Δομές Δεδομένων

Αναδρομή και Δέντρα

Δημήτρης Μιχαήλ

Τμήμα Πληροφορικής και Τηλεματικής
Χαροκόπειο Πανεπιστήμιο

Αναδρομή

Ένας αναδρομικός αλγόριθμος είναι ένας αλγόριθμος που λύνει ένα πρόβλημα λύνοντας ένα ή περισσότερα μικρότερα στιγμιότυπα του ίδιου προβλήματος

Παραγοντικό σε C

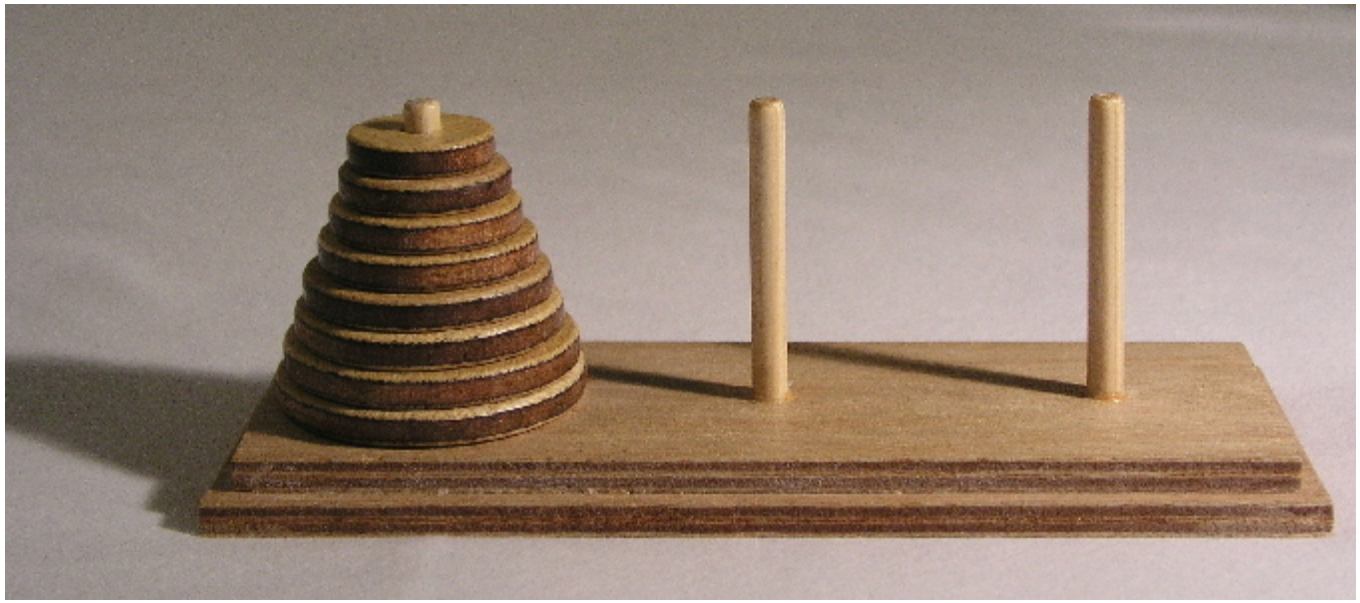
θυμηθείτε τον τύπο για το παραγοντικό:

$$n! = \begin{cases} 1 & \text{εάν n = 0,} \\ n \cdot (n-1)! & \text{διαφορετικά} \end{cases}$$

σε γλώσσα C μπορούμε να γράψουμε:

```
1  int factorial( int n )
2  {
3      if ( n == 0 )
4          return 1;
5      return n * factorial( n - 1 );
6  }
```

Οι πύργοι του Ανόι



- τρεις πάσσαλοι Α,Β,Γ και N δίσκοι
- μετακίνηση μόνο ενός δίσκου τη φορά
- όχι μεγαλύτερος δίσκος πάνω από μικρότερο
- θέλουμε οι δίσκοι να πάνε από τον πάσσαλο Α στον Γ

Οι πύργοι του Ανόι

Αναδρομική λύση

- hanoi(N, start, target, temp): μεταφορά N δίσκων από τον πάσσαλο start στο πάσσαλο target χρησιμοποιώντας τον πάσσαλο temp.
- shift(start, target): μετακίνηση του επάνω δίσκου από τον πάσσαλο start στον πάσσαλο target

Οι πύργοι του Ανόι

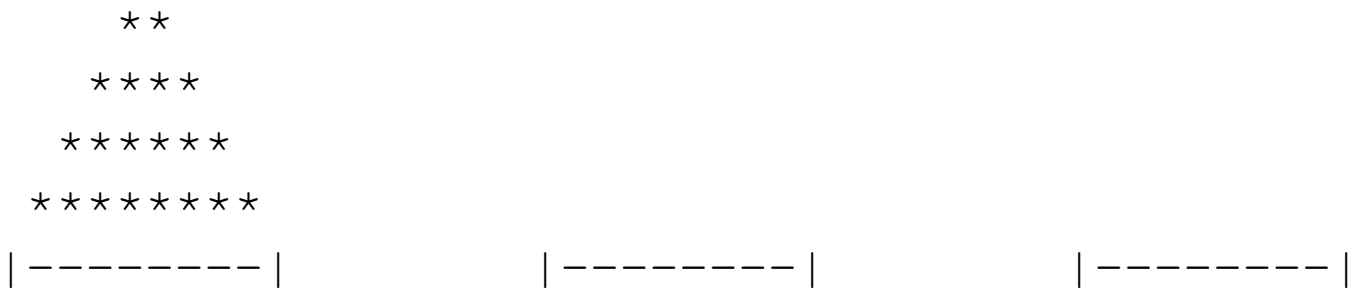
Αναδρομική λύση

- hanoi(N, start, target, temp): μεταφορά N δίσκων από τον πάσσαλο start στο πάσσαλο target χρησιμοποιώντας τον πάσσαλο temp.
- shift(start, target): μετακίνηση του επάνω δίσκου από τον πάσσαλο start στον πάσσαλο target

```
1  void hanoi( int N, int start, int target, int temp )
2  {
3      if ( N == 1 )  {
4          shift( start, target );
5          printf( "%d -> %d\n", start, target );
6          return;
7      }
8      hanoi( N-1, start,   temp, target );
9      hanoi(   1, start, target,   temp );
10     hanoi( N-1,  temp, target,  start );
11 }
```

