Κεφάλαιο 8:
Αφαίρεση Δεδομένων

Η Επιστήμη των Υπολογιστών:
Μια Ολοκληρωμένη Παρουσίαση
(δέκατη αμερικανική έκδοση)
J. Glenn Brookshear

Επικ. Καθ. Αυγερινός Αραμπατζής

avi@ee.duth.gr

www.aviarampatzis.com

Αφαίρεση Δεδομένων

Η κύρια μνήμη ενός υπολογιστή είναι οργανωμένη σειριακά (κελί-προς-κελί).

Θα δούμε: πως μπορούν να προσομοιωθούν άλλες διατάξεις (δομές δεδομένων) π.χ. πίνακες, λίστες, στοίβες, ουρές, δέντρα.

Στόχος: να μπορούν να προσπελάσουν οι χρήστες συλλογές δεδομένων ως **αφηρημένα εργαλεία**, δλδ να μη σκέφτονται με βάση την οργάνωση της κύριας μνήμης.

Κεφάλαιο 8: Αφαίρεση Δεδομένων

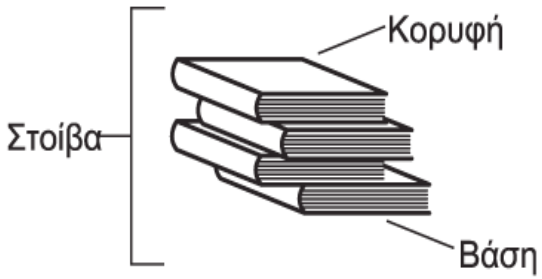
- 8.1 Βασικές δομές δεδομένων
- 8.2 Σχετικές έννοιες
- 8.3 Υλοποίηση δομών δεδομένων
- ~~8.4 Μια σύντομη μελέτη περίπτωσης~~
- 8.5 Προσαρμοσμένοι τύποι δεδομένων
- 8.6 Κλάσεις και αντικείμενα
- ~~8.7 Δείκτες σε γλώσσα μηχανής~~

Βασικές Δομές Δεδομένων

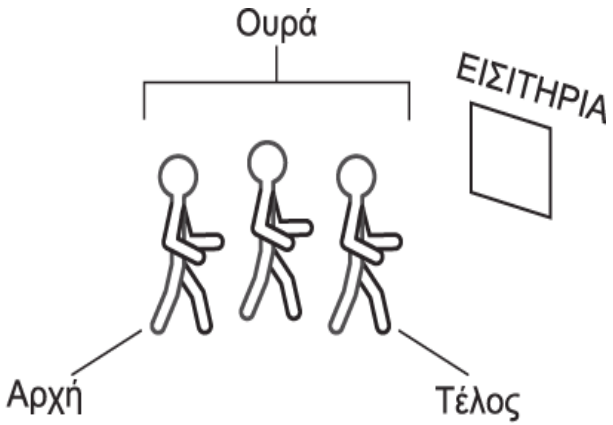
- Ομοιογενής πίνακας
  - πχ μηνιαίες πωλήσεις μιας ομάδας πωλητών
- Ετερογενής πίνακας
  - πχ τα στοιχεία ενός υπαλλήλου
- Λίστα (πχ αγορών, προσκεκλημένων)
  - Στοίβα (πχ βιβλίων)
  - Ουρά (πχ εισιτηρίων)
- Δέντρο
  - πχ οργανόγραμμα εταιρείας

Σχήμα 8.1 Λίστες, στοίβες, και ουρές

α. Λίστα ονομάτων



β. Στοίβα βιβλίων



γ. Ουρά ατόμων

Ορολογία για Λίστες

- **Λίστα:** Μια συλλογή δεδομένων των οποίων οι καταχωρίσεις είναι διατεταγμένες σειριακά
- **Κεφαλή:** Η αρχή της λίστας
- **Ουρά:** Το τέλος της λίστας

Ορολογία για Στοίβες

- **Στοίβα:** Μια λίστα στην οποία οι καταχωρίσεις αφαιρούνται και εισάγονται μόνο στην κορυφή
- **LIFO:** Τελευταίο μέσα, πρώτο έξω (ή FILA)
- **Κορυφή:** Η κεφαλή της λίστας
- **Βάση:** Η ουρά της λίστας
- **Απώθηση:** Αφαίρεση της καταχώρισης από την κορυφή (pop)
- **Ώθηση:** Εισαγωγή μιας καταχώρισης στην κορυφή (push)

Χρησιμότητα Στοίβας

Ιδανική για αποθήκευση στοιχείων που πρέπει να ανακτηθούν με την αντίστροφη σειρά αποθήκευσης.

Οπισθοδρόμηση (backtracking), πχ

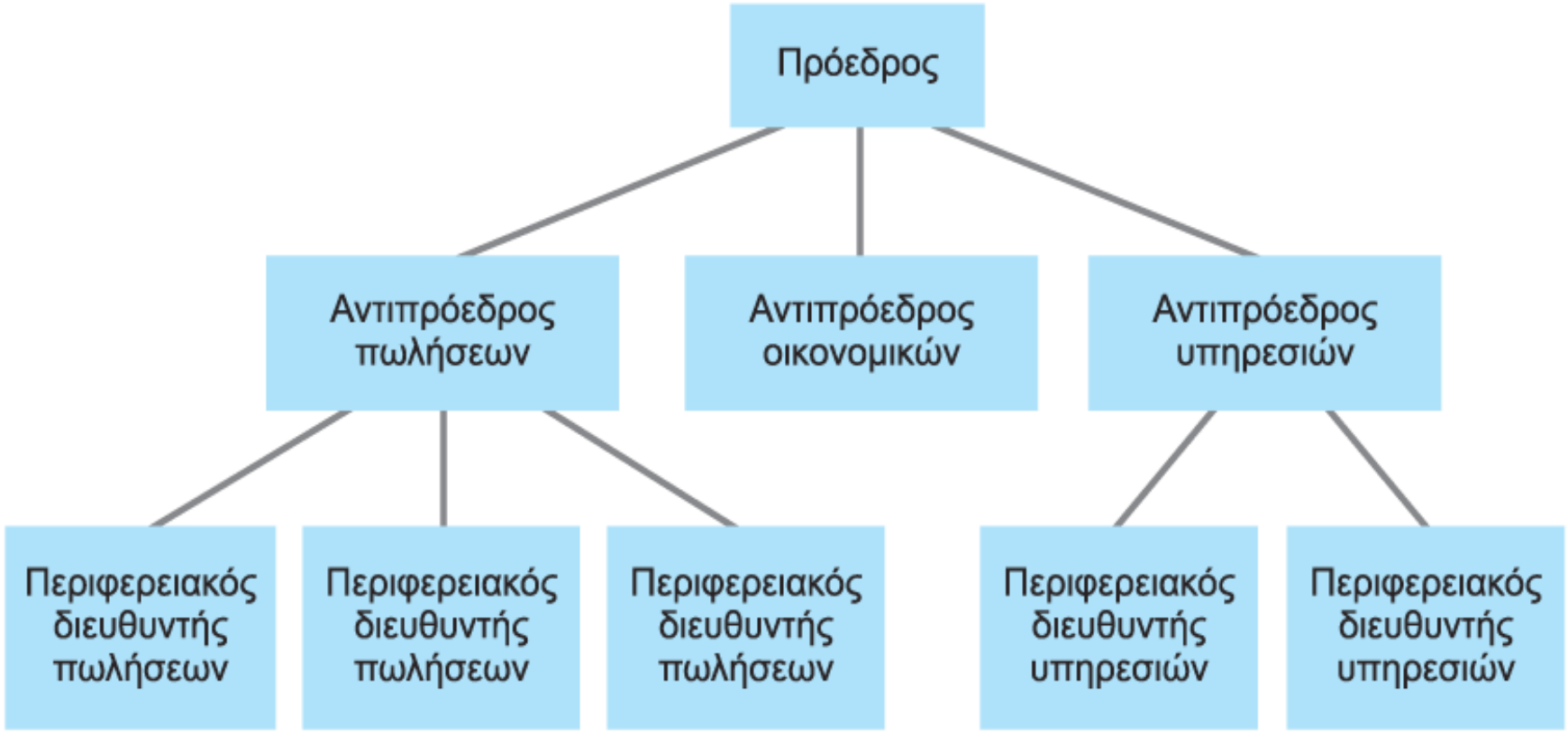
- ακολουθώ τα ίχνη μου για να βγω από το δάσος
- αναδρομικές διαδικασίες (recursion), πχ.

```
int factorial(int x) {
     if (x==1) return 1;
     else      return x*factorial(x-1);
  }
```

