ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΑΛΓΟΡΙΘΜΟΙ

1η άσκηση

Ημερομηνία παράδοσης: 24 Μαρτίου 2022

Η άσκηση είναι ατομική

Σκοπός της άσκησης είναι η εξοικείωση με την χρήση αρχείων στον δίσκο. Η πληροφορία οργανώνεται σε σελίδες δίσκου. Ανάγνωση και εγγραφή στο δίσκο γίνεται μόνο με ολόκληρες σελίδες δίσκου συγκεκριμένου μεγέθους.

Γενικά σχόλια

Η εφαρμογή διαχειρίζεται δεδομένα μιας κλάσης DataClass. Η κλάση περιέχει έναν μοναδικό θετικό ακέραιο αριθμό (κλειδί) και κάποια επιπλέον πληροφορία (δεδομένα). Το κλειδί χρησιμοποιείται για να αναζήτηση μιας εγγραφής στο αρχείο. Τα δεδομένα δεν χρησιμοποιούνται για αναζήτηση πληροφορίας στο αρχείο. Τα δεδομένα είναι μία συμβολοσειρά με ASCII χαρακτήρες.

Θα δημιουργήσετε τυχαία στιγμιότυπα της κλάσης και θα τα αποθηκεύσετε στο δίσκο με διαδοχική εγγραφή σελίδας δίσκου. Θα συγκρίνουμε διαφορετικές μεθόδους αναζήτησης πληροφορίας με βάση το κλειδί. Η απόδοση μιας αναζήτησης αντιστοιχεί στον αριθμό των προσβάσεων δίσκου που χρειάστηκαν μέχρι να βρεθεί η εγγραφή με το κλειδί που δίνεται. Για να μετρήσετε την απόδοση μιας μεθόδου δεν αρκεί να κάνετε μόνο μία αναζήτηση με ένα κλειδί. Θα πρέπει να βρείτε την μέση τιμή της απόδοσης πάνω σε ένα μεγάλο αριθμό αναζητήσεων για διαφορετικά κλειδιά.

Παράδειγμα εγγραφής και αρχείου

Κάθε κλειδί είναι ένας ακέραιος 4 bytes. Η πληροφορία (δεδομένα) που συνοδεύουν κάθε κλειδί είναι μια συμβολοσειρά με 55 λατινικούς ASCII χαρακτήρες. Συνολικά, κάθε εγγραφή (record) έχει μέγεθος 59 bytes. Ο αριθμός των εγγραφών που χωράνε σε μία σελίδα δίσκου εξαρτάται από το μέγεθος της σελίδας.

Στο παρακάτω σχήμα φαίνεται η μορφή μίας σελίδας (του αρχείου δεδομένων) στο δίσκο με ενδεικτικό μέγεθος σελίδας 180 bytes. Το αρχείο στο δίσκο αποτελείται από διαδοχικές τέτοιες σελίδες. Αν οι εγγραφές έχουν μέγεθος 59 bytes κάθε μία, χωράνε έως 3 εγγραφές σε κάθε σελίδα, ενώ περισσεύουν 180 - (59 x 3) = 3 bytes. Ενδεικτικά, αυτά τα 3 bytes μπορούν να χρησιμοποιηθούν για να αποθηκεύσουν τον αριθμό στιγμιοτύπτων (records) στην σελίδα (θα είναι ίδιος αριθμός για όλες τις σελίδες εκτός ίσως από την τελευταία).

key1	data1	key 2	data2	key 3	data3	key 4	data4	
record 1 - 59byte		record 2 - 59byte		record 3 - 59byte		record 4 - 59byte		
Data Page - 180byte								

Πρόσβαση στο αρχείο στο δίσκο γίνεται μόνο ανά σελίδα δίσκου. Οι σελίδες δίσκου διαβάζονται ή γράφονται στο αρχείο σε δυαδική μορφή (όχι σε μορφή text) και μετατρέπονται στην κεντρική μνήμη σε δεδομένα του τύπου της εφαρμογής (μπορεί να είναι είτε text, ή αριθμοί ή οποιοσδήποτε άλλος τύπος).

