Αναφορά Project Δομών Δεδομένων

Ονοματεπώνυμο: Δημήτριος Κωστορρίζος e-mail: kostorrizos@ceid.upatras.gr

A.M:1054419 Γλώσσα Προγραμματισμού: C++

Το project έχει υλοποιηθεί σε 2 cpp αρχεία, το Main Project cpp αρχείο περιέχει την υλοποίηση των 4 πρώτων ερωτημάτων και το Trie Project cpp αρχείο περιέχει την υλοποίηση του 5ου ερωτήματος. Πριν την εκτέλεση του κώδικα, να γίνει η αλλαγή του πλήρους Filepath, των αρχείων “integers.txt” και “executiontimes.csv” αντίστοιχα, στις γραμμές 379 και 347 του Main Project cpp και του αρχείου “words.txt”, στην γραμμή 159 στο Trie Project cpp.

Μέρος Α:

Διαβάζεται το αρχείο “integers.txt”, χρησιμοποιώντας το πλήρες Filepath, αποθηκεύοντας τους αριθμούς σε ένα vector. Καλείται η συνάρτηση MergeSort(), δεχόμενη ως ορίσματα έναν pointer, ο οποίος δείχνει στην αρχή του vector που περιέχει τους ακέραιους και τις 2 τιμές για τα όρια του vector. Έπειτα, ο πίνακας χωρίζεται στα 2 και αναδρομικά καλείται η συνάρτηση MergeSort() για τα δύο μισά μέχρις ότου να φτάσουμε σε πίνακες μοναδιαίου μεγέθους. Όταν ολοκληρωθεί η διαδικασία διαχωρισμού ,καλείται αναδρομικά συνάρτηση Merge(), δημιουργώντας 2 προσωρινούς πίνακες για την αποθήκευση των στοιχείων των διαχωρισμένων πινάκων, ταξινομούνται και αντιγράφονται τα στοιχεία μέχρις ότου να αντιγράφουν και ταξινομηθούν όλοι οι ακέραιοι των 2 προσωρινών πινάκων σε έναν ταξινομημένο πίνακα. Μετά την ολοκλήρωση της συνάρτησης MergeSort(), καλείται η συνάρτηση is_sorted(), ελέγχοντας αν το vector που περιέχει τους ακεραίους έχει ταξινομηθεί σωστά ή όχι, εμφανίζοντας κατάλληλο μήνυμα.

Μέρος Β:

Μετά το διάβασμα του αρχείου και την ταξινόμηση του vector, εμφανίζεται ένα μενού στον χρήστη δίνοντάς του την δυνατότητα να αναζητήσει έναν ακέραιο μέσα στο vector, χρησιμοποιώντας όποιον τρόπο αναζήτησης θέλει(Linear,Binary,Interpolation).

1. Η Linear(σειριακή) αναζήτηση ψάχνει 1 προς 1 τα στοιχεία τους vector, μέχρις ότου είτε να βρεθεί το ζητούμενο στοιχείο είτε να φτάσει στο τέλος του vector, επιστρέφοντας αντίστοιχα την λογική τιμή true αν υπάρχει το ζητούμενο στοιχείο ή false αν δεν υπάρχει.

2. Η Binary(δυαδική) αναζήτηση χωρίζει το vector αναδρομικά σε 2 μισά, αναζητώντας το ζητούμενο στοιχείο στο αριστερό μισό τμήμα αν το ζητούμενο στοιχείο είναι μικρότερο του στοιχείου στην μέση, ή στο δεξί μισό τμήμα αν το ζητούμενο στοιχείο είναι μεγαλύτερο του στοιχείου στην μέση. Η αναζήτηση εκτελείται μέχρις ότου είτε να βρεθεί το ζητούμενο στοιχείο

είτε να φτάσει σε αδύνατη διχοτόμηση του vector, η οποία επιτυγχάνεται όταν το δεξί άκρο είναι μικρότερο του αριστερού, επιστρέφοντας αντίστοιχα την λογική τιμή true αν υπάρχει το ζητούμενο στοιχείο ή false αν δεν υπάρχει.

3. Η Interpolation(παρεμβολής) αναζήτηση εκτελείται επαναληπτικά μέχρις ότου είτε βρεθεί το ζητούμενο στοιχείο είτε αποδειχθεί η απουσία του ζητούμενου στοιχείο στο vector, η οποία διαφαίνεται όταν είτε το ζητούμενο στοιχείο δεν ανήκει στο εύρος τιμών του vector στο οποίο σκοπεύουμε να αναζητήσουμε είτε όταν δεν υπάρχουν άλλα στοιχεία για να ελέγξουμε(όταν το δεξί άκρο είναι μικρότερο του αριστερού), τερματίζοντας την αναζήτηση και επιστρέφοντας την λογική τιμή false. Κατά την αναζήτηση, υπολογίζεται η πιθανή θέση του ζητούμενου στοιχείου, σύμφωνα με τον τύπο: $next$←[[(x−$S[left]$)*($right$−$left$)]/ ($S[right]$- $S[left]$)]+$left$, ελέγχεται η παρουσία του ζητούμενου στοιχείου στην πιθανή θέση, επιστρέφοντας την λογική τιμή true αν υπάρχει το ζητούμενο στοιχείο. Στην περίπτωση αποτυχίας, αλλάζουν οι διαστάσεις του πίνακα στον οποίο γίνεται η αναζήτηση, γεγονός το οποίο επιτυγχάνεται με την αντικατάσταση του δεξιού ορίου με την πιθανή θέση μειωμένη κατά 1, αν το προς αναζήτηση στοιχείο είναι μικρότερο του στοιχείου στην πιθανή θέση, ή με την αντικατάσταση του αριστερού ορίου με την πιθανή θέση αυξημένη κατά 1, αν το προς αναζήτηση στοιχείο είναι μεγαλύτερο του στοιχείου στην πιθανή θέση.

