Μανώλης Πετράκος 2014030009

ΠΕΡΙΓΡΑΦΗ 1ης ΑΣΚΗΣΗΣ

Εισαγωγή στοιχείων στο δίσκο και δημιουργία ερωτήσεων.
Απαιτείται η δημιουργία ενός δυαδικού αρχείου στο δίσκο που περιέχει 10.000.000 κλειδιά. Εάν υπάρχει ήδη το αρχείο στο δίσκο τότε το πρόγραμμα δεν δημιουργεί άλλο. Αλλιώς φτιάχνει ένα αρχείο τυχαίας αναζήτησης εισάγοντας επαναληπτικά σελίδες 128 κλειδιών μέχρι να φτάσει στον επιθυμητό αριθμό στοιχείων. Κάθε κλειδί έχει μέγεθος 4 byte άρα η σελίδα θα αποτελείται από 512. Τα κλειδιά είναι σε αύξουσα σειρά και ξεκινούν από το 1 μέχρι το 10.000.000.

Υπάρχει μία μέθοδος η οποία δημιουργεί δύο πίνακες 10.000 τυχαίων αριθμών. Έχουν τα ίδια στοιχεία αλλά ο ένας είναι ταξινομημένος και ο άλλος αταξινόμητος. Έτσι αφαιρείται ο παράγοντας της τύχης από τα τελικά αποτελέσματα αφού όλοι οι αλγόριθμοι θα έχουν την ίδια είσοδο.

Δυαδική αναζήτηση μέσα σε σελίδα.
Όλοι οι τρόποι αναζήτησης απαιτούν η εύρεση του κλειδιού μέσα σε μια σελίδα να γίνει με δυαδική αναζήτηση, για αυτό έχει δημιουργηθεί μια μέθοδος με είσοδο την σελίδα ως έναν πίνακα 512 bytes και το κλειδί που ψάχνεται. Κάθε φορά που πρέπει να συγκριθεί ένα στοιχείο του πίνακα με το κλειδί απομονώνονται τα 4 bytes που χρειάζονται και μετατρέπονται σε έναν integer. Ακολουθεί η κλασική λειτουργία της δυαδικής αναζήτησης, δηλαδή συγκρίνεται το κλειδί με το μεσαίο στοιχείο,αν είναι το ίδιο βρέθηκε, αν είναι μικρότερο πάει στο 1/4 στοιχείο σε σειρά και αν είναι μεγαλύτερο πάει στο 3/4 στοιχείο κλπ. Αν είχε χρησιμοποιηθεί η έτοιμη συνάρτηση της Java θα έπρεπε να είχε μετατραπεί όλος ο πίνακας των bytes σε πίνακα integers, δηλαδή θα προσπελάζονταν και τα 128 στοιχεία, αντίθετα τώρα μετατρέπονται από byte σε integer το πολύ $\ln_2(128) = 7$ στοιχεία και η αναζήτηση είναι αρκετά πιο γρήγορη.

Α) Σειριακή αναζήτηση στο αρχείο για τυχαίο κλειδί.
Πρώτη μέθοδος αναζήτησης είναι η σειριακή, δηλαδή τραβάει μία μία τις σελίδες από το αρχείο. Οι σελίδες ανεβαίνουν στην κεντρική μνήμη με τη σειρά, από την πρώτη μέχρι τη σελίδα όπου θα βρεθεί το στοιχείο. Αν το κλειδί δεν βρεθεί θα εξαντληθούν όλες οι σελίδες και θα σταματήσει την αναζήτηση.

Β) Δυαδική αναζήτηση στο αρχείο για τυχαίο κλειδί.
Η δυαδική αναζήτηση αντίθετα με την σειριακή δεν χρειάζεται να τραβήξει όλες τις σελίδες μέχρι την μία όπου υπάρχει το κλειδί. Ξεκινάει από την μεσαία,την ψάχνει και αν βρει το κλειδί σταματάει. Αν δεν το βρει κοιτάει το πρώτο στοιχείο της σελίδας και αν το κλειδί είναι μικρότερο από αυτό αλλάζει το πεδίο αναζήτησης στο κάτω μισό του αρχείου. Αντίστοιχα αν το κλειδί είναι μεγαλύτερο από το τελευταίο αριθμό της σελίδας θα περιοριστεί το πεδίο αναζήτησης στο πάνω μισό της σελίδας. Τότε κοιτάει τη μεσαία σελίδα του νέου πεδίου και συνεχίζει την ίδια διαδικασία. Συνεχίζει με αυτόν τον τρόπο μέχρι να βρεθεί το στοιχείο ή να μην υπάρχει πια πεδίο για να αναζητήσει.

