Προηγμένα Θέματα Βάσεων Δεδομένων

Διδάσκων: Άγγελος Μιχάλας

Περιεχόμενα

Ευρετήρια - Κατάλογοι

- Ένα **ευρετήριο/κατάλογος (index)** είναι μια βοηθητική δομή αρχείου που κάνει πιο αποδοτική την αναζήτηση μιας εγγραφής σε ένα αρχείο.
- Το ευρετήριο καθορίζεται (συνήθως) σε ένα **γνώρισμα του αρχείου** που καλείται **πεδίο ευρετηριοποίησης (indexing field)**
- Το ευρετήριο αρχείου είναι ένα διατεταγμένο αρχείο με σταθερού μήκους εγγραφές
- Για ένα αρχείο μπορούμε να ορίσουμε περισσότερα από ένα ευρετήρια

Ευρετήρια - Κατάλογοι

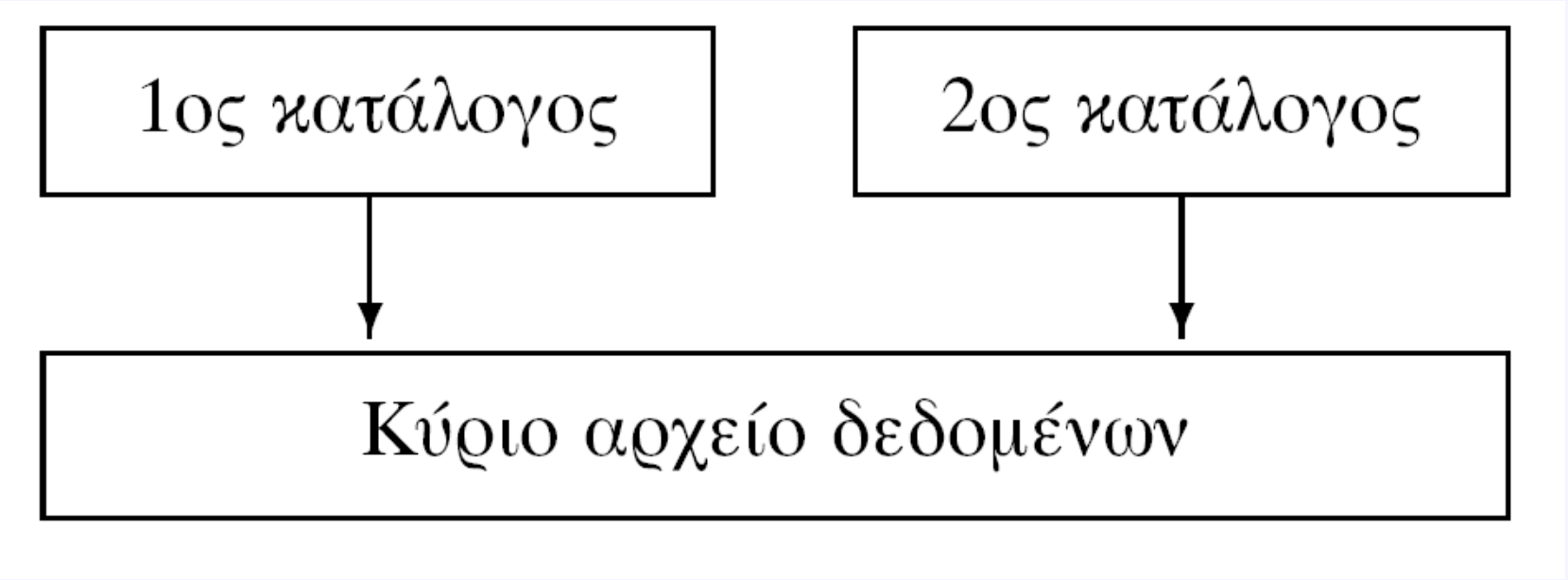
- Ο κατάλογος/ευρετήριο είναι μια ***δομή*** που υλοποιεί μια συνάρτηση
  - το όρισμά της συνάρτησης είναι η τιμή του κλειδιού,
  - η τιμή της συνάρτησης είναι ο αριθμός εγγραφής.
- Ο κατάλογος είναι ανεξάρτητος του αρχείου φυσικά και λογικά.
- Σε έναν κατάλογο το κλειδί μπορεί να είναι ένα χαρακτηριστικό, μέρος του ή συνδυασμός πολλών χαρακτηριστικών.

Ευρετήρια - Κατάλογοι

- Το κόστος αναζήτησης στον κατάλογο είναι μικρό (σε σχέση με την αναζήτηση στο αρχείο) και μπορούμε να βρούμε μια εγγραφή (τον αριθμό ή τη θέση της) με βάση το κλειδί.
- Εναλλακτικά μπορούμε να χρησιμοποιήσουμε τη μέθοδο του ***κατακερματισμού*** που δεν απαιτεί κάποια δομή (όπως ο κατάλογος) αλλά υποφέρει από το πρόβλημα των συνωνυμιών.

Ευρετήρια - Κατάλογοι

- Μπορεί να υπάρχει ένας ή πολλοί κατάλογοι



Ευρετήρια - Κατάλογοι

- Ο απλούστερος τύπος καταλόγου είναι ο **γραμμικός κατάλογος** (linear index), που περιέχει τις εγγραφές ταξινομημένες κατά αύξουσα σειρά χωρίς καμία άλλη δόμηση.
  - Π.χ. Ο κατάλογος ενός βιβλίου
- Ένας γραμμικός κατάλογος μπορεί να ***προσπελασθεί σειριακά*** αλλά αποτελεσματικότερα με ***δυαδική αναζήτηση***.
- Ο κατάλογος αυτός διατηρεί όλα τα **μειονεκτήματα** των **σειριακών αρχείων**, δηλαδή **μη αποτελεσματικές μεθόδους εισαγωγής, διαγραφής και αναδιοργάνωσης**.

Ευρετήρια - Κατάλογοι

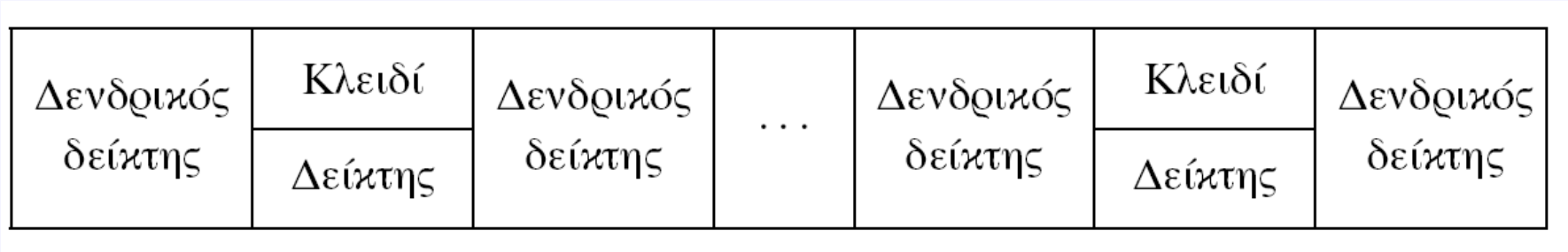
- Οι δενδρικοί κατάλογοι (tree index) έχουν συγκεκριμένη ιεραρχία επιπέδων και διακρίνονται σε 2 είδη:
  - Ομογενή δένδρα
  - Ετερογενή δένδρα
- **Ετερογενή** είναι τα **δένδρα** που **περιέχουν** μόνο **ένα είδος δεικτών**, αλλά οι δείκτες αυτοί είναι διαφορετικοί για τους κόμβους των μεσαίων επιπέδων απ' ότι για το τελευταίο επίπεδο.

Ευρετήρια - Κατάλογοι

- **Ομογενή** είναι τα **δένδρα** που περιέχουν **δύο είδη δεικτών**, οι **δείκτες δεδομένων** (data pointers) και οι **δενδρικοί δείκτες** (tree pointers).
  - *Στα μεσαία επίπεδα και οι δύο τύποι είναι ενεργοί, ενώ στο τελευταίο επίπεδο είναι ενεργοί μόνο οι δείκτες δεδομένων*.

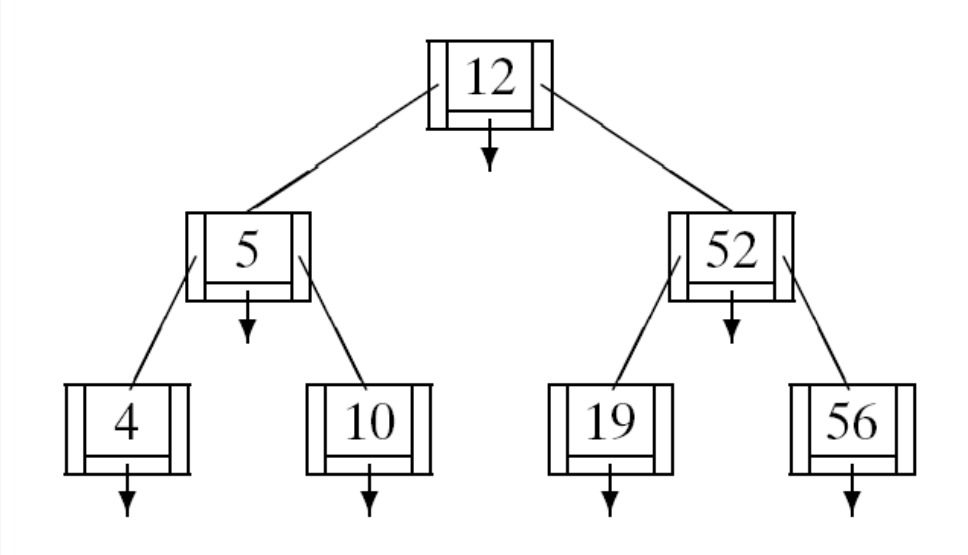
Ευρετήρια - Κατάλογοι

- Σε έναν κόμβο ομογενούς δένδρου, για κάθε κλειδί υπάρχουν δύο δενδρικοί δείκτες
  - ο πρώτος κατευθύνει προς τα κλειδιά που είναι μικρότερα από το πρώτο κλειδί του κόμβου
  - ο δεύτερος κατευθύνει προς τα κλειδιά που είναι μεγαλύτερα από το τελευταίο κλειδί του κόμβου

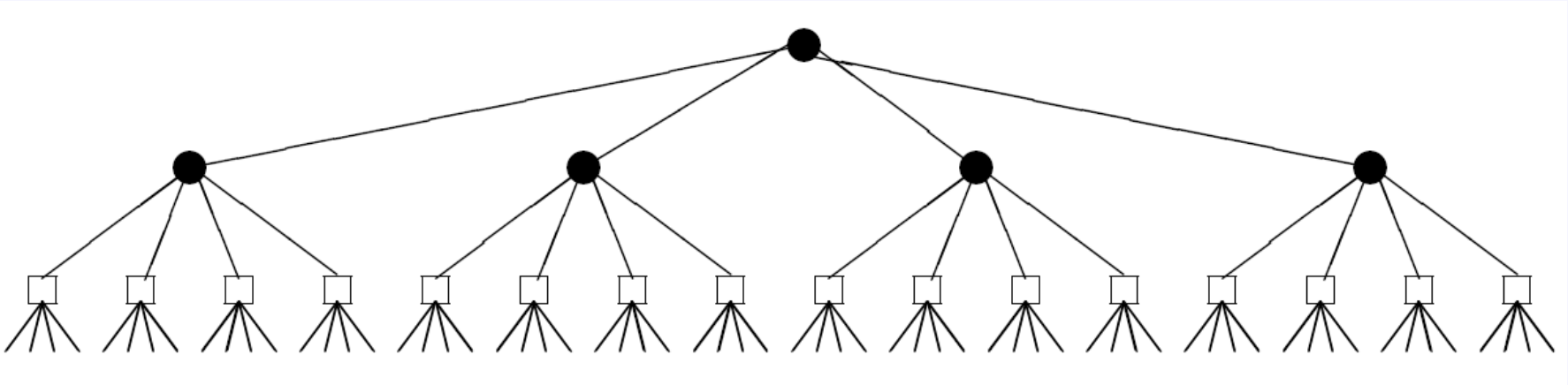


Ευρετήρια - Κατάλογοι

- Ομογενές



- Ετερογενές



Ευρετήρια - Κατάλογοι

- Το ύψος ενός ετερογενούς δένδρου h ορίζεται

$$h \geq \log_m n$$

όπου **n** το **πλήθος** των **εγγραφών** και **m** είναι ο **παράγοντας διακλάδωσης** (*branching factor*).

- Αν m=4 και h=3, τότε n=64.

Ευρετήρια - Κατάλογοι

- Το ύψος ενός ομογενούς δένδρου προκύπτει ως εξής. Γενικά ισχύει:
  - στη ρίζα χωρούν (m-1) κλειδιά,
  - στο δεύτερο επίπεδο m*(m-1) κλειδιά,
  - στο τρίτο επίπεδο m*m*(m-1) κλειδιά.
- Το σύνολο των εγγραφών που δεικτοδοτούνται είναι:

$$(m^0 + m^1 + m^2 + \ldots + m^{h-1}) \times (m-1) \;=\; m^h - 1$$

οπότε: $$n \;\leq\; m^h - 1 \iff h \;\geq\; \log_m(n+1)$$

- Αν m=4 και h=3, τότε n=63.

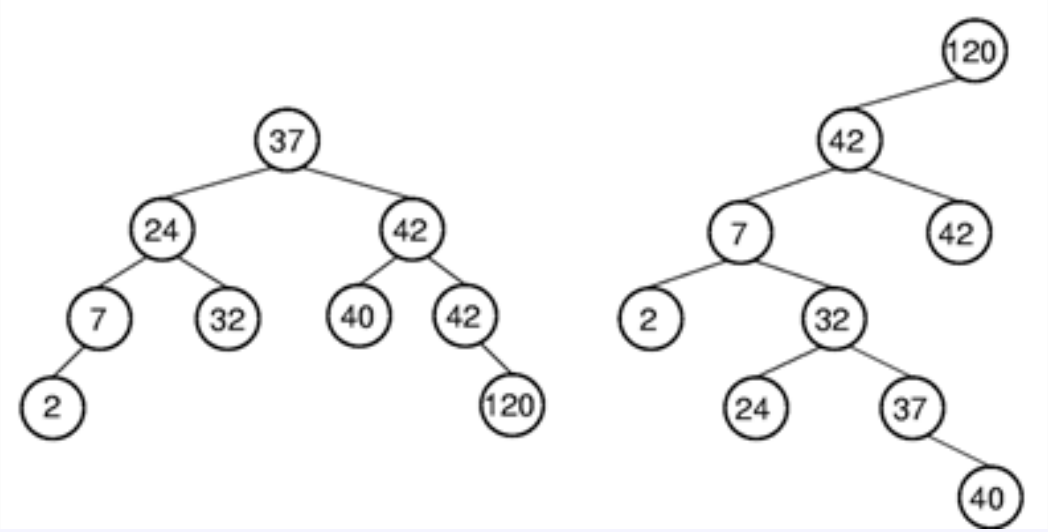
Δένδρα Αναζήτησης

- Δένδρο αναζήτησης είναι ένας ειδικός τύπος δένδρου που χρησιμοποιείται για να καθοδηγήσει την αναζήτηση μιας εγγραφής όταν δίνεται η τιμή ενός πεδίου της.

