ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΑΡΧΕΙΩΝ

1η άσκηση

Ημερομηνία παράδοσης: 24 Μαρτίου 2022

Η άσκηση έχει σκοπό την εξοικείωση με δομές δεδομένων και αρχείων που χειρίζονται δεδομένα δύο διαστάσεων της μορφής **(x,y)**. Οι εφαρμογές είναι πολλές με πιο συνηθισμένη την παράσταση γεωγραφικών πληροφοριών (πόλεις στον χάρτη). Τα δεδομένα είναι σημεία σε μια περιοχή διαστάσεων **NxN** (δηλαδή 0 <= x <= N-1 και 0 <= y <= N-1). **Προσέξτε να μην επαναλαμβάνονται τα ίδια ζεύγη (x,y)**. Για κάθε δομή δεδομένων θα κατασκευάσετε τις μεθόδους εισαγωγής και αναζήτησης στοιχείων για την κεντρική μνήμη και τον δίσκο.

Μέρος Α: Επεξεργασία στην Κεντρική Μνήμη

Α.1 Συνδεδεμένες λίστες (1 μονάδα)

Τα δεδομένα μιας συνδεδεμένης λίστας (linked list) είναι συντεταγμένες (x,y). Κατασκευάστε την κλάση List με τις πράξεις:

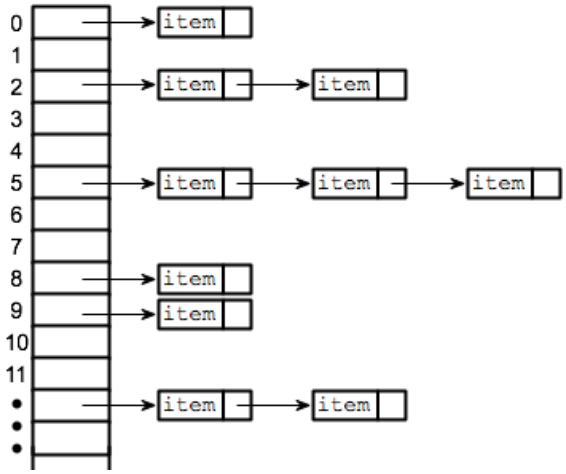
- Εισαγωγή θέσης (x,y) στο τέλος της λίστας. Θα χρειαστείτε δείκτη στη αρχή και στο τέλος της λίστας (ώστε να μην χρειάζεται διάσχιση της λίστας για κάθε εισαγωγή).
- Ερώτηση με όρισμα μία θέση (x, y). Επιστρέφει (α) αν η αναζήτηση είναι επιτυχημένη ή όχι και (β) τον αριθμό συγκρίσεων με (x,y) που έγιναν κατά την διάρκεια της αναζήτησης.

Α.2 Μέθοδος Κατακερματισμού (1 μονάδα)

Τα δεδομένα εισάγονται στην δομή δεδομένων που αποτελείται (α) από ένα πίνακα μεγέθους **M** και (β) συνδεδεμένες λίστες στοιχείων (x,y) σε κάθε θέση του πίνακα. Κάθε συντεταγμένη (x,y), εισάγεται ως τελευταίο στοιχείο της λίστας με αρχή την θέση H(x,y) που υπολογίζεται ως **H(x,y) = (x*N + y)%M**. Κάθε θέση του πίνακα περιέχει (εκτός από την αρχή) και δείκτη στο τελευταίο στοιχείο της λίστας. Κατασκευάστε την κλάση Hash με τις πράξεις

- Εισαγωγή (x,y). Υπολογίστε πρώτα την θέση του στοιχείου στον πίνακα και κάντε μετάβαση την τελευταία θέση της λίστας για την εισαγωγή. Χρησιμοποιείστε το δείκτη προς της τελευταία θέση της λίστας (μην κάνετε διάσχιση).

- Ερώτηση με όρισμα μία θέση (x,y). Επιστρέφει (α) αν η αναζήτηση είναι επιτυχημένη ή όχι και (β) τον αριθμό συγκρίσεων με (x,y) που έγιναν κατά την διάρκεια της αναζήτησης.