Οι πύργοι του Ανόι

Αναδρομική λύση



Οι πύργοι του Ανόι

Αναδρομική λύση

Οι πύργοι του Ανόι

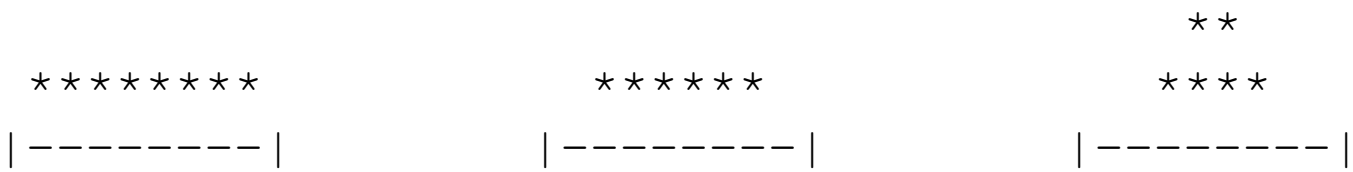
Αναδρομική λύση

Οι πύργοι του Ανόι

Αναδρομική λύση

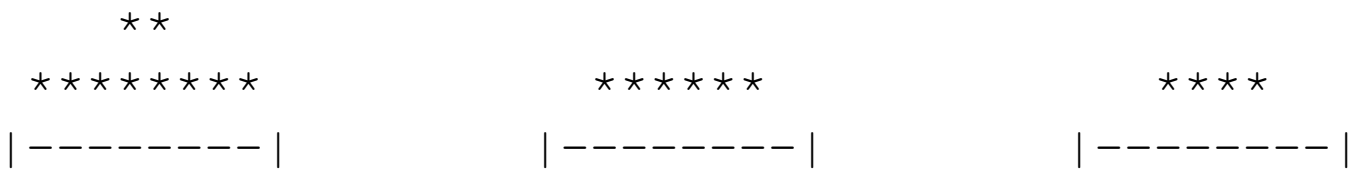
Οι πύργοι του Ανόι

Αναδρομική λύση



Οι πύργοι του Ανόι

Αναδρομική λύση



Οι πύργοι του Ανόι

Αναδρομική λύση

Οι πύργοι του Ανόι

Αναδρομική λύση

Οι πύργοι του Ανόι

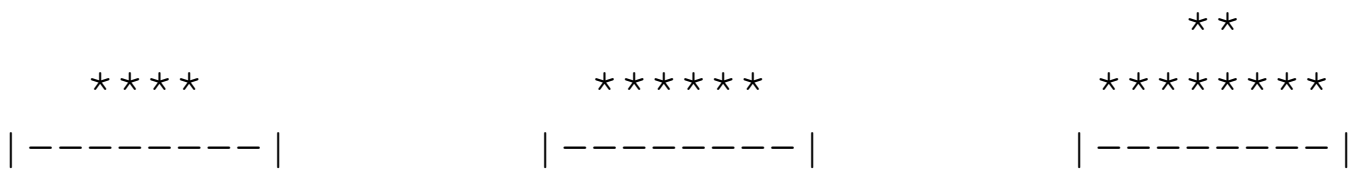
Αναδρομική λύση

Οι πύργοι του Ανόι

Αναδρομική λύση

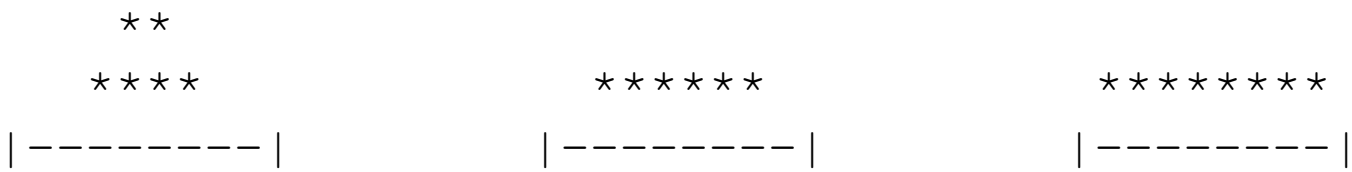
Οι πύργοι του Ανόι

Αναδρομική λύση



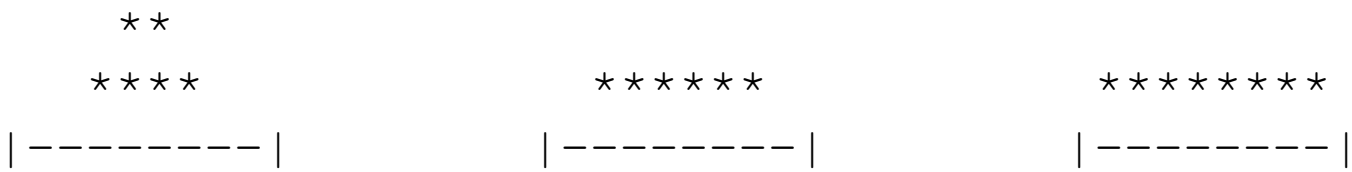
Οι πύργοι του Ανόι

Αναδρομική λύση



Οι πύργοι του Ανόι

Αναδρομική λύση



Οι πύργοι του Ανόι

Αναδρομική λύση

Οι πύργοι του Ανόι

Αναδρομική λύση

Οι πύργοι του Ανόι

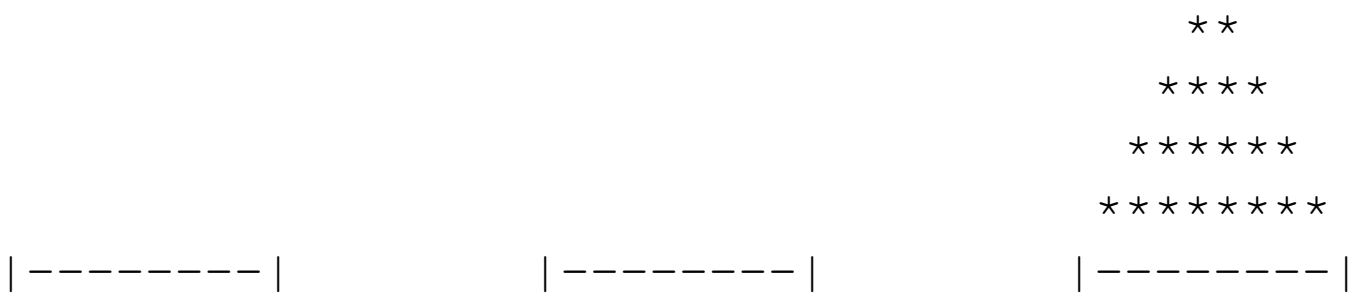
Αναδρομική λύση

Οι πύργοι του Ανόι

Αναδρομική λύση

Οι πύργοι του Ανόι

Αναδρομική λύση

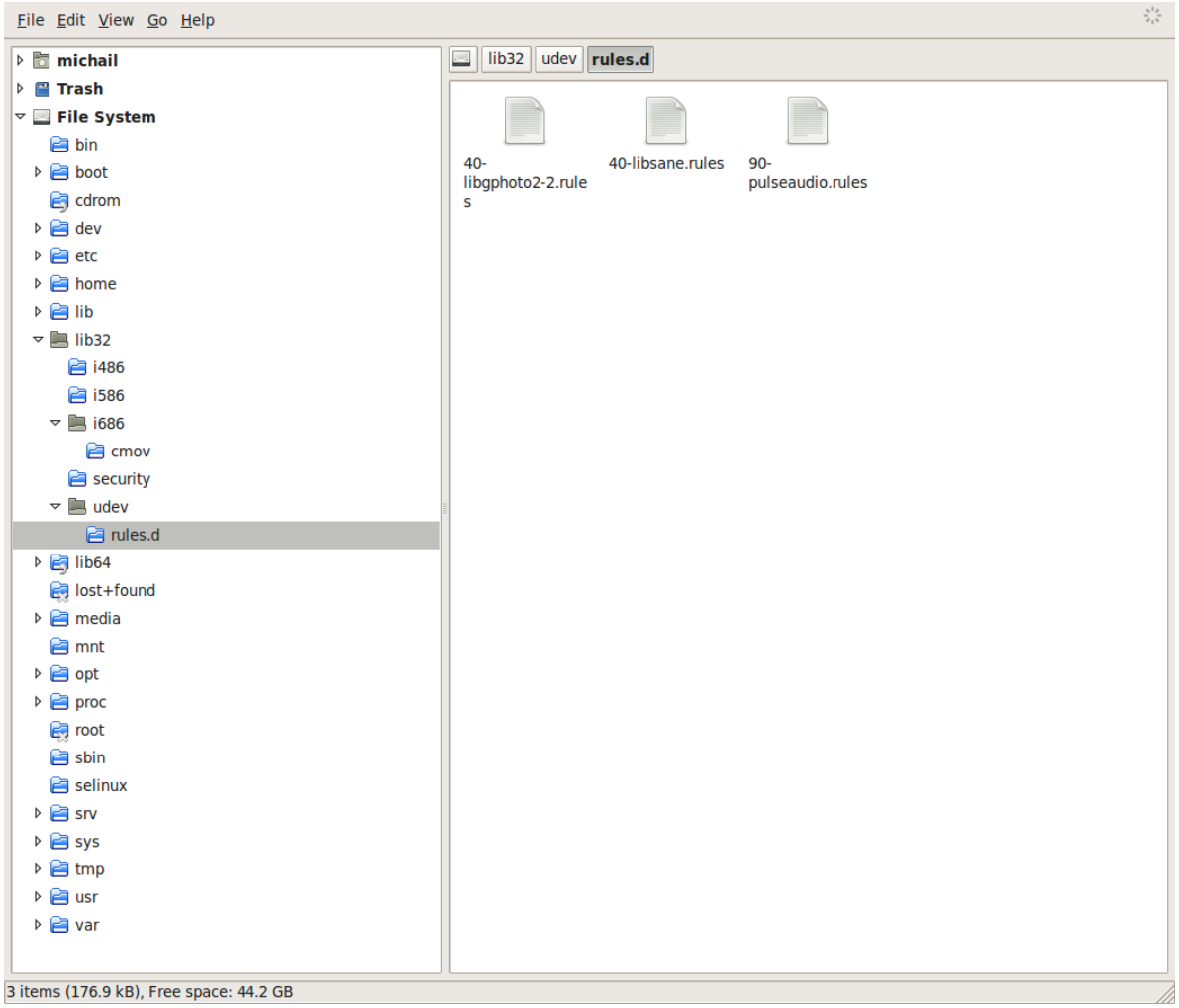


Δέντρα

Μαθηματική αφαίρεση με κεντρικό ρόλο στη σχεδίαση και την ανάλυση αλγορίθμων.

- άμεση σχέση με την αναδρομή
- συχνά στην καθημερινή ζωή
  - γενεαλογικό δέντρο
  - μορφή αγώνων σε knockout πρωτάθλημα

Παράδειγμα



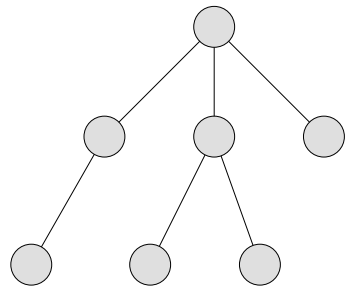
Στις εφαρμογές για υπολογιστές, μια από τις γνωστότερες χρήσεις των δομών δέντρου είναι στην οργάνωση του συστήματος αρχείων.

Τοποθετούμε τα αρχεία σε *καταλόγους* (directories) οι οποίοι ορίζονται αναδρομικώς ως ακολουθίες φακέλων και αρχείων.

Δέντρα

Ένα δέντρο είναι μια συλλογή:

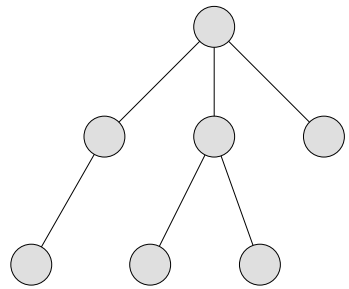
- κόμβων (ή κορυφών)
- μια συλλογή ακμών (κάθε ακμή είναι ένα ζευγάρι κόμβων που συνδέει 2 κόμβους)



Δέντρα

Ένα δέντρο είναι μια συλλογή:

- κόμβων (ή κορυφών)
- μια συλλογή ακμών (κάθε ακμή είναι ένα ζευγάρι κόμβων που συνδέει 2 κόμβους)



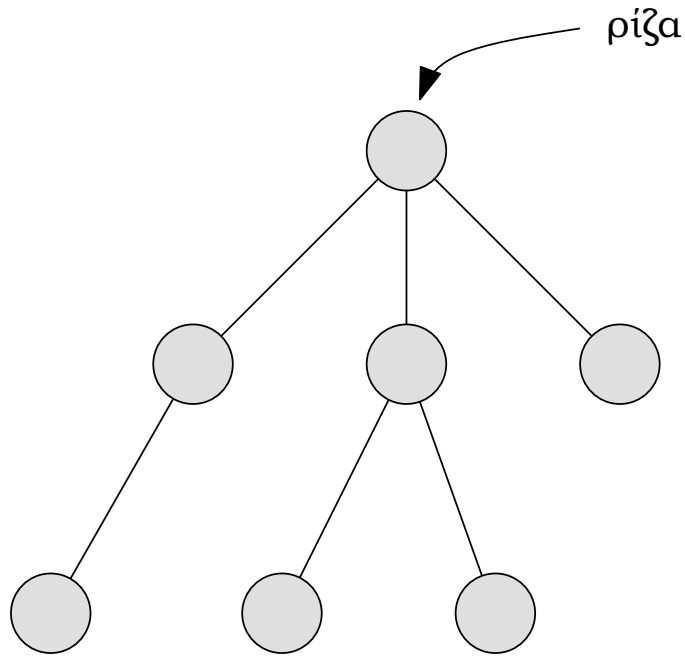
Μια διαδρομή (path) μεταξύ δύο κόμβων είναι μία λίστα από διακεκριμένους κόμβους στην οποία οι διαδοχικοί κόμβοι συνδέονται με ακμές.

