Εισαγωγική Διάλεξη στις Δομές Δεδομένων

Γιάννης Λιβιεράτος

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών
Τμήμα Μαθηματικών

2 Μαρτίου 2024

Υπολογιστικό Πρόβλημα

- Δημιουργία καταλόγου φοιτητών που παρακολουθούν τα μαθήματα "Μαθηματική Λογική" και "Δομές Δεδομένων" στην e-class.
- Θέλουμε αρχείο που να περιλαμβάνει ονοματεπώνυμο, αριθμό μητρώου και το ποιο από τα δύο μαθήματα παρακολουθεί.
- Υποθέτουμε ότι υπάρχουν λίγοι φοιτητές και λίγες φοιτήτριες που θα εμφανιστούν στην μέση του εξαμήνου, λίγοι και λίγες περισσότεροι που θα βαρεθούν και θα αφήσουν το μάθημα.
- Υποθέτουμε ότι θα χρειαστούν συχνές αναζητήσεις στο αρχείο αυτό για το αν ένας φοιτητής ή μία φοιτήτρια παρακολουθεί το εκάστοτε μάθημα (ο διδάσκων έχει δυσκολία στο να θυμάται ονόματα).

Στόχος

Χρήση κατάλληλης δομής ώστε να αποθηκεύσουμε τις πληροφορίες αυτές.

Στόχος

Χρήση κατάλληλης δομής ώστε να αποθηκεύσουμε τις πληροφορίες αυτές.

Με τι κριτήρια;

Στόχος

Χρήση κατάλληλης δομής ώστε να αποθηκεύσουμε τις πληροφορίες αυτές.

Με τι κριτήρια;

- Αποδοτικότητα
- Ταχύτητα
- Αποτελεσματικότητα
- Κατανάλωση πόρων

Υπολογιστής

1. Αποθήκευση των στοιχείων σε κάποιο αρχείο $\longrightarrow$ κατανάλωση μνήμης
2. Προσθήκη στοιχείων, διαγραφή στοιχείων, αναζήτηση στοιχείων $\longrightarrow$ αλγόριθμος
3. Σχεδιασμός αλγορίθμου $\longrightarrow$ αποδοτικότητα αλγορίθμου
4. Υλοποίηση αλγορίθμου $\longrightarrow$ πραγματικός χρόνος εκτέλεσης προγράμματος

Μνήμη

- Στατική: προκαθορισμένος χώρος που δεσμεύεται.
- Δυναμική: αύξηση/μείωση του δεσμευμένου χώρου όταν χρειάζεται.

Αλγόριθμος

★ Πεπερασμένη ακολουθία σαφώς ορισμένων βημάτων.

Παράδειγμα (Αναζήτηση στοιχείου σε λίστα)

1. *Διέτρεξε την λίστα, στοιχείο προς στοιχείο, ξεκινώντας από την αρχή.*
2. *Αν βρεις το ζητούμενο στοιχείο, τερμάτισε δηλώνοντας επιτυχία/επιστρέφοντας την θέση που το βρήκες.*
3. *Αν τελειώσει η λίστα, τερμάτισε δηλώνοντας αποτυχία.*

★ Φιξάρισμα ψευδογλώσσας, ώστε να είναι σαφές ποιες εντολές είναι υπολογιστικά εφικτές.

Αποδοτικότητα Αλγορίθμου

- Μνήμη $\longrightarrow$ μέγεθος λίστας
- Χρόνος $\longrightarrow$ τόσα βήματα όσα χρειάζονται για να φτάσουμε την θέση του ζητούμενου στοιχείου. Χειρότερη περίπτωση: μέγεθος λίστας.

Αποδοτικότητα Αλγορίθμου

- Μνήμη $\longrightarrow$ μέγεθος λίστας
- Χρόνος $\longrightarrow$ τόσα βήματα όσα χρειάζονται για να φτάσουμε την θέση του ζητούμενου στοιχείου. Χειρότερη περίπτωση: μέγεθος λίστας.

★ Σε κάθε περίπτωση, η αποδοτικότητα μετριέται ως προς το μέγεθος της εισόδου.

Αναπαράσταση εισόδου του αλγορίθμου $\longrightarrow$ κωδικοποίηση
Ρυθμός αύξησης χρόνου/χώρου $\longrightarrow$ Ασυμπτωτική Ανάλυση

Υλοποίηση Αλγορίθμου

- Επιλογή επιθυμητής γλώσσας προγραμματισμού (pyhton, C/C++, Java,...).
- Μέτρηση απόδοσης πειραματικά.

★ Η πραγματική απόδοση ενός προγράμματος, εξαρτάται από τεχνολογίες που αλλάζουν συνεχώς. Αυτό κάνει την μαθηματική ανάλυσή τους από πολύ δύσκολη έως αδύνατη.

Δομές Δεδομένων

Πως επιρρεάζουν κάθε ένα από τα παραπάνω;

- Κάθε δομή καταλαμβάνει διαφορετικό χώρο αποθήκευσης.
- Αναλόγως την δομή δεδομένων που θα χρησιμοποιηθεί, λειτουργίες που χρειαζόμαστε μπορεί να γίνονται πιο γρήγορα.

Παράδειγμα (Αναζήτηση στοιχείου σε ταξινομεμένη λίστα)

Δομές Δεδομένων

Πως επιρρεάζουν κάθε ένα από τα παραπάνω;

- Κάθε δομή καταλαμβάνει διαφορετικό χώρο αποθήκευσης.
- Αναλόγως την δομή δεδομένων που θα χρησιμοποιηθεί, λειτουργίες που χρειαζόμαστε μπορεί να γίνονται πιο γρήγορα.

Παράδειγμα (Αναζήτηση στοιχείου σε ταξινομεμένη λίστα)

1. *Έλεγξε το μεσαίο στοιχείο της λίστας.*
2. *Αν βρεις το ζητούμενο στοιχείο, τερμάτισε δηλώνοντας επιτυχία/επιστρέφοντας την θέση που το βρήκες.*
3. *Αν βρήκες μεγαλύτερο στοιχείο, επανέλαβε στο πρώτο μισό της λίστας.*
4. *Αν βρήκες μικρότερο στοιχείο, επανέλαβε στο δεύτερο μισό της λίστας.*
5. *Αν τελειώσει η λίστα, τερμάτισε δηλώνοντας αποτυχία.*

Δομές Δεδομένων, συνέχεια

- Ο παραπάνω αλγόριθμος χρειάζεται στην χειρότερη περίπτωση $\log_2(n)$ επαναλήψεις, με n το μέγεθος της λίστας.
  Πόσα βήματα θα χρειαζόμασταν για να προσθέσουμε στοιχείο σε (ι) μη ταξινομημένη λίστα και (ιι) σε ταξινομημένη λίστα;

Δομές Δεδομένων, συνέχεια

- Ο παραπάνω αλγόριθμος χρειάζεται στην χειρότερη περίπτωση $\log_2(n)$ επαναλήψεις, με n το μέγεθος της λίστας.
  Πόσα βήματα θα χρειαζόμασταν για να προσθέσουμε στοιχείο σε (ι) μη ταξινομημένη λίστα και (ιι) σε ταξινομημένη λίστα;

  1 και $\sim \log_2(n)$ αντίστοιχα.
- Η εκάστοτε γλώσσα προγραμματισμού υποστηρίζει διαφορετικές δομές δεδομένων και χειρίζεται την μνήμη με διαφορετικό τρόπο.