ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS
Σταμάτης Χουλιαράς
3190221
Ιωάννης Βουσβουκής
3190027

Εργασία 3

Insert:

Στην insert δεχόμαστε ένα αντικείμενο τύπου Subject και χρησιμοποιούμε μια βοηθητική συνάρτηση την insertAsRoot. Ουσιαστικά ξεκινάμε από την ρίζα που υπάρχει μια πιθανότητα να τοποθετηθεί ο καινούριος κόμβος. Αν δεν τοποθετηθεί στην ρίζα, θα γίνει σύγκριση με το ΑΦΜ της ρίζας και ανάλογα το αποτέλεσμα θα επαναλάβουμε αυτή την διαδικασία με το head του αριστερού ή δεξιού υποδέντρου. Η εισαγωγή ενός στοιχείου εκτελείται σε O(LogN), όμως πραγματοποιείται και έλεγχος για το αν υπάρχει ήδη το ΑΦΜ στο δέντρο μέσω της SearchByAFM που έχει και αυτή O(LogN) οπότε η τελική πολυπλοκότητα ειναι O(logN^2).

Load:

Στην load δεχόμαστε ως είσοδο ένα αρχείο txt που περιέχει τα στοιχεία των αντικειμένων suspect που θέλουμε να εισάγουμε στο τυχαιοποιημένο δέντρο. Αρχικά, ελέγχουμε αν υπάρχει το αρχείο που μας δίνεται σαν είσοδος. Έπειτα δημιουργούμε το αντικείμενο suspect και με τις μεθόδους setFirstName, setLastName κτλ. εισάγουμε τα στοιχεία του. Τέλος με την βοήθεια της insert το νέο αντικείμενο τοποθετείται κατάλληλα στο δέντρο. Άρα αφού χρησιμοποιεί την insert για κάθε αντικείμενο που περιέχει το αρχείο, η πολυπλοκότητα της είναι O(NlogN).

SearchByLastName:

Σε αυτή την συνάρτηση πρέπει να κάνουμε traverse σε όλο το δέντρο για να βρούμε όλους τους Suspect με το ίδιο επίθετο και να εκτυπώσουμε τα στοιχεία τους. Πραγματοποιούμε ενδοδιατεταγμένη διάσχιση, δηλαδή ξεκινάμε προσπέλαση από το αντικείμενο με το μικρότερο ΑΦΜ και έπειτα συνεχίζουμε την προσπέλαση με αύξουσα σειρά. Αν το αντικείμενο που κάνουμε προσπέλαση έχει το last name που ψάχνουμε τότε το προσθέτουμε σε μία λίστα μονής σύνδεσης. Όταν τελειώνει η διάσχιση, τότε εκτυπώνουμε τα στοιχεία αυτής της λίστας. Η διάσχιση απαιτεί O(N) χρόνο και η εκτύπωση πάλι έχει ως φράγμα το O(N). Οπότε η πολυπλοκότητα είναι O(N).

SearchByAFM:

Σε αυτή την συνάρτηση δεχόμαστε ενα ΑΦΜ και πραγματοποιούμε αναζήτηση στο δέντρο για να επιστρέψουμε τα στοιχεία του Suspect με το συγκεκριμένο ΑΦΜ. Αυτή η αναζήτηση γίνεται όπως και στα κανονικά δέντρα δυαδικής αναζήτησης. Ξεκινάμε από την ρίζα και κάνουμε συγκρίσεις των κλειδιών ΑΦΜ και ανάλογα επιλέγουμε το αριστερό ή το δεξιό υποδέντρο μέχρι να φτάσουμε στο ΑΦΜ που ψάχνουμε. Η πολυπλοκότητα της συγκεκριμένης συνάρτησης είναι O(LogN).

UpdateSavings:

Στην UpdateSavings δεχόμαστε ένα ΑΦΜ και του ανανεώνουμε τα savings. Χρησιμοποιείται η SearchByAFM και έπειτα απλά αλλάζουμε τα στοιχεία του αντικειμένου μέσω της setSavings. Η πολυπλοκότητα είναι ακριβώς ίδια με αυτή της SearchByAFM

Remove:

Η remove λειτουργεί κάπως παρόμοια με την insert όσον αφορά το πως βρίσκουμε το στοιχείο που θέλουμε να διαγράψουμε. Χρησιμοποιούμε και εδώ μια βοηθητική συνάρτηση (RemoveR) αλλά και την joinLR που βοηθάει στην σύνδεση κόμβων όταν αφαιρεθεί το κατάλληλο στοιχείο αλλά και στο να ανανεωθεί ο αριθμός N που αφορά τον αριθμό κόμβων που έχει ένα head από κάτω του. Η removeR εκτελείται σε O(LogN).

GetMeanSavings:

Σε κάθε δέντρο υπάρχει το άθροισμα των Savings από όλα τα στοιχεία του. Επειδή όμως τα Savings μπορεί να ανανεωθούν μέσω της UpdateSavings, κάθε φορά που καλείται η getMeanSavings αυτό το ποσό γίνεται reset και ξαναυπολογίζεται μέσω ενδοδιατεταγμένης διάταξης που προσθέτει στο άθροισμα τα savings των στοιχείων που κανει προσπέλαση. Αφού μιλάμε για διάσχιση ολόκληρου του δέντρου η πολυπλοκότητα θα είναι O(N).

PrintByAFM:

Πραγματοποιεί ενδοδιατεταγμένη διάσχισει όπως και άλλες συναρτήσεις και εκτυπώνει τα στοιχεία των αντικειμένων που κάνει προσπέλαση σε αύξουσα σειρά όσον αφορά το ΑΦΜ τους. Όπως και οι άλλες συναρτήσεις που πραγματοποιούν διάσχιση εκτελείται σε O(N).

PrintTopSuspects:

Σε αυτή την συνάρτηση εκτυπώνουμε τους πιο ύποπτους Suspects που έχουμε στο δέντρο μας. Αυτό γίνεται με την βοήθεια της compareSuspects που παίρνει ως είσοδο δύο Suspects και επιστρέφει πιο είναι πιο ύποπτο με κριτήρια που δόθηκαν στην εκφώνηση.Οι κ (είσοδος του χρήστη) πιο ύποπτοι ανανεώνονται συνεχώς σε μια λίστα μονής σύνδεσης που κρατά μονίμως κ ή λιγότερα στοιχεία ενεργά και στο τέλος τα εκτυπώνει με αύξουσα σειρά ως προς το πόσο ύποπτοι είναι. Πάλι χρησιμοποίουμε διάσχιση αλλά και ταξινόμηση σε κάθε στοιχείο που θέλουμε να προσθέσουμε στην λίστα με τους πιο ύποπτους.Άρα η πολυπλοκότητα της συνάρτησης αυτής έχει ως άνω φράγμα το O(k*N).

Διευκρίνιση:

Ως N που χρησιμοποιούμε για να εκφράσουμε την πολυπλοκότητα των συναρτήσεων εννοούμε τους κόμβους που περιέχει κάθε δέντρο (δηλαδή τα TreeNodes). Επίσης το αρχείο txt πρέπει να βρίσκεται μέσα στον φάκελο src και έπειτα το μόνο που πρέπει να κάνετε για να το φορτώσετε στο δέντρο είναι να πατήσετε 1 στο interface και να γράψετε σαν είσοδο το όνομα του αρχείου με την κατάληξη txt στο τέλος.