ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΑΡΧΕΙΩΝ

1η άσκηση

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΜΙΧΑΗΛ ΚΡΑΤΗΜΕΝΟΣ

ΑΜ: 2018030104

1ο Μέρος

Αρχικά, δημιουργήθηκε η κλάση DoubleLinkedList<E>, η οποία είναι η διπλά συνδεδεμένη λίστα που ζητήθηκε στην εκφώνηση να υλοποιεί το αρχείο κείμενου, οι κόμβοι της οποίας υλοποιήθηκαν με την κλάση Node (εμπεριέχεται μέσα στην κλάση DoubleLinkedList).

Μενού επιλογών

΄Υστερα δημιουργήθηκε ένα στιγμιότυπο της κλάσης Node (temp) που αρχικοποιήθηκε στο head της λίστας, ένα της κλάσης DoubleLinkedList (dll), ένα της κλάσης StandardInputRead (reader), που διαβάζει πληροφορίες από το χρήστη, για τις περισσότερες λειτουργικότητες του μενού επιλογών. Επίσης χρησιμοποιήθηκε η μέθοδος passTxtFileToDLL η οποία αντιγράφει κάθε γραμμή του αρχείου σε ξεχωριστό κόμβο του dll, λαμβάνοντας υπόψιν και το μέγιστο επιτρεπτό μέγεθος κάθε γραμμής.

- Οι λειτουργικότητες ^, $ υλοποιήθηκαν με τις μεθόδους getHead(), getTail() του dll που επιστρέφουν στο temp το πρώτο και το τελευταίο κόμβο της λίστας αντίστοιχα άρα και τις αντίστοιχες γραμμές του αρχείου.
- Στις λειτουργικότητες -, + το temp γίνεται ίσο με τον προηγούμενο και τον επόμενο κόμβο του αντίστοιχα, εφόσον αυτοί δεν είναι κενοί (null), έτσι ώστε το temp να γίνεται η προηγούμενη ή η επόμενη γραμμή (εφόσον υπάρχουν) του αρχείου αντίστοιχα.
- Οι λειτουργικότητες a,t υλοποιήθηκαν με τις μεθόδους addAfterCurrentNode και addBeforeCurrentNode του dll οι οποίες προσθέτουν ένα νέο κόμβο μετά και πριν το temp αντίστοιχα τα στοιχεία του οποίου τα εισάγει ο χρήστης (reader.readString).
- Η λειτουργικότητα d υλοποιήθηκε με τη χρήση της μεθόδου deleteNode του dll η οποία βγάζει ένα κόμβο (temp) από το dll αναλόγως με το πού βρίσκεται αυτός. Μετά το temp γίνεται ίσο με τον επόμενο ή με τον προηγούμενό του κόμβο (αν ο επόμενος δεν υπάρχει) για να μην χαθεί η λίστα ή γίνεται null αν ήταν ο μοναδικός κόμβος της λίστας.
- Στη λειτουργικότητα l χρησιμοποιείται η μέθοδος print του dll η οποία τυπώνει όλους τους κόμβους του και τα indexes (τις γραμμές) τους αν είναι true η μεταβλητή tf (type: boolean), η οποία αλλάζει τιμή μέσω της λειτουργικότητας n.
- Η λειτουργικότητα p τυπώνει το temp (current line) εφόσον υπάρχει, ενώ η = τυπώνει το index του temp (current line number) μέσα στο dll, χρησιμοποιώντας το head του.
- Η λειτουργικότητα q τερματίζει το πρόγραμμα χωρίς να γράφει το dll στο αρχείο. Αντίθετα, η x το γράφει μέσω της μεθόδου saveListToTxtFile και μετά τερματίζοντας το πρόγραμμα, ενώ η w το γράφει με την ίδια μέθοδο χωρίς να τερματίζει το πρόγραμμα.

- Η λειτουργικότητα # τυπώνει τον αριθμό χαρακτήρων και τον αριθμό των indexes (γραμμών) στο dll μέσω της μεθόδου calcLinesAndChars η οποία χρησιμοποιεί το dll.

Μέρος 2ο

Αρχικά, δημιουργήθηκε ένα ArrayList<Information> (al), το οποίο είναι ο πίνακας δεικτοδότησης των λέξεων του προηγούμενου αρχείου. Η κλάση Information περιέχει δύο member variables (ένα String και ένα int), τα οποία χρησιμοποιούνται ως μια λέξη από το αρχείο (έχοντας λάβει υπόψιν να ενδεικτικά όρια για τις λέξεις) και ο αριθμός της γραμμής στην οποία βρίσκεται.

