*Ευφροσύνη Παντελιάδη*
*1115201400301*

*sdi1400301@di.uoa.gr*

## Οδηγίες

Για τη χρήση της εφαρμογής είναι απαραίτητο να προηγηθεί μεταγλώττιση των αρχείων που την αποτελούν. Για το σκοπό αυτό, έχει δημιουργηθεί ένα Makefile, το οποίο εκτελείται όταν ο χρήστης γράψει στη γραμμή εντολών του τερματικού την εντολή *make*.

Για την εκκίνηση της εφαρμογής minisearch χρησιμοποιείται μία από τις παρακάτω εντολές:

- *./minisearch -i docfile -k K*
- *./minisearch -k K -i docfile*
- *./minisearch -i docfile*

όπου docfile είναι το όνομα του αρχείου που περιλαμβάνει τα κείμενα που θέλουμε να εισάγουμε στην εφαρμογή και K ο αριθμός των αποτελεσμάτων που θέλουμε να έχουμε για κάθε ερώτημα.

Όταν ο χρήστης τερματίσει τη λειτουργία της εφαρμογής θα πρέπει να δώσει την εντολή *make clean* στο τερματικό, ωστε να διαγραφθούν όλα τα αντικειμενικά και εκτελέσιμα αρχεία.

## Interface

Κατά την εκκίνηση της εφαρμογής παρουσιάζεται στο χρήστη ένα μήνυμα, που τον ενημερώνει για τις διαθέσιμες λειτουργίες. Ο χρήστης καλείται κάθε φορά να επιλέξει μία από αυτές και στο τέλος να τερματίσει την εφαρμογή χρησιμοποιώντας την κατάλληλη λειτουργία.

## Σχόλια Υλοποίησης

- ***Map*:** Τη συγκεκριμένη δομή αποτελεί ένας δυναμικός πίνακας, που περιέχει τόσες θέσεις όσα και τα κείμενα που δόθηκαν στην είσοδο. Μέσα του αποθηκεύονται δεδομένα τύπου struct map, καθένα από τα οποία περιέχει το id του κάθε κειμένου, το κείμενο και το πλήθος των λέξεων που το αποτελούν.
  Η αρχικοποίησή του γίνεται μέσω της συνάρτησης initializeMap, η οποία χρησιμοποιεί τη συνάρτηση strtok για να σπάσει το κείμενο σε δύο μέρη: το id του και το ίδιο το κείμενο.
  Στο τέλος, αποδεσμεύεται η μνήμη, που δεσμεύσαμε για αυτό, με κλήση της free.

- ***Trie*:** Εδώ, αποθηκεύονται όλες οι λέξεις που υπάρχουν μέσα στο αρχείο που δόθηκε στην είσοδο. Για τη δημιουργία του δεσμεύουμε χώρο στη μνήμη για τη ρίζα του, η οποία αρχικοποιείται μέσω της initializeRoot. Στη συνέχεια, χρησιμοποιούμε τη συνάρτηση initializeTrie για την αρχικοποίησή του.
  Η initializeTrie παίρνει κάθε κείμενο που υπάρχει μέσα στο map και περνά μία λέξη του τη φορά στο Trie με κλήση της insertTrie.

Η συνάρτηση insertTrie είναι υπεύθυνη για την εισαγωγή μίας λέξης μέσα στο trie. Χρησιμοποιεί ένα βρόγχο for, ώστε να εισάγει ένα γράμμα της λέξης κάθε φορά. Διατρέχει τη δομή και ανάλογα με τον ASCII κωδικό του γράμματος, το τοποθετεί στη σωστή θέση. Όποτε τοποθετείται το τελευταίο γράμμα μίας λέξης γίνεται κλήση της συνάρτησης addList, η οποία δημιουργεί την postings list της λέξης, εάν δεν υπάρχει, ή την τροποποιεί.

Η searchTrie ψάχνει τη λέξη που έχει δεχθεί ως όρισμα μέσα στο Trie. Εάν υπάρχει, επιστρέφει δείκτη στον κόμβο που περιέχει πληροφορίες για την postings list της, διαφορετικά επιστρέφει NULL.

Στο τερματισμό λειτουργίας της εφαρμογής αποδεσμεύεται η μνήμη που δεσμεύσαμε για το trie και την κάθε postings list των φύλλων της με κλήση της συνάρτησης destroyTrie, η οποία καλεί την deletePostingsList, η οποία με τη σειρά της καλεί την deleteList.

- ***Postings List*:** Κάθε κόμβος του Trie που περιέχει ένα γράμμα που αποτελεί τέλος μίας λέξης δείχνει σε έναν κόμβο τύπου postingsList. Ο κόμβος αυτός περιέχει ορισμένες πληροφορίες που αφορούν την postings list. Συγκεκριμένα, έχει δείκτη στην κεφαλή της postings list, το document frequency vector της λέξης και την ίδια τη λέξη.
  Οι κόμβοι της postings list είναι τύπου postingsListNode και καθένας από αυτούς περιέχει το ID ενός κειμένου στο οποίο υπάρχει η λέξη, το πλήθος των εμφανίσεων της στο συγκεκριμένο κείμενο και δείκτη στον επόμενο κόμβο.
  Η δημιουργία, η επέκταση και η τροποίησή της γίνεται μέσω της addList. Εάν δεν υπάρχει κόβμος για το κείμενο στο οποίο βρισκόμαστε, τότε δημιουργείται ένας νέος για αυτόν. Διαφορετικά, πηγαίνουμε στον κόμβο που του αντιστοιχεί και αυξάνουμε κατά ένα το πλήθος των εμφανίσεων της λέξης.
  Η αποδέσμευση της μνήμης που δεσμεύσαμε για τη λίστα, αλλά και τον κόμβο που περιέχει πληροφορίες για αυτή, γίνεται μέσω των deletePostingsList και deleteList