Ορολογία για Ουρές

- **Ουρά:** Μια λίστα στην οποία οι καταχωρίσεις αφαιρούνται από την κορυφή και εισάγονται στην ουρά
- **FIFO:** Πρώτο μέσα, πρώτο έξω
- Παράδειγμα χρήσης:
  - Ενταμιευτές (buffers) πχ
    - Εγγραφές στον δίσκο
    - Διαδικτυακά πακέτα

Σχήμα 8.2 **Παράδειγμα οργανογράμματος**



Ορολογία για Δέντρα

- **Δέντρο:** Μια συλλογή για δεδομένα των οποίων οι καταχωρίσεις έχουν ιεραρχική οργάνωση
- **Κόμβος:** Μια καταχώριση σε ένα δέντρο
- **Κόμβος ρίζας:** Ο κόμβος στην κορυφή
- **Τερματικός κόμβος** ή **κόμβος φύλλου:** Ένας κόμβος στη βάση

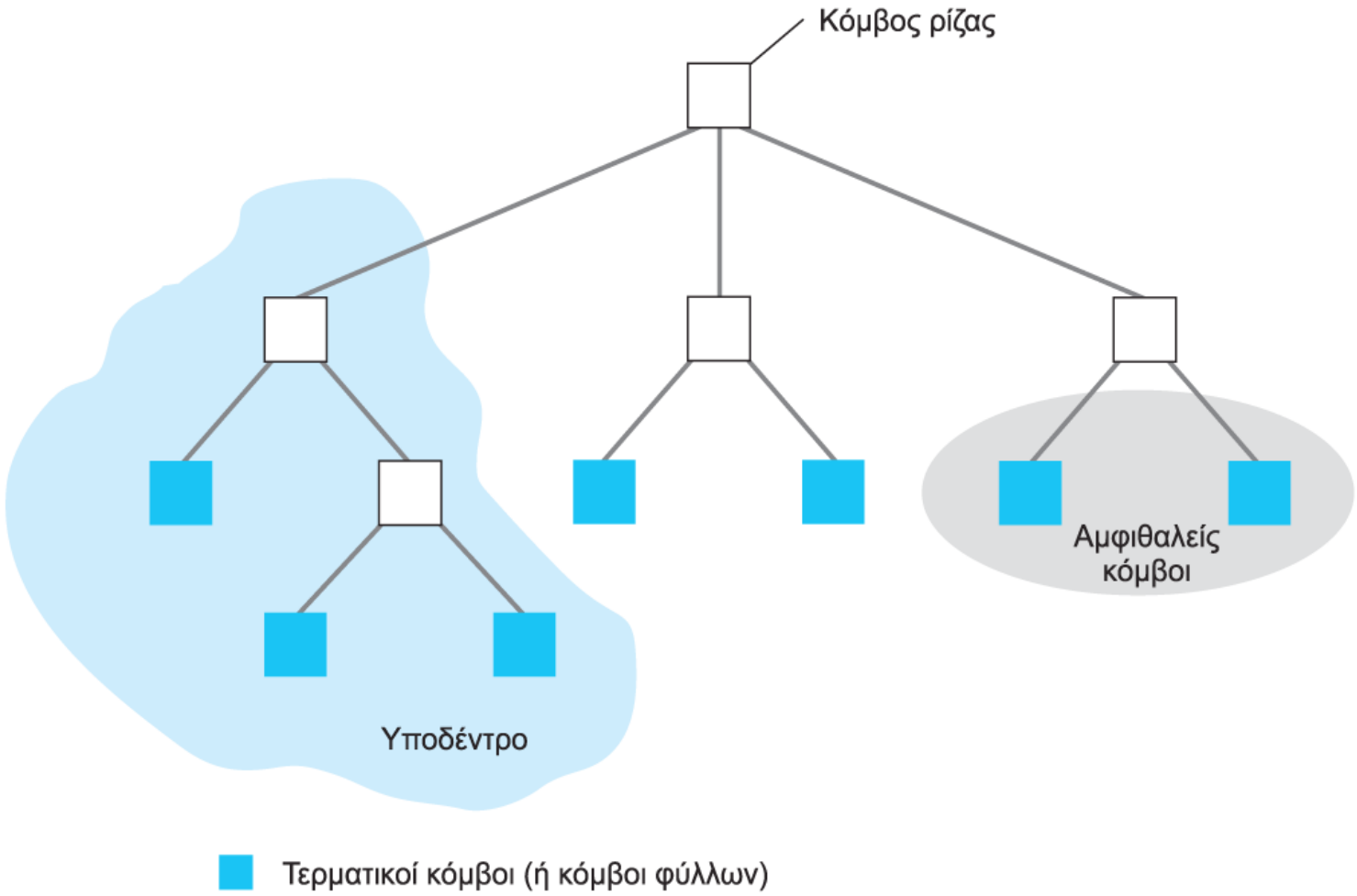
Ορολογία για Δέντρα (συνέχεια)

- **Γονικός κόμβος:** Ο κόμβος ακριβώς πάνω από έναν καθορισμένο κόμβο
  - Ο κάθε κόμβος δεν πρέπει να έχει πάνω από ένα γονικό κόμβο, αλλιώς έχουμε **γράφο** και όχι δέντρο.
- **Θυγατρικός κόμβος:** Ο κόμβος ακριβώς κάτω από έναν καθορισμένο κόμβο
- **Πρόγονος:** Γονικός, γονικός του γονικού, κ.λπ.
- **Απόγονος:** Θυγατρικός, θυγατρικός του θυγατρικού, κ.λπ.
- **Αμφιθαλείς κόμβοι:** Κόμβοι που έχουν έναν κοινό γονικό κόμβο

Ορολογία για Δέντρα (συνέχεια)

- **Δυαδικό δέντρο:** Ένα δέντρο στο οποίο ο κάθε κόμβος έχει το πολύ δύο θυγατρικούς κόμβους
- **Βάθος:** Το πλήθος των κόμβων στη μακρύτερη διαδρομή από τη ρίζα μέχρι κάποιο φύλλο

Σχήμα 8.3 Ορολογία δέντρων



Πρόσθετες Έννοιες

- **Στατικές δομές δεδομένων**: Δεν αλλάζει το μέγεθος και το σχήμα της δομής δεδομένων
- **Δυναμικές δομές δεδομένων**: Μπορεί να αλλάξει το μέγεθος και το σχήμα της δομής δεδομένων
- Γενικά, η διαχείριση στατικών δομών είναι πιο εύκολη από δυναμικών.
  - Για στατική δομή χρειάζεται να παράσχουμε:
    - τρόπο προσπέλασης και μεταβολής τιμών
  - Για δυναμική δομή:
    - οτι και για μια στατική, και επιπλέον:
    - προσθήκη και διαγραφή καταχωρήσεων
    - εύρεση, δέσμευση, και αποδέσμευση μνήμης

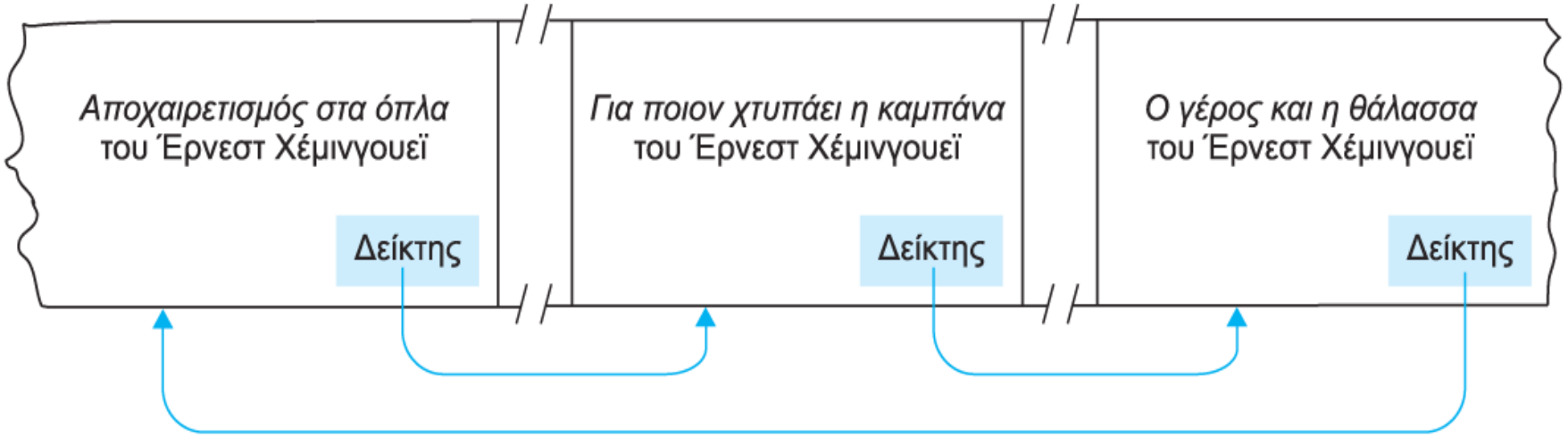
Δείκτες (pointers)

Ένας δείκτης είναι μια περιοχή αποθήκευσης (πχ κελί, καταχωρητής, μεταβλητή) που περιέχει μια διεύθυνση μνήμης.