Μέρος Β: Επεξεργασία στον δίσκο

Θα επαναλάβετε την υλοποίηση του μέρους Α στον δίσκο. Το μέγεθος της σελίδας δίσκου είναι **256 bytes**. Τα δεδομένα αποθηκεύονται σε δυαδική μορφή. Δηλαδή κάθε ζεύγος τιμών καταλαμβάνει χώρο 8bytes. Οι παρενθέσεις και τα κόμμα κάθε ζεύγους (x,y) δεν αποθηκεύονται.

Β.1 Συνδεδεμένες λίστες στον δίσκο (2 μονάδες)

Επαναλάβετε το ερώτημα Α.1 για την εισαγωγή στοιχείου (x,y). Όταν δοθεί ένα στοιχείο (x,y) πρέπει να διαβάσετε την τελευταία σελίδα του αρχείου. Η μετάβαση στην τελευταία σελίδα γίνεται με την εντολή seek. Αμέσως μετά διαβάζεται η τελευταία σελίδα σε ένα buffer και μεταφέρεται στην κεντρική μνήμη. Αυτό γίνεται με την εντολή read. Αν η σελίδα δεν είναι γεμάτη, προστίθεται το νέο στοιχείο και η σελίδα αντιγράφεται πίσω στο αρχείο με την εντολή write (θα αντιγράψει όλο το buffer στο αρχείο). Αν η σελίδα είναι γεμάτη, τότε θα δημιουργηθεί μια νέα σελίδα που θα προστεθεί (θα γραφτεί) στο τέλος του αρχείου.

Κατά την δημιουργία του αρχείου, χρησιμοποιήστε ένα buffer μεγέθους σελίδας δίσκου. Τα στοιχεία εισάγονται πρώτα σε ένα buffer στην κεντρική μνήμη. Κάθε φορά που γεμίζει ο buffer τον αντιγράφετε στον δίσκο σε δυαδική μορφή με την εντολή write. Οι σελίδες αποθηκεύονται διαδοχικά η μια μετά την προηγούμενη.

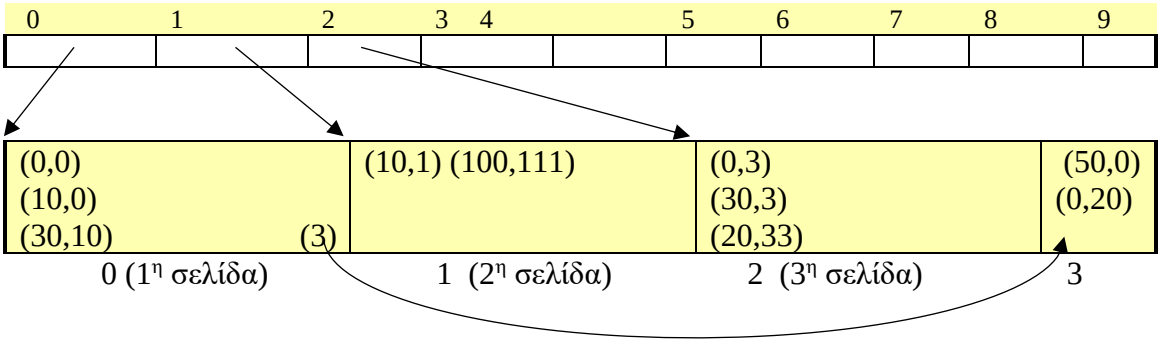
Επαναλάβετε το ερώτημα Α1. Για την αναζήτηση στοιχείων στον δίσκο, διαβάστε την μία σελίδα μετά την άλλη (σειριακά) στην κεντρική μνήμη. Η αναζήτηση θα γίνει με μια

επανάληψη while που θα διαβάζει τις σελίδες του αρχείου με την εντολή read (με όρισμα ένα buffer) μέχρι να βρεθεί το στοιχείο αναζήτησης. Αν δεν βρεθεί, τότε θα διαβαστούν όλες οι σελίδες του αρχείου. Κάθε κλήση της συνάρτησης read μετακινεί τον δείκτη του αρχείου στην επόμενη σελίδα.

Ο έλεγχος (ή σύγκριση) με το στοιχείο αναζήτησης γίνεται στην κεντρική μνήμη στα στοιχεία του buffer. Αν το στοιχείο δεν βρεθεί στα στοιχεία της σελίδας που φέρατε στην κεντρική μνήμη, τότε θα διαβάσετε την επόμενη σελίδα. Κάθε διάβασμα σελίδας είναι μια πρόσβαση στον δίσκο. Σε αυτή την περίπτωση αντί για μέσο αριθμό συγκρίσεων θα μετρήσετε μέσο αριθμό προσβάσεων στον δίσκο (disk accesses) που έγιναν μέχρι να βρεθεί το στοιχείο.

