Πανεπιστήμιο Ιωαννίνων – Τμήμα Μηχανικών Η/Υ και Πληροφορικής
Δομές Δεδομένων [ΜΥΥ303] – Χειμερινό Εξάμηνο 2020

1η Εργαστηριακή Άσκηση
Συνδεδεμένες Λίστες
Παράδοση έως Πέμπτη 29/10, 12:00 από το eCourse

ΠΡΟΣΟΧΗ: Γράψτε σε κάθε αρχείο που παραδίδετε τα ονόματα και τους Α.Μ. των μελών της ομάδας σας. Συμπεριλάβετε όλα τα αρχεία σας (κώδικας Java και lab1results.txt) σε ένα zip αρχείο το οποίο να έχει το όνομα ενός μέλους της ομάδας σας με λατινικούς χαρακτήρες.

Ο σκοπός της άσκησης είναι να υλοποιήσουμε μια συνδεδεμένη λίστα η οποία αποθηκεύει αλφαριθμητικά (αντικείμενα τύπου String). Συγκεκριμένα, κάθε κόμβος της συνδεδεμένης λίστας, τύπου Node, αποθηκεύει ένα αλφαριθμητικό str, το πλήθος των εμφανίσεων του str στο αρχείο εισόδου (int count) και μια αναφορά στον επόμενο κόμβο της λίστας (Node next).

Η δομή μας πρέπει να υποστηρίζει τις ακόλουθες βασικές μεθόδους που καλείστε να υλοποιήσετε:

int contains(String s)	Επιστρέφει το πλήθος των εμφανίσεων του αλφαριθμητικού s. Αν το s δεν έχει εισαχθεί στη δομή τότε επιστρέφει μηδέν.
void insert(String s)	Εισάγει το αλφαριθμητικό s στη συνδεδεμένη λίστα. Αν το s υπάρχει ήδη στη δομή, τότε αυξάνουμε κατά ένα το μετρητή των εμφανίσεων του.
void delete(String s)	Διαγράφει το αλφαριθμητικό s από τη συνδεδεμένη λίστα.

Για να ελέγξουμε αν το String s είναι αποθηκευμένο σε ένα κόμβο x, μπορούμε να γράψουμε $x.str.\text{equals}(s) == \text{true}$.

Στη συνέχεια υλοποιήστε τις ακόλουθες μεθόδους:

void lexOrder()	Ταξινομεί τη συνδεδεμένη λίστα σε λεξικογραφική διάταξη (π.χ., το “and” προηγείται του “because”).
void freqOrder()	Ταξινομεί τη συνδεδεμένη λίστα σε φθίνουσα σειρά ως προς το πλήθος εμφανίσεων κάθε αλφαριθμητικού. (Το αλφαριθμητικό με το μέγιστο πλήθος εμφανίσεων θα πρέπει να βρίσκεται στην αρχή της λίστας.)
String print(int k)	Τυπώνει τα k πρώτα αλφαριθμητικά ως προς την τρέχουσα διάταξη της συνδεδεμένης λίστας, μαζί με το πλήθος των εμφανίσεων τους.
String select(int k)	Επιστρέφει το k-οστό αλφαριθμητικό s ως προς την τρέχουσα διάταξη της συνδεδεμένης λίστας. (Δηλαδή, υπάρχουν $k-1$ αλφαριθμητικά που προηγούνται του s στη συνδεδεμένη λίστα.)

Εκτέλεση Προγράμματος

Η main() μέθοδος του προγράμματος WordList.java χρησιμοποιεί το πρόγραμμα In.java για να διαβάσει από ένα αρχείο εισόδου μία ακολουθία λέξεων, τα οποία εισάγει στη συνδεδεμένη λίστα. Για την εκτέλεση του προγράμματος θα χρησιμοποιήσουμε για αρχείο εισόδου το κείμενο TomSawyerB.txt.

Για την εκτέλεση των προγραμμάτων γράψτε

java WordList < TomSawyerB.txt.

Το πρόγραμμα αρχικά τυπώνει το χρόνο κατασκευής της συνδεδεμένης λίστας, αφού ολοκληρωθεί η ανάγνωση του αρχείου εισόδου και τυπώνει το πλήθος των εμφανίσεων των λέξεων and, astonished, boat, path, the, train, tom και wondered.

Στη συνέχεια, ταξινομεί τη συνδεδεμένη λίστα σε λεξικογραφική διάταξη και τυπώνει τις πρώτες 10 λέξεις, καθώς και την εκατοστή λέξη ως προς τη λεξικογραφική διάταξη.

Μετά, ταξινομεί τη συνδεδεμένη λίστα σε φθίνουσα σειρά ως προς το πλήθος εμφανίσεων κάθε λέξης, τυπώνει τις 10 πιο συχνές λέξεις, καθώς και την εκατοστή πιο συχνή λέξη *s* του κειμένου εισόδου. Τέλος, διαγράφει την *s* και τυπώνει την αμέσως συχνότερη λέξη μετά την *s*.

Αποθηκεύστε τα παραπάνω αποτελέσματα στο αρχείο lab1results.txt.

Παραδοτέα

Ανεβάστε στο eCourse ένα zip αρχείο με το τελικό σας πρόγραμμα WordList.java, το έτοιμο πρόγραμμα In.java, καθώς και το αρχείο των αποτελεσμάτων lab1results.txt. Το zip αρχείο πρέπει να έχει το όνομα ενός μέλους της ομάδας σας με λατινικούς χαρακτήρες.