Καθοριστική ιδιότητα δέντρου: μεταξύ κάθε ζεύγους κόμβων υπάρχει ακριβώς μια διαδρομή

Δέντρα

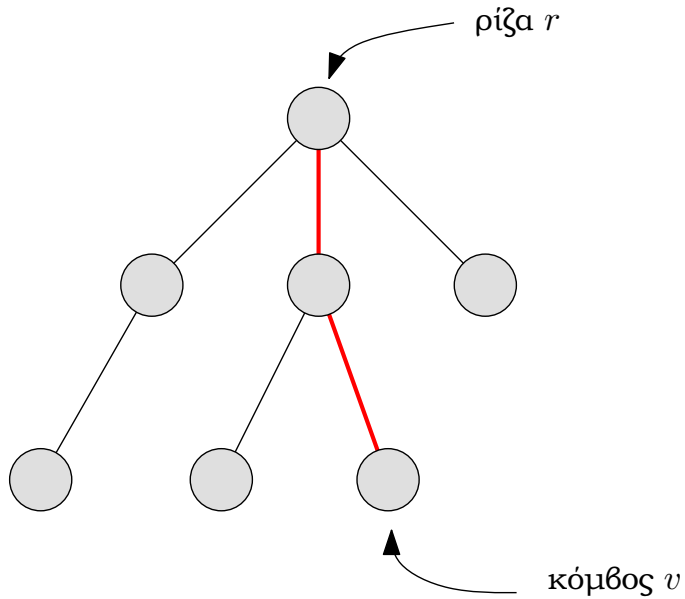
δέντρο με ρίζα (rooted tree)

Ορίζουμε ένα κόμβο ως ρίζα. Συνήθως ζωγραφίζουμε αυτόν τον κόμβο πιο ψηλά.



Δέντρα

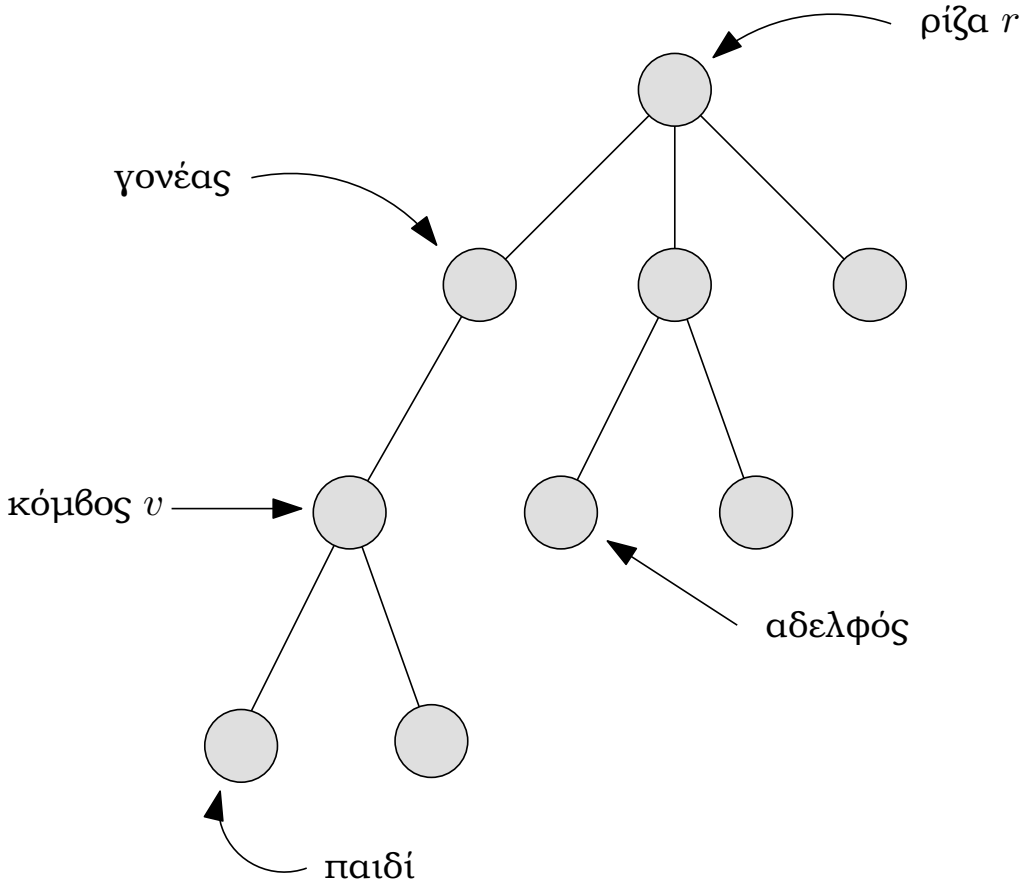
Μεταξύ ενός κόμβου *v* και της ρίζας *r* υπάρχει ένα μοναδικό μονοπάτι.



Ο αριθμός των ακμών του μονοπατιού αυτού καθορίζει το ύψος του κόμβου *v* στο δέντρο.

Δέντρα

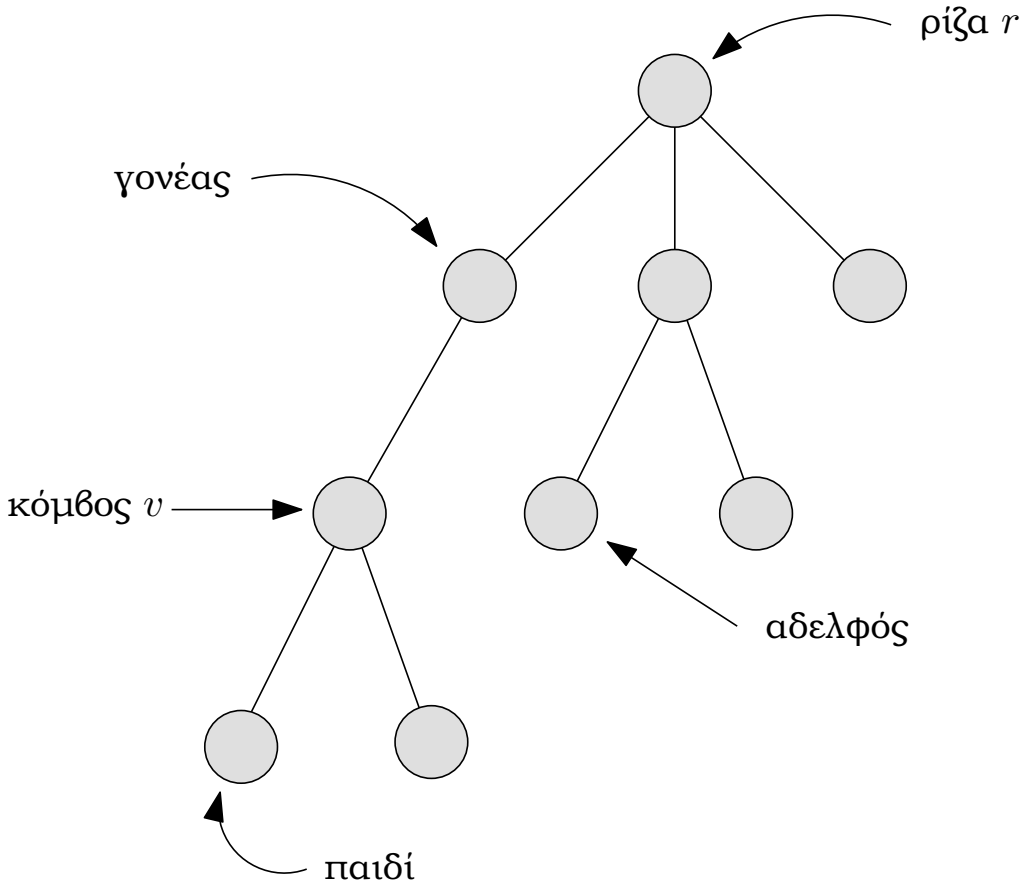
Έστω ένας κόμβος *v* και η ρίζα *r*.



Ο κόμβος ακριβώς πριν τον *v* στο μονοπάτι $r \rightsquigarrow v$ λέγεται πατέρας του *v*.

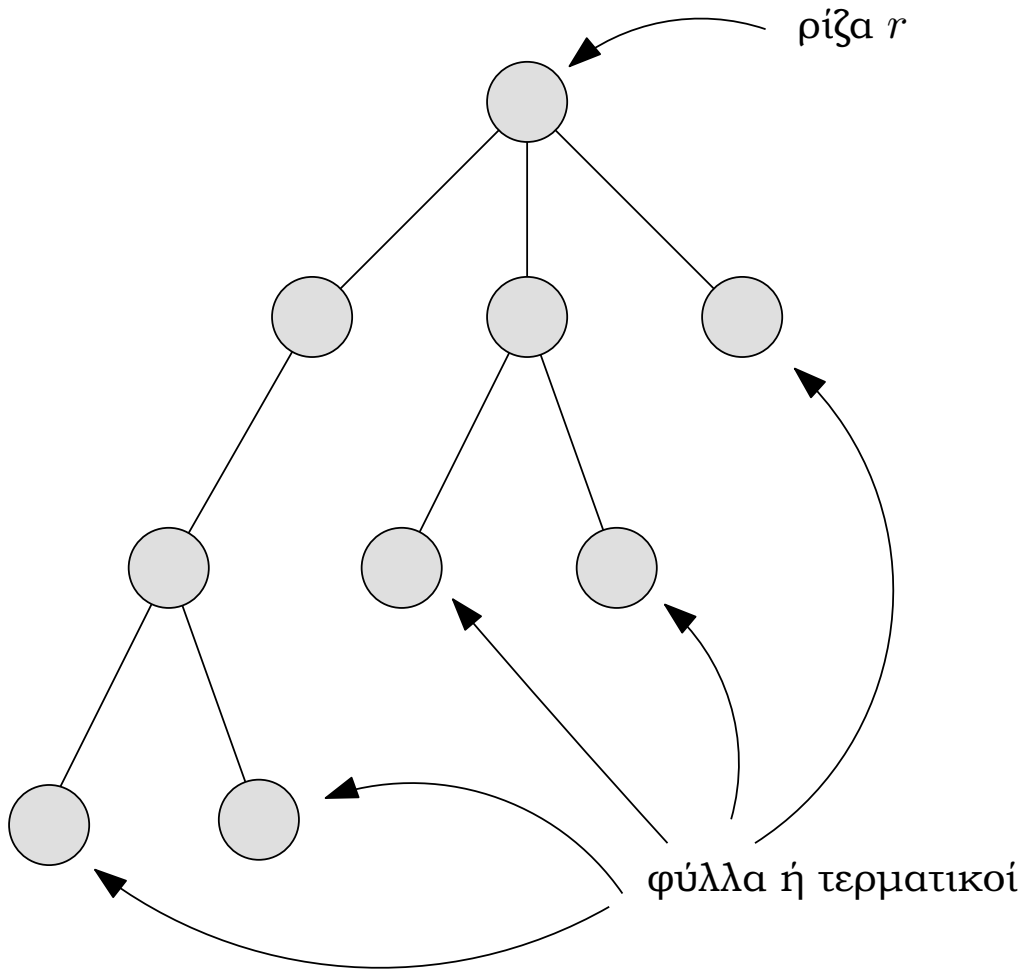
Δέντρα

Έστω ένας κόμβος *v* και η ρίζα *r*.



