ΑΝΑΦΟΡΑ 2ΟΥ PROJECT ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΑΡΧΕΙΩΝ

ΟΝ.:ΑΝΑΣΤΑΣΙΟΣ ΚΑΡΑΓΕΩΡΓΙΑΔΗΣ ΑΜ.:2013030135 ΕΡΓ.ΟΜΑΔΑ: LAB21124955

ΥΛΟΠΟΙΗΣΗ B+TREE

Εισαγωγή Ν κλειδιών: Η υλοποίηση της διαδικασίας της εισαγωγής έγινε από μια συνάρτηση **insert()** στην οποία καλείται μια συνάρτηση **orderArray()** ,η οποία ταξινομεί τον πίνακα στον οποίο κρατάμε τα τυχαία κλειδιά, αφού με την σειρά της καλέσει την **randomInt()** για να πάρει τους τυχαίους αριθμούς που θέλουμε. Έπειτα επιστρέφουμε στην **insert()** για να καλέσουμε για κάθε κλειδί μια συνάρτηση **insertKey(key),** για να διακρίνει τις περιπτώσεις εισαγωγής κλειδιών και να γίνει σωστή εισαγωγή στους κόμβους.

Αναζήτηση Κλειδιού: Χρησιμοποιούμε μια συνάρτηση **searchAkey(key)** για να βρούμε που βρίσκεται το κλειδί που ζητάμε (αν υπάρχει) και να εκτυπώσουμε τα αντίστοιχα μηνύματα, αφού όμως πρώτα έχουμε καλέσει τη συνάρτηση **search4node(rootNode,key)** η οποία μας επιστρέφει σε ποιον κόμβο πιθανόν να βρίσκεται το κλειδί και μετά προσπελαύνουμε ένα-ένα τα κλειδιά του κόμβου αυτού.

Αναζήτηση Εύρους Κλειδιών:

Καλούμε την συνάρτηση **searchRageKeys(downLimit,upLimit)** και επαναλαμβάνουμε την παραπάνω διαδικασία της αναζήτησης κλειδιού μονό που τώρα κάνουμε την διαδικασία για ***key=downLimitKey*** και μετά μέχρις ότου βρούμε το πάνω όριο ή τελειώσουν τα κλειδιά προσπελαύνουμε ένα-ένα τα κλειδιά του κόμβου και όσα είναι εντός εύρους τα αποθηκεύουμε σε ένα πίνακα.

Διαγραφή Κλειδιού*:* Καλούμε την συνάρτηση **deleteKey(key)** και αφού βρούμε το κλειδί κάνοντας την διαδικασία της αναζήτησης κλειδιού, διαγράφουμε το κλειδί από το φύλλο που βρίσκεται και μετά ψάχνουμε και στον πατέρα του για να το σβήσουμε. Τέλος αντικαθιστούμε την θέση του στον πατέρα με το δεξιότερο κλειδί του αριστερού παιδιού ώστε να διατηρηθεί η ομοιομορφία του δέντρου. Για να βρούμε σε ποια θέση στον πατέρα βρίσκεται το κλειδί που θέλουμε να διαγράψουμε χρησιμοποιούμε την μέθοδο **searchKinN(node,key).**

Σελιδοποίηση κόμβων στο αρχείο: Η σελίδα μέσα στην οποία θα βρισκόταν ο κάθε κόμβος βρέθηκε με τυχαίο τρόπο, δηλαδή πιο συγκεκριμένα μετρούσα κάθε φορά σε μία τύπου ***int*** μεταβλητή ,ορατή σε ολόκληρη την κλάση **BPtree**, τον αριθμό των κόμβων (πλην τις ρίζα η οποία θα βρίσκεται πάντα στη θέση 0) κάθε φορά που γινόταν **split** αυτή σταθερά αυξανόταν κατά 1 και αυτός αριθμός γινόταν η σελίδα τους ενός από τους δυο κόμβους που δημιουργούνταν κατά το **split** και του άλλου κόμβου η σελίδα ήταν αυτός ο αριθμός -1. Επειδή ο κόμβος στον οποίο γινόταν το **split** ουσιαστικά άλλαζε ένας από τους δύο *<<καινούργιους>> κόμβους(παιδιά)* που δημιουργούταν ξαναγραφόταν στην ίδια σελίδα με τον αρχικό (τον οποίο έκανα **split**). Τέλος κατά το διάβασμα ή το γράψιμο μιας σελίδας του αρχείου κάναμε **seek(PageNumber*PageSize)** για βρούμε την σωστή θέση του κόμβου με στο αρχείο.

++Αναζήτηση: Την αναζήτηση την έσπασα σε δυο κομμάτια, δηλαδή πρώτα με την συνάρτηση **search4node(node,key)** βρίσκουμε σε ποιον κόμβο θα πρέπει να εισαχθεί το κλειδί (αυτό χρησιμεύει πολύ κατά την εισαγωγή) και μετά σε αυτόν τον κόμβο ψάχνουμε για το κλειδί (απλή αναζήτηση) ή είναι ο κόμβος αφετηρίας μας(για αναζήτηση εύρους).

+Αλγόριθμός συνάρτησης search4node(node,key) : Ξεκινώντας από το κόμβο της ρίζας διασχίζουμε το δέντρο μέχρι το φύλλο που ψάχνουμε για το κλειδί (δηλ. το φύλλο που μπορεί να βρίσκεται το κλειδί που ζητάμε ή εκεί που πρέπει να εισάγουμε το κλειδί) ,ελέγχουμε τα κλειδιά κάθε κόμβου μέχρι να βρούμε το κλειδί στο κόμβο που είναι αμέσως μεγαλύτερο από το κλειδί που θέλουμε, μόλις ισχύσει η παραπάνω συνθήκη παίρνουμε *το αριστερό παιδί* αυτού του κλειδιού και επαναλαμβάνουμε την διαδικασία μέχρι το ζητούμενο φύλλο(φορτώναμε το αντίστοιχο παιδί από το αρχείο). Στην περίπτωση που το κλειδί που θέλουμε είναι μεγαλύτερο από όλα τα κλειδιά του κόμβου τότε παίρνουμε *το δεξί παιδί το τελευταίου κλειδιού του κόμβου*.