Παράδειγμα:
Ο **μετρητής προγράμματος** (program counter) της ΚΜΕ (CPU). Λέγεται και **δείκτης εντολών** (instruction pointer).

Χρησιμοποιούνται και για τον εντοπισμό δεδομένων.

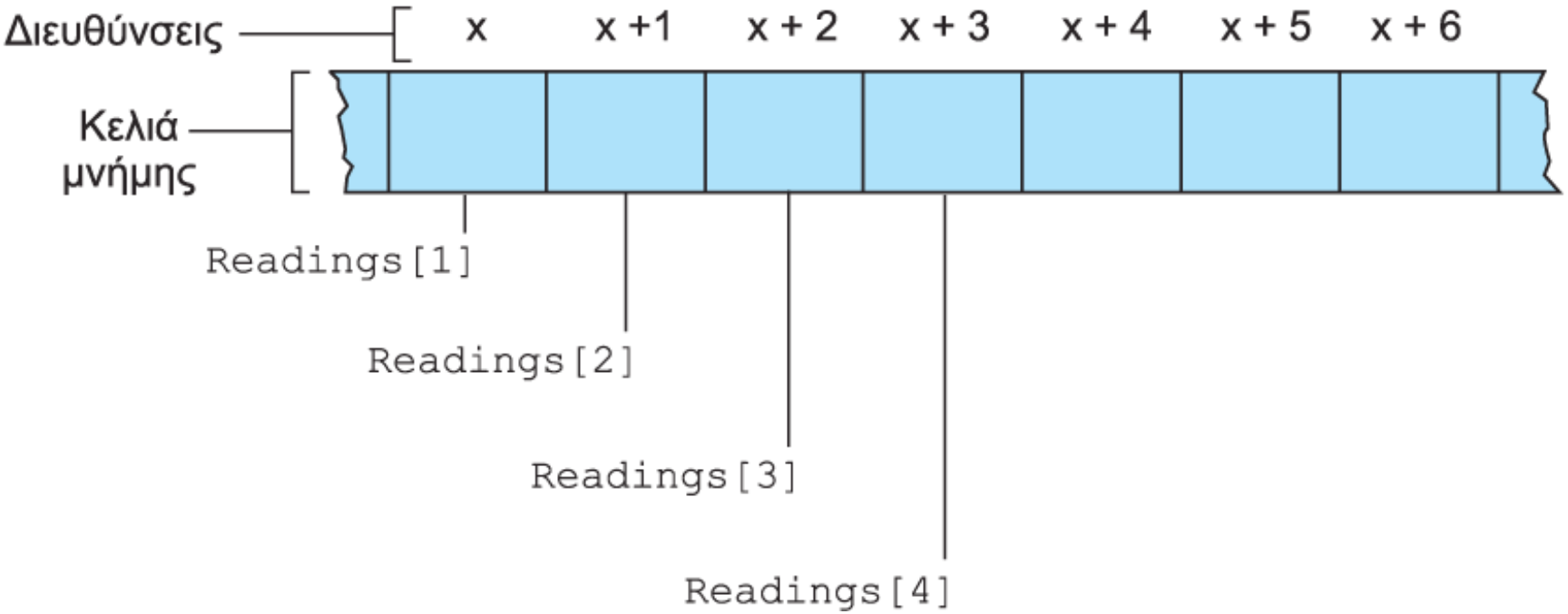
Σχήμα 8.4 **Μυθιστορήματα ταξινομημένα κατά τίτλο, αλλά συνδεδεμένα σύμφωνα με το συγγραφέα**



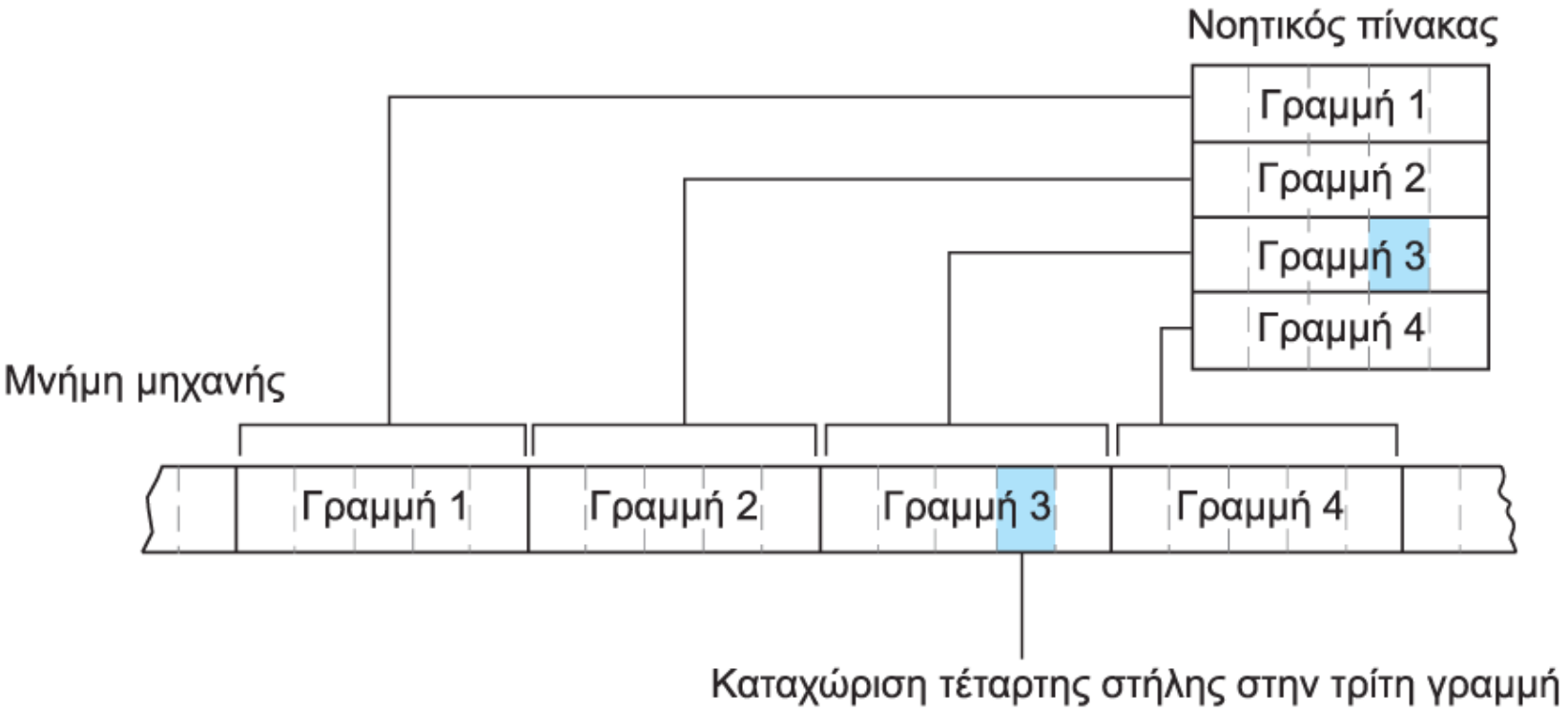
Αποθήκευση πινάκων

- Ομοιογενείς πίνακες
  - **Διάταξη κατά γραμμές** και **διάταξη κατά στήλες**
  - Πολυώνυμο διεύθυνσης
- Ετερογενείς πίνακες
  - Τα συστατικά στοιχεία μπορούν να αποθηκεύονται το ένα μετά το άλλο σε ένα συνεχόμενο μπλοκ
  - Τα συστατικά στοιχεία μπορούν να αποθηκεύονται σε ξεχωριστές θέσεις που προσδιορίζονται με δείκτες

Σχήμα 8.5 Αποθήκευση του πίνακα ενδείξεων θερμοκρασίας στη μνήμη με αρχή τη διεύθυνση x



Σχήμα 8.6 Ένας δισδιάστατος πίνακας με τέσσερις γραμμές και πέντε στήλες, αποθηκευμένος με διάταξη κατά γραμμές