- Στην λειτουργικότητα c η μέθοδος passToArrayList του dll γεμίζει το al με τις λέξεις του αρχείου που πληρούν τα επιτρεπόμενα όρια και με τους αριθμούς των γραμμών που βρίσκονται αυτές στο αρχείο, ενώ η μέθοδος sortArrayList βάζει τις λέξεις σε αλφαβητική σειρά. Ύστερα, ζητείται από το χρήστη (reader.readString) να δοθεί το όνομα του δυαδικού αρχείου στο οποίο θα περαστεί το al. Αφού δοθεί, δημιουργείται (file) και ανοίγεται για διάβασμα και γράψιμο (“rw”) με το όνομα που έδωσε ο χρήστης και με κατάληξη “.ndx” και μέσω της μεθόδου createRAFile αντιγράφονται στοιχεία από το al στο file μεγέθους buffer size κάθε φορά (μέγεθος σελίδας), μέχρι να αντιγραφούν όλα τα στοιχεία του (οι αριθμοί των γραμμών των λέξεων μετατρέπονται πρώτα σε String πριν μετατραπούν σε bytes και γραφούν στο αρχείο κάτι που σημαίνει ότι το αρχικό αρχείο μπορεί να έχει μέχρι 9999 γραμμές, λόγω του intSize από την εκφώνηση (4), γιαυτό και η τιμή του intSize έγινε 7).
- Αντίστοιχα και η λειτουργικότητα v ανοίγει το ίδιο αρχείο (File) για διάβασμα και γράψιμο (“rw”) και μέσω της μεθόδου printRAFile το διαβάζει σελίδα σελίδα και τυπώνει τα στοιχεία τους κάθε φορά.
- Η λειτουργικότητα s ξανανοίγει το αρχείο (FILE) για τους ίδιους λόγους (“rw”) και μέσω της μεθόδου searchWordSerial ψάχνει μια λέξη που εισάγει ο χρήστης (reader.readString) σειριακά και σελίδα σελίδα. Αν τη βρει τυπώνει στην οθόνη τις γραμμές στις οποίες ήταν στο αρχικό αρχείο αλλά και το πόσες προσβάσεις στο δίσκο έγιναν μέχρι να βρεθεί.
- Η λειτουργικότητα b κάνει το ίδιο με την s, με τη διαφορά ότι χρησιμοποιεί τη μέθοδο searchWordBin, η οποία ψάχνει αντίστοιχη λέξη, αλλά δυαδικά, διαβάζοντας μια σελίδα τη φορά, όπως και η s. Στην περίπτωση που βρεθεί σελίδα γεμάτη με λέξεις ίδιες με την λέξη που έγραψε ο χρήστης, τότε γίνεται σειριακός έλεγχος στις προηγούμενες και στις επόμενες σελίδες. Σε κάποιες υποπεριπτώσεις της περίπτωσης αυτής, το πρόγραμμα διαβάζει κάποιες σελίδες πάνω από μία φορά.

Μέθοδοι που χρησιμοποιήθηκαν αλλά δεν αναφέρθηκαν

Στην κλάση MyEditor:

- removeSpaces: Η μέθοδος αυτή χρησιμοποιήθηκε για να αφαιρεί τα κενά στις λέξεις που δεν έφταναν το μέγιστο μέγεθος, αφού αρχικά τοποθετήθηκαν για να έχουν όλες οι λέξεις το ίδιο μέγεθος (χρησιμοποιήθηκε στις μεθόδους searchWordBin, searchWordSerial, printRAFile).
- printMenu: Η μέθοδος αυτή τυπώνει το μενού επιλογών (χρησιμοποιήθηκε στην αρχή του προγράμματος στη μέθοδο menu).

Στην κλάση DoubleLinkedList:

- addLast: Η μέθοδος αυτή προσθέτει έναν κόμβο στο τέλος της λίστας (χρησιμοποιήθηκε στη μέθοδο passTxtFileToDLL).

Αποτελέσματα πειράματος:

μέσοι όροι προσβάσεων στο δίσκο για serial και binary search:

- testfile_x2.txt: 182 , 8 αντίστοιχα
- testfile_x5.txt: 456 , 10 αντίστοιχα
- testfile_x10.txt: 911 , 12 αντίστοιχα
- testfile_x1000.txt: 91133 , 178 αντίστοιχα

Παρατηρώ ότι οι προσβάσεις στο δίσκο στο serial search είναι πολύ περισσότερες από αυτές στο binary search, άρα το binary search είναι πιο αποδοτικό από το serial search. Επίσης παρατηρώ ότι ο μέσος όρος προσβάσεων στο δίσκο στο serial search μεταβάλλεται ανάλογα με το πόσες φορές έχει αντιγραφεί το testfile.txt.

Πηγές:

- menu: http://tutorials.jenkov.com/java-io/randomaccessfile.html
- searchWordBin: https://www.youtube.com/watch?v=1ruyrpTKGZk
- passTxtFileToDLL: https://stackoverflow.com/questions/5343689/java-reading-a-file-into-an-arraylist (6η απάντηση)
- saveListToTxtFile: http://www.java2s.com/Tutorials/Java/IO/Text_File/Save_to_a_text_File_with_PrintStream_in_Java.htm
- createRAFile: https://www.geeksforgeeks.org/bytebuffer-clear-methods-in-java-with-examples/ , https://stackoverflow.com/questions/44455987/resetting-byte-buffer-to-zeros (2η απάντηση)
- sortArrayList: https://www.youtube.com/watch?v=wzWFQTLn8hI
- DoubleLinkedList, Node: https://www.java2novice.com/data-structures-in-java/linked-list/doubly-linked-list/
- passToArrayList: https://stackoverflow.com/questions/11726023/split-string-into-individual-words-java (6η απάντηση)
- StandardInputRead: αυτή η κλάση μας δόθηκε στο 2ο εξάμηνο από τους εργαστηριακούς βοηθούς στο μάθημα Δομημένος Προγραμματισμός

Οι πηγές αναφέρονται και στα σχόλια σε Javadoc των κλάσεων και συναρτήσεων στο πεδίο @version.