Δένδρα Αναζήτησης

Ορισμός: Ένα ΔΔΑ είναι δυαδικό δένδρο με διακριτά κλειδιά (τιμές) και τις εξής ιδιότητες:

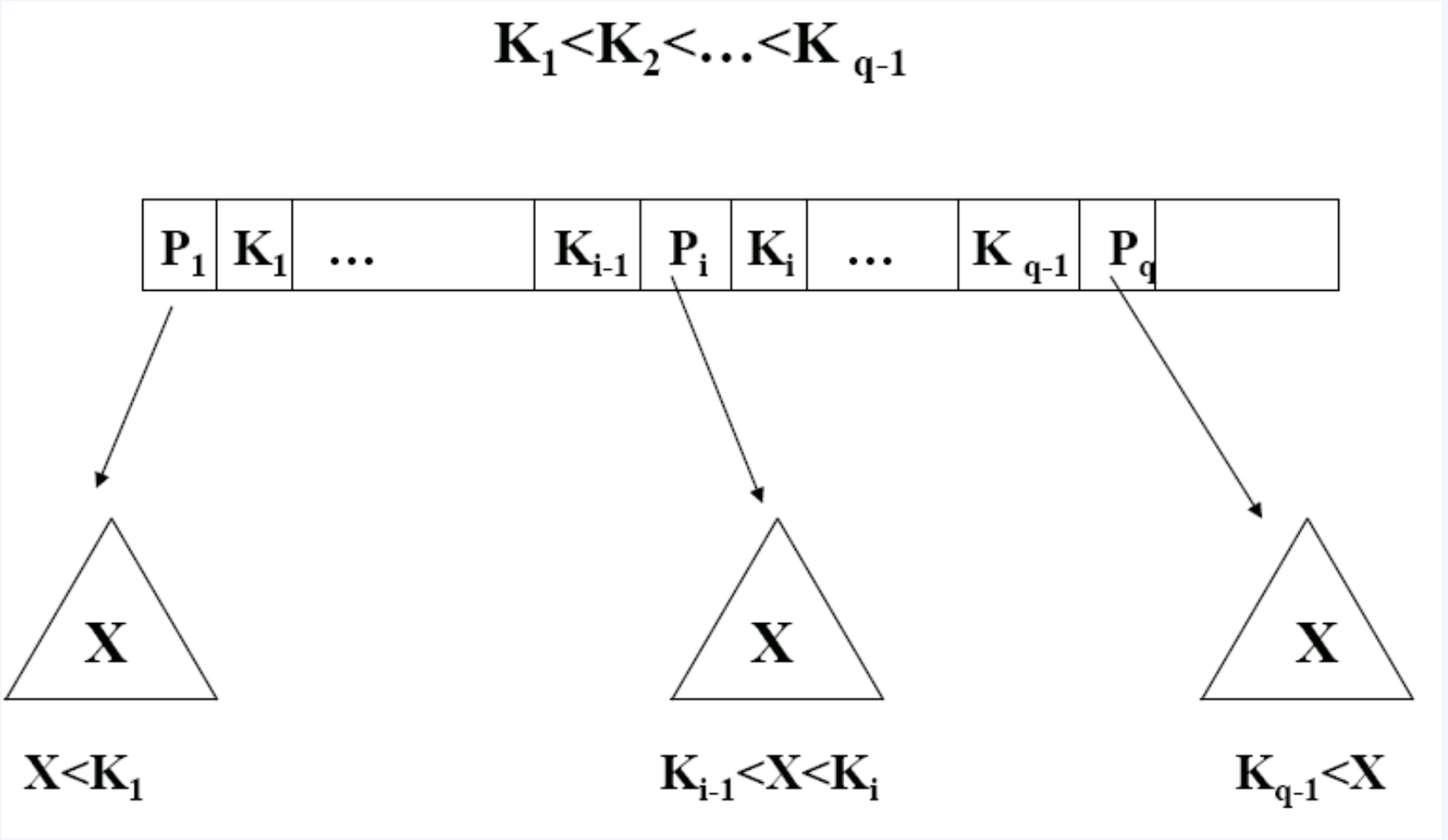
- Τα κλειδιά (αν υπάρχουν) στο αριστερό υποδένδρο της ρίζας είναι μικρότερα από το κλειδί της ρίζας
- Τα κλειδιά (αν υπάρχουν) στο δεξιό υποδένδρο της ρίζας είναι μεγαλύτερα από το κλειδί της ρίζας
- Το αριστερό και το δεξιό υποδένδρο είναι επίσης ΔΔΑ



Κίνητρο:

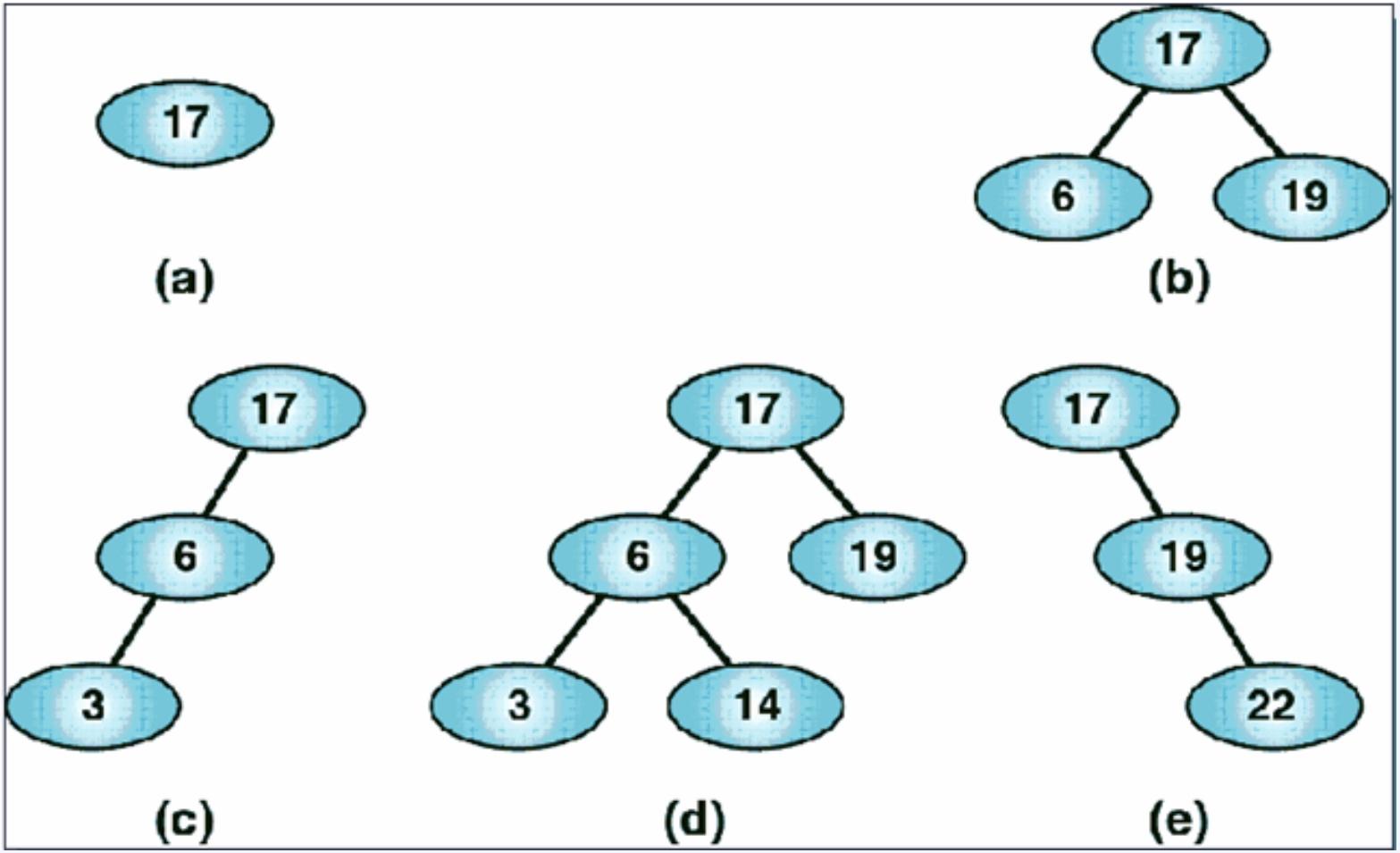
να μειώσουμε τους χρόνους ενημέρωσης και αναζήτησης σε λιγότερο από Θ(n)

Δένδρα Αναζήτησης



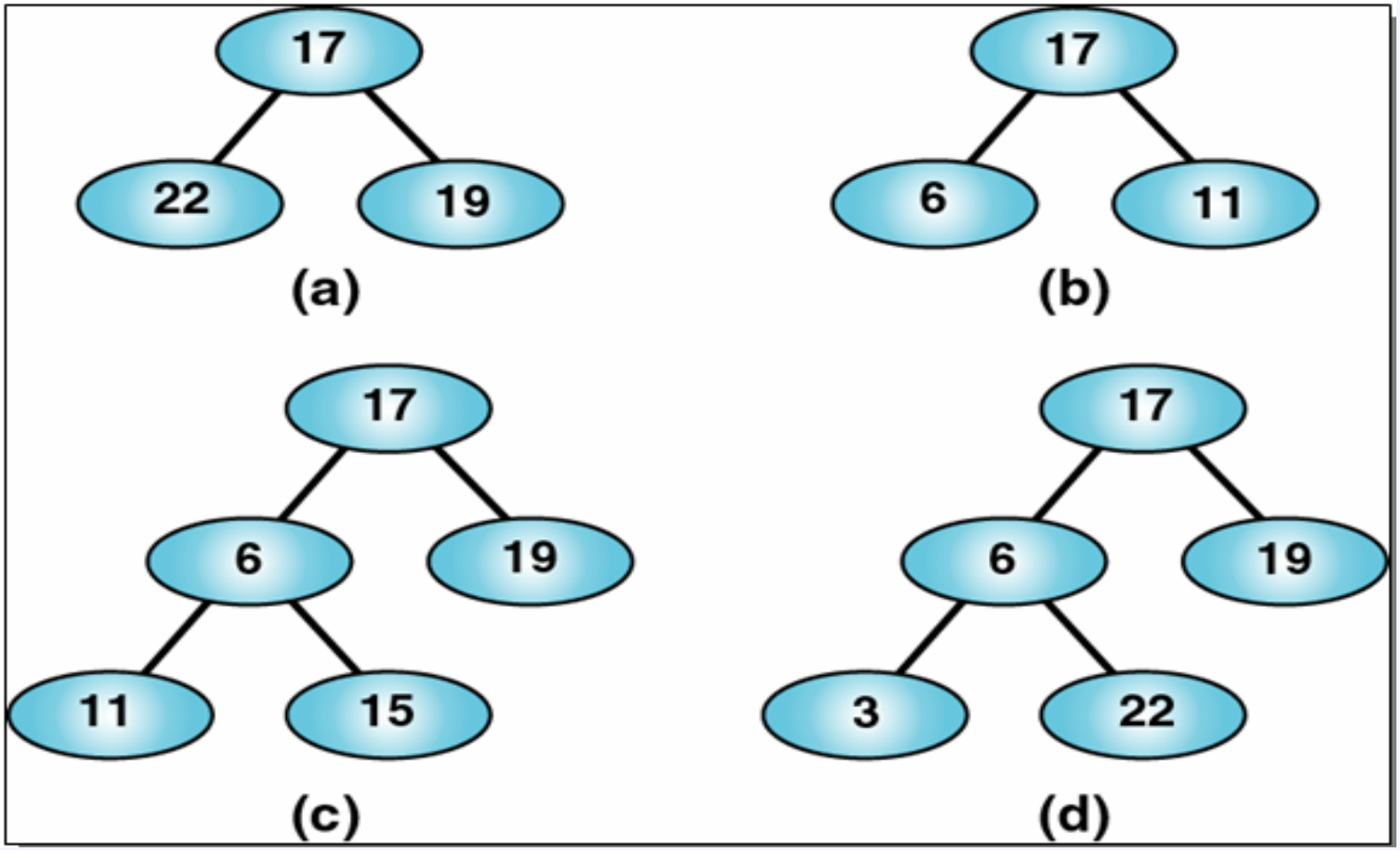
Δένδρα Αναζήτησης

- Έγκυρα δέντρα



Δένδρα Αναζήτησης

- Μη-έγκυρα δέντρα



Δένδρα Αναζήτησης - ADT

AbstractDataType *BSTree* {

instances: binary trees, each node has an element with a key field; all keys are distinct; keys in the left subtree of any node smaller than the key in the node; those in the right subtree are larger;

operations

Create (): create an empty binary search tree

Search (*k*, *e*): return in *e* the element with key *k*; return `false` if the operation fails, return `true` if it succeeds

Insert (*e*): insert element *e* into the search tree

Delete (*k*, *e*): delete the element with key *k* and return it in *e*

Ascend (): Output all elements in ascending order of key

}

Δένδρα Αναζήτησης - Αναζήτηση

```
bool BSTree<E,K>::Search(const K& k, E &e) const
{// Search for element that matches k.
   // pointer p starts at the root and moves through
   // the tree looking for an element with key k
   BinaryTreeNode<E> *p = root;
   while (p)                          // examine p->data
      if (k < p->data) p = p->LeftChild;
      else if (k > p->data) p = p->RightChild;
           else {                     // found element
                  e = p->data;
                  return true;}
   return false;
}
```

Κόστος αναζήτησης = κόστος κατάβασης = O(h)

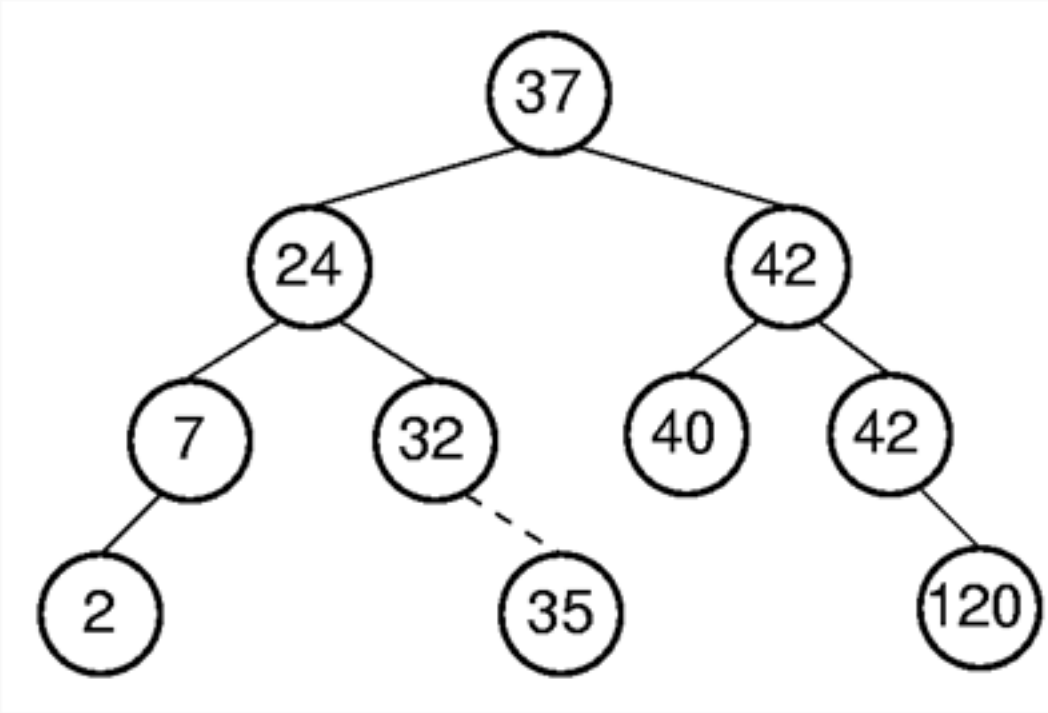
Δένδρα Αναζήτησης - Εισαγωγή

Βασική ιδιότητα ΔΔΑ:

- Η εισαγωγή γίνεται πάντα σε κάποιο (νέο) φύλλο

Διαδικασία:

- Αναζήτηση του στοιχείου (οπότε καταλήγουμε σε κόμβο-φύλλο)
- Εισαγωγή του ως παιδί εκείνου του κόμβου



Δένδρα Αναζήτησης - Εισαγωγή

```
BSTree<E,K>& BSTree<E,K>::Insert(const E& e)
{// Insert e if not duplicate.
   BinaryTreeNode<E> *p = root,   // search pointer
                     *pp = 0;     // parent of p
   // find place to insert
   while (p) {                    // examine p->data
      pp = p;
      // move p to a child
      if (e < p->data) p = p->LeftChild;
      else if (e > p->data) p = p->RightChild;
           else throw BadInput(); // duplicate
      }

   // get a node for e and attach to pp
   …
```

