο της εργασίας είναι να κατασκευάσετε ένα Μετρητή Λέξεων. Ο Μετρητής Λέξεων θα "φορτώνει" ένα αρχείο κειμένου και θα μετρά πόσες φορές εμφανίζεται η κάθε λέξη. Για παράδειγμα, αν το αρχείο περιέχει το παρακάτω κείμενο

- Hello, how are you?
- Very well, thank you. How about you?
- Fine, thank you.

η λέξη «you» εμφανίζεται 4 φορές, η λέξη «thank» 2 φορές, η λέξη how επίσης 2, ενώ από 1 φορά εμφανίζονται οι λέξεις hello, are, very, well, what, about και fine. Ο Μετρητής Λέξεων θα πρέπει να αφαιρεί τα σημεία στίξης , . ? ; ! - : " ", από τις λέξεις, καθώς και παρενθέσεις, αγκύλες, σύμβολα πράξεων, ή οποιονδήποτε άλλο χαρακτήρα που δεν είναι γράμμα του αγγλικού αλφαβήτου. Από το κείμενο παραπάνω, όταν διαβάσει π.χ. το string "well," θα πρέπει να βγάλει το κόμμα για να μείνει μόνο η λέξη well. Αν υπήρχε φράση σε παρενθέσεις, π.χ. "(Maria asked me)", θα πρέπει να αφαιρεθούν οι παρενθέσεις και να μείνουν οι λέξεις Maria, asked και me. Επιτρέπεται μόνο η μονή απόστροφος μέσα σε λέξη, π.χ. το don't θα το θεωρούμε σαν 1 λέξη. Επίσης θα πρέπει να αγνοούνται πλήρως όλες οι λέξεις που περιέχουν αριθμούς π.χ. 17:25 ή 1980's. Τέλος θα πρέπει να δίνεται η δυνατότητα στο χρήστη της βιβλιοθήκης να ορίζει ειδικές λέξεις (stop words) που θέλει να αγνοούνται πλήρως, π.χ. τα άρθρα "a", "an" και "the". Το πρόγραμμα θα είναι case in-sensitive. Για παράδειγμα οι λέξεις Hello και hello είναι ίδιες.

Μέρος Α [10 μονάδες]. Για να ξεκινήσουμε, θα πρέπει πρώτα να φτιάξετε τους εξής 2 τύπους δεδομένων.

Η κλάση WordFreq. Για κάθε διαφορετική λέξη που διαβάζετε, θα πρέπει να υπάρχει στο δέντρο σας ένας κόμβος με κλειδί αυτή τη λέξη. Συγκεκριμένα, σε κάθε λέξη θα αντιστοιχεί ένα αντικείμενο τύπου WordFreq. Η κλάση αυτή περιέχει (τουλάχιστον) 2 πεδία: την ίδια τη λέξη (private String) και τον αριθμό εμφανίσεων (private int). Μπορείτε εδώ να υπερφορτώσετε κατάλληλα την μέθοδο toString για να σας

χρησιμεύσει για την εκτύπωση αποτελεσμάτων. Η WordFreq θα πρέπει να περιέχει και μια μέθοδο key() που θα επιστρέφει το κλειδί, δηλαδή τη λέξη.

Η κλάση TreeNode. Κάθε κόμβος του δέντρου είναι ένα αντικείμενο τύπου TreeNode, και πρέπει να περιέχει ένα αντικείμενο τύπου WordFreq, όπως περιγράφεται παραπάνω. Επιπλέον, πρέπει να περιέχει τους δείκτες προς το αριστερό και δεξιό υποδέντρο, και ένα πεδίο που θα δηλώνει πόσους κόμβους έχει το υποδέντρο που ξεκινά από αυτόν τον κόμβο. Επομένως στην κλάση Treenode πρέπει να υπάρχουν τουλάχιστον τα εξής πεδία (και ενδεχομένως ό,τι άλλο θέλετε εσείς να προσθέσετε):

```
private class Treenode {
   WordFreq item //
   Treenode l // pointer to left subtree
   Treenode r // pointer to right subtree
   int number  //number of nodes in the subtree starting at this node
   ...
}
```

Η πρόσβαση στο κλειδί του κόμβου θα πρέπει να γίνεται μέσω της μεθόδου key() του item. Αν h είναι ένα αντικείμενο τύπου Treenode, το κλειδί του κόμβου θα το παίρνετε από την κλήση `h.item.key()`, όπως και στις διαφάνειες του μαθήματος.