- ***Search Operation*:** Υπεύθυνη για τη λειτουργία search της εφαρμογής είναι η συνάρτηση searchOperation.
  Αρχικά, σπάει το όρισμα που έδωσε ο χρήστης σε αυτόνομες λέξεις και τις τοποθετεί σε έναν δυναμικό πίνακα, καθώς πρέπει να τις θυμόμαστε, ώστε να ξερούμε κατά την εκτύπωση του αποτελέσματος εάν πρέπει να τις υπογραμμίσουμε ή όχι.
  Με χρήση της συνάρτησης searchTrie διακρίνουμε εάν μία λέξη βρίσκεται μέσα στο trie (και άρα σε κάποιο από τα κείμενα που περιέχονται στο map) ή όχι και αποκτάμε πρόσβαση στην postings list της.
  Καθώς θέλουμε τα αποτελέσματα να εκτυπώνονται σε φθίνουσα σειρά με βάση ένα σκορ (το οποίο υπολογίζεται από τις getScore και getIDF), έχει δημιουργηθεί μία λίστα score list. Κάθε κόμβος της περιέχει το id του κειμένου, το σκορ και δείκτη στον επόμενο κόμβο. Η δημιουργία και τροποποίησή της γίνεται μέσω της συνάρτησης addScoreList. Κάθε φορά που πρόκειται να γίνει μία καινούρια εισαγωγή, διατρέχουμε τη λίστα για να δούμε εάν υπάρχει ήδη κόμβος για το κείμενο στο οποίο αναφερόμαστε. Εάν υπάρχει, αυξάνουμε το σκορ που θέλουμε να εισάγουμε προσθέτοντας το σκορ του κόμβου, διαγράφουμε τον κόμβο και κάνουμε εισαγωγή με τη νέα τιμή. Για την εισαγωγή μίας τιμής διατρέχουμε ξανά τη λίστα ώστε, μέσω αριθμητικών συγκρίσεων, να την τοποθετήσουμε στη σωστή θέση.
  Έτσι, θα οδηγηθούμε σε μία ταξινομημένη λίστα και διαβάζοντας τα πρώτα K στοιχεία της θα εκτυπώσουμε τα αποτελέσματα με τη σειρά που πρέπει.
  Η λίστα διαγράφεται στο τέλος της λειτουργίας search, μέσω της deleteScoreList.

Για την υπογράμμιση των λέξεων που δόθηκαν ως όρισμα κατά την εκτύπωση των αποτελεσμάτων έχει δημιουργηθεί μία λίστα underlineList. Κάθε κόμβος της περιέχει το νούμερο της στήλης, που βρίσκεται το πρώτο γράμμα της λέξης, το μήκος της λέξης και δείκτη σε επόμενο κόμβο και δημιουργείται όποτε συναντάμε κατά την εκτύπωση λέξη που να βρίσκεται στο δυναμικό πίνακα που δημιουργήσαμε στην αρχή.

- ***Df Operation*:** Υπεύθυνη για τη συγκεκριμένη λειτουργία είναι η συνάρτηση dfOperation, η οποία καλώντας τη searchTrie αποκτά πρόσβαση στον κόμβο που περιέχει πληροφορία και δείχνει στην postings list της λέξης, εφόσον εκείνη υπάρχει μέσα στο trie.
  Στην περίπτωση που θέλουμε να εκτυπώσουμε το document frequency vector για όλες τις λέξεις του αρχείου, καλείται η printAllDF, η οποία διατρέχει το trie και εκτυπώνει τα αποτελέσματα σε σειρά με βάση τους ASCII κωδικούς.

- ***Tf Operation*:** Υπεύθυνη για τη συκγκεκριμένη λειτουργία είναι η συνάρτηση tfOperation, η οποία μέσω της searchTrie εντοπίζει την postings list της λέξης στην οποία αναφερόμαστε και διατρέχοντάς την βρίσκει την πληροφορία και την εκτυπώνει.

**Σχόλια:**
Σε όλη την έκταση της υλοποίησης έχουν τοποθετηθεί σχόλια, προκειμένου να γίνεται περισσότερο κατανοητή.

Εκτός από τις συναρτήσεις που αναφέρθηκαν παραπάνω, έχουν δημιουργηθεί και ορισμένες ακόμα οι οποίες συμβάλλουν είτε στη λειτουργία των προηγούμενων είτε είναι βοηθητικές για σκοπούς debugging.

Η εργασία δοκιμάστηκε σε Ubuntu 16.04.4, καθώς και στα μηχανήματα της σχολής, με τα datasets που δόθηκαν.

Συνολικά αποτελείται από τρία αρχεία: *main.c*, *functions.c* και *functions.h*