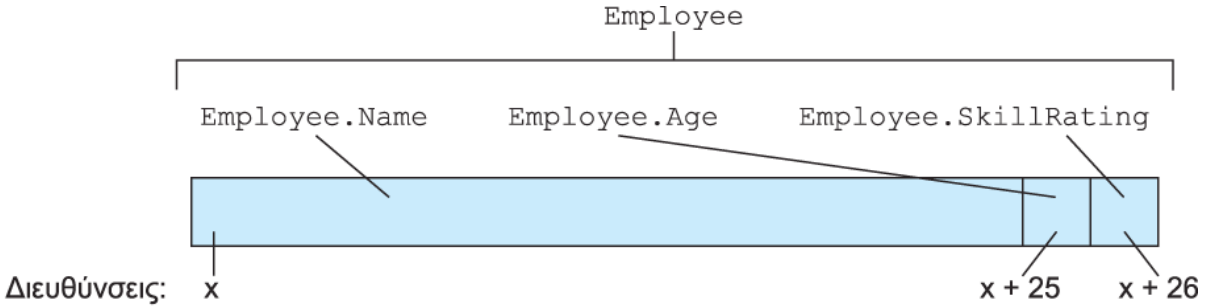
Πολυώνυμο Διεύθυνσης

- Έστω ο πίνακας Readings με στοιχεία Readings[1], ..., Readings[20] με αρχή την διεύθυνση x.
  - Το Readings[i] βρίσκεται στην διεύθυνση:

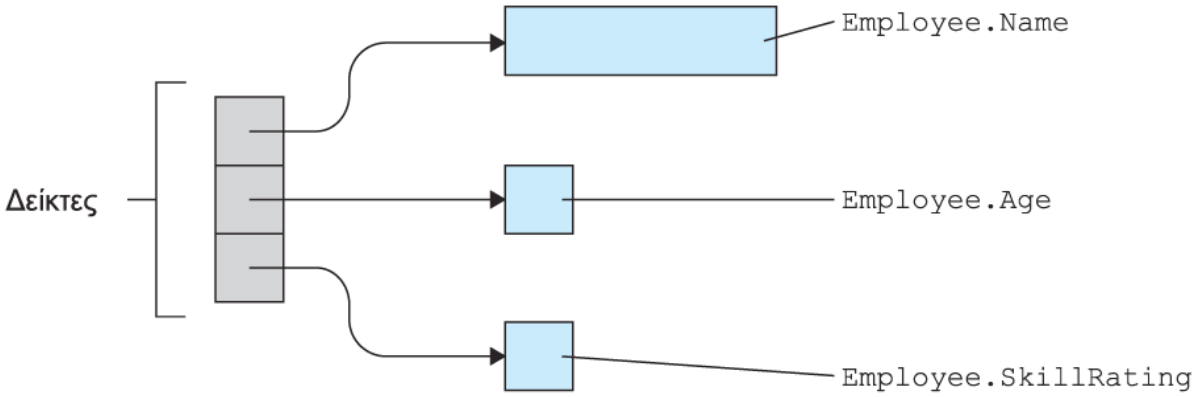
    y = x + (i-1)
- Έστω ο πίνακας R με στοιχεία R[1][1], R[1][2], ..., R[2,1], ..., R[r][c] είναι αποθηκευμένος κατά γραμμές από την x.
  - Το R[i][j] βρίσκεται στην διεύθυνση:

    y = x + (i-1)*c + (j-1)
  - Για αποθήκευση κατά στήλες, ποιο είναι το πολυώνυμο διεύθυνσης;

Σχήμα 8.7 Αποθήκευση του ετερογενούς πίνακα Employee



α. Πίνακας αποθηκευμένος σε συνεχόμενο τμήμα

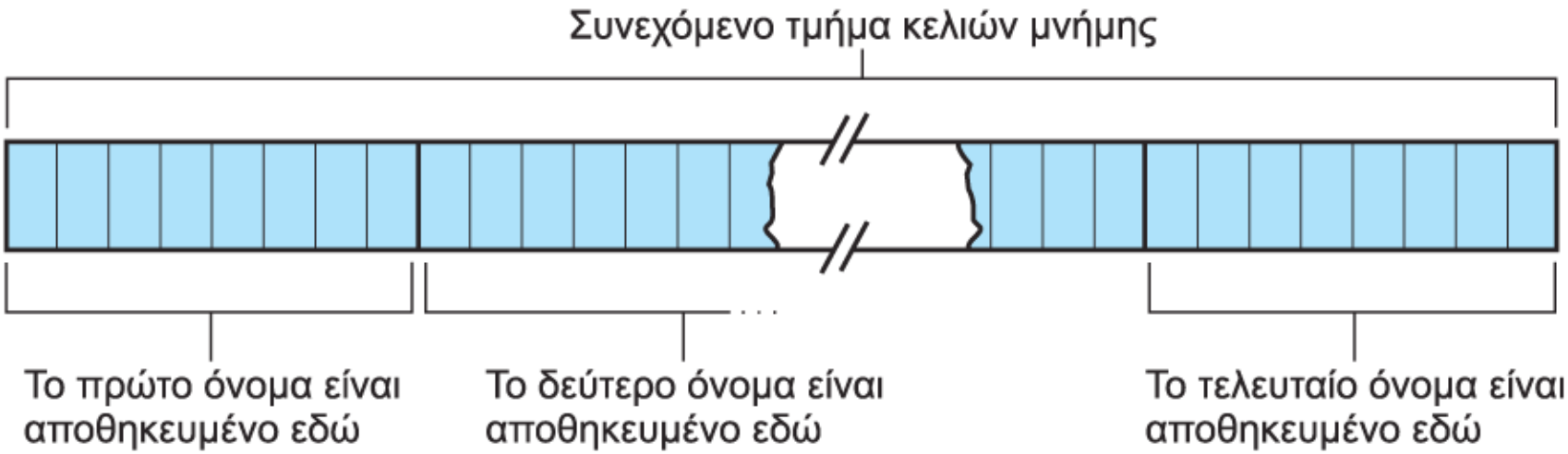


β. Στοιχεία πίνακα αποθηκευμένα σε διαφορετικές θέσεις

Αποθήκευση Λίστας

- **Συνεχόμενη λίστα:** Λίστα αποθηκευμένη σε έναν ομοιογενή πίνακα
- **Συνδεδεμένη λίστα:** Λίστα στην οποία οι καταχωρίσεις είναι συνδεδεμένες με δείκτες
  - **Δείκτης κεφαλής:** Δείκτης στην πρώτη εγγραφή της λίστας
  - **Δείκτης NIL (ή NULL):** Μια τιμή “μη δείκτη” που χρησιμοποιείται για να προσδιορίσει το τέλος της λίστας