Μέρος Β [80 μονάδες]. Η κλάση του πίνακα συμβόλων. Η δομή σας για τον πίνακα συμβόλων θα πρέπει να έχει τουλάχιστον τα πεδία και τις μεθόδους που φαίνονται παρακάτω (ακολουθούν επεξηγήσεις):

```
class ST    {
private class TreeNode {
...
};
private TreeNode head; //ρίζα στο δέντρο
private List stopwords; // λίστα με τα stopwords

WordFreq search(String w);
void insert(WordFreq item);
void update(String w);
void remove(String w)
void load(String filename);
int getTotalWords();
int getDistinctWords();
int getFrequency(String w);
WordFreq getMaximumFrequency();
double getMeanFrequency();
void addStopWord(String w);
void removeStopWord(String w);
void printTreeAlphabetically(PrintStream stream);
void printTreeByFrequency(PrintStream stream);
}
```

Ακολουθεί συνοπτική περιγραφή των απαιτούμενων μεθόδων:

- `void insert(WordFreq item)`: εισάγει στο δέντρο έναν νέο κόμβο με το αντικείμενο item. Η εισαγωγή θα είναι η απλή εισαγωγή ως φύλλο.
- `void update(String w)`: ψάχνει να βρει αν υπάρχει ήδη στο δέντρο κόμβος με κλειδί w. Αν ναι, τότε του αυξάνει τη συχνότητα κατά 1. Αν όχι, τότε εισάγει ένα νέο κόμβο στο δέντρο (ως φύλλο), με αυτό το κλειδί και με συχνότητα ίση με 1.
- `WordFreq search(String w)`: Ψάχνει στο δέντρο για την ύπαρξη της λέξης w (επιστρέφει null αν δεν υπάρχει). Η μέθοδος search θα δουλεύει όπως και η αντίστοιχη μέθοδος

που έχουμε δει στο μάθημα με την εξής διαφοροποίηση: όταν βρίσκει τη λέξη w μέσα στο δέντρο, αν η συχνότητα της w είναι μεγαλύτερη της μέσης συχνότητας (από την `getMeanFrequency()`), τότε με χρήση περιστροφών θα φέρνει τη λέξη αυτη στη ρίζα του δέντρου (ένας τρόπος να γίνει αυτό, αλλά όχι και ο μοναδικός, είναι να καλείτε πρώτα τη `remove` για να βγάλει τον κόμβο αυτό από το ΔΔΑ και στη συνέχεια να κάνετε εισαγωγή στη ρίζα για τον συγκεκριμένο κόμβο). Το σκεπτικό είναι ότι λέξεις με μεγάλη συχνότητα είναι λέξεις για τις οποίες μπορεί να γίνουν πολλές αναζητήσεις και επομένως θέλουμε να τις έχουμε όσο το δυνατόν πιο ψηλά στο δέντρο.