Οι κόμβοι ακριβώς κάτω από τον *v* είναι τα παιδιά του *v*.

Δέντρα

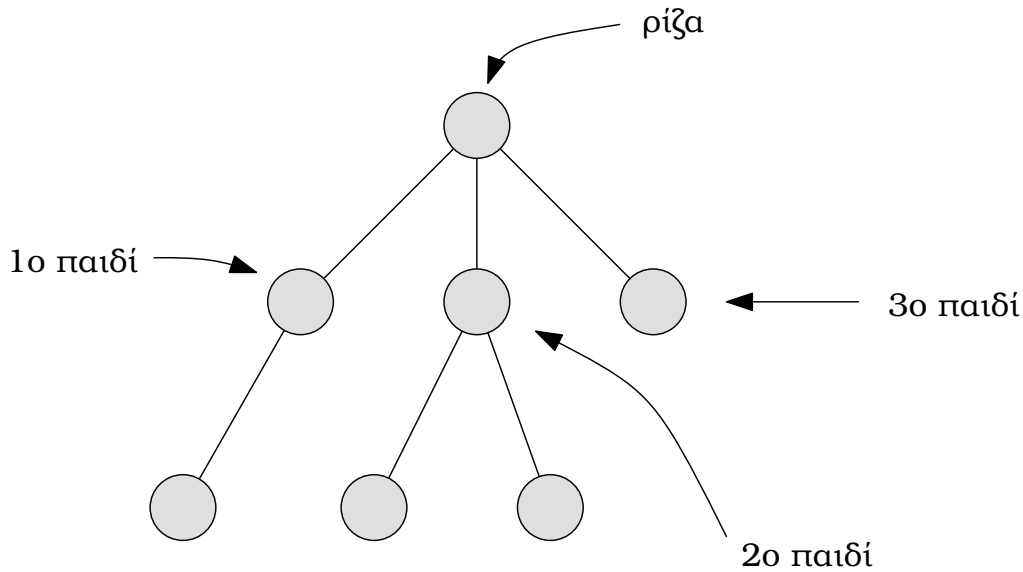


Κόμβοι χωρίς παιδιά ονομάζονται φύλλα ή τερματικοί. Κόμβοι με τουλάχιστον ένα παιδί ονομάζονται μη-τερματικοί.

Δέντρα

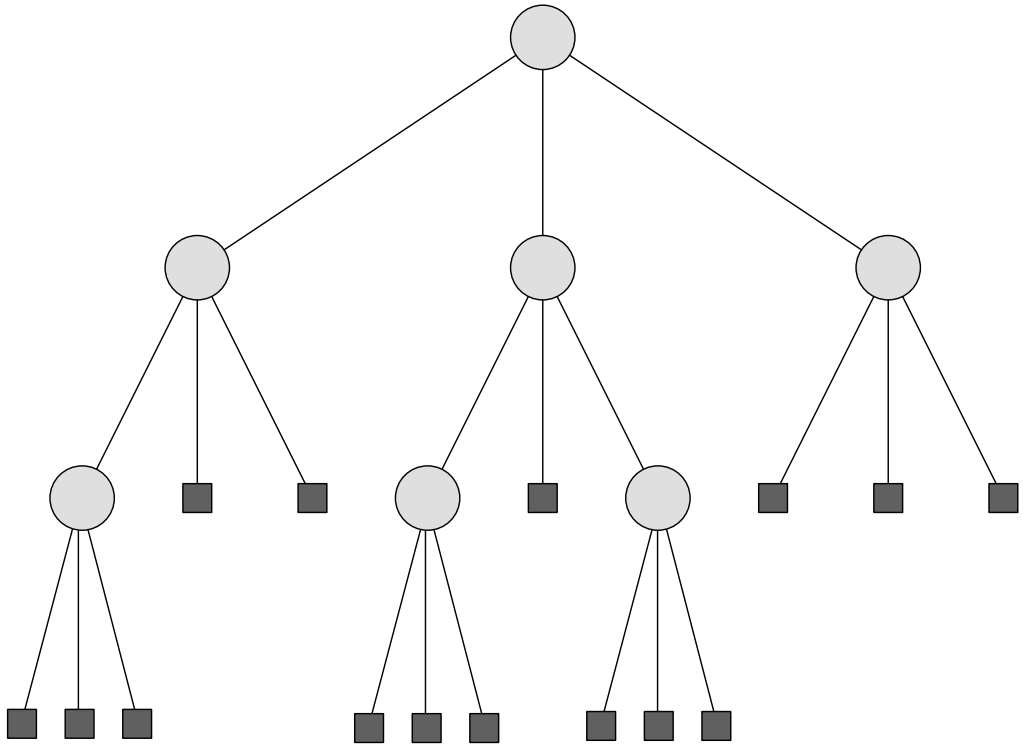
διατεταγμένα δέντρα (ordered trees)

είναι δέντρα με ρίζα στα οποία η σειρά των παιδιών κάθε κόμβου είναι καθορισμένη



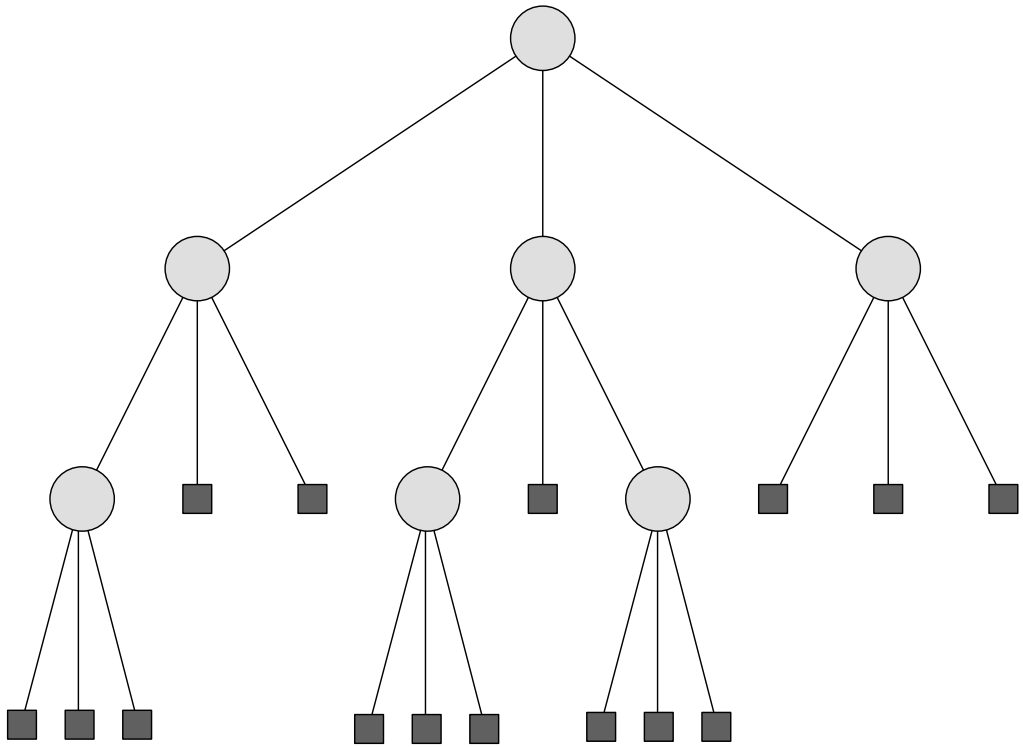
Μ-αδικά Δέντρα

Είναι διατεταγμένα δέντρα στα οποία κάθε κόμβος έχει υποχρεωτικά κάποιο καθορισμένο πλήθος παιδιών.



Μ-αδικά Δέντρα

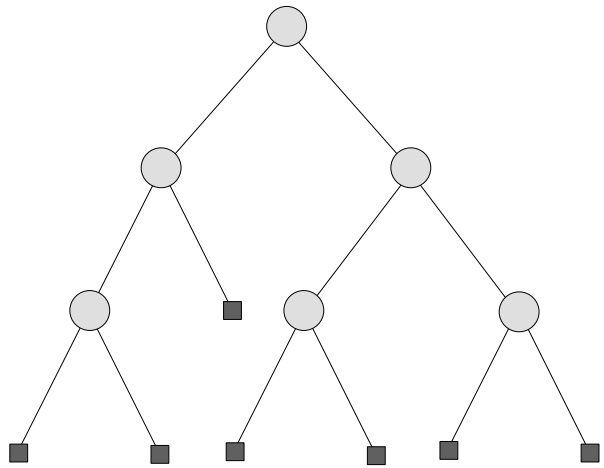
Ορίζουμε ειδικούς κόμβους που δεν έχουν παιδιά και τους ονομάζουμε *εξωτερικούς* κόμβους.



Κόμβοι χωρίς αρκετά παιδιά χρησιμοποιούν εξωτερικούς κόμβους ως ψευδοκόμβους.

Δυαδικά Δέντρα

Είναι διατεταγμένα δέντρα στα οποία κάθε κόμβος έχει ένα αριστερό και ένα δεξί παιδί.

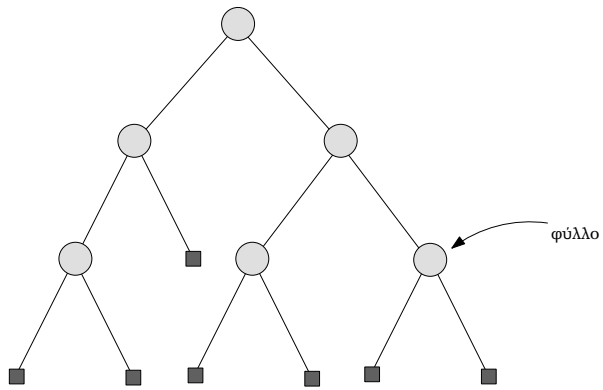


Έχει 2 ειδών κόμβους:

- εξωτερικούς κόμβους χωρίς παιδιά
- εσωτερικούς κόμβους με ακριβώς 2 παιδιά ο καθένας

Δυαδικά Δέντρα

Κάθε εσωτερικός κόμβος πρέπει να έχει ένα αριστερό και ένα δεξί παιδί, παρόλο που ένα από τα δύο ή και τα δύο μπορούν να είναι εξωτερικοί κόμβοι.



Ένα *φύλλο* σε ένα Μ-αδικό δέντρο είναι ένας εσωτερικός κόμβος του οποίου τα παιδιά είναι εξωτερικοί κόμβοι.