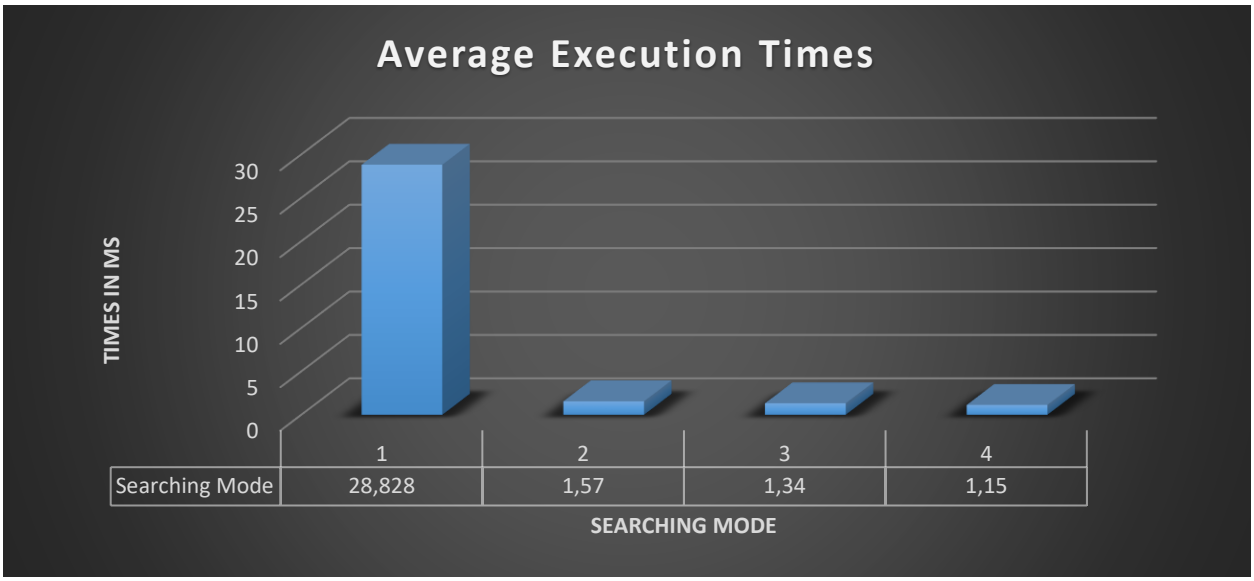
Μέρος Γ:

Για την υλοποίηση των κόμβων του Red-Black Tree, έχει οριστεί ένα struct(red_black_tree_node) που περιέχει ως μεταβλητές, μία ακέραια τιμή value, μία λογική τιμή colour (false για το κόκκινο, true για το μαύρο). Για κάθε αριθμό που υπάρχει στο vector, δημιουργείται ένας νέος κόμβος, ενημερώνονται οι μεταβλητές του κόμβου με τις αντίστοιχες τιμές και καλείται η συνάρτηση Red_Black_Tree_Insertion, η οποία εισάγει τους κόμβους, δημιουργώντας έτσι το δέντρο. Μετά την εισαγωγή του κόμβου καλείται η συνάρτηση Red_Black_Tree_Fixing, η οποία ελέγχει για την ορθότητα του δέντρου. Σε περίπτωση, ανάγκης για περιστροφή, καλούνται η συναρτήσεις για δεξιά ή αριστερή περιστροφή, Red_Black_Tree_Right_Rotation και Red_Black_Tree_Left_Rotation αντίστοιχα. Μέσα από το κύριο μενού δίνεται στον χρήστη η δυνατότητα προσθήκης νέου στοιχείου στο δέντρο και η αναζήτηση ενός στοιχείου σε αυτό. Καλώντας την συνάρτηση Red_Black_Tree_Search, ξεκινάει η αναζήτηση του ζητούμενου αριθμού μέχρι είτε να βρεθεί το στοιχείο επιστρέφοντας true είτε τον τερματισμό της αναζήτησης φτάνοντας σε φύλλο του δέντρου, οπότε επιστρέφεται και false.