Σχήμα 8.8 Ονόματα αποθηκευμένα στη μνήμη με τη μορφή συνεχόμενης λίστας



Συνεχόμενες Λίστες

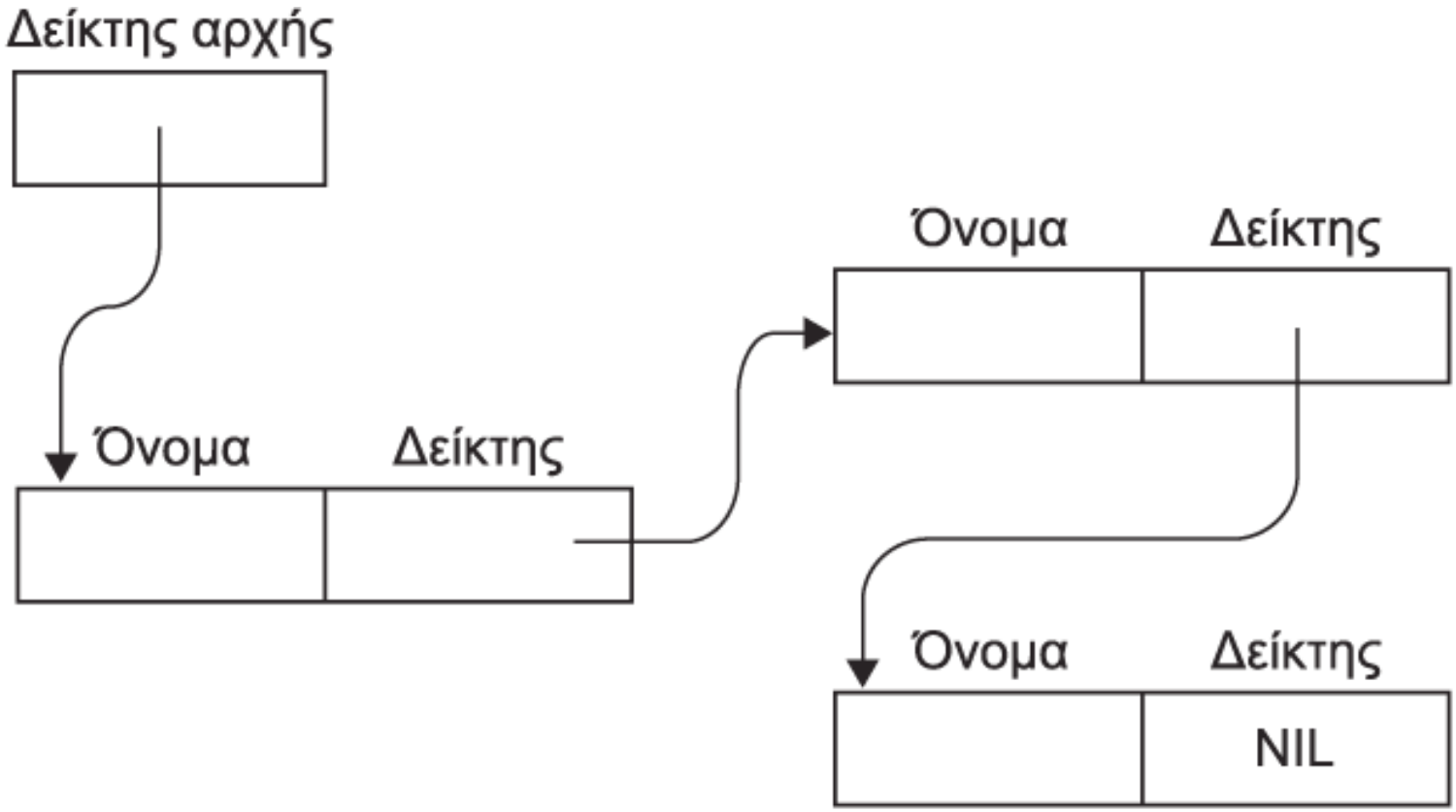
Πλεονεκτήματα:

Βολικές για την υλοποίηση στατικών λιστών.

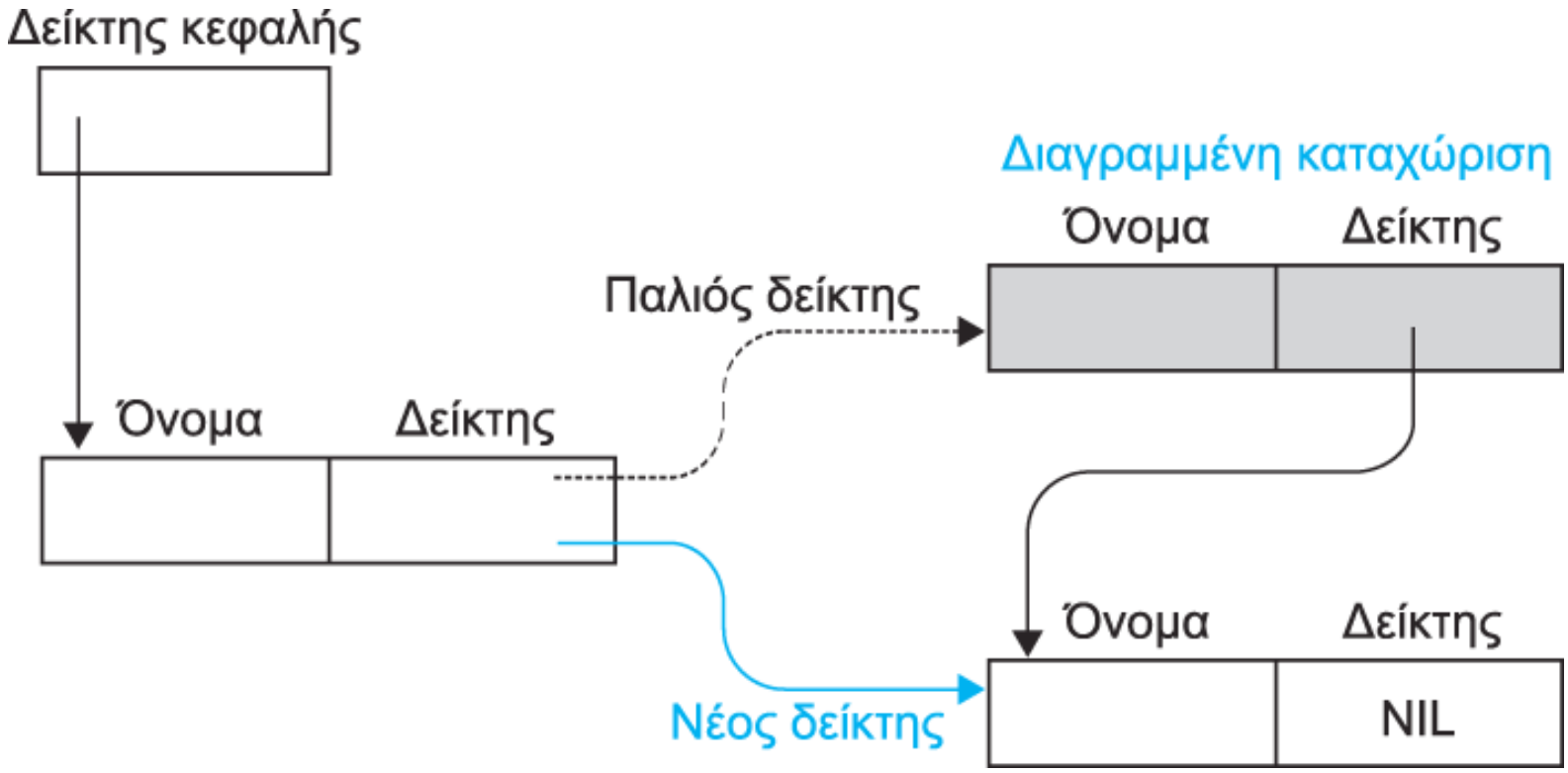
Μειονεκτήματα:

Για δυναμικές λίστες, η διαγραφή και η εισαγωγή νέων στοιχείων μπορούν να προκαλέσουν χρονοβόρες αναδιατάξεις των καταχωρήσεων.

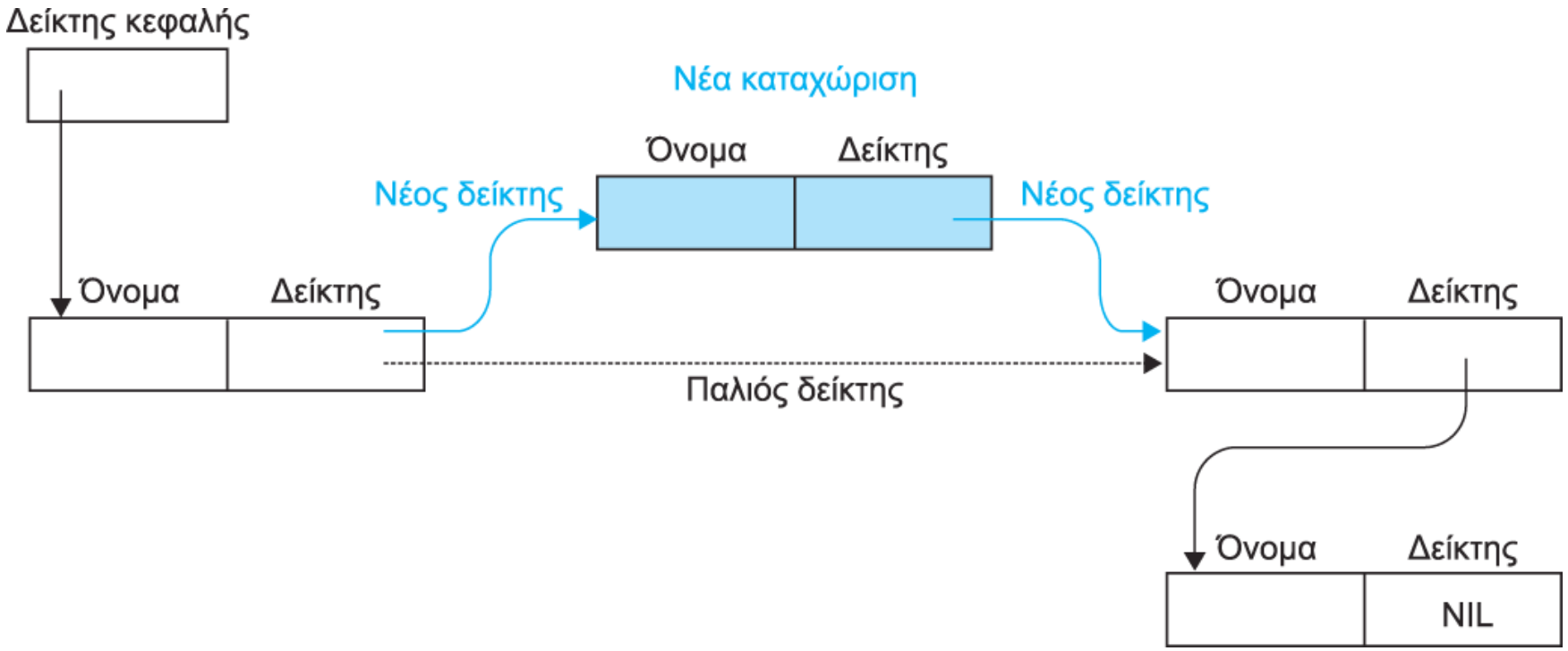
Σχήμα 8.9 Η δομή μιας συνδεδεμένης λίστας