Β.2 Μέθοδος Κατακερματισμού στον δίσκο (4 μονάδες)

Ο πίνακας των Μ θέσεων εξακολουθεί να υπάρχει στην κεντρική μνήμη. Όλες οι λίστες του πίνακα αποθηκεύονται σε ένα αρχείο. Κάθε θέση του πίνακα έχει τιμή (α) την διεύθυνση της πρώτης σελίδας του αρχείου όπου αποθηκεύονται τα στοιχεία της λίστας (β) την τελευταία σελίδα της αλυσίδας για την συγκεκριμένη θέση. Στο παράδειγμα, ο πίνακας έχει μέγεθος Μ=10 και τα x, y παίρνουν τιμές στο διάστημα [0..N].



Αν μία σελίδα γεμίσει, θα χρησιμοποιήσει μια νέα σελίδα που θα αποθηκευτεί στο τέλος του αρχείου (σελίδα υπερχείλισης ή overflow page). Αν και αυτή η σελίδα γεμίσει θα χρησιμοποιηθεί ακόμα μια σελίδα υπερχείλισης που θα αποθηκευτεί στο τέλος του αρχείου. Κάθε σελίδα που υπερχειλίζει αποθηκεύει την διεύθυνση της επόμενης σελίδας υπερχείλισης. Δηλαδή οι σελίδες υπερχείλισης σχηματίζουν λίστα. Στο σχήμα, η σελίδα 0 έχει υπερχειλίσει και δείχνει στην σελίδα 3 που αποθηκεύει στοιχεία που δεν χωράνε στην σελίδα 0. Κάθε θέση του πίνακα θα μπορούσε να δείχνει σε μία λίστα με σελίδες υπερχείλισης.

Επαναλάβετε το ερώτημα Α.1 για τον πίνακα κατακερματισμού στον δίσκο. Για την εισαγωγή ενός νέου στοιχείου (x,y) πρέπει να διαβάσετε την τελευταία σελίδα της αλυσίδας. Την διεύθυνσή της θα την μάθετε από τον πίνακα. Η μετάβαση στην τελευταία σελίδα γίνεται με την εντολή seek. Αμέσως μετά διαβάζεται η τελευταία σελίδα σε ένα buffer και μεταφέρεται στην κεντρική μνήμη. Αυτό γίνεται με την εντολή read. Αν η σελίδα δεν είναι γεμάτη, προστίθεται το νέο στοιχείο και η σελίδα αντιγράφεται πίσω στο αρχείο με την εντολή write (θα αντιγράψει όλο το buffer στο αρχείο). Αν η σελίδα είναι

γεμάτη, τότε θα δημιουργηθεί μια νέα σελίδα που θα (α) προστεθεί στην αλυσίδα και (β) θα γραφτεί στο τέλος του αρχείου.

Επαναλάβετε το ερώτημα Α.2 για αναζήτηση στον πίνακα κατακερματισμού στον δίσκο. Για την αναζήτηση στοιχείων στον δίσκο, διαβάστε με read την πρώτη σελίδα της αλυσίδας. Την διεύθυνσή της θα την μάθετε από τον πίνακα. Η αναζήτηση για το στοιχείο (x,y) γίνεται διαβάζοντας τις σελίδες της αλυσίδας την μία μετά την άλλη (σειριακά) στην κεντρική μνήμη. Η αναζήτηση θα γίνει με μια επανάληψη while που θα διαβάζει τις σελίδες του αρχείου με την εντολή read (με όρισμα ένα buffer) μέχρι να βρεθεί το στοιχείο αναζήτησης. Αν δεν βρεθεί, τότε θα διαβαστούν όλες οι σελίδες του αρχείου. Κάθε κλήση της συνάρτησης read μετακινεί τον δείκτη του αρχείου στην επόμενη σελίδα.

Σκεφθείτε πως θα μπορούσατε να αποθηκεύεται σε ένα ξεχωριστό αρχείο στον δίσκο και ο πίνακα. Περιγράψτε πως θα γινόταν αυτό στην αναφορά της άσκησης. Αν το υλοποιήσετε και αυτό θα πάρετε επιπλέον 2 μονάδες.