Αν θέλουμε να διαβάσουμε όλες τις εγγραφές του αρχείου, και έχουμε σελίδες δίσκου μεγέθους X (εδώ 180 bytes)

- Διαβάζουμε την πρώτη σελίδα του αρχείου ένα πεδίο (array) μεγέθους X bytes στη κεντρική μνήμη. Αυτή η πράξη μετράει ως μία πρόσβαση στον δίσκο. Όλες οι άλλες πράξεις γίνονται στην κεντρική μνήμη και δεν επιβαρύνουν την απόδοση.
- Στην κεντρική μνήμη, μετατρέπουμε τα X bytes σε πεδίο με στιγμιότυπα της κλάσης μας.
- Επαναλαμβάνουμε την ίδια διαδικασία διαβάζοντας κάθε φορά την επόμενη σελίδα δίσκου (με τα επόμενα X bytestου αρχείου) μέχρι να εξαντληθεί το αρχείο.

Αν θέλουμε να γράψουμε συνολικά Y στιγμιότυπα στο δίσκο. Έστω Z το πλήθος των στιγμιοτύπτων που χωράνε σε κάθε σελίδα δίσκου.

- Χρησιμοποιούμε ένα πεδίο X bytes στη μνήμη (με μέγεθος όσο η σελίδα δίσκου).
- Μετατρέπουμε κάθε μία τις Z εγγραφές σε array από bytes, και τις προσθέτουμε στο πεδίο των X bytes.
- Γράφουμε τα X bytes σε μία σελίδα δίσκου στο τέλος του αρχείου. Αυτή η πράξη μετράει ως μία πρόσβαση στον δίσκο. Όλες οι άλλες πράξεις γίνονται στην κεντρική μνήμη και δεν επιβαρύνουν την απόδοση.
- Επαναλαμβάνουμε με τα επόμενα Z στιγμιότυπα, μέχρι να έχουμε γράψει και τα Y στιγμιότυπα

Α τρόπος οργάνωσης: Οργάνωση πληροφορίας σε σειριακό αρχείο με τυχαία σειρά (2 μονάδες)

Το αρχείο περιέχει εγγραφές με τυχαία σειρά κλειδιών (δεν είναι ταξινομημένο). Τα κλειδιά είναι μοναδικά. Ένα αρχείο μπορεί να περιέχει έναν πολύ μεγάλο αριθμό από τέτοιες εγγραφές που αποθηκεύονται σε διαδοχικές σελίδες δίσκου (η μία μετά την άλλη).

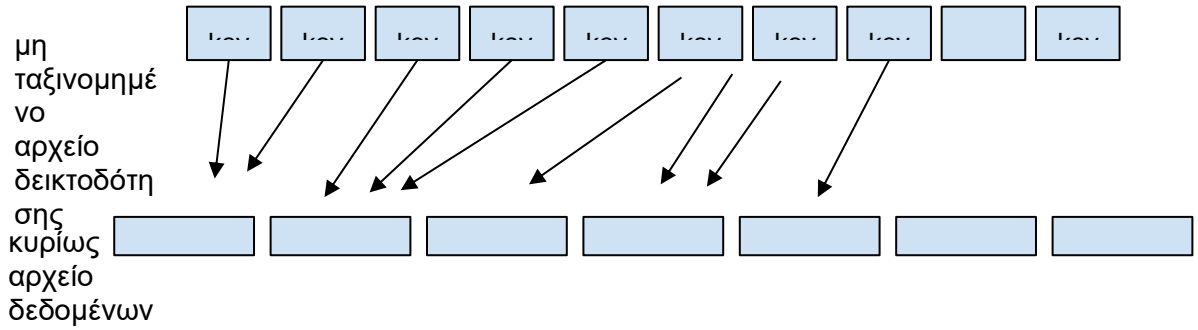
Για την αναζήτηση εγγραφής που έχει κάποιο κλειδί, πρέπει να διαβάσουμε σειριακά το αρχείο, από την αρχή και ανά σελίδα δίσκου (μία σελίδα κάθε φορά) και την φέρνουμε στην