Σχήμα 8.10 Διαγραφή καταχώρισης από συνδεδεμένη λίστα



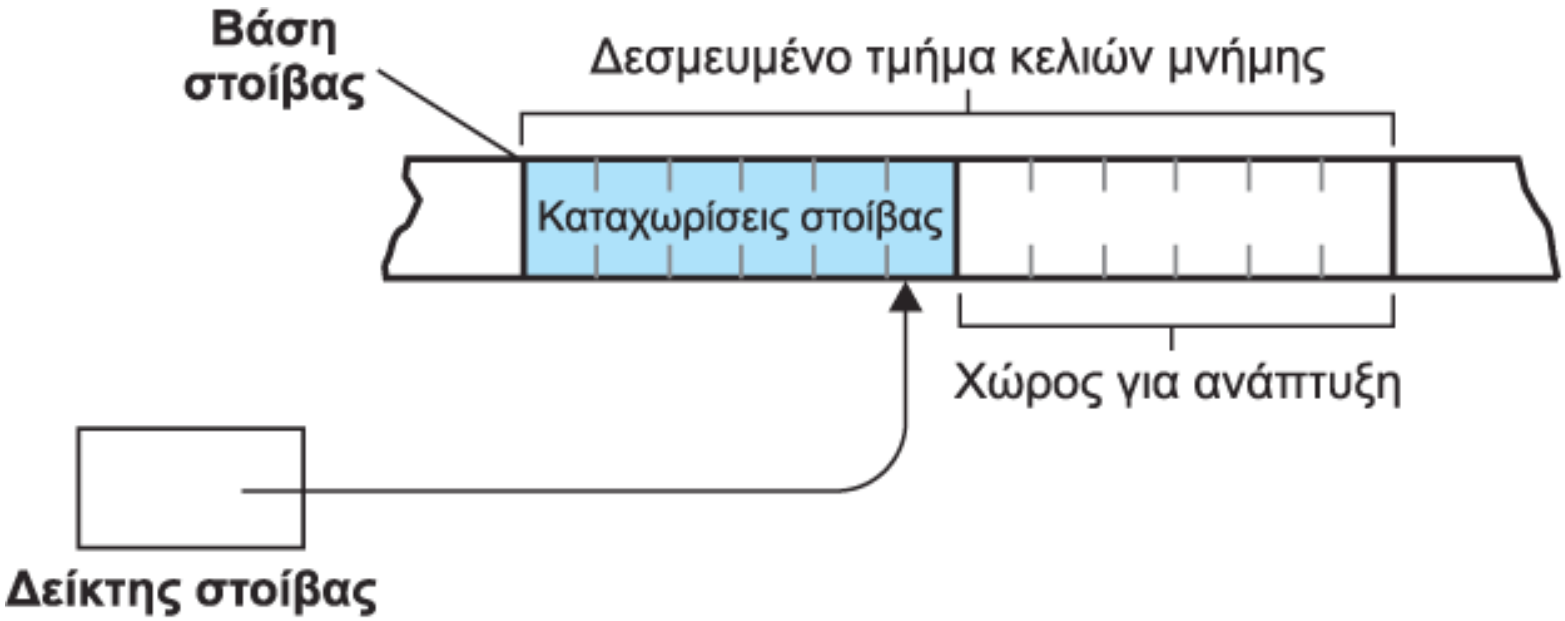
Σχήμα 8.11 Προσθήκη καταχώρισης σε συνδεδεμένη λίστα



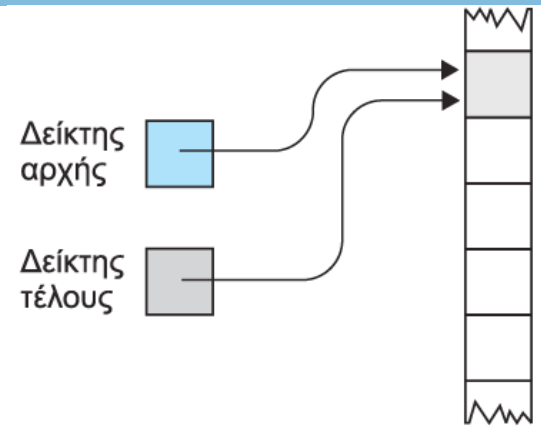
Αποθήκευση Στοιβών και Ουρών

- Οι στοίβες και ουρές αποθηκεύονται συνήθως ως συνεχόμενες λίστες.
- Κρίσιμη απόφαση:
  - ο προσδιορισμός του μέγιστου μεγέθους
- Οι ουρές αποθηκεύονται συνήθως ως **κυκλικές ουρές**
  - Αποθηκεύονται σε ένα συνεχόμενο τμήμα όπου η πρώτη καταχώριση θεωρείται ότι ακολουθεί την τελευταία καταχώριση
  - Αποφεύγει το να “ξεγλιστρήσει” η ουρά έξω από τον εκχωρημένο χώρο αποθήκευσης

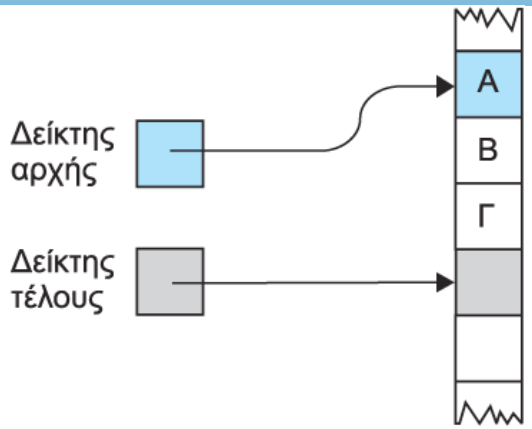
Σχήμα 8.12 **Μια στοίβα στη μνήμη**



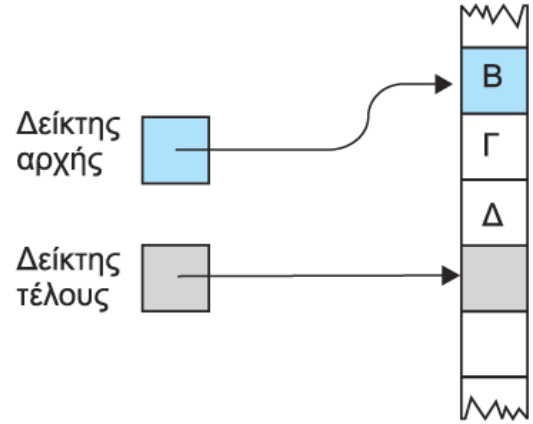
Σχήμα 8.13 Μια υλοποίηση ουράς με δείκτες αρχής και τέλους