Μέρος Γ: Σύγκριση μεθόδων και τεκμηρίωση (2 μονάδες)

Εισάγετε σε κάθε μια από τις παραπάνω δομές **Κ δεδομένα** της μορφής (**x,y**) όπου τα **x**, **y** είναι τυχαίοι ακέραιοι αριθμοί στο **διάστημα NxN** και για **$N=2^{16}$**. Κατασκευάστε κάθε δομή δεδομένων για **Κ = 1.000, 10.000, 30.000, 50.000, 70.000, 100.000**.

Μετρήστε μέσο αριθμό συγκρίσεων για 100 τυχαίες ερωτήσεις. Σε κάθε δομή δεδομένων και για διαφορετικό αριθμό δεδομένων **Κ**, κάντε **100** αναζητήσεις για δεδομένα (**x,y**) που ξέρετε εκ των προτέρων ότι υπάρχουν ήδη (πχ έχετε αποθηκεύσει 100 τυχαία κατά την διάρκεια της εισαγωγής).

Για κάθε αναζήτηση του μέρους Α υπολογίστε τον μέσο αριθμό συγκρίσεων με (x,y) που απαιτούνται. Για κάθε αναζήτηση του μέρους Β υπολογίστε τον μέσο αριθμό προσβάσεων στον δίσκο. Για κάθε **Κ** και να κάνετε τις αναζητήσεις που ζητούνται και να υπολογίζετε τον αριθμό συγκρίσεων ή τον αριθμό προσβάσεων στον δίσκο.

Για το Μέρος Α: Κατασκευάστε το διάγραμμα που δείχνει πώς μεταβάλλεται ο αριθμός συγκρίσεων που απαιτούνται για την εύρεση ενός στοιχείου ως συνάρτηση του **Κ** για κάθε μια από τις παραπάνω δύο περιπτώσεις αναζήτησης στις δύο δομές. Για την αναζήτηση να φτιάξετε δύο καμπύλες που αντιστοιχούν στις περιπτώσεις επιτυχημένης αναζήτησης (το στοιχείο υπάρχει) και αποτυχημένης αναζήτησης (το στοιχείο δεν βρέθηκε). Δηλαδή το διάγραμμα έχει δύο καμπύλες (που δείχνουν εξάρτηση από τον πλήθος των στοιχείων Κ) για το Α.1 και μια για τον Α.2.

Για το Μέρος Β: Κατασκευάστε το διάγραμμα που δείχνει πώς μεταβάλλεται ο αριθμός προσβάσεων στον δίσκο που απαιτούνται για την εύρεση ενός στοιχείου ως συνάρτηση του **Κ** για κάθε μια από τις παραπάνω δύο περιπτώσεις αναζήτησης στις δύο δομές. Για την αναζήτηση να φτιάξετε δύο καμπύλες που αντιστοιχούν στις περιπτώσεις επιτυχημένης αναζήτησης (το στοιχείο υπάρχει) και αποτυχημένης αναζήτησης (το

στοιχείο δεν βρέθηκε). Για το μέρος Β το διάγραμμα έχει δύο καμπύλες εξάρτησης για το Β.1 και μια για τον Β.2.

Προσπαθήστε να δικαιολογήστε την απόδοση κάθε μεθόδου. Εξηγήστε για ποιο λόγο μία μέθοδος είναι πιο αποδοτική από μία άλλη για αναζήτηση. Συνολικά δεν χρειάζεται να γράψετε πάνω από 1 ή 2 σελίδες αρκεί κάθε απάντηση να είναι σαφής και αιτιολογημένη σωστά. Δεν χρειάζεται να γράψετε περισσότερα σχόλια για την υλοποίηση του κώδικα εκτός από ότι ζητείται παρακάτω (δείτε την ενότητα «παραδοτέα»).