κεντρική μνήμη. Με κάθε ανάγνωση σελίδας, μετατρέπουμε τα bytes του αρχείου (σε δυαδική μορφή) σε array με στιγμιότυπα. Στην κεντρική μνήμη ψάχνουμε σειριακά στα στιγμιότυπα να βρούμε αν υπάρχει το κλειδί που αναζητάμε. Συνεχίζουμε μέχρι να βρούμε το στιγμιότυπο με το κλειδί αναζήτησης ή μέχρι να εξαντλήσουμε το αρχείο (αν δεν το βρούμε).

Β τρόπος οργάνωσης: Οργάνωση πληροφορίας σε σειριακό αρχείο με τυχαία σειρά, με χρήση αρχείου δεικτοδότησης τυχαίας σειράς (2 μονάδες)

Ένας άλλος τρόπος οργάνωσης της ίδιας πληροφορίας στον δίσκο είναι με την χρήση 2 αρχείων. Το πρώτο αρχείο είναι το ίδιο όπως και ο Α τρόπος οργάνωσης, και αποτελεί το κύριο αρχείο δεδομένων.

Το 2ο αρχείο είναι αρχείο δεικτοδότησης (index). Περιέχει μόνο ζεύγη αριθμών, όπου ο πρώτος αριθμός αντιστοιχεί στο κλειδί της κάθε εγγραφής, και ο δεύτερος αριθμός στο αριθμό σελίδας που περιέχει την εγγραφή στο 1ο αρχείο.

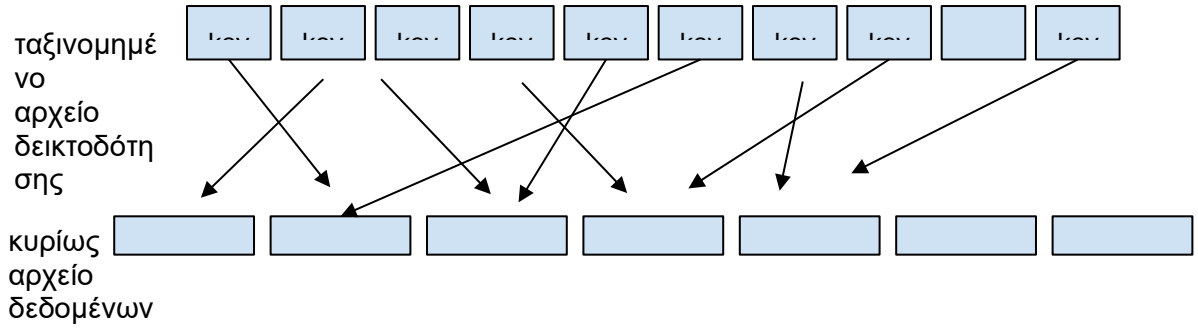


Όπως και στο κύριο αρχείο δεδομένων, τα κλειδιά στο αρχείο δεικτοδότησης δεν είναι ταξινομημένα. Η πρόσβαση στο αρχείο δεικτοδότησης γίνεται πάλι ανά σελίδα. Μιας και κάθε εγγραφή αποτελείται από μόνο 2 ακέραιους αριθμούς, χρειαζόμαστε μόνο 8 bytes ανά εγγραφή, οπότε σε κάθε σελίδα χωράνε πολύ περισσότερες εγγραφές σε σχέση με το κύριο αρχείο δεδομένων.