Γ) Δυαδική αναζήτηση με ομαδοποίηση των ερωτήσεων
Η αναζήτηση βασίζεται στον προηγούμενο αλγόριθμο αλλά επειδή χρησιμοποιείται ο ταξινομημένος πίνακας ερωτήσεων υπάρχει η πιθανότητα οι επόμενες ερωτήσεις να βρίσκονται στην ίδια σελίδα που έχει ανέβει στο δίσκο και έχει βρεθεί το κλειδί. Το πρόγραμμα αφού κάνει μια επιτυχημένη αναζήτηση κοιτάει αν οι αμέσως επόμενες ερωτήσεις υπάρχουν στην σελίδα μέχρι να έρθει μία ερώτηση που δεν περιέχεται. Τότε συνεχίζει η αναζήτηση κανονικά από αυτήν. Θα μπορούσε να γίνει η ίδια διαδικασία και σε έναν αταξινόμητο πίνακα αλλά θα ήταν σχεδόν αδύνατο να έρθουν 2 κλειδιά που ανήκουνε στην ίδια σελίδα συνεχόμενα.

Δ) Δυαδική αναζήτηση με χρήση προσωρινής μνήμης.
Η μέθοδος διατηρεί μια ουρά στη κεντρική μνήμη προδιαγεγραμμένου μεγέθους η οποία περιέχει σελίδες που έχουν έρθει από τον δίσκο. Πριν γίνει αναζήτηση για ένα κλειδί στο δίσκο η μέθοδος κοιτάει αν είναι εκεί βγάζοντας το head από τη στοίβα, ελέγχοντας το και ξαναβάζοντας το μέσα. Αυτό γίνεται σε όλες τις σελίδες και έτσι κρατιέται η παλαιότητα τους σωστή. Αν δεν βρεθεί το κλειδί τότε γίνεται αναζήτηση στο δίσκο με την μέθοδο της δυαδικής αναζήτησης. Όταν βρεθεί το κλειδί προστίθεται η σελίδα του στη στοίβα και εάν αυτή έχει γεμίσει διαγράφεται η πιο παλιά. Επειδή εισάγονται μόνο σελίδες που έχουν μόλις βρεθεί σε αυτές κλειδιά, είναι σίγουρο πως δεν θα υπάρχει η ίδια σελίδα δύο φορές στην στοίβα, γιατί δεν εισάγονται οι ενδιάμεσες.

Τεκμηρίωση των αποτελεσμάτων

Μέθοδος	Α	Β	Γ	Δ(Κ=1)	Δ(Κ=50)	Δ(Κ=100)
Απόδοση	39357	15.341	14.3916	15.341	15.3378	15.333

Η συριακή αναζήτηση επειδή πρέπει να ψάξει όλες τις σελίδες μέχρι να βρει την σωστή είναι πολύ πιο αργή από τις υπόλοιπες. Αν ένα κλειδί είναι στη πρώτη σελίδα θα το βρει αμέσως αλλά αν είναι στην τελευταία θα πρέπει να ανεβάσει όλο το αρχείο. Εφόσον τα κλειδιά είναι σωστά κατανεμημένα ο μέσος όρος προσβάσεων στο δίσκο θα είναι περίπου ίσος με τον αριθμό των σελίδων δια δύο. Στη προκειμένη περίπτωση οι σελίδες είναι 78125 και ο μέσος όρος 39357, δηλαδή ισχύει.

Αντίθετα η δυαδική αναζήτηση δεν χρειάζεται στη χειρότερη περίπτωση να ψάξει όλες τις σελίδες για να βρει ένα κλειδί αλλά το πολύ $\log_2(78125)$. Σε κάθε αναζήτηση ξεκινάει από τη μέση και κόβει πάντα το πεδίο αναζήτησης στο μισό και συνεχίζει στο νέο πεδίο με τον ίδιο τρόπο. Έτσι μετά το πρώτο τράβηγμα σελίδας από το δίσκο οι πιθανές σελίδες έχουν μειωθεί στο 1/2, στο δεύτερο στο 1/4 κ.λπ.