Δένδρα Αναζήτησης - Εισαγωγή

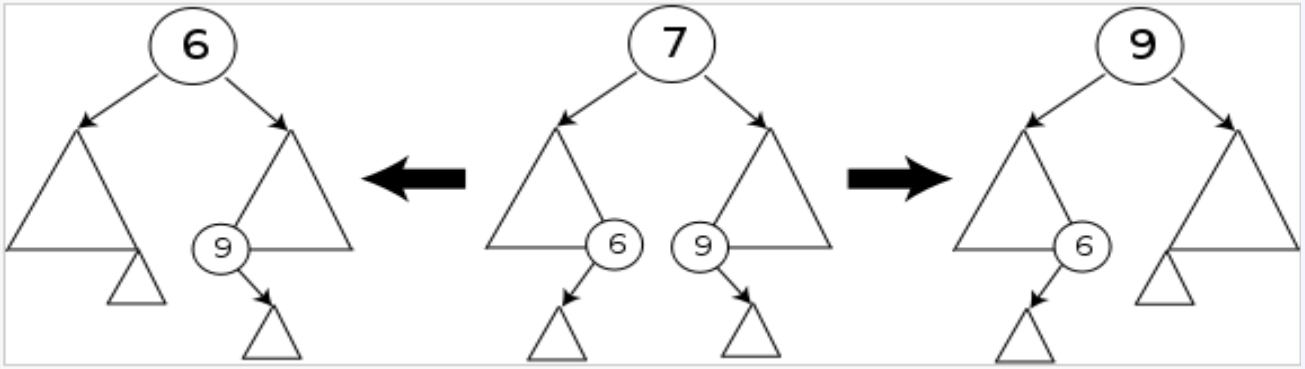
```
…
  // get a node for e and attach to pp
  BinaryTreeNode<E> *r = new
BinaryTreeNode<E> (e);
  if (root) {// tree not empty
     if (e < pp->data) pp->LeftChild = r;
     else pp->RightChild = r;}
  else // insertion into empty tree
       root = r;

  return *this;
}
```

Κόστος = Κόστος αναζήτησης + κόστος 'συγκόλλησης'
νέου κόμβου στον πατέρα-κόμβο = O(h) + O(1) = O(h)

Δένδρα Αναζήτησης - Διαγραφή

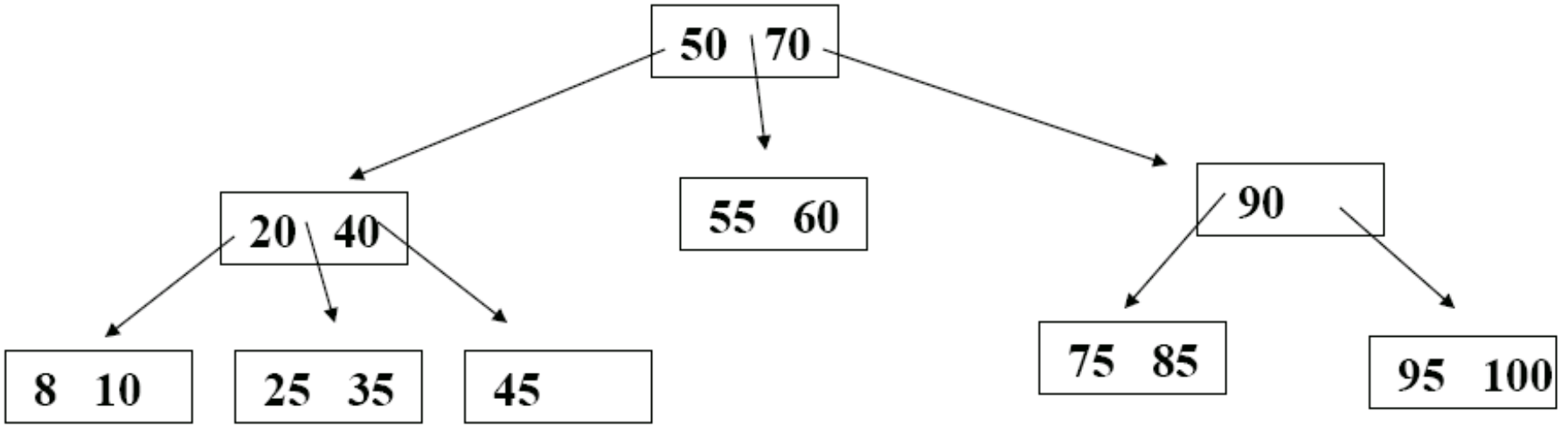
- **Διαγραφή φύλλου (κόμβος χωρίς παιδιά)**:
  - Απλά το απομακρύνουμε από το δένδρο.
- **Διαγραφή κόμβου με ένα παιδί**:
  - Απομακρύνουμε το κόμβο και τον αντικαθιστούμε με το παιδί του.
- **Διαγραφή κόμβου με δύο παιδιά**:
  - Έστω ο κόμβος προς διαγραφή είναι ο N. Μη διαγράψετε το κόμβο N. Αντίθετα επιλέξτε **είτε τον in-order επόμενο κόμβο είτε τον in-order προηγούμενο κόμβο**, R. Αντικαθιστούμε την τιμή του N με την τιμή του κόμβου R και μετά διαγράφουμε τον κόμβο R.



Δένδρα Αναζήτησης

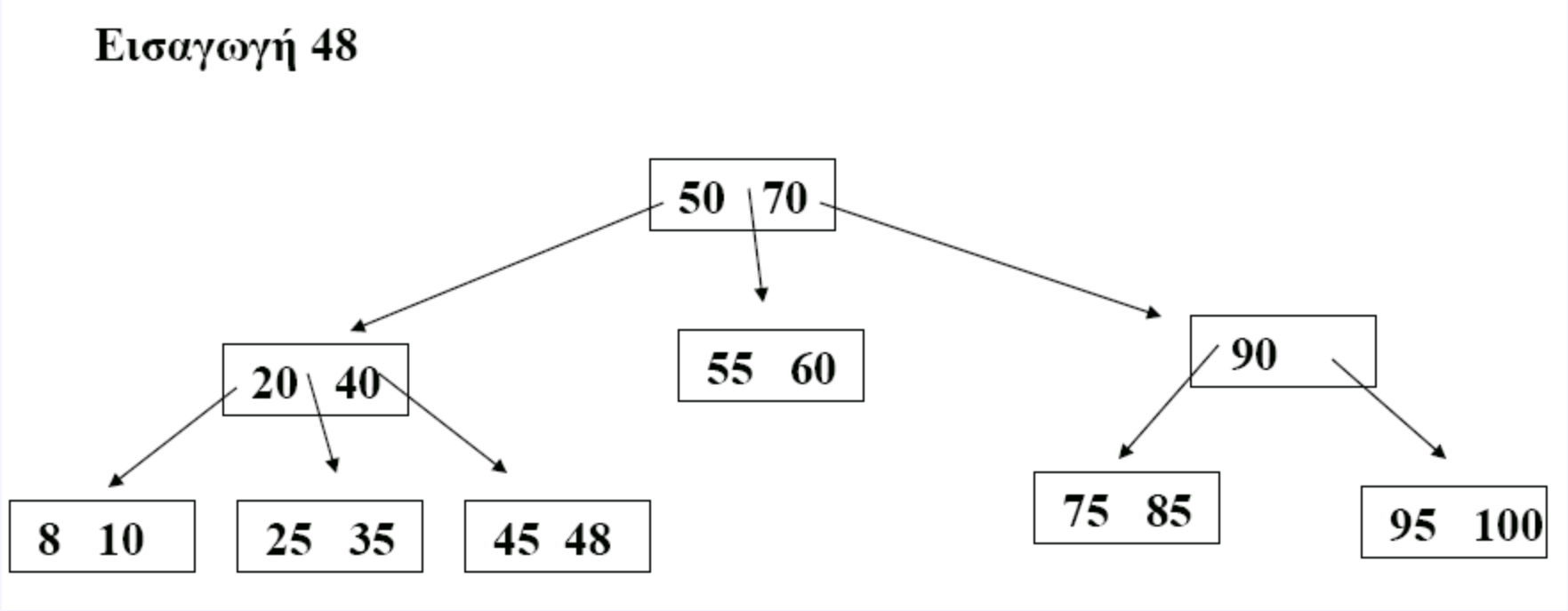
- Τα δυαδικά δένδρα αναζήτησης διευκολύνουν την αναζήτηση στοιχείων
- Αναδρομική αναζήτηση
  - αν η τιμή που ζητείται είναι στη ρίζα, βρέθηκε
  - αν είναι μικρότερη από την τιμή της ρίζας, αρκεί να αναζητηθεί στο αριστερό παιδί
  - αν είναι μεγαλύτερη από την τιμή της ρίζας, αρκεί να αναζητηθεί στο δεξί παιδί
- Κόστος αναζήτησης: O(log n)
  - υπό την προϋπόθεση το δένδρο να είναι **ισοζυγισμένο**
  - **n** είναι το πλήθος των κλειδιών

Δένδρα Αναζήτησης - Παράδειγμα

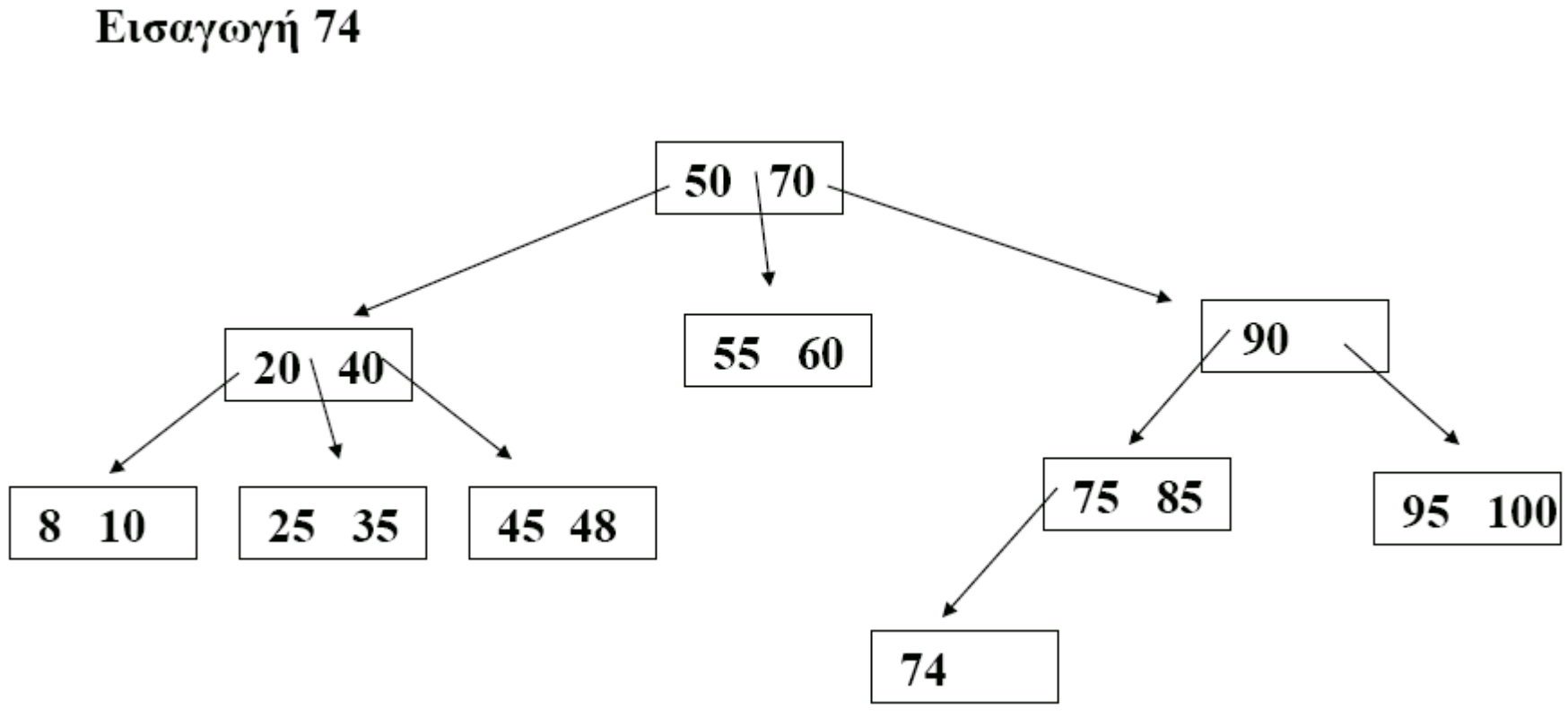


Έστω ότι ο κόμβος χωράει 2 κλειδιά

Δένδρα Αναζήτησης - Παράδειγμα

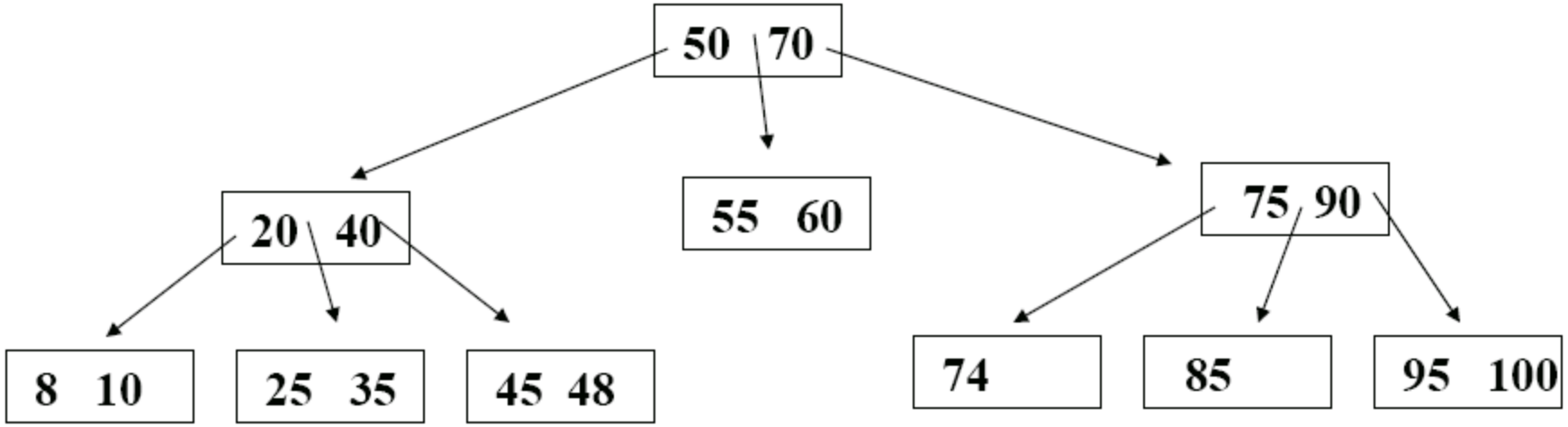


Δένδρα Αναζήτησης - Παράδειγμα



Δένδρα Αναζήτησης - Παράδειγμα

Όμως έτσι μπορεί να μην έχουμε καλή απόδοση ενώ θα μπορούσαμε να έχουμε



Δένδρα Αναζήτησης

Αν το δένδρο είναι τάξεως m σε κάθε κόμβο μπορεί να περιέχει το μέγιστο m-1 κλειδιά. Επομένως ένα δένδρο αναζήτησης ύψους h περιέχει το μέγιστο

$$(m-1)+m(m-1)+m^2(m-1)+\ldots m^{h-1}(m-1)=(m-1)(1+m+m^2+\ldots+m^{h-1})$$

$$=m^h-1$$

Επομένως για N κλειδιά το καλλίτερο δένδρο που μπορεί να δημιουργηθεί είναι N= m^h-1 δηλαδή το πιο κοντό θα έχει ύψος

$$h=\log_m(N+1)$$

Δένδρα Αναζήτησης

- ***Όμως επειδή η χρήση είναι δυναμική δεν είναι πρακτικό με κάθε εισαγωγή και διαγραφή να γίνεται αναδιοργάνωση του δένδρου ώστε να έχω την καλύτερη δυνατή απόδοση***

- **Θέλουμε να μπορεί να γίνεται εύκολα η εισαγωγή** αλλά ταυτόχρονα το δένδρο που δημιουργείται να είναι όσο το δυνατόν **ισοζυγισμένο** με την έννοια οι κόμβοι φύλα να είναι στο ίδιο επίπεδο. **Το να γέρνει προς μια πλευρά το δένδρο οδηγεί σε απόδοση που πλησιάζει αυτήν του ταξινομημένου αρχείου.**

Δένδρα Αναζήτησης

- Επίσης **θέλουμε να γίνεται η καλύτερη δυνατή χρήση του χώρου**. **Οι διαγραφές δημιουργούν κενά στους κόμβους** που μπορεί να αφήσουν το δένδρο με σχεδόν κενούς κόμβους.