Για την αναζήτηση εγγραφής με κλειδί, χρησιμοποιούμε το αρχείο δεικτοδότησης. Διαβάζουμε σειριακά το αρχείο δεικτοδότησης (από την αρχή του αρχείου), ανά σελίδα δίσκου. Με κάθε ανάγνωση σελίδας, μετατρέπουμε τα bytes σε array ζευγών (κλειδί, θέση), και ψάχνουμε σειριακά στα ζεύγη να βρούμε αν υπάρχει το κλειδί που αναζητάμε. Εφόσον το βρούμε, γνωρίζουμε σε ποια σελίδα του κυρίως αρχείου βρίσκονται τα δεδομένα. Οπότε διαβάζουμεν μόνο τη συγκεκριμένη σελίδα του αρχείου δεδομένων, μετατρέπουμε τη σελίδα σε array από στιγμιότυπα, και ψάχνουμε το στιγμιότυπο με το κλειδί που αναζητάμε. Η απόδοση μίας αναζήτησης ισούται με τον αριθμό προσβάσεων στον αρχείο δεικτοδότησης συν μια πρόσβαση στο αρχείο δεδομένων.

Γ τρόπος οργάνωσης: Οργάνωση πληροφορίας σε σειριακό αρχείο με τυχαία σειρά, με χρήση ταξινομημένου αρχείου δεικτοδότησης (2 μονάδες)

Όπως και στον Β τρόπο οργάνωσης, έχουμε και ένα αρχείο δεικτοδότησης. Η διαφορά από το προηγούμενο ερώτημα είναι ότι το αρχείο δεικτοδότησης είναι ταξινομημένο ως προς το κλειδί. Οπότε τα κλειδιά εμφανίζονται κατά αύξουσα σειρά. Μόνο το αρχείο δεικτοδότησης είναι ταξινομημένο. Το κυρίως αρχείο είναι το ίδιο όπως στο Β και Α τρόπο οργάνωσης.



Εκμεταλλευόμαστε την ταξινόμηση του αρχείου δεικτοδότησης για να εφαρμόσουμε δυαδική αναζήτηση για την αναζήτηση κλειδιού.

Δυαδική αναζήτηση στο αρχείο

Για την δυαδική αναζήτηση, ανοίγετε το αρχείο και υπολογίζετε τον αριθμό σελίδων του αρχείου των κλειδιών. Για κάθε αναζήτηση, εκμεταλλευτείτε ότι το αρχείο των κλειδιών είναι ταξινομημένο: διαβάστε αρχικά την μεσαία σελίδα του αρχείου στην κεντρική μνήμη. Εξετάστε αν το κλειδί είναι στην σελίδα (μπορεί να γίνει σειριακά η αναζήτηση στη μνήμη). Αν υπάρχει, η αναζήτηση σταματάει. Αν το κλειδί που ψάχνετε είναι μικρότερο από τον μικρότερο αριθμό της σελίδας, τότε φέρτε στην κεντρική μνήμη την μεσαία σελίδα του αριστερού μισού του αρχείου. Αν το κλειδί είναι μεγαλύτερο από τον μεγαλύτερο αριθμό της σελίδας, φέρτε στην κεντρική μνήμη την μεσαία σελίδα του δεξιού μισού του αρχείου. Συνεχίστε με τον ίδιο τρόπο την αναζήτηση.

Εφόσον κατά τη δυαδική αναζήτηση στο αρχείο δεικτοδότησης βρούμε το κλειδί που ψάχνουμε, γνωρίζουμε σε ποια σελίδα του κυρίως αρχείου βρίσκονται τα δεδομένα. Οπότε διαβάζουμε τη συγκεκριμένη μόνο σελίδα του αρχείου δεδομένων, μετατρέπουμε τη σελίδα σε array από στιγμιότυπα, και ψάχνουμε το στιγμιότυπο με το κλειδί που αναζητάμε.

Υλοποίηση και πειραματικές μετρήσεις:

Υλοποιήστε τα παρακάτω με μία μόνο main( ) μέθοδο, η οποία εκτελεί ολόκληρο το πρόγραμμα χωρίς να ζητάει κάτι από το χρήστη (όχι μενού, όχι ερωτήσεις !!).