Ορισμός

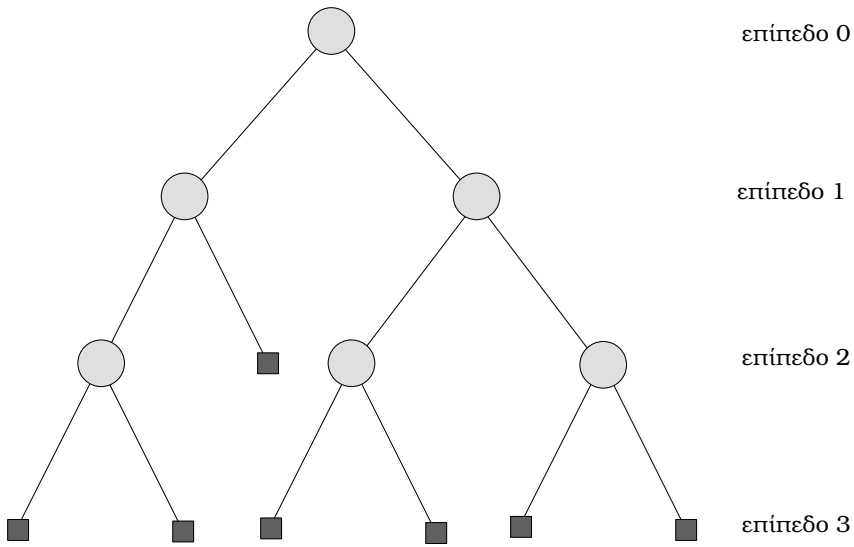
Ένα **δυαδικό δέντρο** (binary tree) είναι είτε κάποιος εξωτερικός κόμβος είτε κάποιος εσωτερικός κόμβος που συνδέεται με ένα ζεύγος δυαδικών δέντρων, τα οποία ονομάζονται αριστερό και δεξιό υποδέντρο αυτού του κόμβου.

Ορισμός

Ένα **Μ-αδικό δέντρο** (M-ary tree) είναι είτε κάποιος εξωτερικός κόμβος είτε κάποιος εσωτερικός κόμβος που συνδέεται με μια διατεταγμένη ακολουθία Μ δέντρων τα οποία είναι επίσης Μ-αδικά δέντρα.

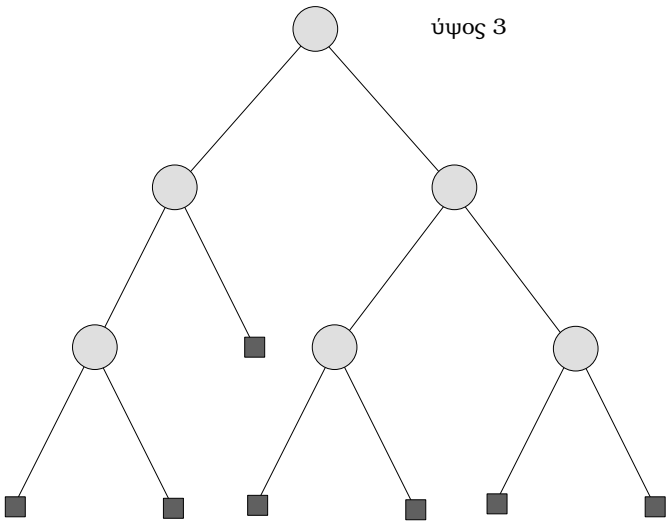
Δυαδικά Δέντρα

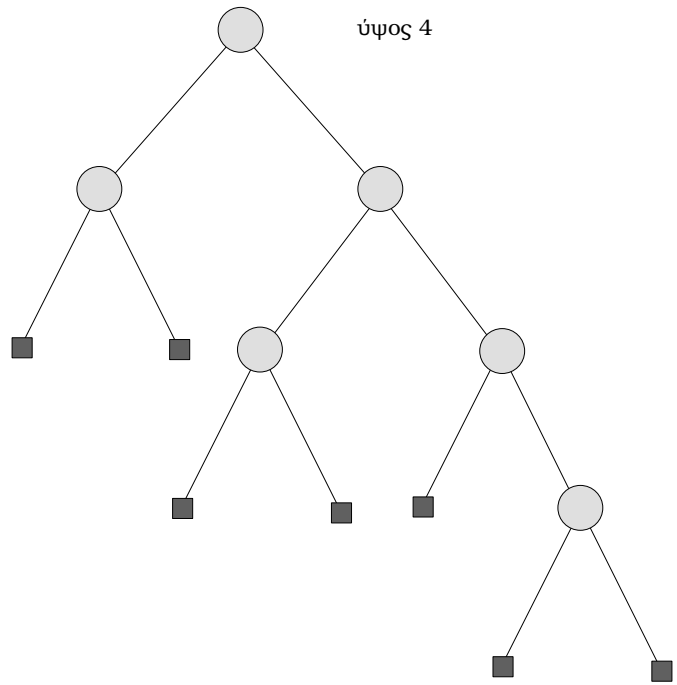
Το **επίπεδο** (level) ενός κόμβου σε κάποιο δέντρο είναι κατά ένα μεγαλύτερο από το επίπεδο του γονέα του (με την ρίζα να βρίσκεται στο επίπεδο 0).



Δυαδικά Δέντρα

Το **ύψος** (height) ενός δέντρου είναι ίσο με το μέγιστο επίπεδο των κόμβων του.

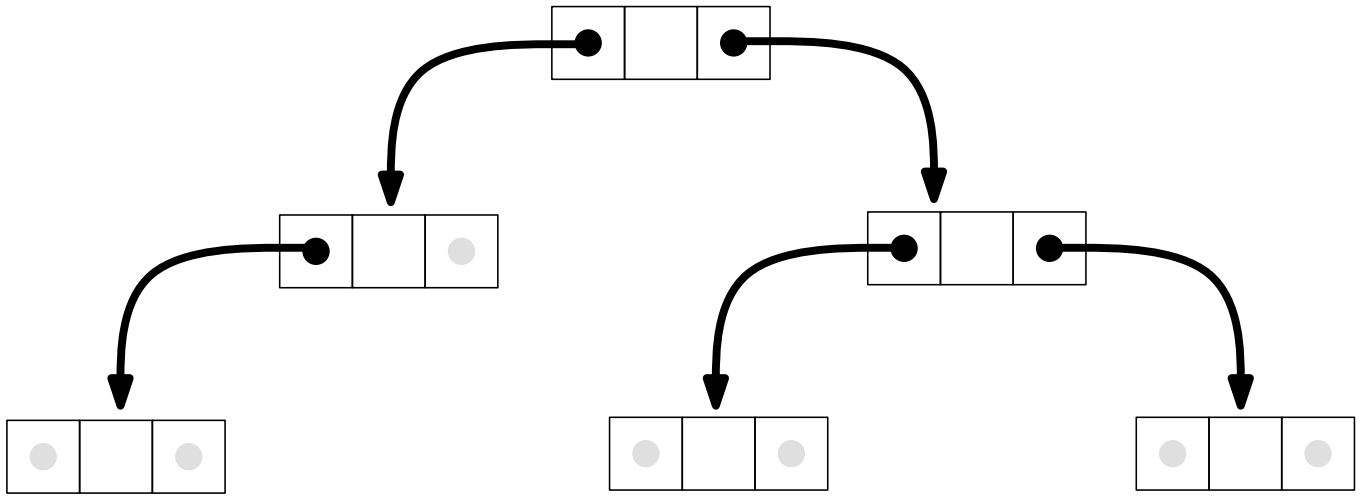




Αναπαράσταση Δέντρων

Δυαδικά δέντρα στον υπολογιστή

Χρησιμοποιούμε παρόμοια τεχνική με τις λίστες. Συνήθως χρησιμοποιούμε τιμή `NULL` για να αναπαραστήσουμε εξωτερικούς κόμβους.



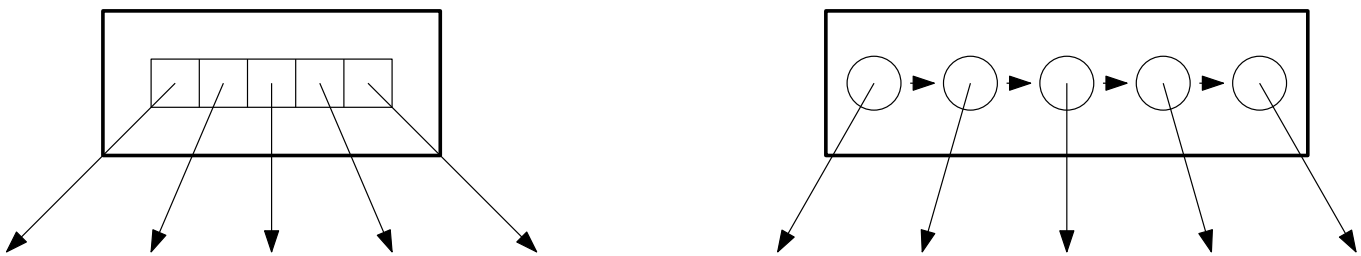
Μπορούμε να έχουμε και συνδέσμους από τα παιδιά προς τον πατέρα (ανάλογα με την χρήση του δέντρου).

Αναπαράσταση Δέντρων

M-αδικά δέντρα στον υπολογιστή

Για να αναπαραστήσουμε *M*-αδικά δέντρα, μπορούμε να χρησιμοποιήσουμε ως κόμβους του δέντρου εγγραφές που περιέχουν:

- έναν πίνακα με *M* δείκτες στα παιδιά του κόμβου
- την πληροφορία του κόμβου.



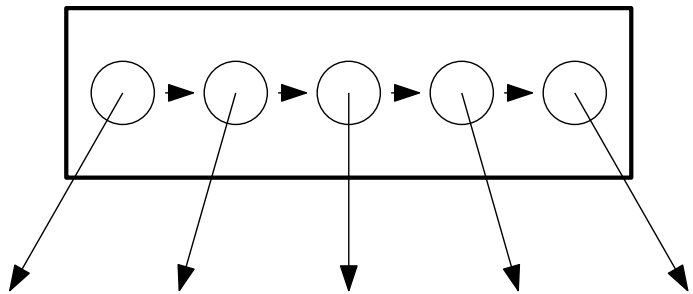
Επίσης αντί για πίνακα μπορούμε να χρησιμοποιήσουμε λίστα για τα παιδιά ενός κόμβου.