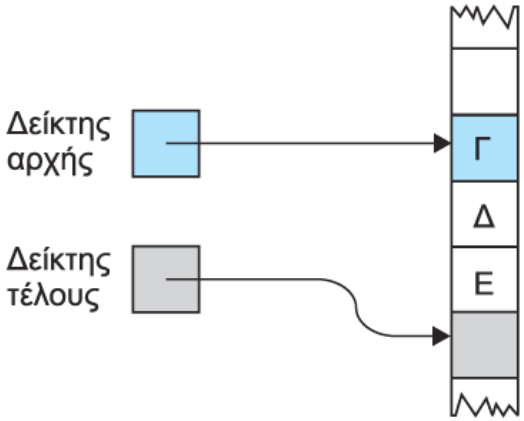
α. Άδεια ουρά



β. Μετά την εισαγωγή των καταχωρίσεων Α, Β, και Γ

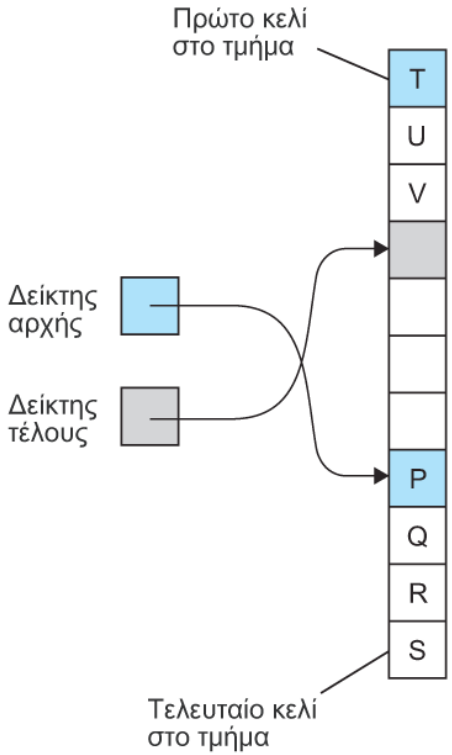


γ. Μετά την αφαίρεση του Α και την εισαγωγή του Δ

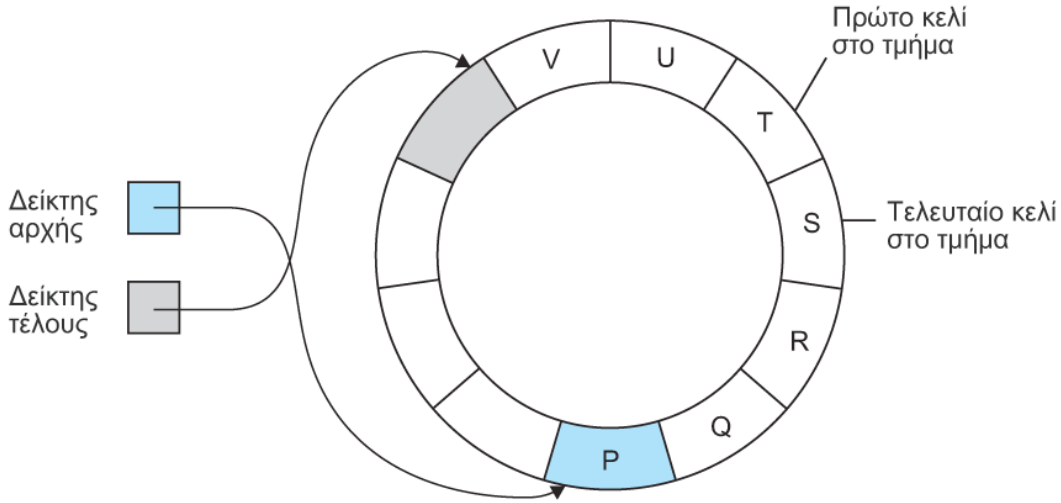


δ. Μετά την αφαίρεση του Β και την εισαγωγή του Ε

Σχήμα 8.14 Μια κυκλική ουρά που περιέχει τα γράμματα Π έως Χ



α. Ουρά όπως είναι πραγματικά αποθηκευμένη



β. Σχηματική αποθήκευση με το τελευταίο κελί να είναι "γειτονικό" του πρώτου κελιού

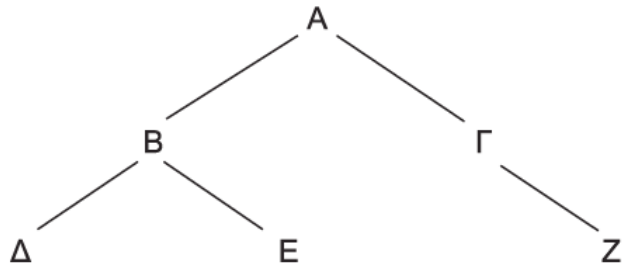
Αποθήκευση Δυαδικών Δέντρων

- Συνδεδεμένη δομή
  - Κάθε κόμβος = κελιά δεδομένων + δύο θυγατρικοί δείκτες
  - Προσπελάζεται μέσω δείκτη στον κόμβο ρίζας
- Δομή συνεχόμενου πίνακα
  - A[1] = κόμβος ρίζας
  - A[2],A[3] = θυγατρικοί κόμβοι A[1]
  - A[4],A[5],A[6],A[7] = θυγατρικοί κόμβοι του A[2] και του A[3]

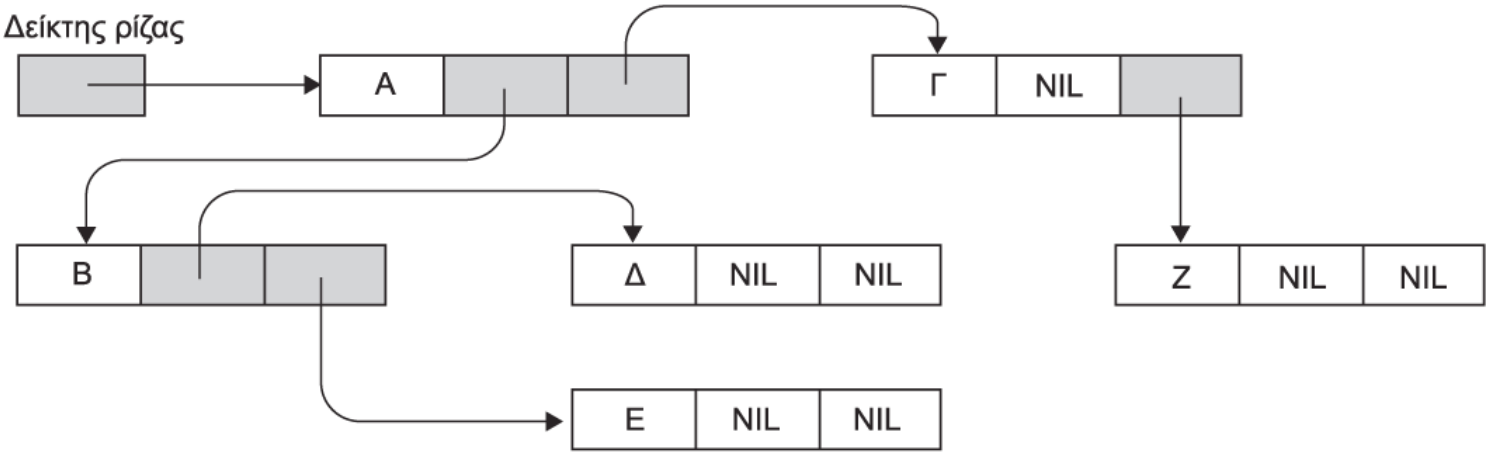
Σχήμα 8.15 Η δομή ενός κόμβου σε δυαδικό δέντρο

Σχήμα 8.16 Η σχηματική και η πραγματική οργάνωση ενός δυαδικού δέντρου με τη χρήση ενός συστήματος αποθήκευσης με συνδέσεις