Για τις μετρήσεις που ζητούνται παρακάτω, χρησιμοποιήστε τα παρακάτω στοιχεία για το πλήθος στιγμιοτύπων και μέγεθος σελίδας 256 bytes.

Πλήθος στιγμιότυπων (records) N: 50, 100, 200, 500, 800, 1.000, 2.000, 5.000, 10.000, 50.000, 100.000, 200.000

Μέγεθος στιγμιότυπων (records):
α) 59 bytes (δεδομένα 55 bytes και κλειδί 4 bytes)
β) 31 bytes (δεδομένα 27 bytes και κλειδί 4 bytes)

Για κάθε πλήθος N και μέγεθος στιγμιότυπων επαναλάβετε τα παρακάτω βήματα:

Δημιουργήστε N τυχαία μοναδικά κλειδιά στη μνήμη με τιμές κλειδιών από 1 έως 2xN. Για κάθε κλειδί δημιουργήστε μία τυχαία συμβολοσειρά ASCII χαρακτήρων για το πεδίο δεδομένων (55 ή 27 bytes για κάθε περίπτωση). Για κάθε κλειδί και συμβολοσειρά, δημιουργήστε ένα στιγμιότυπο της κλάσης DataClass. Οπότε έχετε ένα array με N στιγμιότυπα.

1. Δημιουργήστε το κύριο αρχείο των τρόπων Α, Β και Γ στο δίσκο από τα N στιγμιότυπα. Το αρχείο θα δημιουργηθεί κάνοντας διαδοχικές εγγραφές στο δίσκο σελίδων δίσκου.
2. Δημιουργήστε στη μνήμη τα ζεύγη για το αρχείο δεικτοδότησης του τρόπου Β. Γράψτε το αρχείο δεικτοδότησης στο δίσκο. Χρησιμοποιήστε διαδοχικές εγγραφές σελίδων δίσκου
3. Ταξινομήστε τα ζεύγη στη μνήμη για να αποκτήσετε στη μνήμη τα δεδομένα για το αρχείο δεικτοδότησης του τρόπου Γ. Γράψτε το αρχείο δεικτοδότησης στο δίσκο. Χρησιμοποιήστε διαδοχικές εγγραφές σελίδων δίσκου
4. Για κάθε τρόπο οργάνωσης (Α, Β, Γ), κάντε 1000 αναζητήσεις με τυχαία κλειδιά από 1 έως 2xN. Για κάθε σύνολο 1000 αναζητήσεων, μετρήστε
   a. το μέσο αριθμό προσβάσεων στο δίσκο ανα αναζήτηση (πόσες φορές χρειάζεται να κάνετε ανάγνωση σελίδας δίσκου για να αποκτήσετε τα δεδομένα που αντιστοιχούν στο κλειδί που αναζητάτε, ή να διαπιστώσετε ότι δεν υπάρχει το κλειδί που αναζητάτε). Χρησιμοποιήστε 1 δεκαδικό ψηφίο
   b. το μέσο χρόνο αναζήτησης ανα αναζήτηση (πόσος χρόνος περνάει για να αποκτήσετε τα δεδομένα που αντιστοιχούν στο κλειδί που αναζητάτε, ή να διαπιστώσετε ότι δεν υπάρχει το κλειδί που αναζητάτε). Μετρήστε σε nanosecond

Το αποτέλεσμα είναι συμπληρωμένοι οι 2 παρακάτω πίνακες (**2 μονάδες)**

	Τρόπος Α		Τρόπος Β		Τρόπος Γ	
N	Αριθμός προσβάσεων ανά αναζήτηση	Αριθμός προσβάσεων ανά αναζήτηση	Αριθμός προσβάσεων ανά αναζήτηση	Αριθμός προσβάσεων ανά αναζήτηση	Αριθμός προσβάσεων ανά αναζήτηση	Αριθμός προσβάσεων ανά αναζήτηση