Αναπαράσταση Δέντρων

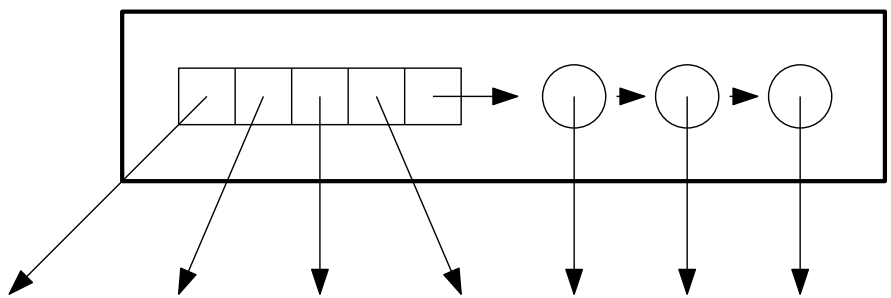
multiway δέντρα στον υπολογιστή

Για να αναπαραστήσουμε δέντρα που δεν έχουν όριο παιδιών, χρησιμοποιούμε ως κόμβους εγγραφές που περιέχουν:

- μια λίστα με δείκτες στα παιδιά του κόμβου
- την πληροφορία του κόμβου.



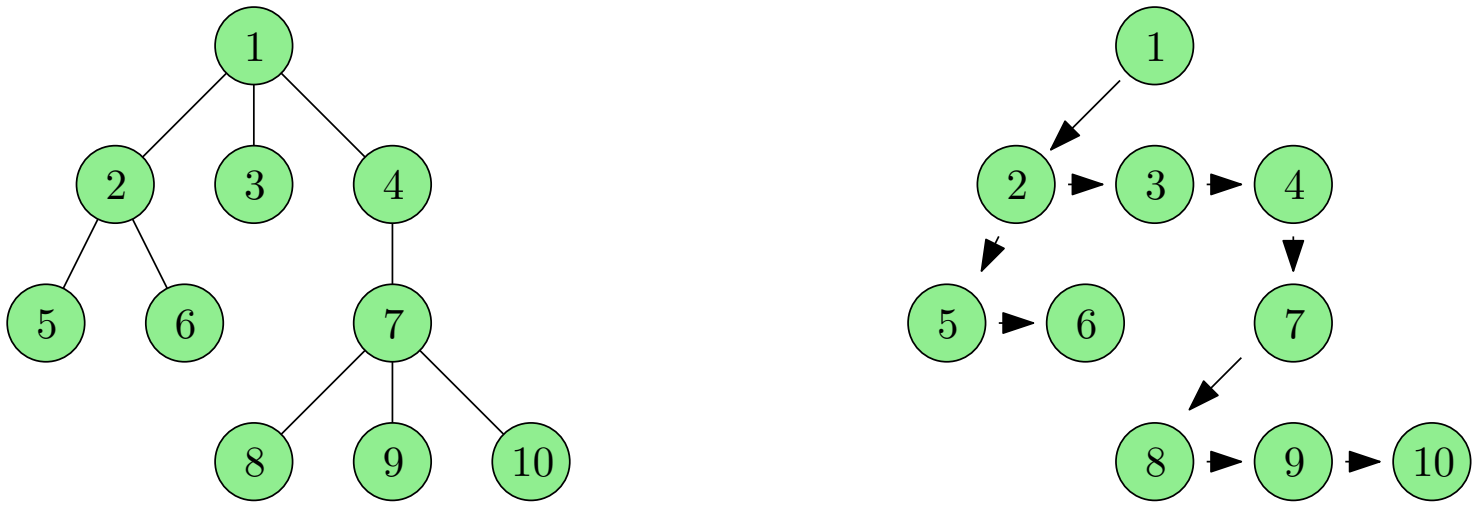
Μια πιο περίτεχνη αναπαράσταση είναι να έχουμε έναν πίνακα για τα πρώτα παιδιά και μια λίστα σε περίπτωση που ένας κόμβος έχει περισσότερα παιδιά.



Αναπαράσταση Δέντρων ως Δυαδικά

first child-next sibling binary tree

Μπορούμε να αναπαραστήσουμε ένα multiway ή ένα *M*-αδικό δέντρο ως δυαδικό χρησιμοποιώντας την τεχνική **πρώτου παιδιού-επόμενου αδελφού**.



Κάθε κόμβος έχει

- έναν δείκτη στο αριστερότερο παιδί του (leftmost child),
- έναν δείκτη στον αμέσως επόμενο αδελφό του (next sibling).

Διάσχιση Δέντρων

Έχοντας ένα σύνδεσμο για έναν κόμβο ενός δέντρου (συνήθως τη ρίζα) θέλουμε να διασχίσουμε όλους τους κόμβους ενός δέντρου.

Υπάρχουν διάφορες επιλογές:

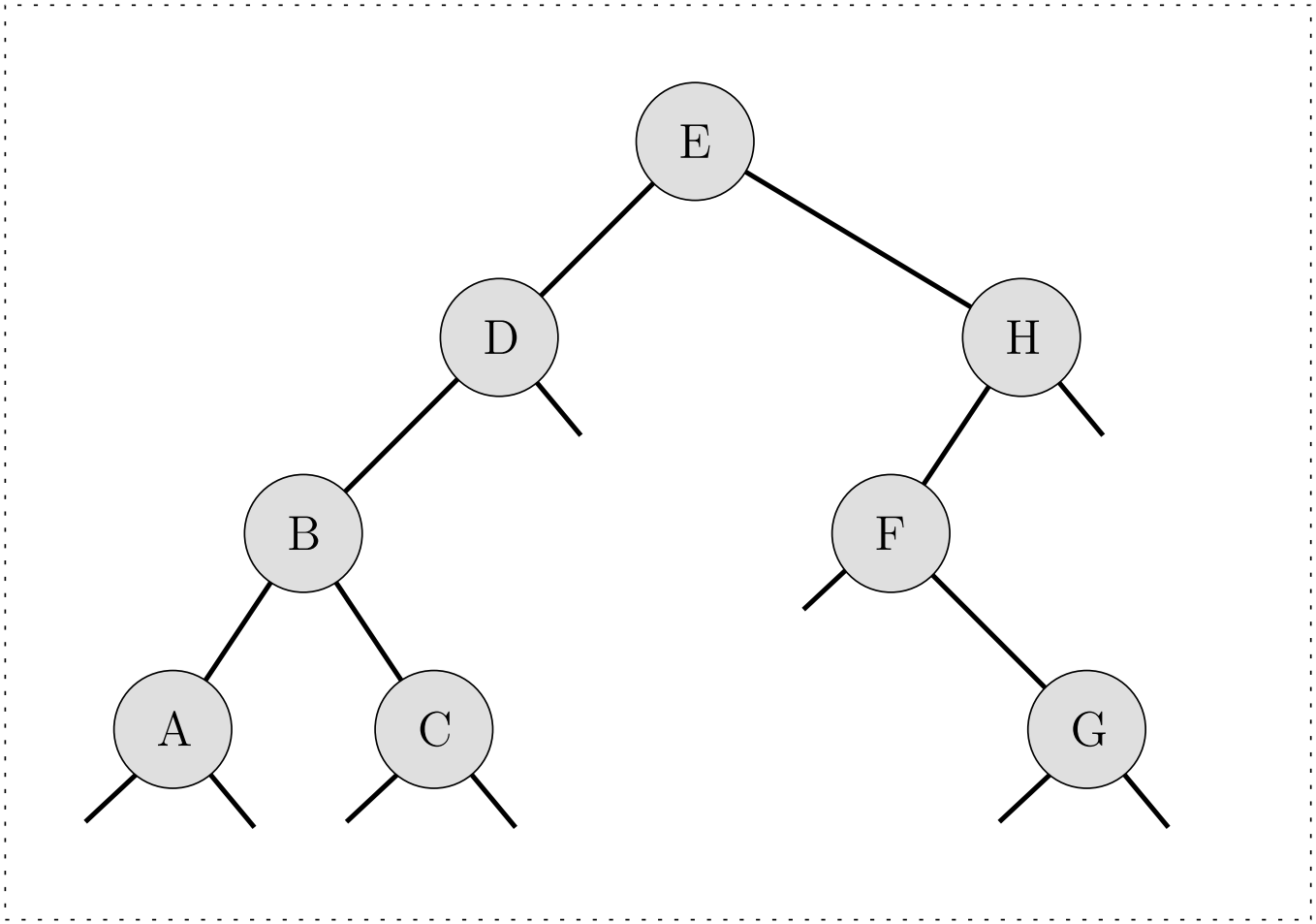
- προδιατεταγμένη (preorder)
- ενδοδιατεταγμένη (inorder)
- μεταδιατεταγμένη (postorder)
- επιπεδοδιατεταγμένη (levelorder)

Οι περισσότερες διασχίσεις υλοποιούνται εύκολα με αναδρομή.