- `void remove(String w)`: αφαιρεί τον κόμβο με κλειδί w (αν υπάρχει τέτοιος κόμβος). Μπορείτε να χρησιμοποιήσετε τη μέθοδο αφαίρεσης που έχουμε δει στο μάθημα.
- `void load(String filename)`: φορτώνει το αρχείο filename που περιέχει κείμενο Αγγλικής γλώσσας, και φτιάχνει το δέντρο. Κατά τη φόρτωση του αρχείου θα πρέπει να τηρούνται οι προϋποθέσεις που αναφέρθηκαν προηγουμένως, δηλαδή, αφαιρούνται τα σημεία στίξης και άλλοι χαρακτήρες από λέξεις, και αγνοούνται πλήρως λέξεις που έχουν δοθεί ως stop words ή λέξεις που περιέχουν αριθμούς.
- `int getTotalWords()`: επιστρέφει τον συνολικό αριθμό λέξεων του αρχείου που φορτώθηκε στην τελευταία κλήση της load. (αφού αγνοήσουμε πρώτα τις stop words και τις λέξεις που περιέχουν αριθμούς). Μπορεί να γίνει με μια απλή διάσχιση του δέντρου (όποια διάσχιση θέλετε).
- `int getDistinctWords()`: επιστρέφει τον αριθμό διαφορετικών λέξεων του αρχείου (και πάλι αφού αγνοήσουμε πρώτα τις stop words και τις λέξεις που περιέχουν αριθμούς). Πρέπει να τρέχει σε O(1).
- `int getFrequency(String w)`: επιστρέφει τον αριθμό εμφανίσεων της λέξης w (αν η λέξη δεν υπάρχει στο δέντρο, επιστρέφει 0).
- `WordFreq getMaximumFrequency()`: επιστρέφει ένα αντικείμενο τύπου WordFreq που περιέχει τη λέξη με τις περισσότερες εμφανίσεις (δεν χρειάζεται να κάνετε ταξινόμηση για να λύσετε το πρόβλημα αυτό).
- `double getMeanFrequency()`: υπολογίζει και επιστρέφει την μέση συχνότητα. Ο μέσος όρος παράγεται από τις συχνότητες όλων των διαφορετικών λέξεων μέσα στο κείμενο.
- `void addStopWord(String w)`: προσθέτει στη λίστα stopwords με τις λέξεις που θα αγνοούνται, τη λέξη w.
- `void removeStopWord(String w)`: αφαιρεί τη λέξη w από τη λίστα stopwords.
- `printAlphabetically(PrintStream)`: εκτυπώνει τις λέξεις του δέντρου, μαζί με τον αριθμό εμφανίσεων κάθε λέξης, με αλφαβητική σειρά. Πρέπει να υλοποιηθεί με κάποια μέθοδο διάσχισης του δέντρου.
- `printByFrequency(PrintStream)`: εκτυπώνει τις λέξεις και τον αριθμό εμφανίσεων ταξινομημένες σε αύξουσα σειρά ως προς τον αριθμό εμφανίσεων.

Σχόλια και πρόσθετες οδηγίες υλοποίησης:

- Μπορείτε να βασιστείτε στον κώδικα από τις διαφάνειες και από τα εργαστήρια για ΔΔΑ.
- Χρησιμοποιήστε τις μεθόδους της βασικής βιβλιοθήκης της Java για την μετατροπή κεφαλαίων σε μικρούς χαρακτήρες, την αφαίρεση των σημείων στίξης και όποια άλλη επεξεργασία κάνετε για τις λέξεις.
- Δεν θα θεωρήσετε κάποια stop word ως δεδομένη. Η λίστα με τα stopwords θα δημιουργείται μεσα από κλήσεις της addStopWord. Για τη λίστα stopwords διαλέξτε όποια υλοποίηση λίστας θέλετε: μονής ή διπλής σύνδεσης.
- Το κλειδί των αντικειμένων που αποθηκεύουμε στο δέντρο είναι τύπου String. Επομένως σύγκριση κλειδιών εδώ σημαίνει σύγκριση μεταξύ strings. Επίσης, κάθε κλειδί θα εμφανίζεται το πολύ σε έναν κόμβο του δέντρου.
- Αρκετές μέθοδοι χρειάζονται διάσχιση του δέντρου. Θα πρέπει λοιπόν να υλοποιήσετε μέσα στον πίνακα συμβόλων και κάποια ή κάποιες μεθόδους διάσχισης.
- Προσοχή να γίνεται σωστή ενημέρωση του πεδίου number, της κλάσης TreeNode.