	. Μεγεθος δεδομένων 55 bytes	. Μεγεθος δεδομένων 27 bytes	. Μεγεθος δεδομένων 55 bytes	. Μεγεθος δεδομένων 27 bytes	. Μεγεθος δεδομένων 55 bytes	. Μεγεθος δεδομένων 27 bytes
50						
100						
200						
500						
800						
1000						
2000						
5000						
10000						
50000						
100000						
200000						

	Τρόπος Α		Τρόπος Β		Τρόπος Γ	
Ν	Χρόνος ανά αναζήτηση . Μεγεθος δεδομένων 55 bytes	Χρόνος ανά αναζήτηση . Μεγεθος δεδομένων 27 bytes	Χρόνος ανά αναζήτηση . Μεγεθος δεδομένων 55 bytes	Χρόνος ανά αναζήτηση . Μεγεθος δεδομένων 27 bytes	Χρόνος ανά αναζήτηση . Μεγεθος δεδομένων 55 bytes	Χρόνος ανά αναζήτηση . Μεγεθος δεδομένων 27 bytes
50						
100						
200						
500						
800						
1000						

2000						
5000						
10000						
50000						
100000						
200000						

Από τα παραπάνω δεδομένα, δημιουργήστε διαγράμματα όπου στο X άξονα είναι το πλήθος στιγμιοτύπτων, και στον Y άξονα ο μέσος αριθμός προσβάσεων και ο μέσος χρόνος εκτέλεσης για μία αναζήτηση.

Προσοχή στις κλίμακες των διαγραμμάτων όσον αφορά το X άξονα! Μη χρησιμοποιήσετε απλά διαγράμματα (linechart / barchart) χωρίς να λάβετε υπόψη τις αποστάσεις μεταξύ των σημείων στο X άξονα. Στο excel λέγονται “Scatter Chart”. Στο LibreOffice “XY Scatter”.

Σχολιάστε (**2 μονάδες)**

- Τη μορφή των καμπυλών (αν ειναι γραμμική, πολυωνυμική, εκθετική κλπ και γιατί)
- Πως μεταβάλλεται ο αριθμός προσβάσεων στο δίσκο και ο χρόνος εκτέλεσης σε σχέση με το πλήθος των εγγραφών στο δίσκο
- Τη σχέση του αριθμού προσβάσεων στο δίσκο με το χρόνο εκτέλεσης
- Πως συγκρίνονται οι τρόποι Α, Β, Γ μεταξύ τους

Για την υλοποίηση των παραπάνω μετρήσεων, θα χρειαστεί μεταξύ άλλων να υλοποιήσετε

- Την κλάση DataClass και
- Μεθόδους μετατροπής DataClass instance σε array από bytes
- Μεθόδους μετατροπής array από bytes σε DataClass instance
- Την κλάση αναπαράστασης ζεύγους κλειδί-θέση DataPage
- Μεθόδους μετατροπής ζεύγους σε array από bytes
- Μεθόδους μετατροπής array από bytes σε ζεύγος instance
- Κλάση αναπαράστασης DataPage
- Ανάγνωση DataPage από συγκεκριμένη θέση σε αρχείο στο δίσκο και μετατροπή σε array από DataClass instances ή instances ζεύγους
- Εγγραφή DataPage σε συγκεκριμένη θέση σε αρχείο στο δίσκο (θα χρειαστείτε μόνο προσθήκη DataPage και όχι εγγραφή σε ήδη κατειλημμένη θέση)

Παραδοτέα

Ένα συμπιεσμένο zip αρχείο που περιέχει ό,τι ζητείται παρακάτω (όχι zip αρχείο μέσα σε zip αρχείο!):

- τον πηγαίο κώδικα
- μία αναφορά με τα διαγράμματα και τα σχόλια που ζητούνται παραπάνω, και
    - πολύ σύντομη περιγραφή του κώδικα (π.χ. απαρίθμηση κλάσεων και μεθόδων και περιγραφή 1-2 γραμμές για κάθε μία)
    - εφόσον απαιτούνται ειδικές οδηγίες για την μεταγλώττιση/εκτέλεση, τις ειδικές οδηγίες
    - τι λάθη έχει (αν έχει περιπτώσεις που δεν δουλεύει το πρόγραμμα) ή περιπτώσεις που κάνει περισσότερα από όσα σας ζητεί η άσκηση
    - πλήρης και λεπτομερείς αναφορές εξωτερικών πηγών (π.χ. αν χρησιμοποιήσατε κώδικα που βρήκατε στο διαδίκτυο, ακριβής διεύθυνση σχετικής σελίδας και σημείο στη σελίδα)