Συμπεράσματα:

- Εισαγωγή 100000 τυχαίων κλειδιών: Οι προσβάσεις που έγιναν κατά την εισαγωγή ήταν 30.6198.
- Αναζήτηση 20 τυχαίων κλειδιών: Κατά την αναζήτηση των παρακάτω τυχαίων κλειδιών προκύπτει ο εξής **μέσος αριθμός προσβάσεων=3** δηλαδή όσο και τα επίπεδα του δέντρου αν αναλογιστούμε πόσα κλειδιά εισάγουμε και πόσα κλειδιά μπορεί να αποθηκεύσει ένας κόμβος(προκύπτει ότι 3 επίπεδα αρκούν για να χωρέσουν όλα τα κλειδιά).

Κλειδιά	5	24	65	220	243	425	996	2.230	10.295	15.058
Προσβάσεις	3	3	3	3	3	3	3	3	3	3

Κλειδιά	15.905	31.136	44.951	56.117	62.275	76.416	78.652	79.001	83.954	97.321
Προσβάσεις	3	3	3	3	3	3	3	3	3	3

* Σημείωση: τα κλειδιά με κόκκινο χρώμα δεν υπήρχαν στο δέντρο.

- Αναζήτηση Εύρους κλειδιών: Ο μέσος όρος προσβάσεων κατά την αναζήτηση κλειδιών με συγκεκριμένο εύρος, όπως προκύπτει από τον παρακάτω πίνακα είναι **μ.ο.π=(3+4+4+7+10)/5=5,6.** Σε αυτήν την περίπτωση η πρόσβαση σε ένα εύρος υπολογίζεται από τις προσβάσεις που κάνουμε μέχρι να βρούμε το κάτω όριο οι οποίες είναι όσα και τα επίπεδα του δέντρου, προσθέτοντας τις όποιες προσβάσεις θα κάνουμε σε συγγενικούς κόμβους που ατό εξαρτάται από το πόσα κλειδιά έχουμε σε αυτό το εύρος και πόσα κλειδιά χωράει ο κόμβος.

Εύρος Κλειδιών	[10,50]	[10,101]	[10, 212]	[25,527]	[16,25000]
Προσβάσεις	3	4	4	7	10

- Διαγραφή 20 τυχαίων κλειδιών: Κατά την διαγραφή ενός τυχαίου κλειδιού έχουμε **τουλάχιστον** όσες προσβάσεις και με την αναζήτηση (αριθμός των επιπέδων του δέντρου),στην συγκεκριμένη περίπτωση 3,αν ο μίσος κόμβος(τουλάχιστον) είναι γεμάτος αλλιώς αν δεν είναι γεμάτος ο μισός κόμβος κάνουν ***borrow*** ή ***merge*** με συγγενικούς κόμβους που αυτό μας στοιχίζει άλλες τρεις προσβάσεις το πολύ(διάβασμα γειτονικού κόμβου, εγγραφή κόμβου πατέρα(μετά τις αλλαγές),εγγραφή καινούργιου κόμβου(με τις αλλαγές)).

κλειδιά	5	24	65	220	243	425	996	2230	10295	15058
προσβάσεις	4	4	4	4	6	4	4	6	4	6

Κλειδιά	15905	31136	44951	56117	62275	76416	78652	79001	83954	97321
προσβάσεις	6	6	6	4	6	4	3	3	3	3

*Σημείωση: τα κλειδιά με κόκκινο χρώμα δε βρέθηκαν στο δέντρο.

Μ.ο.π1=(4*4+6+4*2+6+4+6*4+4+6+4)/16=4,875 (χωρίς τα κλειδιά που δε βρέθηκαν)
Μ.ο.π2=(Μ.ο.π*16+3*4)/20=4.5(με αυτά που δε βρέθηκαν)

Πηγές:

- Wikipedia.com (για αλγορίθμους και δομή B+ δέντρου),
- Σημειώσεις μαθήματος (φροντιστήριο, διαλέξεις)
- Ιστοσελίδα της oracle σχετικά με μεθόδους και κλάσεις της Java

Extras!!: Εκτύπωση δεδομένων μιας σελίδας του αρχείου δίνοντας έναν πιθανό αριθμό σελίδας εκτυπώνει τα δεδομένα του κόμβου οποίος έχει γραφτεί εκεί.

Υ.γ.: Επειδή στο πρόγραμμα δε δούλευε σωστά η διαγραφή κλειδιού(κυρίως επειδή έκανα κάποιες παραδοχές κατά την κατασκευή του αλγορίθμου που θέλουν επανεξέταση αλγόριθμός βρίσκεται στην κλάση BPtree), αναγκαστικά είδα ποια κλειδιά περιέχονται μέσα στο δέντρο(όχι όλα κάποια από αυτά) και με βάσει της συναρτήσεις που έγραψα προσπαθούσα θεωρητικά να υπολογίσω τις προσβάσεις με βάσει τα read και write στο αρχείο. Τα επίπεδα του δέντρου υπολογίστηκαν από τον μέγιστο αριθμό κλειδιών κάθε κόμβου και το πλήθος των κλειδιών που εισάγουμε.