Η μέθοδος ομαδοποίησης των ερωτήσεων εκμεταλλεύεται ότι οι ερωτήσεις είναι σε αύξουσα σειρά και κάθε φορά που βρίσκει ένα κλειδί με τον τρόπο της δυαδικής αναζήτησης τσεκάρει αν τα αμέσως επόμενα βρίσκονται και αυτά στην ίδια σελίδα. Υπάρχουν 78125 σελίδες και 10000 ερωτήσεις άρα οι περισσότερες σελίδες δεν θα ικανοποιούν καμία, και οι πιο πολλές από αυτές το πολύ μία. Όμως, περίπου το 6% των ερωτήσεων βρίσκονται στην ίδια σελίδα με μία άλλη και θα υπάρξει μια ελαφριά μείωση στον μέσο αριθμό προσβάσεων στο δίσκο. Είναι η αποδοτικότερη μέθοδος από όλες.

Τέλος η μέθοδος με την προσωρινή μνήμη προσπαθεί να εκμεταλλευτεί ότι μπορούμε να κρατήσουμε μια σελίδα στη μνήμη για περισσότερη ώρα. Τσεκάροντας τις ερωτήσεις με τις σελίδες που έχουν ανέβει πιο πρόσφατα υπάρχει η πιθανότητα να βρεθούν και να μην χρειαστεί να γίνει δυαδική αναζήτηση στο δίσκο. Όσες περισσότερες σελίδες κρατιούνται τόσο ανεβαίνουν και οι πιθανότητες να βρεθεί ένα κλειδί άρα μειώνεται και ο μέσος αριθμός προσβάσεων στο δίσκο. Στην περίπτωση όπου κρατιέται μόνο μία σελίδα είναι σχεδόν αδύνατο να βρεθεί κάποιο κλειδί πιο γρήγορα γιατί πρακτικά προσπαθεί να κάνει το προηγούμενο ερώτημα αλλά χωρίς ομαδοποιημένες ερωτήσεις. Αντίθετα όταν μένουν στην μνήμη περισσότερες σελίδες υπάρχει μια αναλογική με τον αριθμό τους μείωση τού μέσου όρου προσβάσεων στο δίσκο.

Υπάρχουν δύο ακόμα τρόποι υλοποίησης της μεθόδου. Πρώτον θα μπορούσαμε να κρατήσουμε όλες τις σελίδες που ανεβαίνουν στο δίσκο και όχι μόνο τις τελικές, με αποτέλεσμα να μειωθούν ακόμα περισσότερο οι προσβάσεις. Η στοίβα θα ανανεωνόταν πιο συχνά, θα έπρεπε να γίνεται έλεγχος να μην υπάρχει η ίδια σελίδα δυο φορές και αν δεν βρει το στοιχείο στη στοίβα να μην ξαναδιαβαστούν οι ίδιες ενδιάμεσες σελίδες στο δίσκο. Δεύτερος τρόπος θα ήταν να κρατιούνται οι σελίδες με σειρά χρήσης και όχι παλαιότητας. Δηλαδή αντί να διαγράφεται η πιο παλιά σελίδα θα διαγράφεται αυτή που έχει χρησιμοποιηθεί λιγότερο από όλες. Σελίδες όπως η κεντρική θα μέναν πάντα στη στοίβα, καθώς έχουν την μεγαλύτερη χρήση κατά τη διάρκεια λειτουργίας του συστήματος. Οι προσβάσεις στο δίσκο θα μειωνόντουσαν αισθητά καθώς οι πιο χρήσιμες σελίδες θα ήταν πάντα διαθέσιμες.

Πηγές

1. Κώδικας φροντιστηρίου : Δημιουργία, διάβασμα και διαχείριση αρχείων.
2. docs.oracle.com/ : BufferedReader class
3. Wikipedia.com : Δυαδική αναζήτηση
4. stackoverflow.com/ : Διαχείριση bytes.