Μέρος Δ:

Επιλέγοντας την επιλογή με τον αριθμό 5 από το μενού, εκτελούνται 1000 αναζητήσεις για 1000 τυχαία παραγόμενους ακεραίους, με τιμές από το 0 έως 999999, σε κάθε ένα εκ των 4 είδών αναζήτησης (Linear,Binary,Interpolation,Red-Black Tree).Ο χρόνος εκτέλεσης της κάθε αναζήτησης καταγράφεται και αποθηκεύεται σε ένα πίνακα με διαστάσεις 4Χ1000. Μετά την εκτέλεση των αναζητήσεων αυτών, ο πίνακας αυτός αποθηκεύεται σε ένα αρχείο CSV με όνομα “executiontimes.csv”. Έπειτα, ανοίγοντας το αρχείο “graphs.xlsx” και πατώντας το κουμπί Refresh All(Ctrl+Alt+F5) στην καρτέλα Data, το αρχείο ενημερώνεται με τις νέες τιμές των χρόνων εκτέλεσης των αναζητήσεων, ενημερώνοντας με την σειρά

τους, τις τιμές των γραφημάτων, των μεγίστων και των μέσων χρόνων αναζητήσεων για κάθε τύπο αναζήτησης. Για τον υπολογισμό, των μεγίστων και των μέσων χρόνων αναζητήσεων για κάθε τύπο αναζήτησης έχουν χρησιμοποιηθεί οι έτοιμες συναρτήσεις που παρέχει το Excel. Παραθέτω, 2 γραφήματα για ένα δείγμα 1000 αναζητήσεων.



Μέρος Ε:

Διαβάζεται το αρχείο “words.txt”, χρησιμοποιώντας το πλήρες Filepath, αποθηκεύοντας τις λέξεις σε ένα vector. Για την υλοποίηση των κόμβων του Digital Tree, έχει οριστεί ένα struct(Digital_Tree_Node) που περιέχει ως μεταβλητές, , μία λογική τιμή end_of_word (true αν ο κόμβος αντιστοιχεί στο τελευταίο γράμμα της λέξης) και ένα πίνακα δεικτών 26 θέσεων(μια θέση για κάθε γράμμα το λατινικού αλφαβήτου). Για κάθε λέξη που υπάρχει στο vector, δημιουργείται ένας νέος κόμβος,

ενημερώνονται οι μεταβλητές του κόμβου με τις αντίστοιχες τιμές και καλείται η συνάρτηση Digital_Tree_Insertion, η οποία δημιουργεί τους κόμβους χρησιμοποιώντας την συνάρτηση Create_Digital_Tree_Node, αντιστοιχίζει τον κόμβο στην σωστή θέση τοποθετώντας την αντίστοιχη θέση του πίνακα, δείκτη προς τον κόμβο αυτό. Μετά την δημιουργία του δέντρου, εμφανίζεται στον χρήστη ένα μενού, δίνοντάς του την δυνατότητα προσθήκης μίας λέξης στο δέντρο, αναζήτησης και διαγραφής. Κατά την αναζήτηση, ακολουθείται η σειρά των κόμβων ξεκινώντας από την αρχή της λέξης. Αν υπάρχει η προς αναζήτηση λέξη, επιστρέφεται true αλλιώς επιστρέφεται false. Κατά την διαγραφή μίας λέξης από το δέντρο, εκτελείται η συνάρτηση Digital_Tree _Delete, ελέγχεται σε ποια από τις οι παρακάτω περιπτώσεις ανήκει η λέξη προς διαγραφή, οπότε και εκτελούνται οι ανάλογες ενέργειες. Για την διαγραφή των κόμβων καλείται η αναδρομική συνάρτηση Digital_Tree_Node_Delete, σε συνδυασμό με την συνάρτηση Is_Leaf_Node, η οποία ελέγχει αν ένα κόμβος είναι φύλο επιστρέφοντας την λογική τιμή true αν είναι ή την λογική τιμή false, εάν δεν είναι και την συνάρτηση Is_Fork_Node, η οποία ελέγχει αν ένα κόμβος έχει 2 ή και περισσότερα παιδιά επιστρέφοντας την λογική τιμή true αν έχει ή την λογική τιμή false εάν έχει 1 ή 0 παιδιά.

1. Η ζητούμενη λέξη δεν υπάρχει στο δέντρο, οπότε το δέντρο παραμένει αμετάβλητο και επιστρέφεται η τιμή false.

2. Η ζητούμενη λέξη δεν περιέχει άλλη λέξη και δεν περιέχεται σε άλλη ήδη υπάρχουσα λέξη, οπότε διαγράφονται όλοι οι κόμβοι της λέξης αυτής και επιστρέφεται η τιμή true.

3. Η ζητούμενη λέξη περιέχεται σε άλλη ήδη υπάρχουσα λέξη, οπότε η μεταβλητή end_of_word του κόμβου του τελευταίου γράμματος της προς διαγραφής λέξης αλλάζει από true σε false και επιστρέφεται η τιμή true.

4. Η ζητούμενη λέξη περιέχει άλλη λέξη, οπότε διαγράφονται οι κόμβοι της λέξης ξεκινώντας από τον προηγούμενο (κατά την διαπέραση των κόμβων της λέξης) κόμβο του οποίου η μεταβλητή end_of_word ισούται με την τιμή true, και επιστρέφεται η τιμή true.