- Το **Β-δένδρο** (και οι παραλλαγές του) **αντιμετωπίζει με επιτυχία αυτά τα δύο προβλήματα**

B-Δένδρο

- Τα B-δένδρα είναι η βασική δομή για καταλόγους.
  - Τα δένδρα της οικογένειας αυτής χρησιμοποιούνται σε όλα τα εμπορικά ΣΔΒΔ, όπως στο σύστημα SQL Server της Microsoft, στο σύστημα της Oracle, στο DB2 της IBM κ.α.
- Προτάθηκε από τους Bayer και McCreight το 1972.
- Η ονομασία προέρχεται:
  - Από το επίθετο του Bayer ή
  - Από το όνομα της εταιρίας Boeing όπου εργαζόταν ο Bayer ή
  - Από τη λέξη balanced.
    - Όχι πάντως από τη λέξη binary (δυαδικό).

B-Δένδρο

B-δένδρο τάξεως p

1. **Κάθε εσωτερικός κόμβος είναι της μορφής $<P_1,<k_1,Pr_1>,P_2,<K_2,Pr_2>,\ldots,<k_{q-1},Pr_{q-1}>,P_q>$ $q \leq p$. Τα P_i είναι δείκτες δένδρου ενώ τα Pr_i δείκτες δεδομένων.**
2. **Σε κάθε κόμβο ισχύει: $K_1<K_2<\ldots K_{q-1}$.**
3. **Για όλα τα κλειδιά X στο υποδένδρο που δείχνει το Pi ισχύει $K_{i-1}<X<K_i$ $1<I<q$, $X<K_i$ για $i=1$ και $K_{i-1}<X$ για $i=q$**
4. **Κάθε κόμβος έχει το πολύ p δείκτες δένδρου.**
5. **Κάθε κόμβος εκτός της ρίζας και των φύλων έχει τουλάχιστον $\lceil p/2 \rceil$ δείκτες δένδρου. Ο κόμβος της ρίζας έχει τουλάχιστον δύο δείκτες δένδρου εκτός αν είναι ο μοναδικός κόμβος του δένδρου.**
6. **Ένας κόμβος με q δείκτες δένδρου περιέχει q-1 κλειδιά.**
7. **Όλοι οι κόμβοι φύλα είναι στο ίδιο επίπεδο. Οι κόμβοι φύλα έχουν την ίδια δομή με τους εσωτερικούς κόμβους μόνο που οι δείκτες δένδρου έχουν τιμή null.**

B-Δένδρο

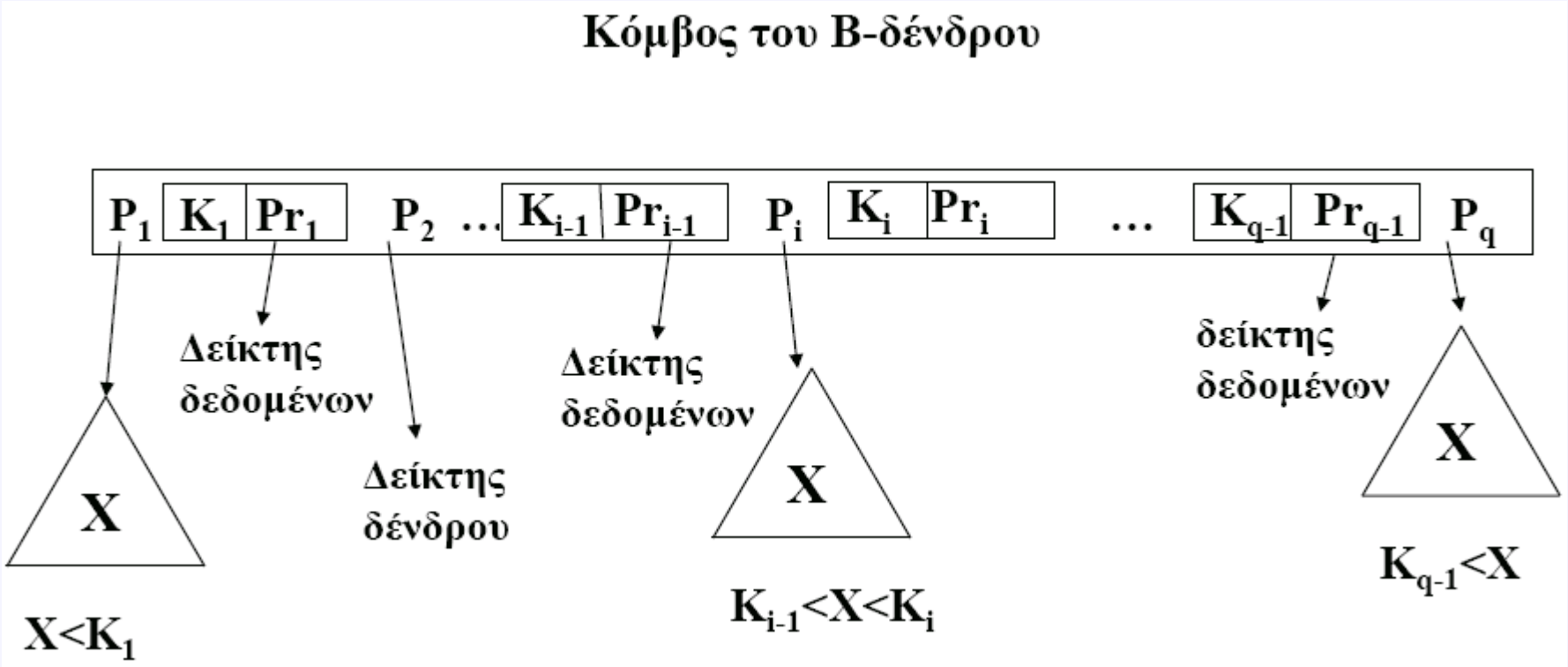
- Σύμφωνα με έναν άλλο ορισμό, ως B-δένδρο βαθμού d ορίζεται το ομογενές δένδρο με τα εξής χαρακτηριστικά:
  - **η ρίζα έχει το ελάχιστο δύο παιδιά,**
  - **οι εσωτερικοί κόμβοι (εκτός της ρίζας) έχουν το ελάχιστο d κλειδιά και το μέγιστο 2d κλειδιά,**
  - **ένας εσωτερικός κόμβος με k κλειδιά έχει k+1 παιδιά, και**
  - **οι εξωτερικοί κόμβοι βρίσκονται στο ίδιο επίπεδο.**

Θεωρήσεις

- Κάθε κόμβος του δέντρου είναι ένα block στο δίσκο

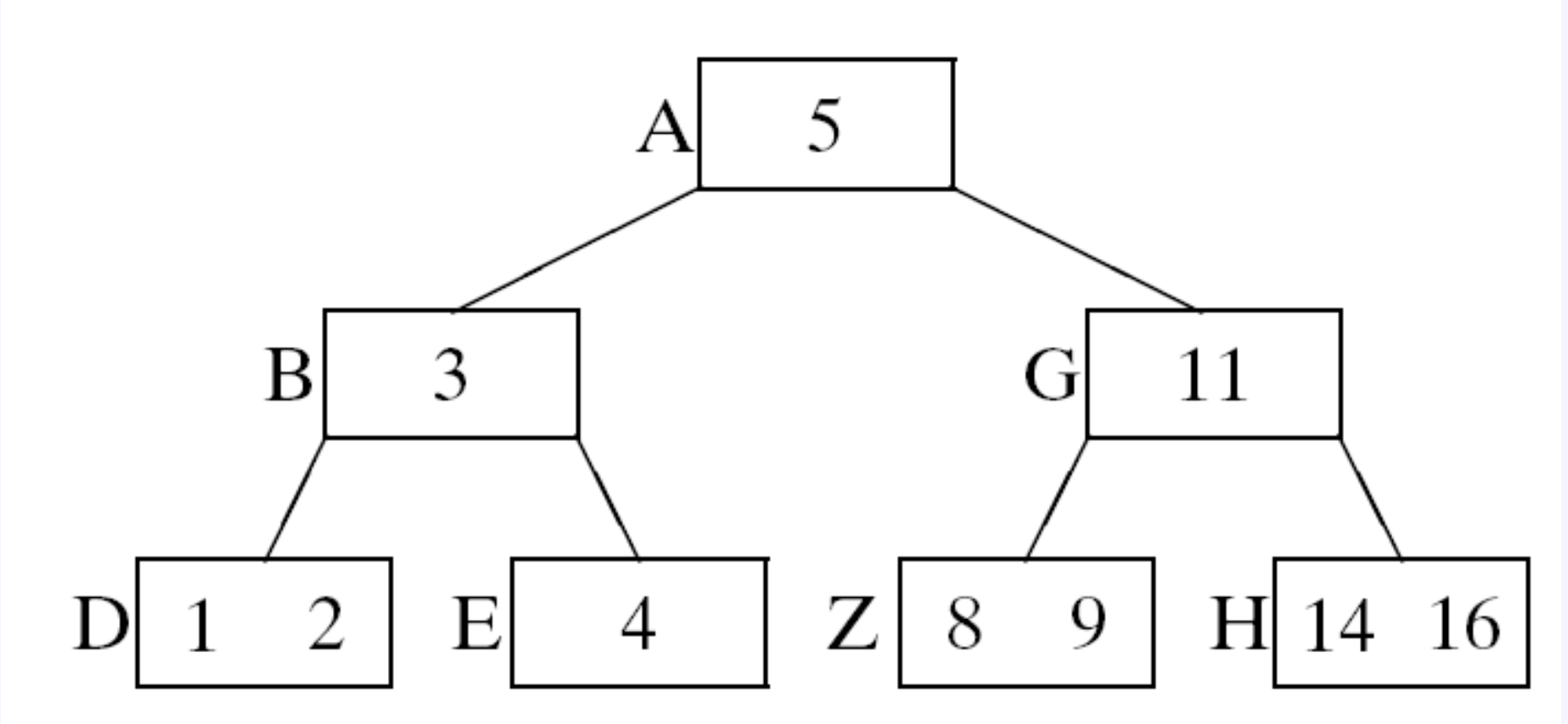
- Ισοζυγισμένο: όλοι οι κόμβοι-φύλλα στο ίδιο επίπεδο

B-Δένδρο


Κόμβος του B-δένδρου
P1
K1
Pr1
P2
…
Ki-1
Pri-1
Pi
Ki
Pri
…
Kq-1
Prq-1
Pq
Δείκτης δεδομένων
Δείκτης δεδομένων
Δείκτης δεδομένων
Δείκτης δένδρου
X
X
X
X<K1
Ki-1<X<Ki
Kq-1<X

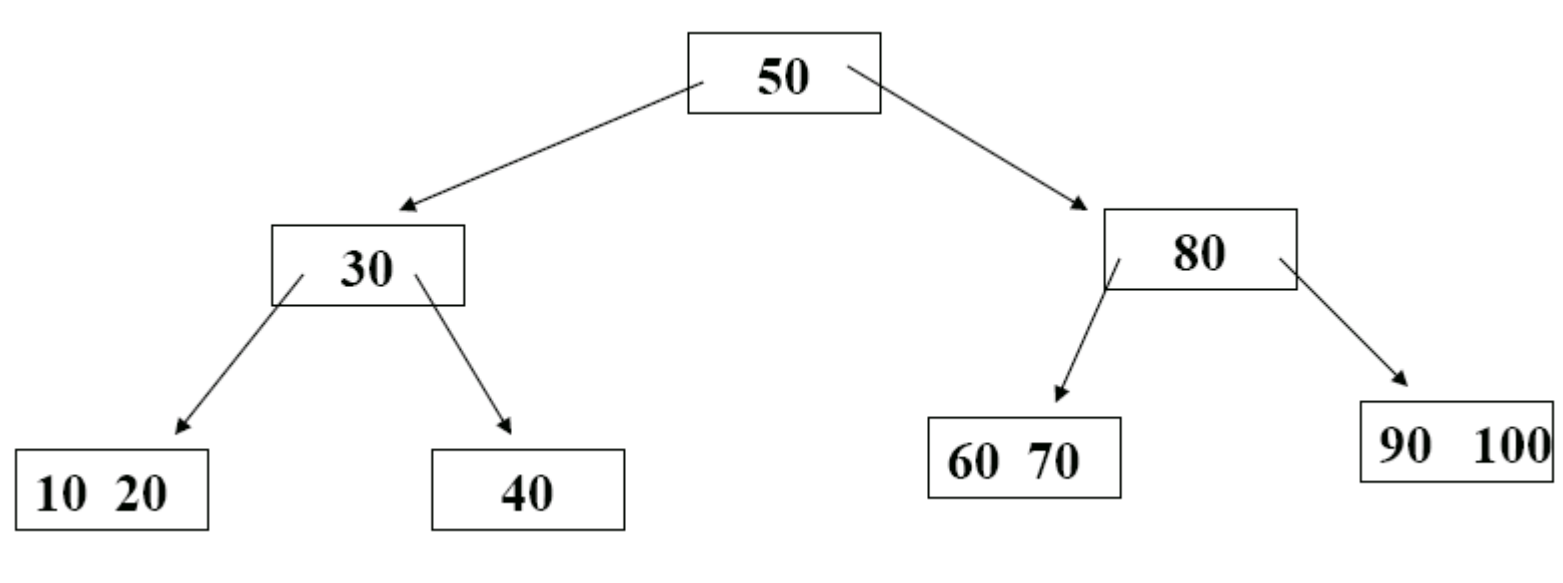
B-Δένδρο

- Παράδειγμα B-δένδρου τάξης m=3 ή βαθμού d=1.



Β-Δένδρο

- Το δένδρο τάξεως 3 που σημαίνει ότι κάθε κόμβος έχει 1 ή 2 κλειδιά και 2 ή 3 παιδιά (δείκτες δενδρικούς).



B-Δένδρο

- **Παράγοντας χρησιμοποίησης χώρου**, U, ορίζεται **ο λόγος του αριθμού των αποθηκευμένων κλειδιών προς το σύνολο των διαθέσιμων θέσεων**.
- Γενικώς ισχύει: Umin= 50% και Umax= 100%.
- Από το δεύτερο ορισμό ισχύει:

$$U = \frac{n}{2d \times nod}$$

- όπου n ο αριθμός των κλειδιών, nod ο αριθμός των κόμβων, d ο βαθμός του δένδρου.

B-Δένδρο

- Η ελάχιστη τιμή του αριθμού των κόμβων προκύπτει από:

$$nod_{min} = \frac{n}{2d}$$

- Η μέγιστη τιμή του αριθμού των κόμβων προκύπτει από:

$$nod_{max} = \frac{n}{2d \times U_{min}}$$

Β-Δένδρο

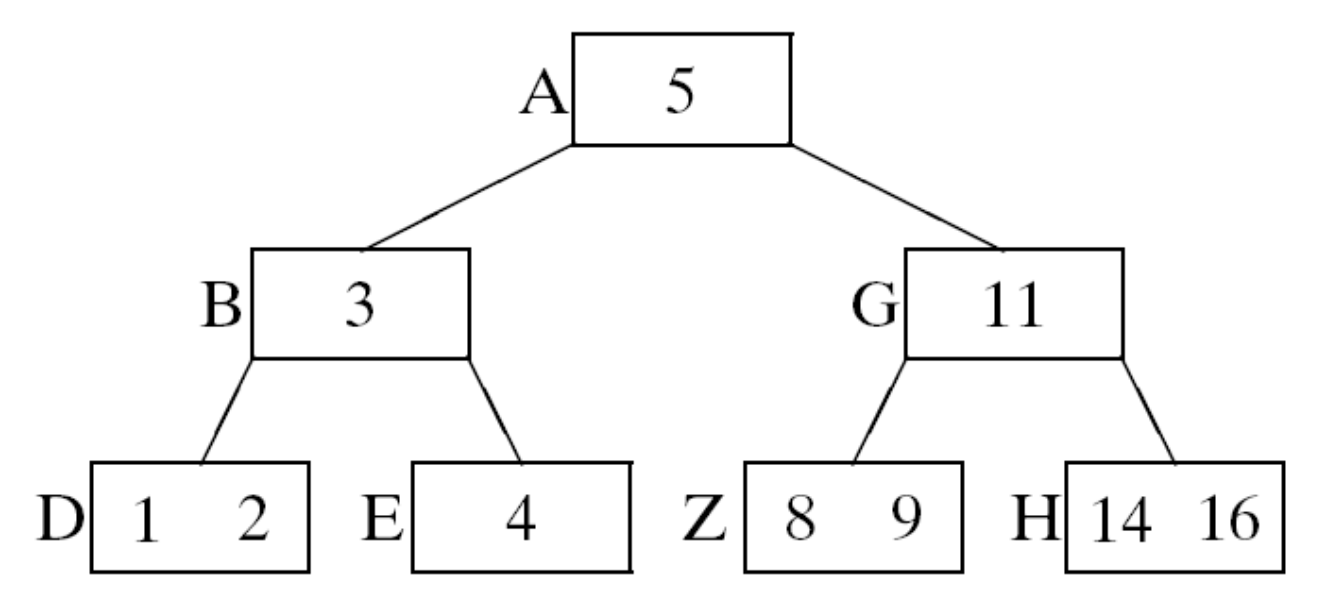
- Ο ελάχιστος αριθμός των κλειδιών ενός Β-δένδρου τάξης m και ύψους h είναι:

$$n = 2 \times \left\lceil \frac{m}{2} \right\rceil^{h-1} - 1$$

ενώ ο μέγιστος είναι:

$$n = m^h - 1$$

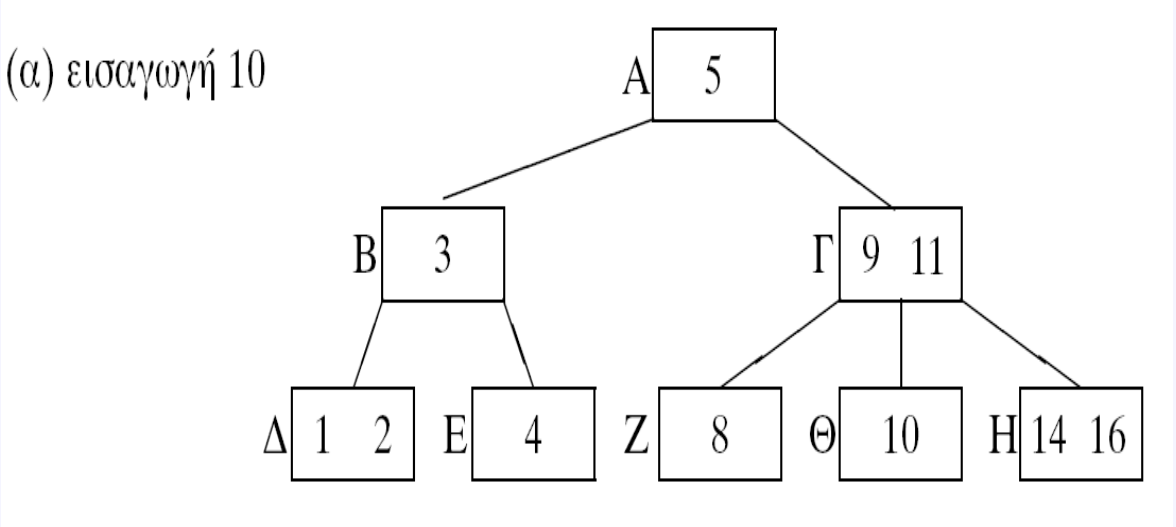
B-Δένδρο - Εισαγωγή



- Στο δένδρο πρέπει να εισαχθεί το κλειδί 10.
- Αν εισαχθεί στον κόμβο Z παραβιάζεται ο ορισμός, γιατί ο μέγιστος αριθμός κλειδιών είναι 2 (2d).

Β-Δένδρο - Εισαγωγή

- ***Όταν τα κλειδιά είναι περισσότερα από το επιτρεπτό όριο κλειδιών (υπερχείλιση) δημιουργείται νέος κόμβος. Το μεσαίο κλειδί ανεβαίνει στο πατρικό κόμβο.***
- Το μεσαίο από τα κλειδιά 8,9,10 (στην περίπτωση αυτή το 9) ανεβαίνει στον πατρικό κόμβο.
- Το κλειδί 8 επανα-αποθηκεύεται στον κόμβο Ζ, ενώ το 10 στο νέο κόμβο Θ.

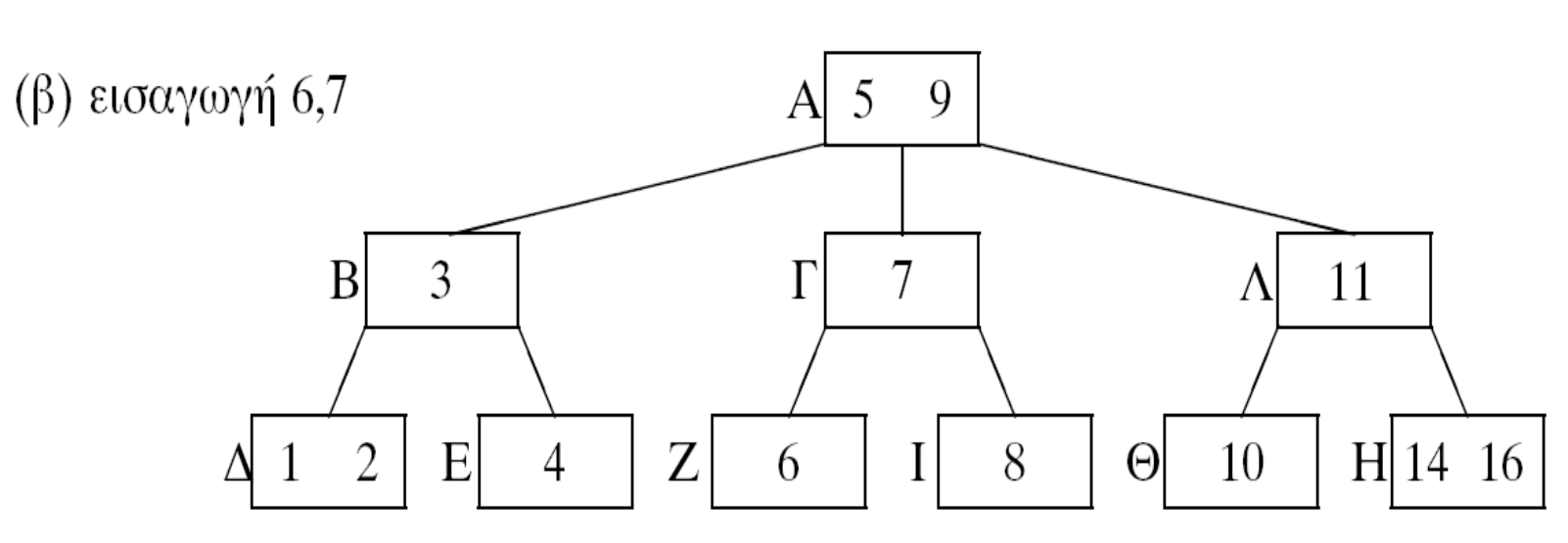


Β-Δένδρο - Εισαγωγή

- Το 6 τοποθετείται στον κόμβο Ζ του προηγούμενου σχήματος.
- Το 7 τοποθετείται στον κόμβο Ζ (6,8)➔ παραβίαση ορισμού ➔ δημιουργία κόμβου Ι.

  Το 7 προχωρεί ένα επίπεδο στο κόμβο Γ (9,11) ➔ παραβίαση ορισμού ➔ δημιουργία κόμβου Λ

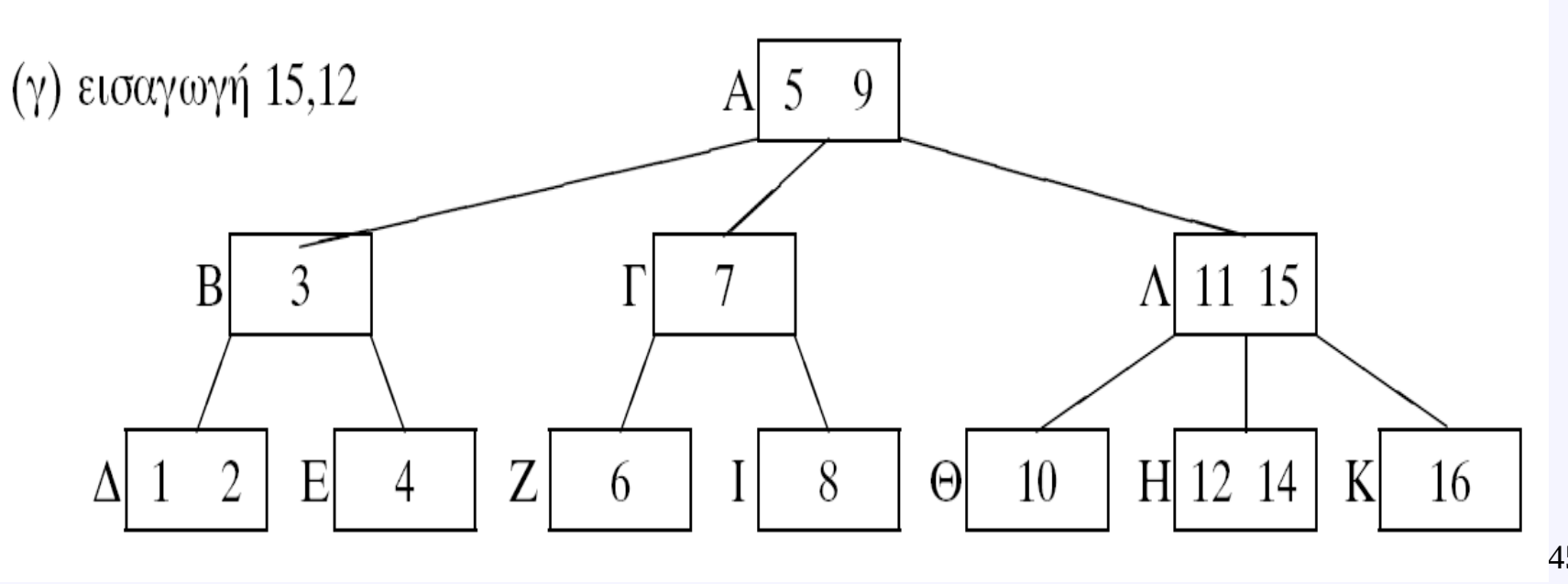
  Το 9 προχωρεί ένα επίπεδο στο κόμβο Α.



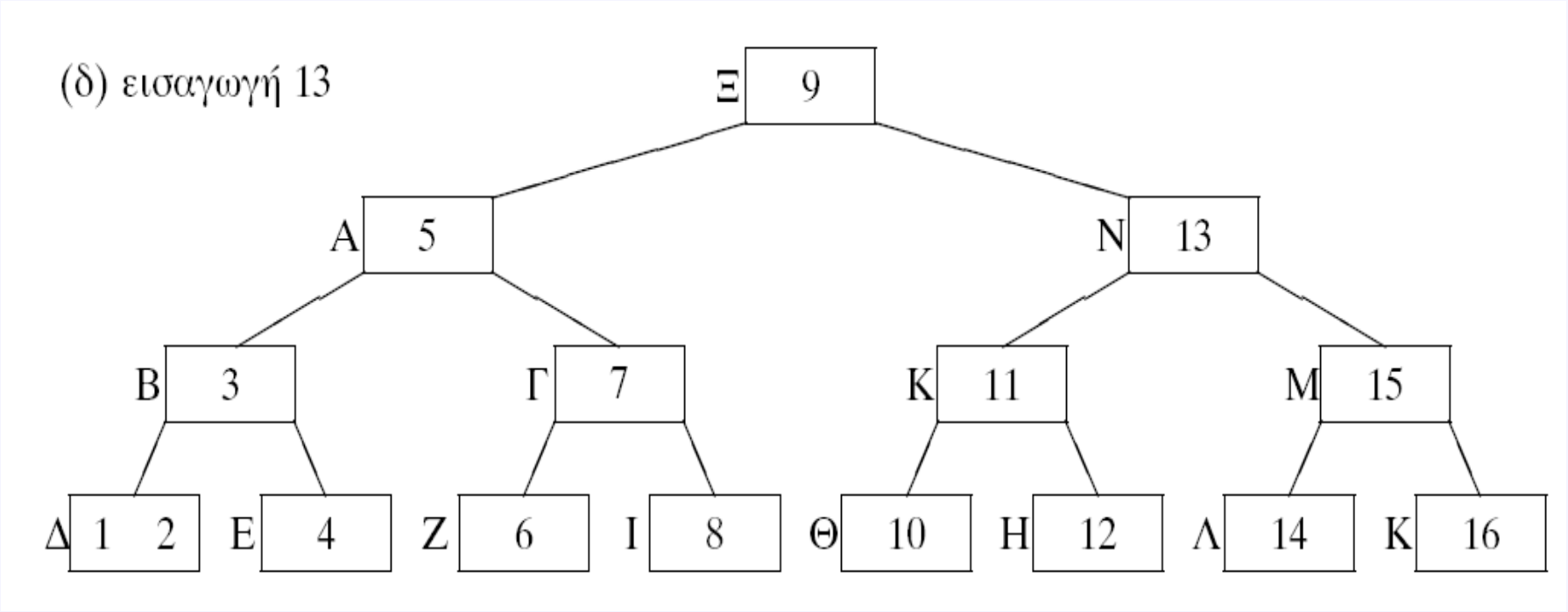
Β-Δένδρο - Εισαγωγή

- Το 15 τοποθετείται στον κόμβο Η (14,16)➔παραβίαση ορισμού

  Το 15 προχωρεί ένα επίπεδο στο κόμβο Λ (11) ➔ δημιουργείται ο κόμβος Κ
- Το 12 τοποθετείται στο κόμβο Η.



Β-Δένδρο - Εισαγωγή

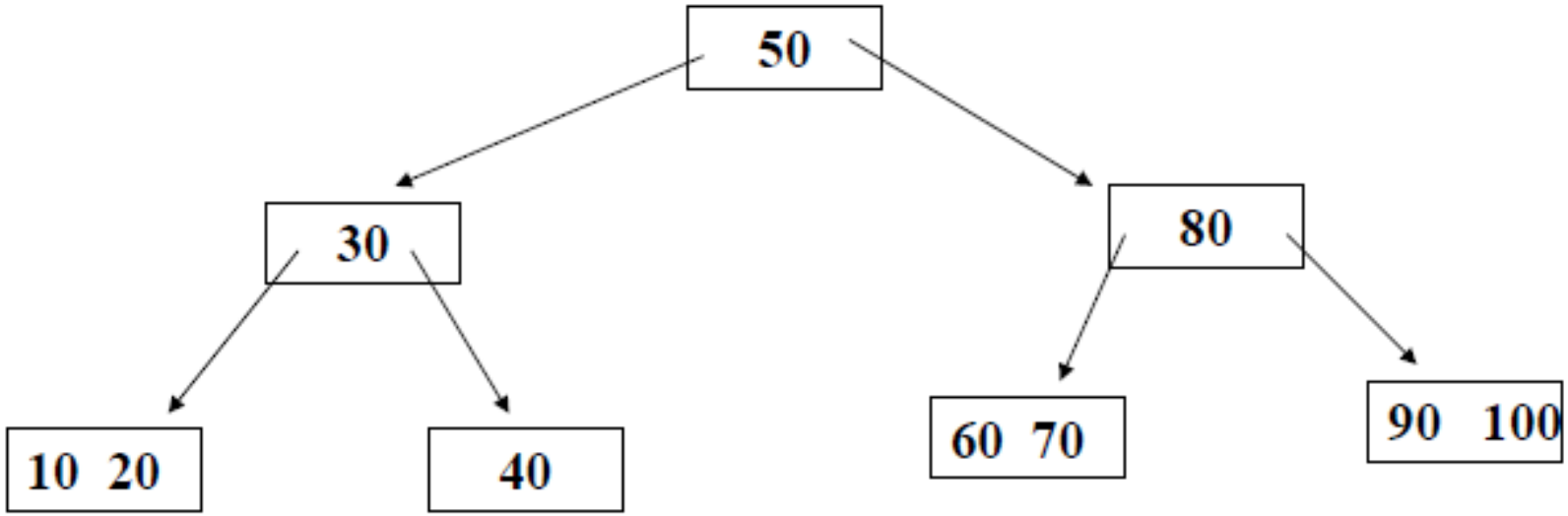


B-Δένδρο - Εισαγωγή

Εισαγωγή του κλειδιού X στο B-δένδρο

- **Σαρώνουμε το B-δένδρο μέχρι που ή θα βρούμε την τιμή X ήθα φθάσουμε σε ένα φύλο που θα έπρεπε να βρίσκεται η τιμή X.**
- **Αν ο κόμβος αυτός έχει έχει λιγότερα από p-1 κλειδιά τότε προσθέτουμε το κλειδί στον κόμβο αυτό και τελείωσε η εισαγωγή.**
- **Αν ο κόμβος έχει ήδη p κλειδιά τότε δεν μπορεί να προστεθεί στον κόμβο αυτό. Στην περίπτωση αυτή χωρίζουμε τα p κλειδιά ($K_1<K_2<K_3<\ldots<K_q$)σε δύο κόμβους όπου ο πρώτος έχει τα πρώτα $\lceil q/2 \rceil$ ο δεύτερος τα τελευταία $\lceil q/2 \rceil$ και το μεσαίο στοιχείο ανεβαίνει στον κόμβο γονέα σαν διαχωριστικό.**
- **Αν ο κόμβος γονέας έχει χώρο τότε τελειώσαμε. Αν δεν έχει τότε χωρίζεται με τον ίδιο τρόπο και αυτός και προχωράμε προς τη ρίζα. Στην χειρότερη περίπτωση θα διασπασθεί και η ρίζα και θα ανέβουμε ένα επίπεδο (θα ψηλώσει το δένδρο).**