preorder

κανόνας

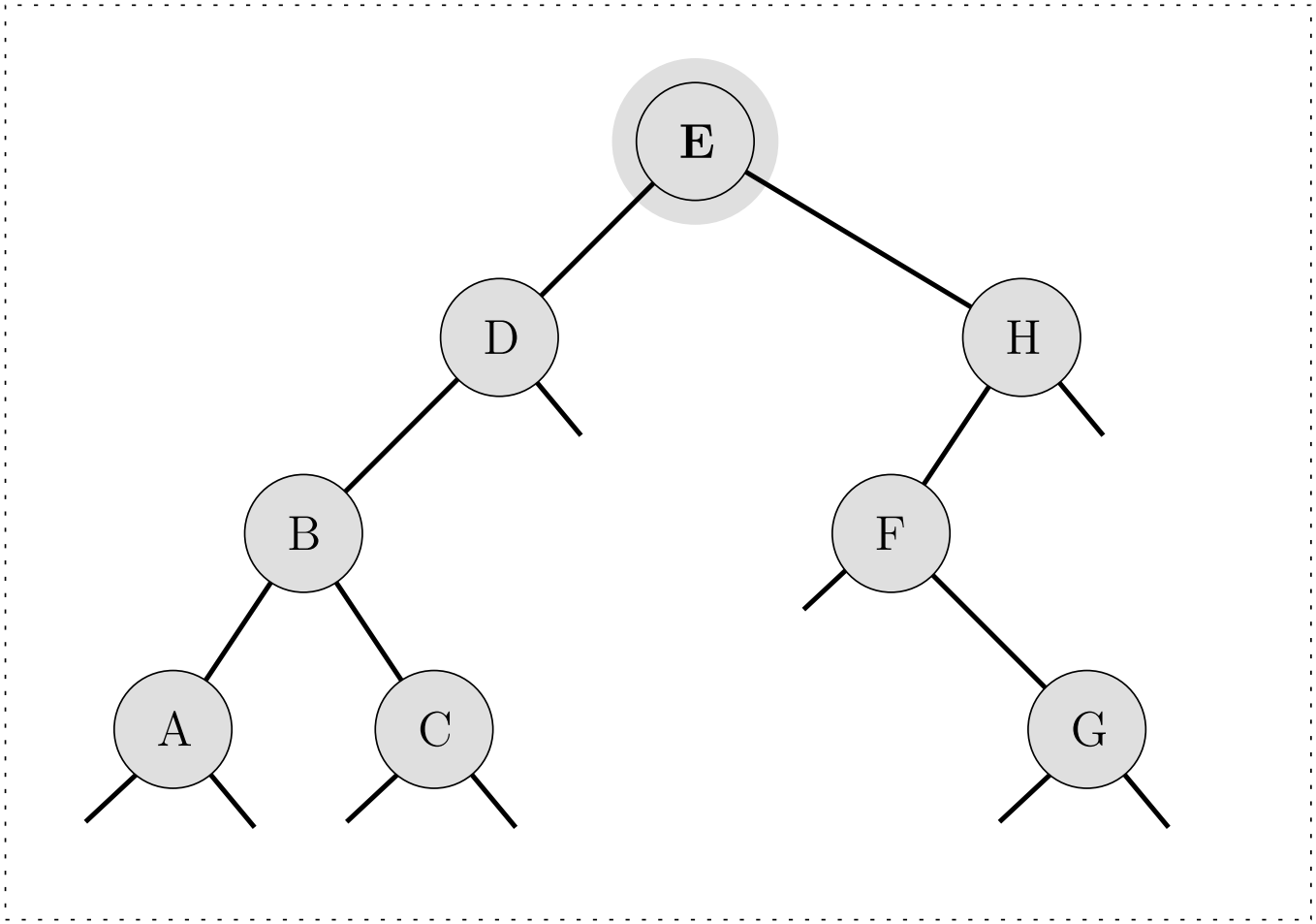
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

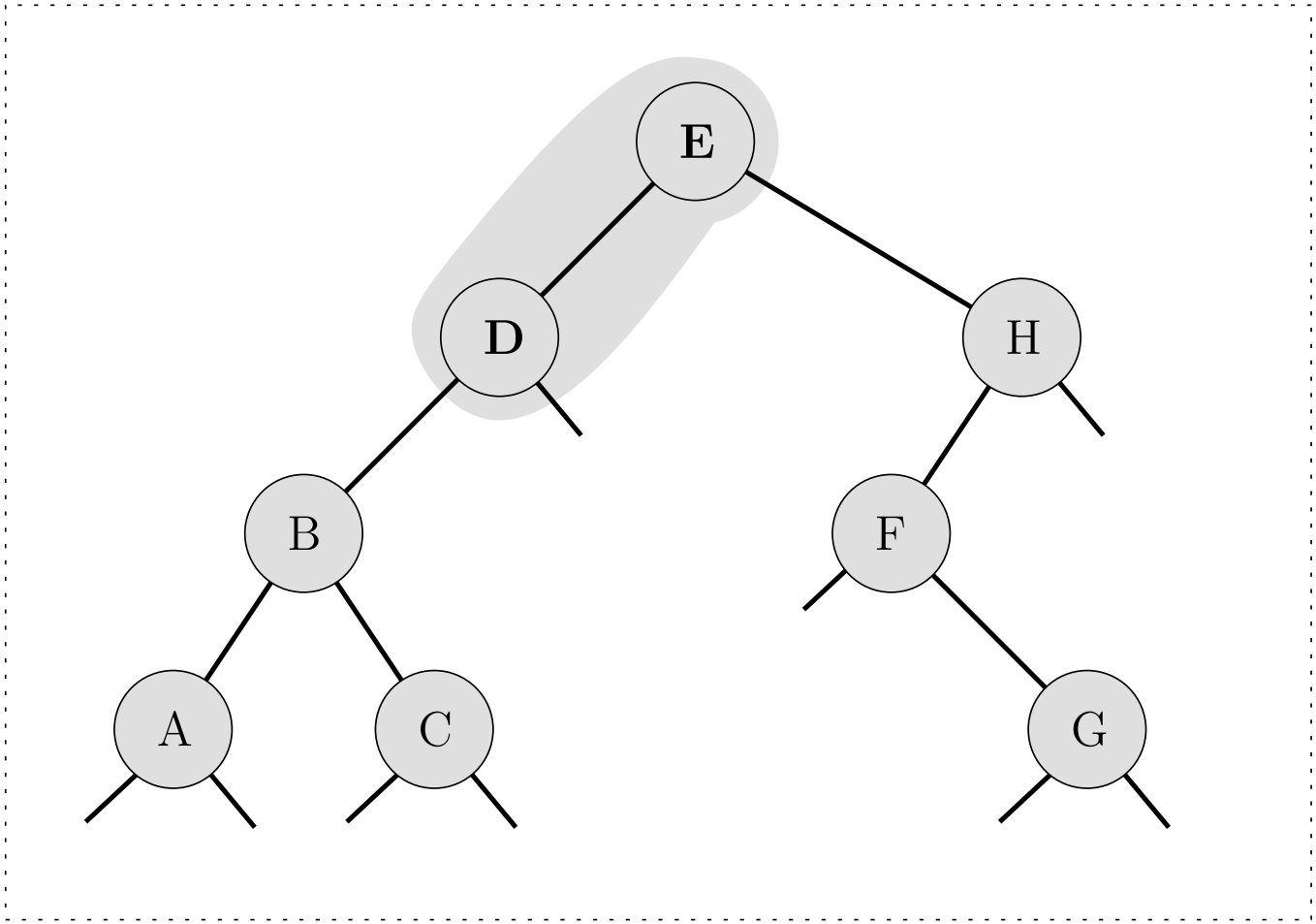
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

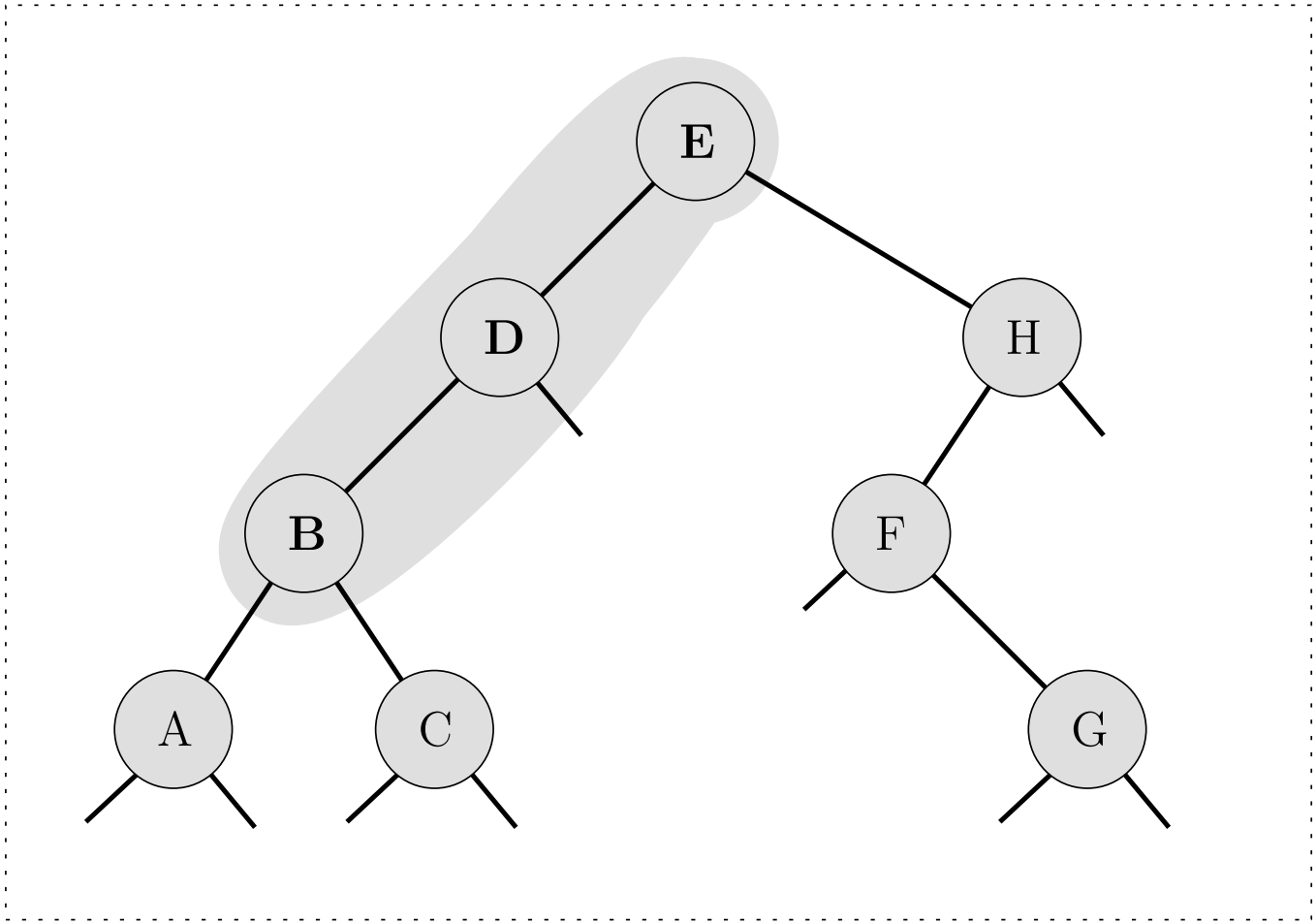
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