Οι ασκήσεις υποβάλλονται ηλεκτρονικά στον ιστοχώρο του μαθήματος και όχι με e-mail.

Οι αντιγραφές (ακόμη και μέρους της υλοποίησης) μηδενίζονται.

Χρήσιμα

Ταξινόμηση δυάδων

Για την ταξινόμηση των δυάδων <λέξη,γραμμή>, μπορείτε να κάνετε χρήση του Interface Comparable. Δημιουργώντας μία κλάση για τις δυάδες σας, η οποία υλοποιεί το Comparable interface, μπορείτε να χρησιμοποιήσετε τη static μέθοδο Arrays.sort(Object[] a) για ταξινόμηση των δυάδων.

Παραγωγή τυχαίων string

Μπορείτε να παράγετε τυχαία strings συγκεκριμένου μεγέθους για τα δεδομένα σας. Ενδεικτική υλοποίηση μπορείτε να βρείτε στη διεύθυνση https://www.geeksforgeeks.org/generate-random-string-of-given-size-in-java/

Εγγραφή/Ανάγνωση από δίσκο ανά σελίδες

Θα πρέπει να διαβάζετε και να γράφετε στο δίσκο ανά σελίδες. Αυτό σημαίνει ότι για να γράψετε κάτι, θα πρέπει πρώτα στη μνήμη να δημιουργήσετε από τα δεδομένα σας ένα byte array με το μέγεθος της σελίδας, και να γράψετε μετά αυτό το byte array στο δίσκο. Αντίστοιχα, θα πρέπει να διαβάζετε ένα byte array από το δίσκο, και να το μετατρέψετε μετά στα δεδομένα σας.

Για τη δημιουργία του byte array, μπορείτε να χρησιμοποιήσετε την κλάση
java.nio.ByteBuffer.

```
int someInt = 10;
String someString = "test";
java.nio.ByteBuffer bb = java.nio.ByteBuffer.allocate(10);
```

```
bb.putInt(someInt);
bb.put(someString.getBytes(java.nio.charset.StandardCharsets.US_ASCII));
byte byteArray[] = bb.array();
```

Έχοντας ένα byte Array, η αντίστροφη διαδικασία (μετατροπή κάποιων bytes σε δεδομένα), γίνεται με τον ανάποδο τρόπο

```
byte[] buffer; // read from disk
ByteBuffer bb = ByteBuffer.wrap(buffer);
int someInt = bb.getInt();
byte byteArray[] = new byte[20];
bb.get(byteArray, 10, 20); // fills byteArray with 20 bytes from ByteBuffer bb, starting from offset 10
String someString = new String(byteArray, java.nio.charset.StandardCharsets.US_ASCII);
```

Η καθαυτή εγγραφή ενός Byte array στο δίσκο, ή η ανάγνωση, μπορεί να γίνει μεταξύ άλλων με τη μέθοδο RandomAccessFile.read() και RandomAccessFile.write()

Παραγωγή τυχαίων μοναδικών ακεραίων

Για την παραγωγή numberOfNumbers ακεραίων μεταξύ minIntNumber και maxIntNumber μπορείτε να χρησιμοποιήσετε τον παρακάτω κώδικα.

```
Random randomGenerator = new Random();
int[] randomInts = randomGenerator.ints(minIntNumber,
maxIntNumber+1).distinct().limit(numberOfNumbers).toArray();
```

Για τη μέτρηση του χρόνου

Τρέχουσα ώρα σε nanosecond.

```
System.nanoTime()
```