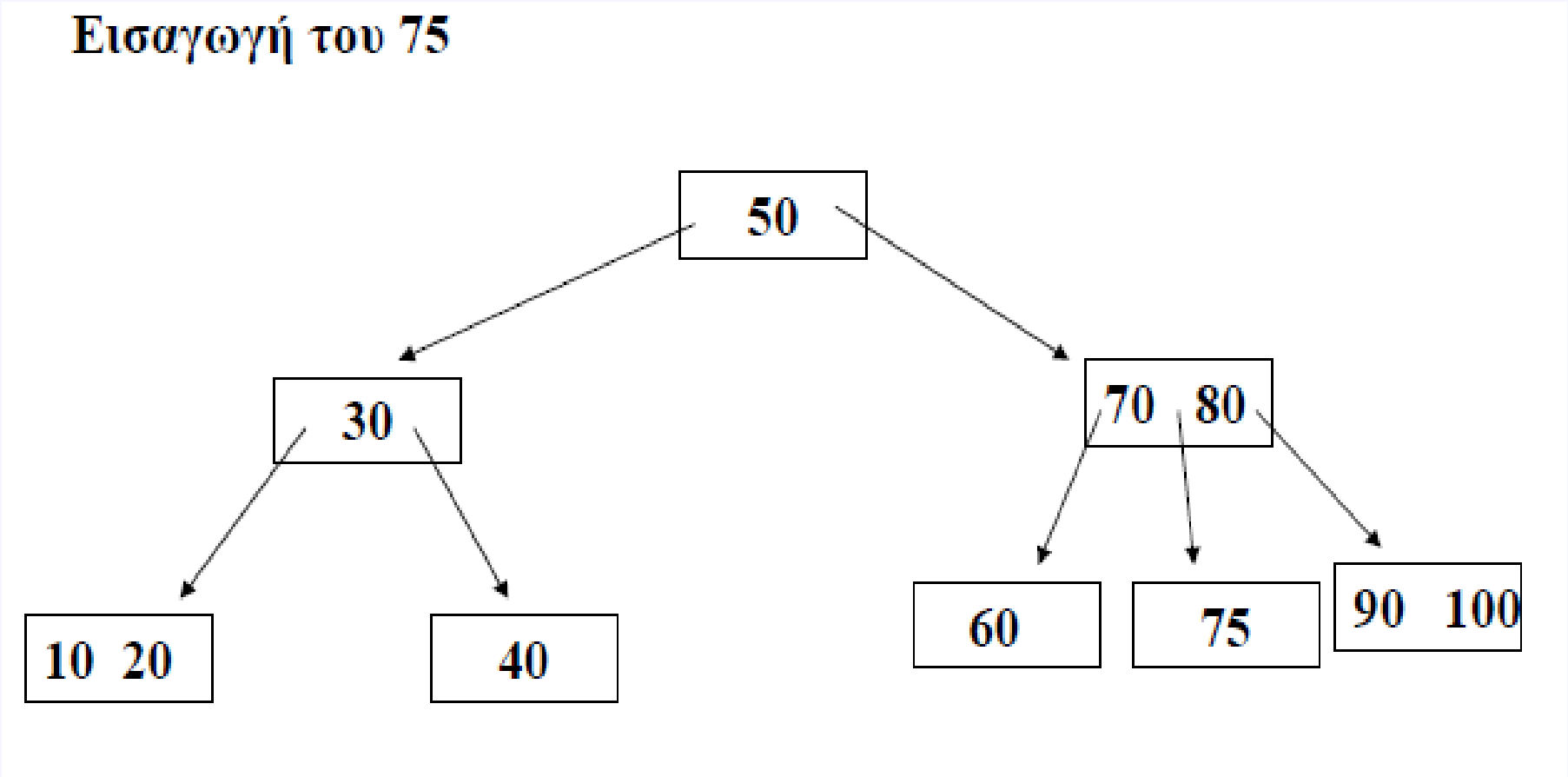
Β-Δένδρο - Εισαγωγή



Το δένδρο τάξεως 3 που σημαίνει ότι κάθε κόμβος έχει 1 ή 2 κλειδιά και 2 ή 3 παιδιά