Παραδοτέα: Ένα συμπιεσμένο zip αρχείο που περιέχει ότι ζητείται παρακάτω:

- Ο κώδικας περιέχει συνοπτικά σχόλια που εξηγούν την υλοποίηση.
- Μία έκθεση που περιγράφει σε 1-2 σελίδες πως φτιάχτηκε ο κώδικας (δηλ. για κάθε ερώτημα ποια είναι η γενική ιδέα της λύσης σε 3-4 προτάσεις), υπάρχουν σαφείς οδηγίες μετάφρασης από compiler και εκτέλεσης, τι λάθη έχει (αν έχει, περιπτώσεις που δεν δουλεύει το πρόγραμμα, ή περιπτώσεις που κάνει περισσότερα από όσα σας ζητεί η άσκηση, τι χρησιμοποιήσατε από έτοιμα προγράμματα ή πηγές πληροφόρησης. Υποδείξετε ακόμα και πηγές στο WWW όπως Wikipedia (πλήρεις διευθύνσεις) ή ακόμα και συναδέλφους που σας βοήθησαν στην άσκηση.
- Στην ενότητα τεκμηρίωσης των αποτελεσμάτων πρέπει να υπάρχει απόλυτη σαφήνεια
- Εκτός των παραπάνω, οι ασκήσεις βαθμολογούνται με άριστα εφόσον:
  - Το zip είναι πλήρες
  - Οι κώδικες περνούν από compiler και εκτελούνται κανονικά και σωστά σε windows ή Linux περιβάλλον
  - Ο κώδικάς σας δουλεύει για οποιοδήποτε αρχείο δεδομένων που θα σας δοθεί ως είσοδος, και όχι μόνο το αρχείο που παράγετε εσείς

Βασικά στοιχεία θεωρίας αρχείων: Το αρχείο είναι οργανωμένο σε σελίδες δίσκου. Τα δεδομένα γράφονται σε σελίδες δίσκου που έχουν σταθερό μέγεθος (πχ 256 bytes). Η ανάγνωση σελίδων γίνεται με την εντολή read. Η εγγραφή σελίδων γίνεται με την εντολή write. Κάθε ανάγνωση (κλήση της read) διαβάζει πληροφορία μεγέθους σελίδας δίσκου. Κάθε εγγραφή (κλήση της write) γράφει στο αρχείο πληροφορία μεγέθους σελίδας δίσκου. Κάθε σελίδα στον αποκτά μια διεύθυνση που είναι η θέση της στο αρχείο (ο αριθμός σε bytes από την αρχή του αρχείου).

Για να κάνετε πρόσβαση σε μια σελίδα δίσκου, θα χρησιμοποιήσετε την εντολή seek με ορίσματα την θέση της σελίδας που θέλετε και τον buffer μεγέθους σελίδας δίσκου (256 bytes). Μετά μπορείτε να διαβάσετε την σελίδα με μια εντολή read. Η εντολή read θα φέρει την σελίδα στην κεντρική μνήμη για επεξεργασία. Η επεξεργασία των δεδομένων της σελίδας γίνεται πάνω στα στοιχεία του buffer. Κάθε εντολή read φέρνει στην κεντρική μνήμη ένα buffer και κοστίζει μια πρόσβαση στον δίσκο (disk access). Αντίστοιχα, η εγγραφή στο αρχείο γίνεται πάλι σε σελίδες δίσκου. Δηλαδή κάθε εγγραφή θα γράψει στο αρχείο μία σελίδα δίσκου μεγέθους ενός buffer με δεδομένα. Με την εντολή seek γίνεται μετάβαση στην θέση του αρχείου όπου θα γραφτεί μια σελίδα.

Η εντολή write γράφει τον buffer (ως μια σελίδα δίσκου) στο τέλος του αρχείου. Κάθε εντολή write κοστίζει μια πρόσβαση στον δίσκο (disk access). Κάθε αποθηκευμένη σελίδα

έχει διεύθυνση που ορίζεται ως το μέγεθος σε bytes από την αρχή του αρχείου. Κάθε κλήση των συναρτήσεων read ή write προϋποθέτει (α) ότι έχετε τοποθετήσει τον δείκτη του αρχείου στην θέση στην οποία θέλετε να διαβάσετε ή να γράψετε, (β) έχετε δώσει όρισμα τον δείκτη σε μεταβλητή buffer που έχει μέγεθος όσο ή σελίδα του δίσκου.

Για την εκτίμηση της απόδοσης μιας μεθόδου μετρήσετε πόσες φορές καλέσατε τις εντολές read και write αγνοώντας πόσες πράξεις έγιναν στην κεντρική μνήμη. Η εντολή seek δεν κοστίζει σε προσβάσεις δίσκου.