- Η υλοποίηση της printByFrequency μπορεί να γίνει είτε μέσω κάποιας διάσχισης είτε με άλλους τρόπους. Μπορείτε π.χ. (δεν είναι απαραίτητο) όταν καλείται η printByFrequency να δημιουργείτε επί τόπου ένα προσωρινό ΔΔΑ με τον κατάλληλο τύπο κλειδιού και έναν κατάλληλο Comparator (σκεφτείτε τι πρέπει να συγκρίνετε) για να διατρέξετε το δέντρο με βάση τον αριθμό εμφάνισης της κάθε λέξης.
- Δεν υπάρχει κάποια αυστηρή απαίτηση για την πολυπλοκότητα των μεθόδων, εκτός από την μέθοδο `getDistinctWords()`. Μπορείτε να έχετε ως γνώμονα ό,τι έχουμε πει και στο μάθημα. Αν όμως κάνετε κάτι υπερβολικά χρονοβόρο, π.χ. αν υλοποιήσετε την printAlphabetically σε χρόνο $O(N^2)$, τότε δεν θα πάρετε όλες τις μονάδες που αντιστοιχουν στη μέθοδο αυτή. Γενικά θα πρέπει να αποφύγετε το $O(N^2)$ για όλες τις μεθόδους εκτός της load.
- **Δεν επιτρέπεται να χρησιμοποιήσετε έτοιμες υλοποιήσεις δομών δεδομένων από την βιβλιοθήκη της Java** (π.χ. Vector, ArrayList κλπ).

Προαιρετικά: Όποιος θέλει, μπορεί αντί για ΔΔΑ να χρησιμοποιήσει κάποια από τις δομές του Κεφαλαίου 13, που έχουν ως αποτέλεσμα το δέντρο να είναι πιο ισοζυγισμένο, π.χ. τυχαιοποιημένα ΔΔΑ ή δέντρα κόκκινου-μαύρου. Επίσης, μπορείτε αν θέλετε να έχετε και μια μέθοδο main η οποία να κάνει κάποιο ενδεικτικό τρέξιμο, π.χ. να προσθέτει κάποια stopwords, μετά να καλεί τη μέθοδο load, και μετά να τυπώνει τη μέση συχνότητα, ή ακόμα και να εμφανίζει κάποιο μενού διαχείρισης.

Μέρος Γ - Αναφορά παράδοσης [10 μονάδες]. Ετοιμάστε μία σύντομη αναφορά σε pdf αρχείο (μην παραδώσετε Word ή txt αρχεία!) με όνομα project3-report.pdf, στην οποία θα αναφερθείτε στα εξής:

a. Εξηγήστε συνοπτικά πώς υλοποιήσατε κάθε μέθοδο του Μέρους Β (αρκούν το πολύ 4-5 γραμμές για κάθε μέθοδο).
b. Σχολιάστε την πολυπλοκότητα των μεθόδων αυτών.

Το συνολικό μέγεθος της αναφοράς θα πρέπει να είναι τουλάχιστον 2 σελίδες. Μην ξεχνάτε τα ονοματεπώνυμα και τους ΑΜ σας στην αναφορά.

Οδηγίες Παράδοσης

Η εργασία σας θα πρέπει να μην έχει συντακτικά λάθη και να μπορεί να μεταγλωττίζεται. Εργασίες που δεν μεταγλωττίζονται χάνουν το **50%** της συνολικής αξίας.

Η εργασία θα αποτελείται από:

1. Τον πηγαίο κώδικα (source code). Τοποθετήστε σε ένα φάκελο με όνομα **src** τα αρχεία java που έχετε φτιάξει. Ενδεικτικά θα πρέπει να περιέχει (χρησιμοποιήστε τα όνοματα των κλάσεων όπως ακριβώς δίνονται στην εκφώνηση):
   - Τα αρχεία ST.java, WordFreq.java.
   - Οποιαδήποτε άλλα αρχεία πηγαίου κώδικα φτιάξατε και απαιτούνται για να μεταγλωττίζεται η εργασία σας.

   Φροντίστε επίσης να προσθέσετε επεξηγηματικά σχόλια όπου κρίνετε απαραίτητο στον κώδικά σας.
2. Την αναφορά παράδοσης

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 υποβάλλετε το zip αρχείο σας στην περιοχή του μαθήματος «Εργασίες» στο e-class. Δεν χρειάζεται υποβολή και από τους 2 φοιτητές μιας ομάδας.

Η προθεσμία παράδοσης της εργασίας είναι Παρασκευή, 11 Ιανουαρίου 2018 και ώρα 23:59.