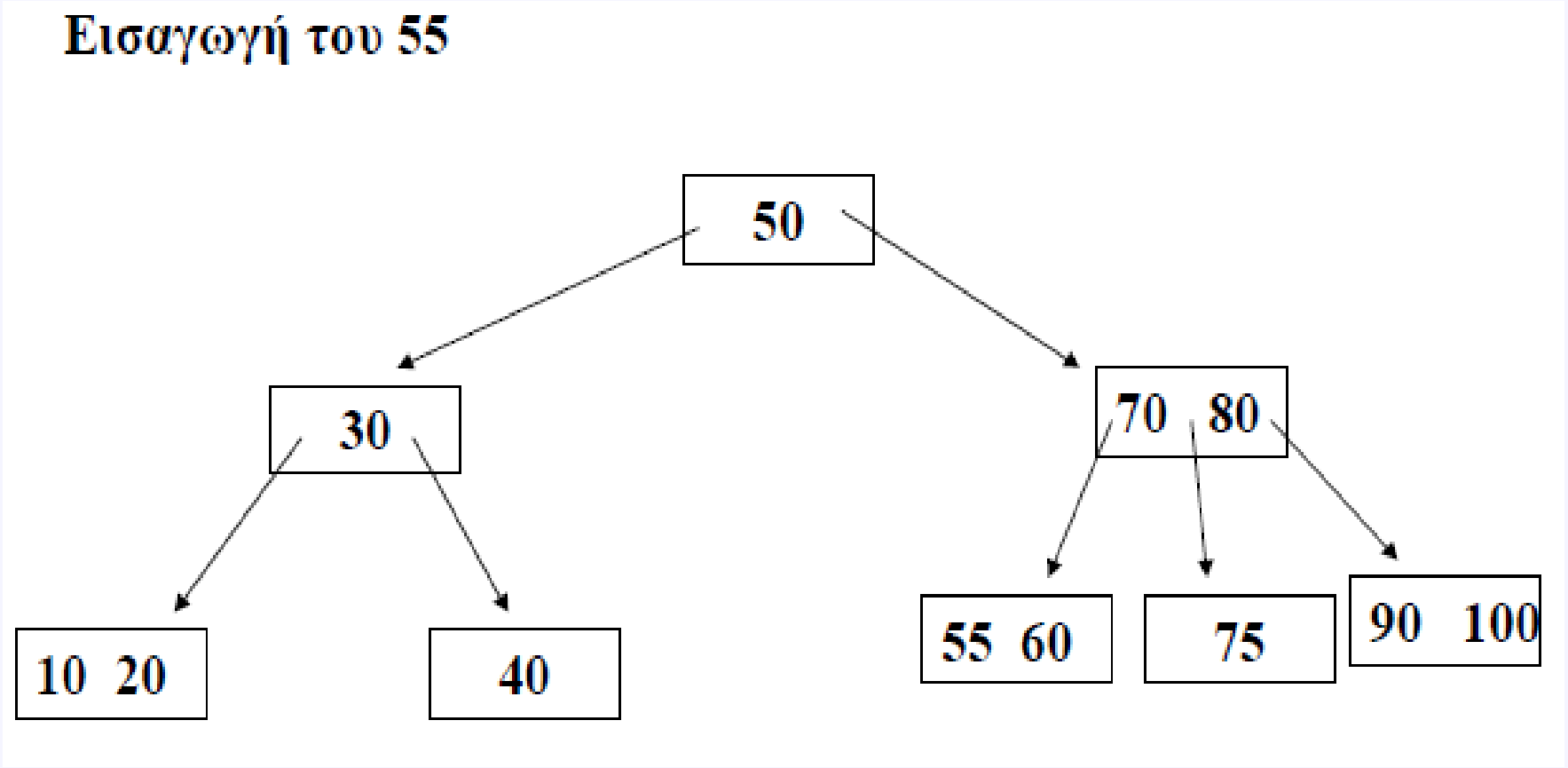
- Στη συνέχεια εισάγουμε το 75

Β-Δένδρο - Εισαγωγή



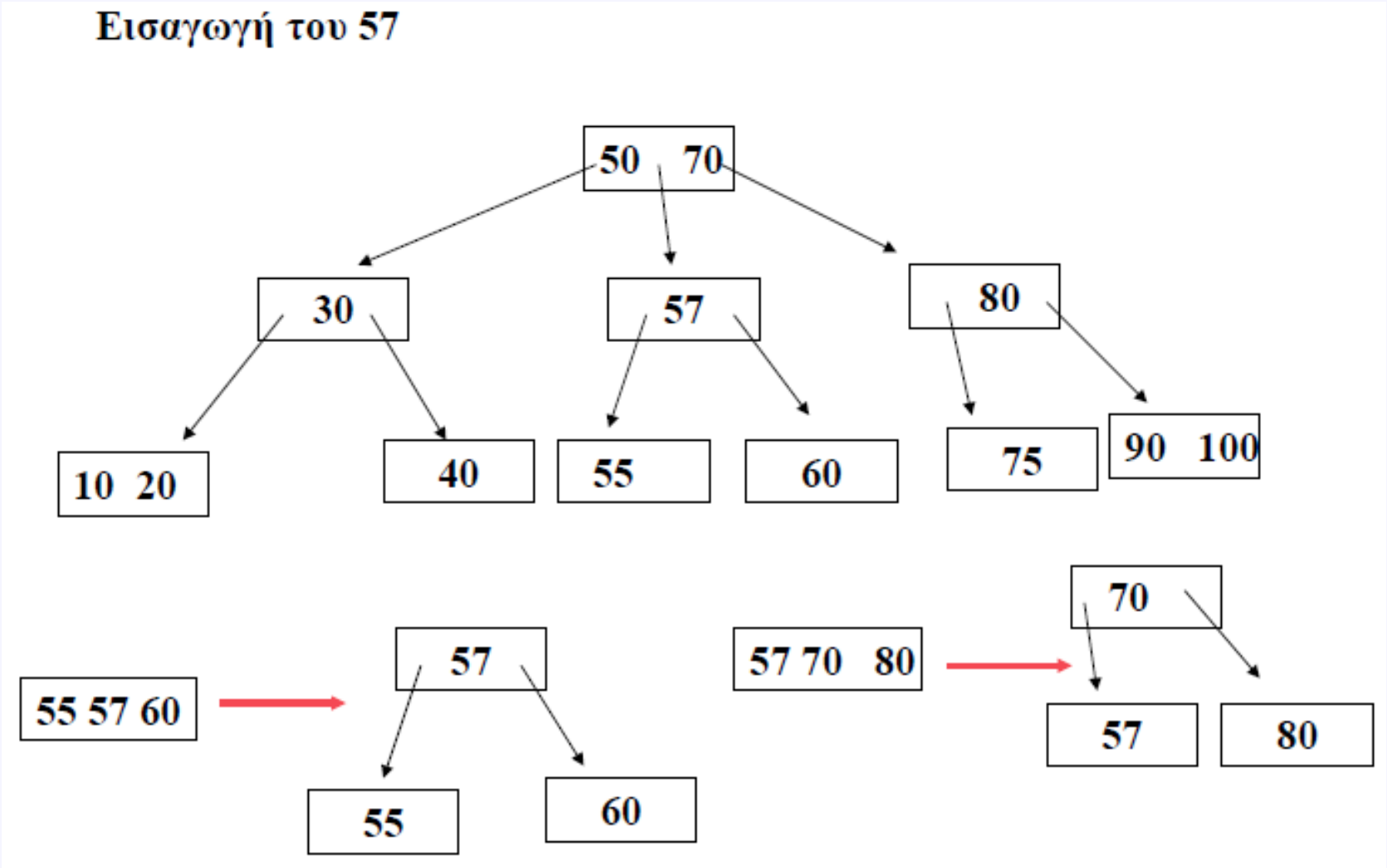
- Στη συνέχεια εισάγουμε το 55

Β-Δένδρο - Εισαγωγή



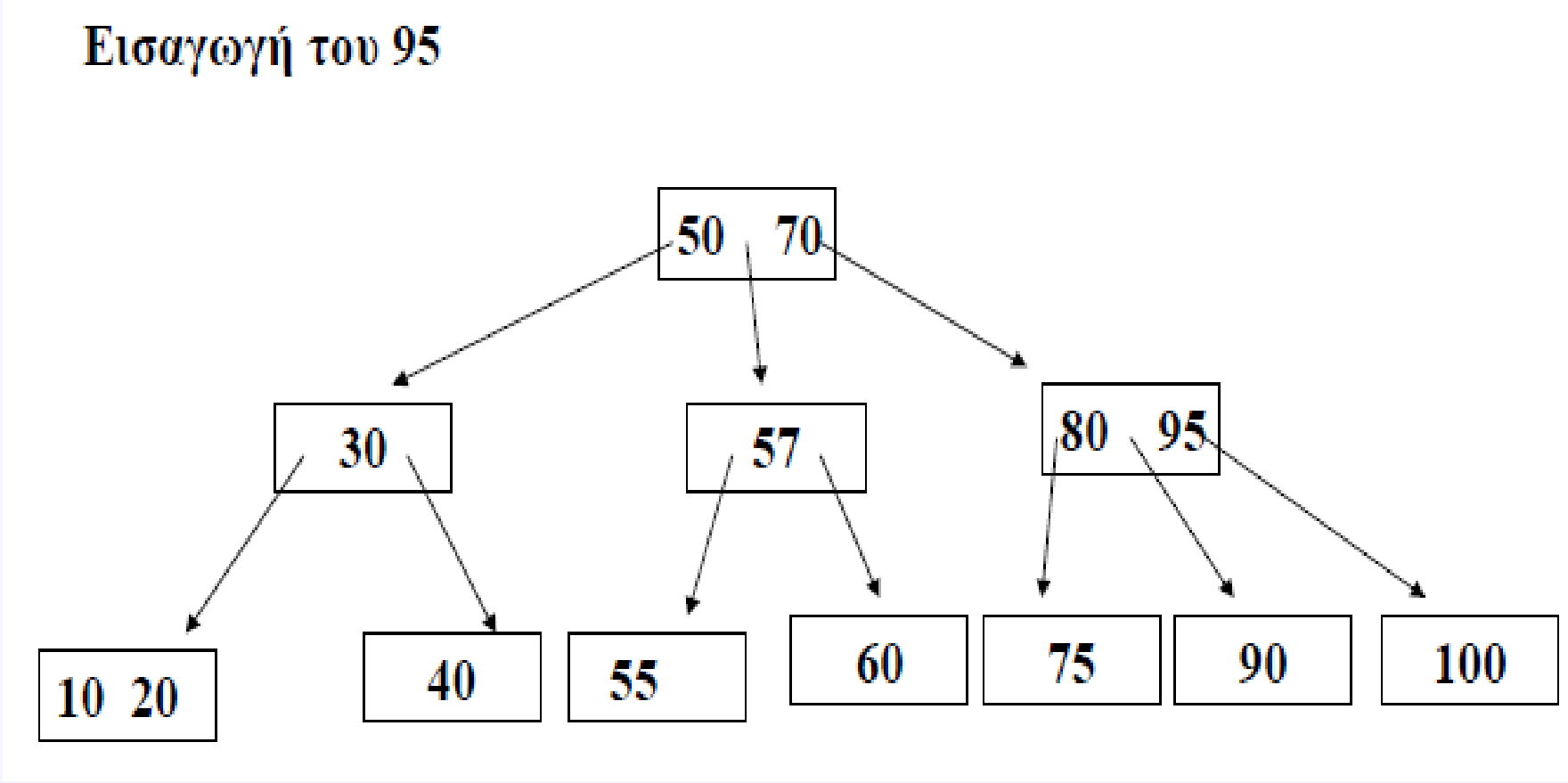
- Στη συνέχεια εισάγουμε το 57

B-Δένδρο - Εισαγωγή



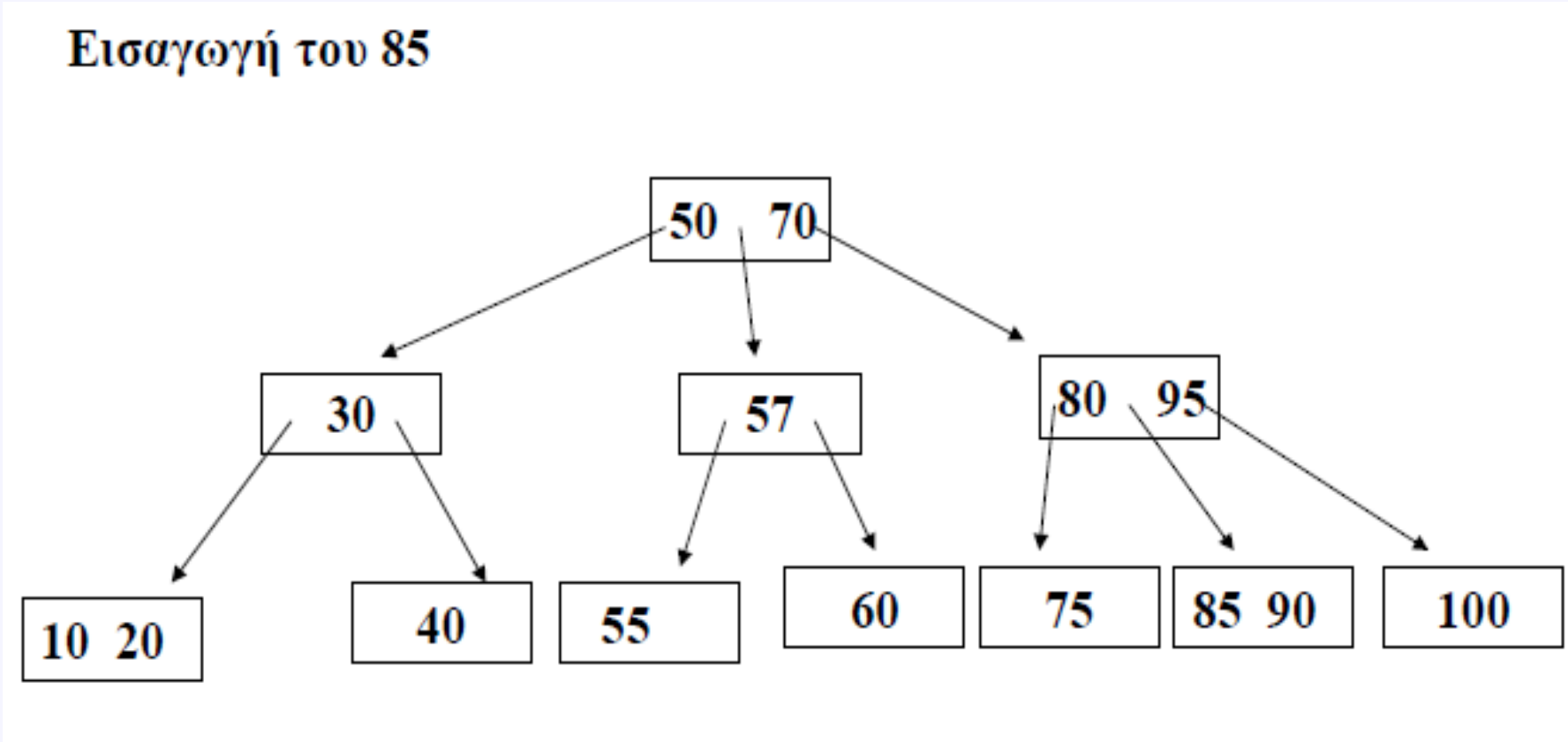
- Στη συνέχεια εισάγουμε το 95

Β-Δένδρο - Εισαγωγή



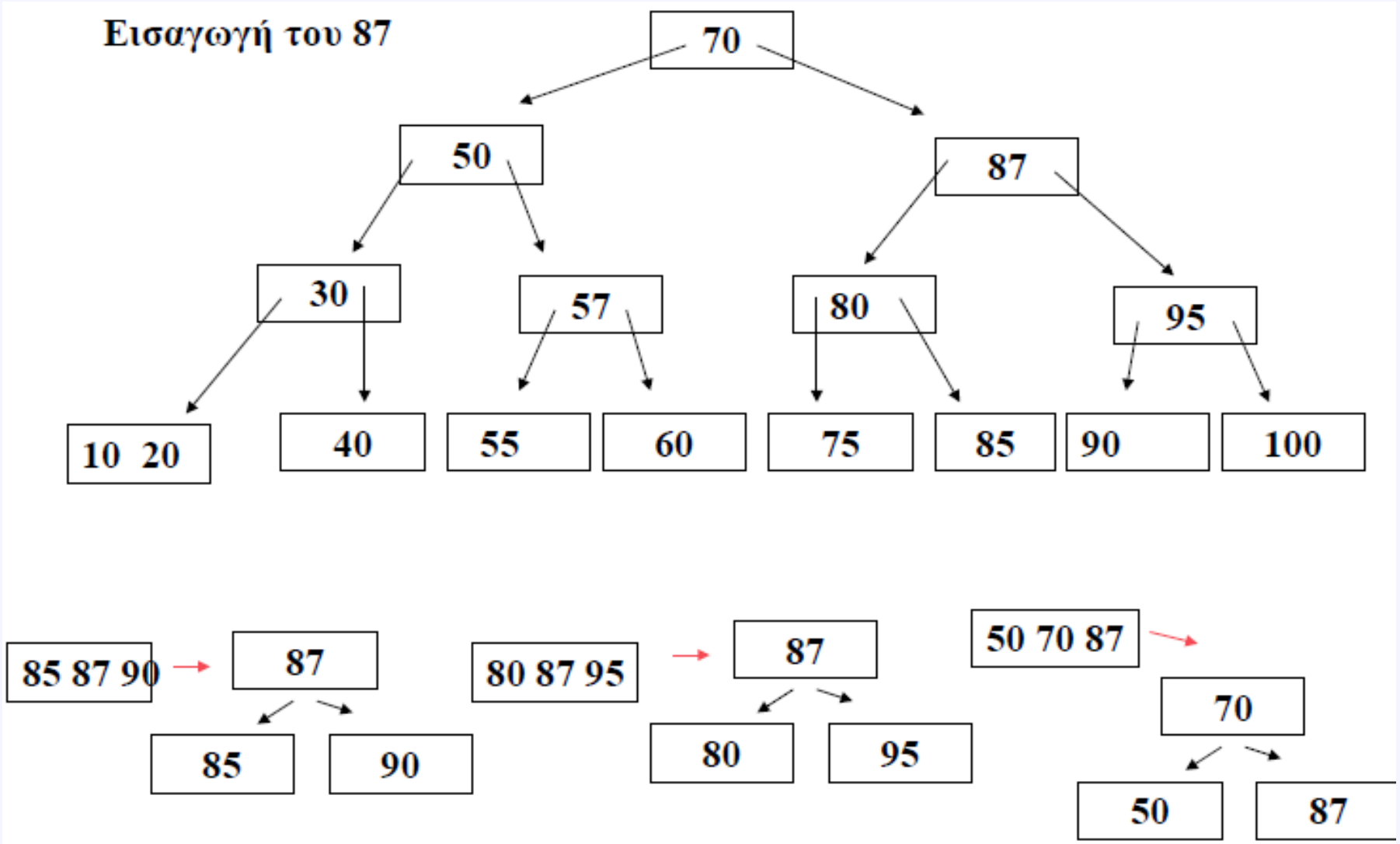
- Στη συνέχεια εισάγουμε το 85

Β-Δένδρο - Εισαγωγή



- Στη συνέχεια εισάγουμε το 87

Β-Δένδρο - Εισαγωγή



Β-Δένδρο - Διαγραφή

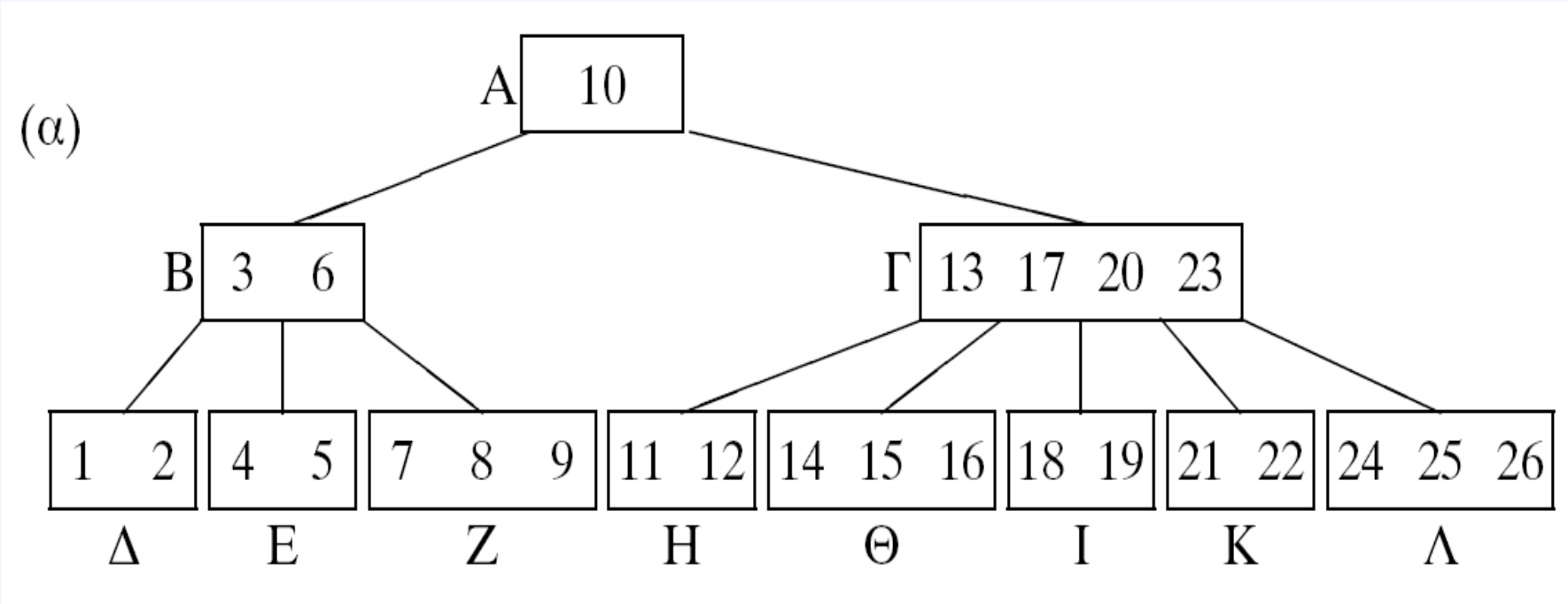
- Αν το υπό διαγραφή κλειδί βρίσκεται σε **εξωτερικό κόμβο** και παραβιάζεται ο ορισμός του Β-δένδρου τότε:
  - ***ελέγχεται αν ο διπλανός κόμβος (δεξιά και αριστερά) έχει περισσότερα από τον ελάχιστο αριθμό κλειδιών και γίνεται ανακατανομή των κλειδιών.***
  - *ενημερώνεται ο γονέας ώστε να αντανακλά τη νέα κατανομή*.
- Όταν οι διπλανοί κόμβοι έχουν τον ελάχιστο αριθμό κλειδιών, τότε γίνεται συγχώνευση κόμβων.
  - ***Η συγχώνευση αφορά 2 κόμβους από το κατώτερο επίπεδο, και***
  - ***Το κλειδί από το πατρικό κόμβο που βρίσκεται ανάμεσά τους***

Β-Δένδρο - Διαγραφή

- Αν το υπό διαγραφή κλειδί βρίσκεται σε **εσωτερικό κόμβο**, τότε:
  - ***εντοπίζεται το λεξικογραφικά αμέσως προηγούμενο κλειδί για να καταλάβει τη θέση του διαγραφόμενου***, και
  - η διαδικασία συνεχίζει σύμφωνα με τα προηγούμενα.

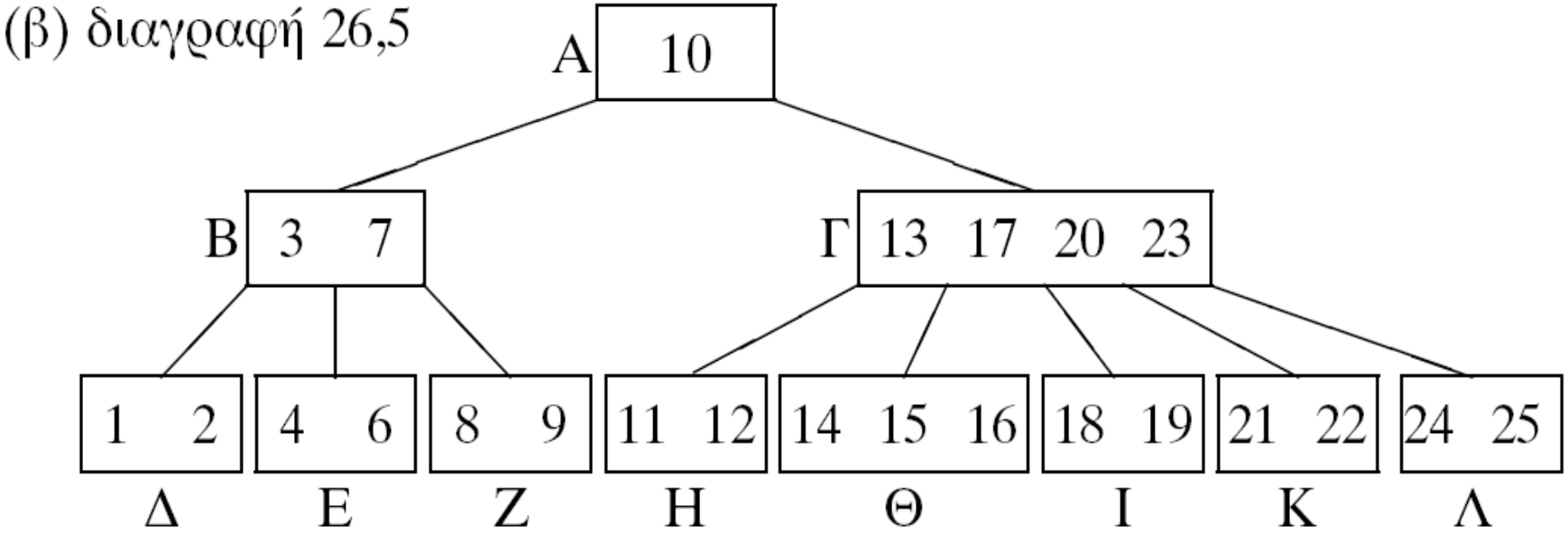
Β-Δένδρο - Διαγραφή

- Έστω το Β-δένδρο 5ης τάξης – Διαγραφή κλειδιού που ανήκει σε εξωτερικό κόμβο



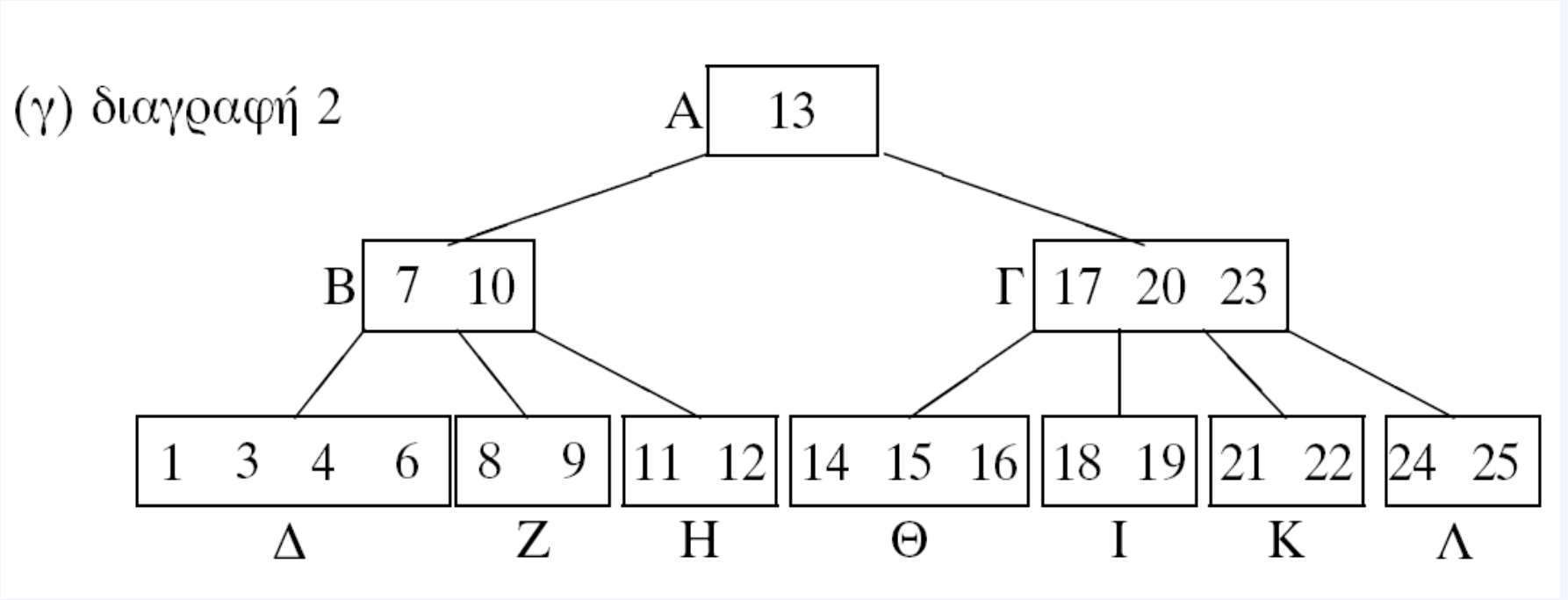
Β-Δένδρο - Διαγραφή

- Διαγραφή 26 – Απλά διαγράφετε από το κόμβο Λ
- Διαγραφή 5 ➔ Παραβιάζεται ο ορισμός αφού ο κόμβος Ε θα έχει ένα κλειδί. Ελέγχεται αν ο γειτονικός κόμβος Ζ έχει > 2 κλειδιά. Τότε άνοδος του 7 στο Β και κάθοδος του 6 στο Ε.