Σχηματική αναπαράσταση του δέντρου

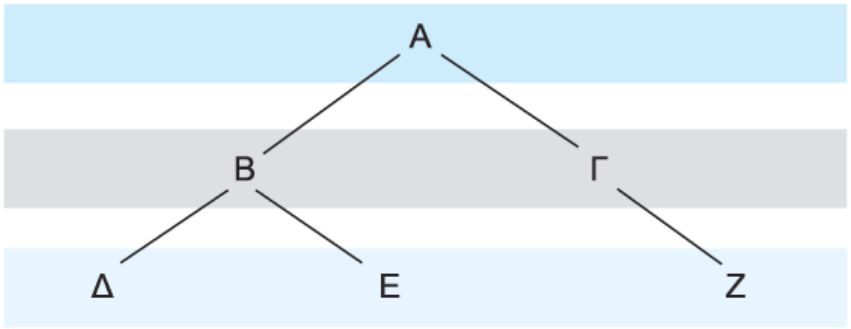


Πραγματική οργάνωση της αποθήκευσης

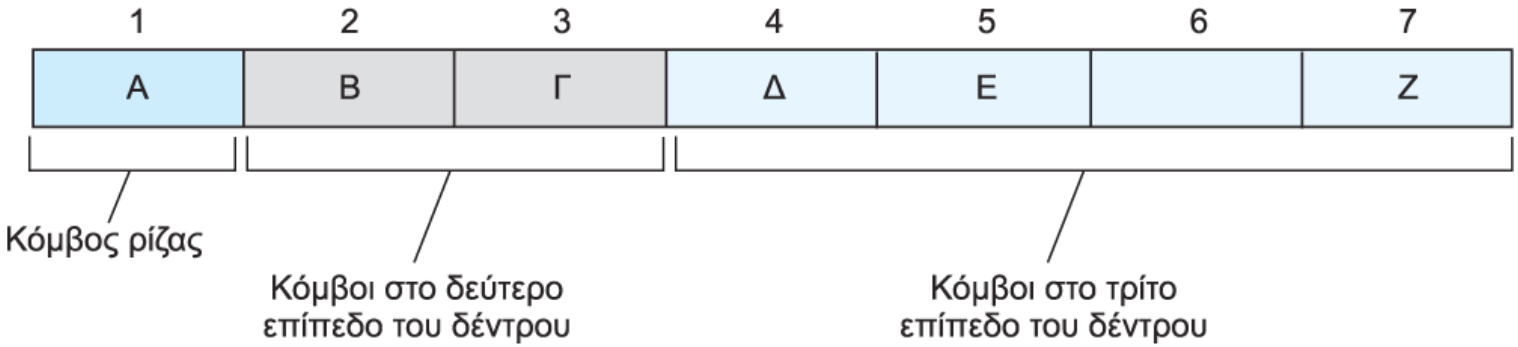


Σχήμα 8.17 Ένα δέντρο που έχει αποθηκευτεί χωρίς δείκτες

Σχηματική αναπαράσταση του δέντρου

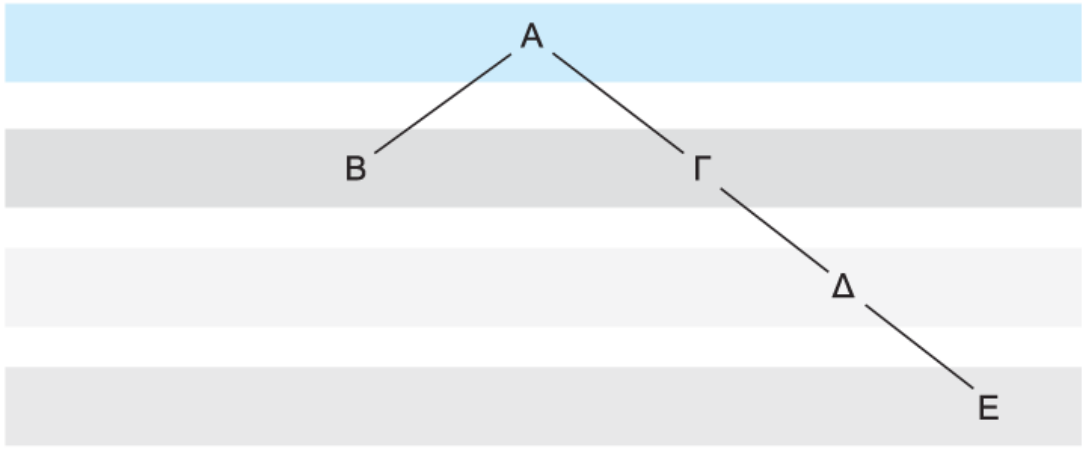


Πραγματική οργάνωση της αποθήκευσης

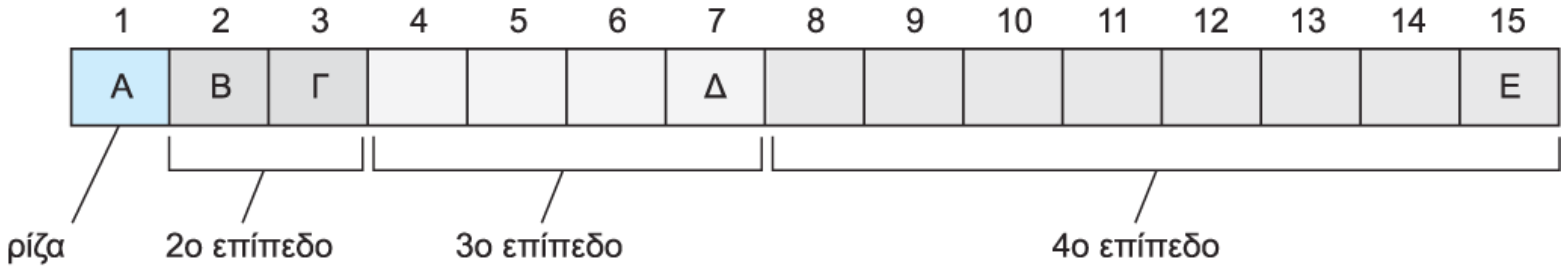


Σχήμα 8.18 Η σχηματική αναπαράσταση ενός αραιού, μη ισορροπημένου δέντρου, και ο τρόπος που θα αποθηκευόταν χωρίς δείκτες

Σχηματική αναπαράσταση του δέντρου



Πραγματική οργάνωση της αποθήκευσης



Χειρισμός Δομών Δεδομένων

- Ιδανικά, ο χειρισμός μιας δομής δεδομένων θα πρέπει να γίνεται αποκλειστικά με προκαθορισμένες διαδικασίες.
  - Παράδειγμα: Μια στοίβα τυπικά χρειάζεται τουλάχιστον διαδικασίες `push` και `pop`.
  - Οι δομές δεδομένων μαζί με τις διαδικασίες αυτές συνιστούν ένα πλήρες αφηρημένο εργαλείο.

Σχήμα 8.19 **Διαδικασία για την εκτύπωση μιας συνδεδεμένης λίστας**

διαδικασία PrintList(Λίστα)
ΤρέχωνΔείκτης ← δείκτης κεφαλής της Λίστας
όσο (ΤρέχωνΔείκτης διάφορος του NIL) **κάνε**
(Τύπωσε το όνομα της καταχώρισης στην οποία δείχνει ο ΤρέχωνΔείκτης.
Βρες την τιμή στο κελί δείκτη της καταχώρισης της Λίστας όπου
δείχνει ο ΤρέχωνΔείκτης, και ανάθεσε εκ νέου στον ΤρέχονταΔείκτη
την τιμή αυτή.)

Οριζόμενοι από το Χρήστη Τύποι Δεδομένων (user-defined datatypes)

- Για την καλύτερη κάλυψη των αναγκών μιας εφαρμογής, μπορούμε να ορίσουμε δικούς μας τύπους δεδομένων χρησιμοποιώντας τους βασικούς.
- **Τύπος οριζόμενος από τον χρήστη = πρότυπο για μια ετερογενή δομή**
- Παράδειγμα:

```
όρισε τύπο ΤύποςΥπαλλήλου ως
{
 char    Όνομα[25];
 int     Ηλικία;
 real    ΒαθμόςΙκανότητας;
}
```

User-defined datatypes

Παρόλο που τα user-defined datatypes είναι χρήσιμα, δεν είναι τύποι με όλη τη σημασία της λέξης (όπως πχ οι βασικοί τύποι):

-- Δεν μπορούν να εφαρμοστούν οι στοιχειώδεις πράξεις (πχ πρόσθεση).

-- Πρέπει να οριστούν και οι σχετικές λειτουργίες/διαδικασίες για προσπέλαση και χειρισμό,
πχ. push/pop στις στοίβες.

Αφηρημένοι Τύποι Δεδομένων

- Ένας οριζόμενος από το χρήστη τύπος δεδομένων με διαδικασίες για προσπέλαση και χειρισμό
- Παράδειγμα:

```
όρισε τύπο ΤύποςΣτοίβας ως
{
 int ΚαταχωρίσειςΣτοίβας[20];
 int ΔείκτηςΣτοίβας = 0;
 διαδικασία push(τιμή)
     {
      ΚαταχωρίσειςΣτοίβας[ΔείκτηςΣτοίβας] ← τιμή;
      ΔείκτηςΣτοίβας ← ΔείκτηςΣτοίβας + 1;
     }
 διαδικασία pop . . .
}
```

Κλάση

- Ένας αφηρημένος τύπος δεδομένων με πρόσθετες λειτουργίες:
  - Τα χαρακτηριστικά μπορούν να κληρονομηθούν
  - Τα περιεχόμενα μπορούν να ενθυλακωθούν (πχ να κρυφτούν από τον χρήστη)
  - Μέθοδοι κατασκευής για απόδοση αρχικών τιμών σε νέα αντικείμενα (δομητές/constructors).

Βασικό δομικό στοιχείο του αντικειμενοστρεφούς προγραμματισμού. (C++, C#, Java, κ.α.)

Σχήμα 8.27 **Στοίβα ακεραίων αριθμών υλοποιημένη σε Java και C#**

```
class StackOfIntegers
{private int[] StackEntries = new int[20];
 private int StackPointer = 0;


 public void push(int NewEntry)
 {if (StackPointer < 20)
    StackEntries[StackPointer++] = NewEntry;
 }

 public int pop()
 {if (StackPointer > 0) return StackEntries[--StackPointer];
  else return 0;
 }
}
```