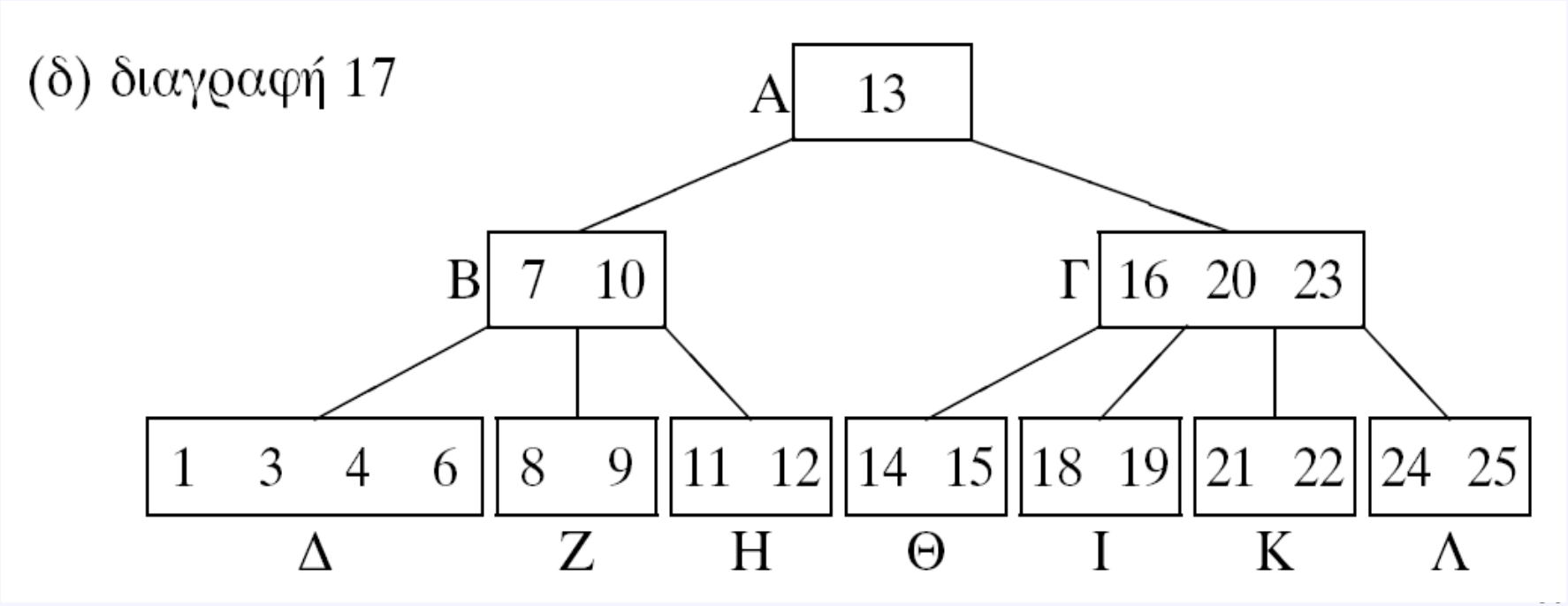
Β-Δένδρο - Διαγραφή

- Διαγραφή 2 – Συγχώνευση των κλειδιών 1 (Δ), 3 (Β), 4 και 6 (Ε) και αποθήκευση στον κόμβο Δ. Ο κόμβος Ε αποδίδεται στο σύστημα
  - Το Β όμως έχει 1 κλειδί. Ελέγχεται αν ο δεξιός γείτονας έχει > 2 κλειδιά ➔ 13 άνοδος στο Α και 10 κάθοδος στο Β.



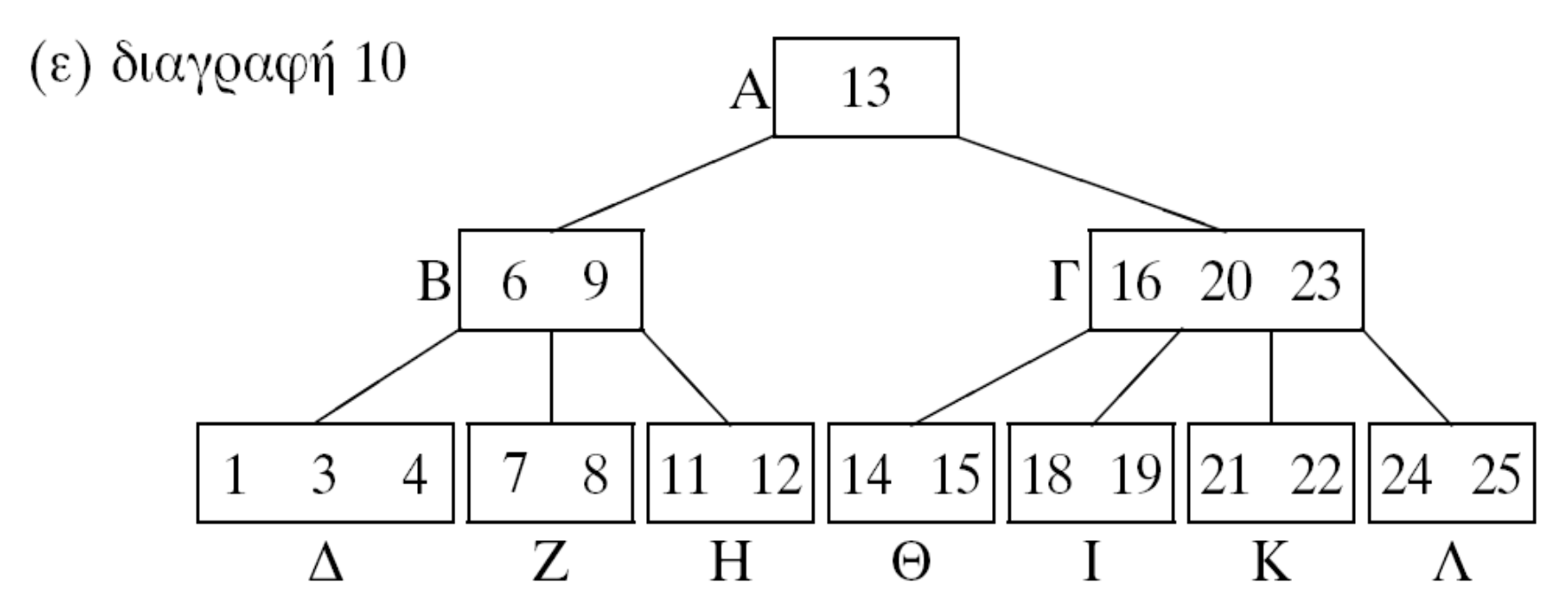
Β-Δένδρο - Διαγραφή

- Διαγραφή σε εσωτερικό κόμβο
- Διαγραφή 17 – Ελέγχεται αν ο κόμβος Θ έχει > 2 κλειδιά. Το 16 ανεβαίνει στο κόμβο Γ.



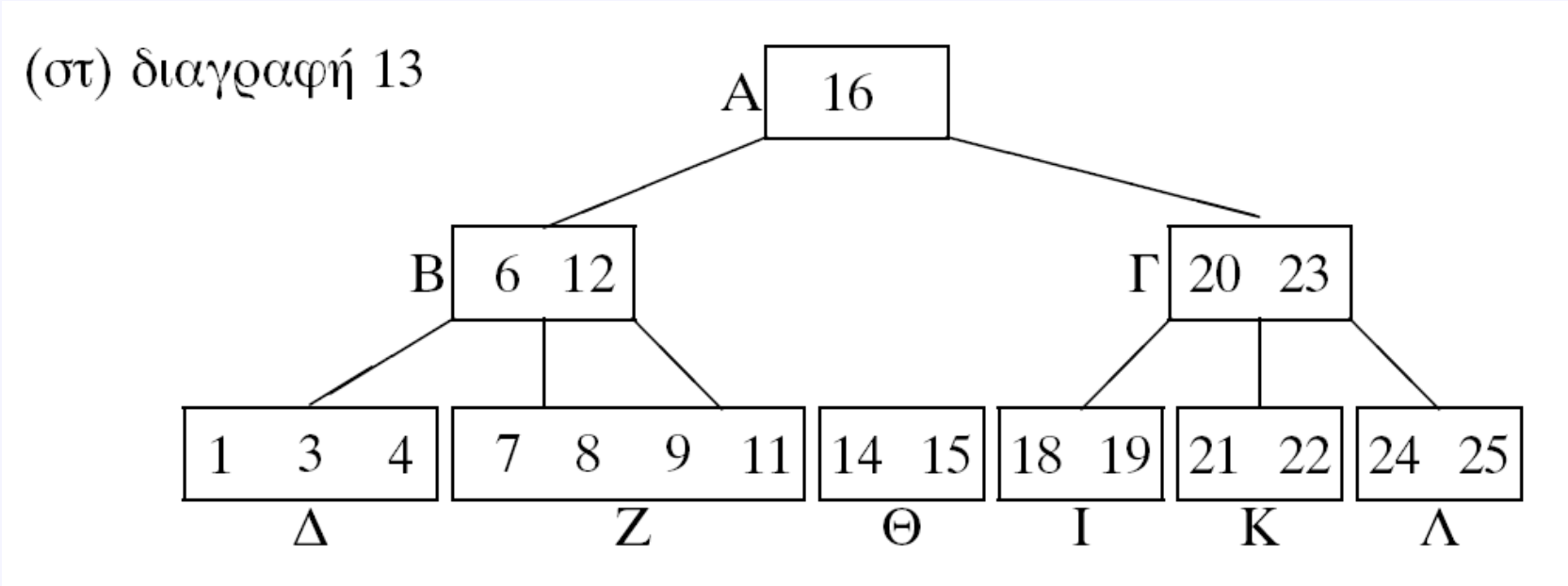
Β-Δένδρο - Διαγραφή

- Διαγραφή 10 – Άνοδος του 9. Ο κόμβος Ζ έχει όμως μόνο ένα κλειδί. Ο δεξιός κόμβος δεν μπορεί να δανείσει κάποιο κλειδί.
  - Το 6 από το κόμβο Δ ανεβαίνει και το 7 κατεβαίνει.



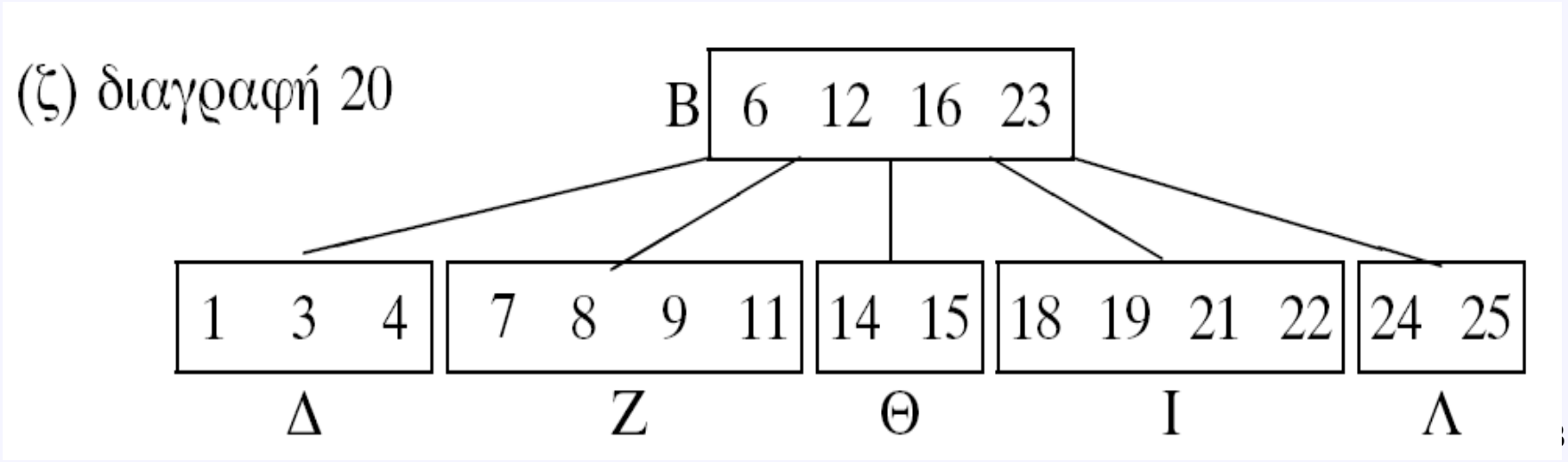
Β-Δένδρο - Διαγραφή

- Διαγραφή 13 – Το αμέσως μικρότερο κλειδί (12) λαμβάνει τη θέση του. Τότε ο κόμβος Η μένει με 1 κλειδί.
  - Ο κόμβος Ζ, Η και το κλειδί μεταξύ των δύο κόμβων (9) συγχωνεύονται στο κόμβο Ζ.
  - Ο κόμβος Β έχει ένα κλειδί (6). Από το κόμβο Γ ανέρχεται το 16 και το 12 κατέρχεται στο κόμβο Β.



Β-Δένδρο - Διαγραφή

- Διαγραφή 20 ➔ Ελάττωση του ύψους του δένδρου
  - Το κλειδί 19 ανεβαίνει στο κόμβο Γ.
  - Ο κόμβος Ι έχει μόνο ένα κλειδί. Ο κόμβος Ι, Κ και το 19 συγχωνεύονται στο κόμβο Ι.
  - Ο κόμβος Γ έχει ένα κλειδί (23). Ο αριστερός αδερφός και η ρίζα έχουν το ελάχιστο αριθμό κλειδιών ➔ συγχώνευση των κόμβων Α, Β και Γ.



Β-Δένδρο

Όμως έχουμε $\mathbf{h \geq \log_m (N+1)}$

$\mathbf{Άρα\ \log_m (N+1) \leq h \leq \log_{\lceil m/2 \rceil} ((N+1)/2)}$

B-Δένδρο

Μήκος κλειδιού V=9 byte
Μέγεθος block B=512byte
Μήκος δείκτη υποδένδρου P=6byte
Μήκος δείκτη εγγραφής Pr=7byte
Για την τάξη p του δένδρου πρέπει να ισχύει

$$(p*P)+((p-1)*(Pr+V)) \leq B$$

Δηλαδή στην περίπτωση μας

$$(p*6)+((p-1)*(7+9)) \leq 512$$
$$22*p \leq 521$$
$$p=23$$