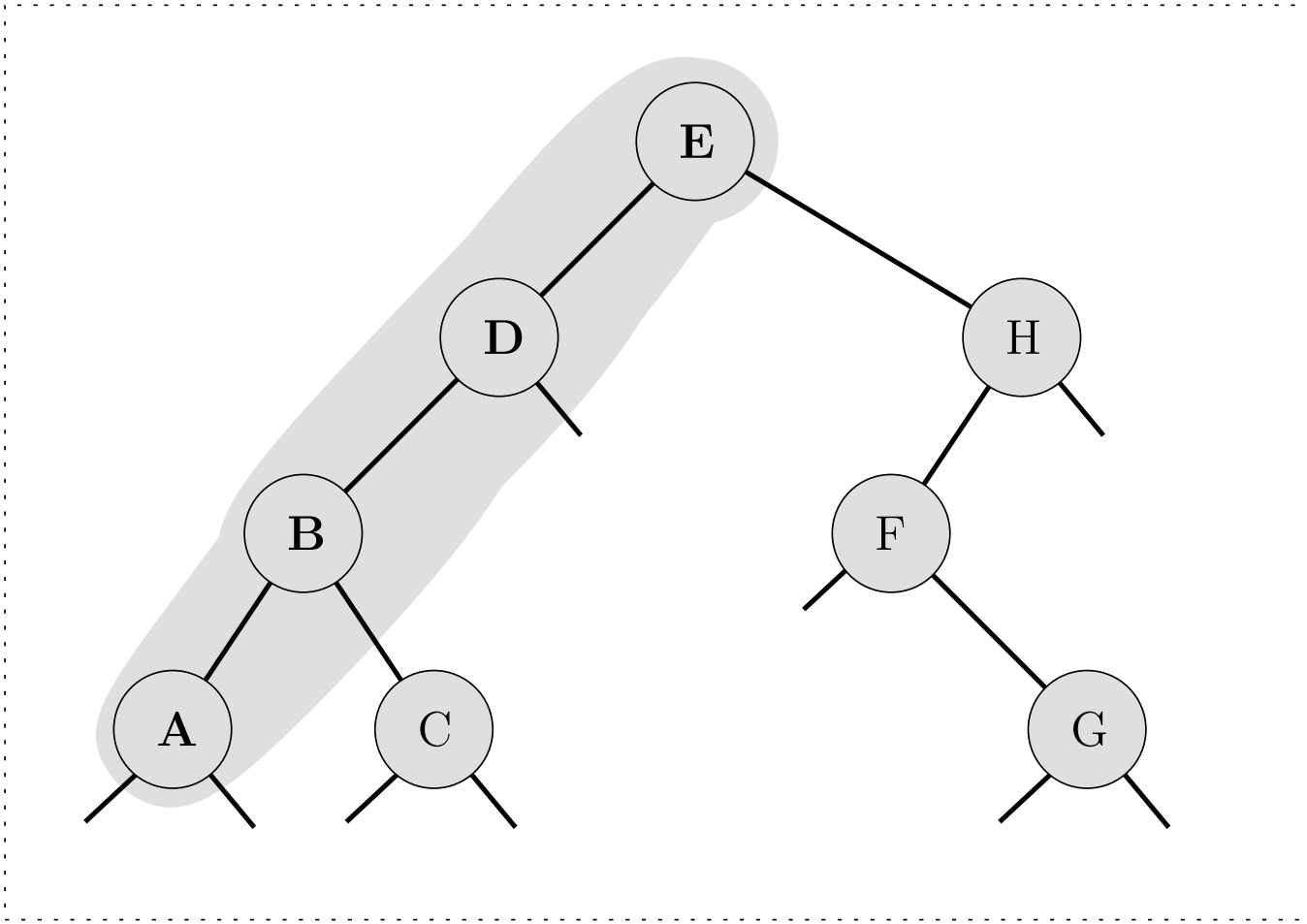
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

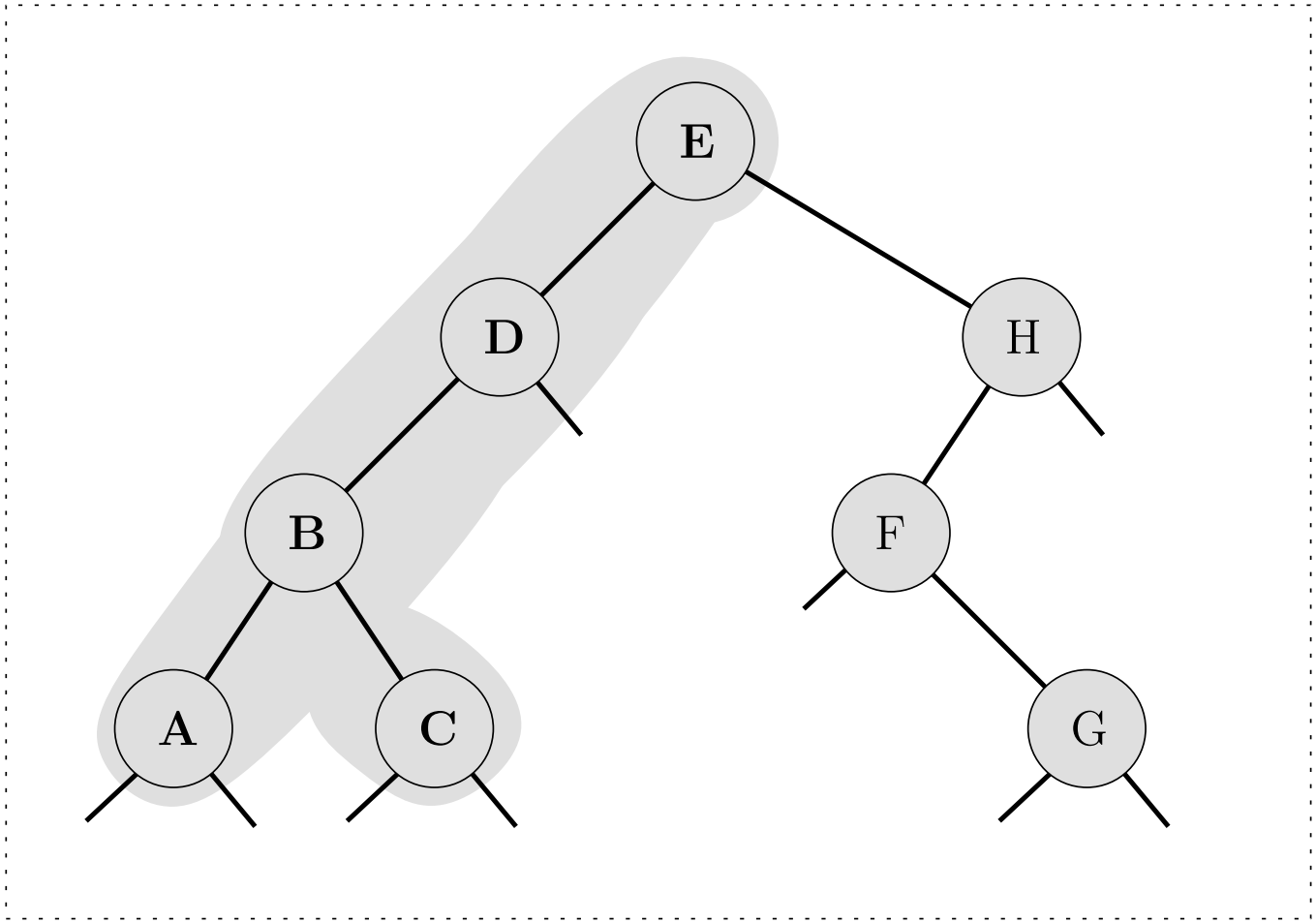
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

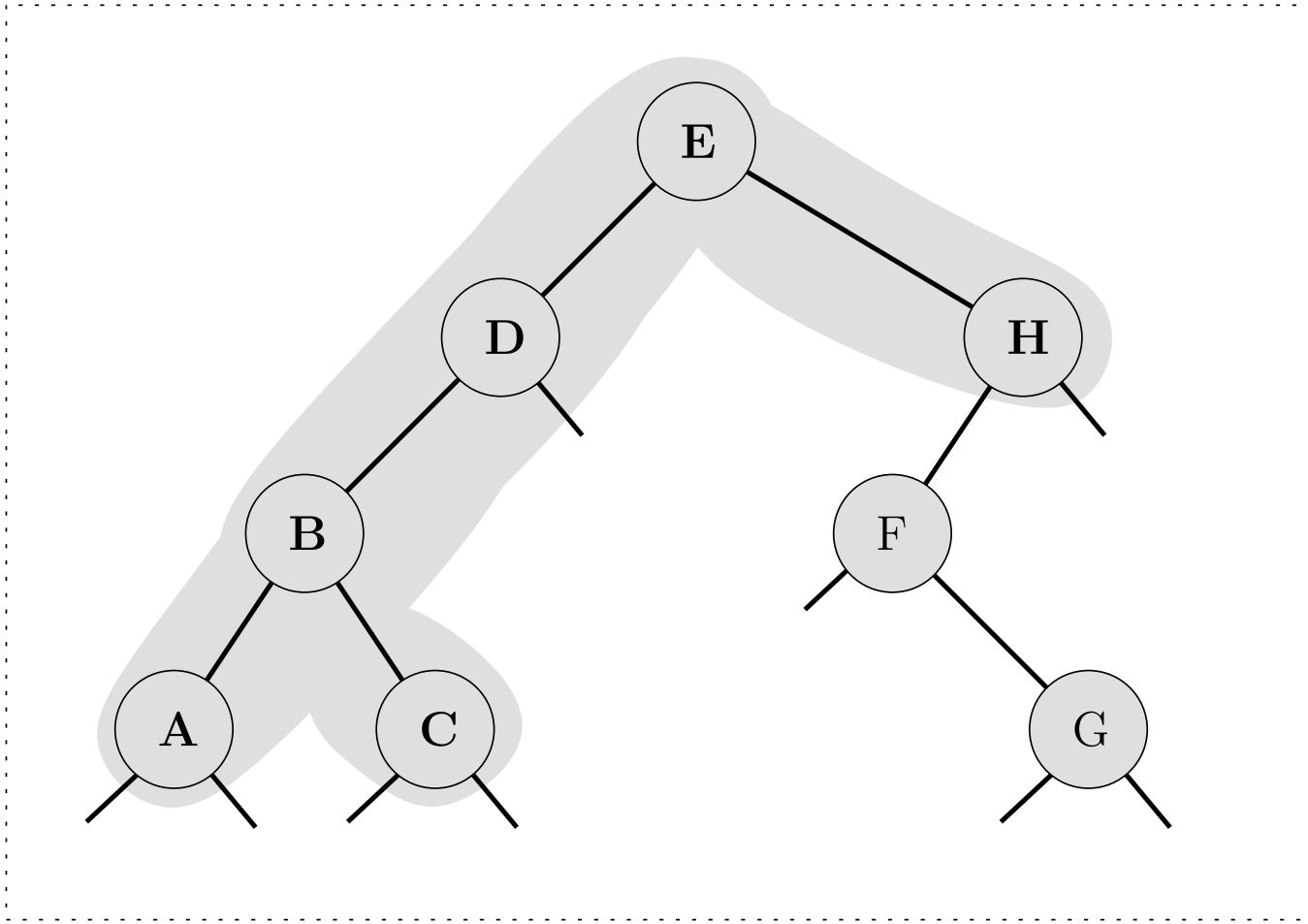
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

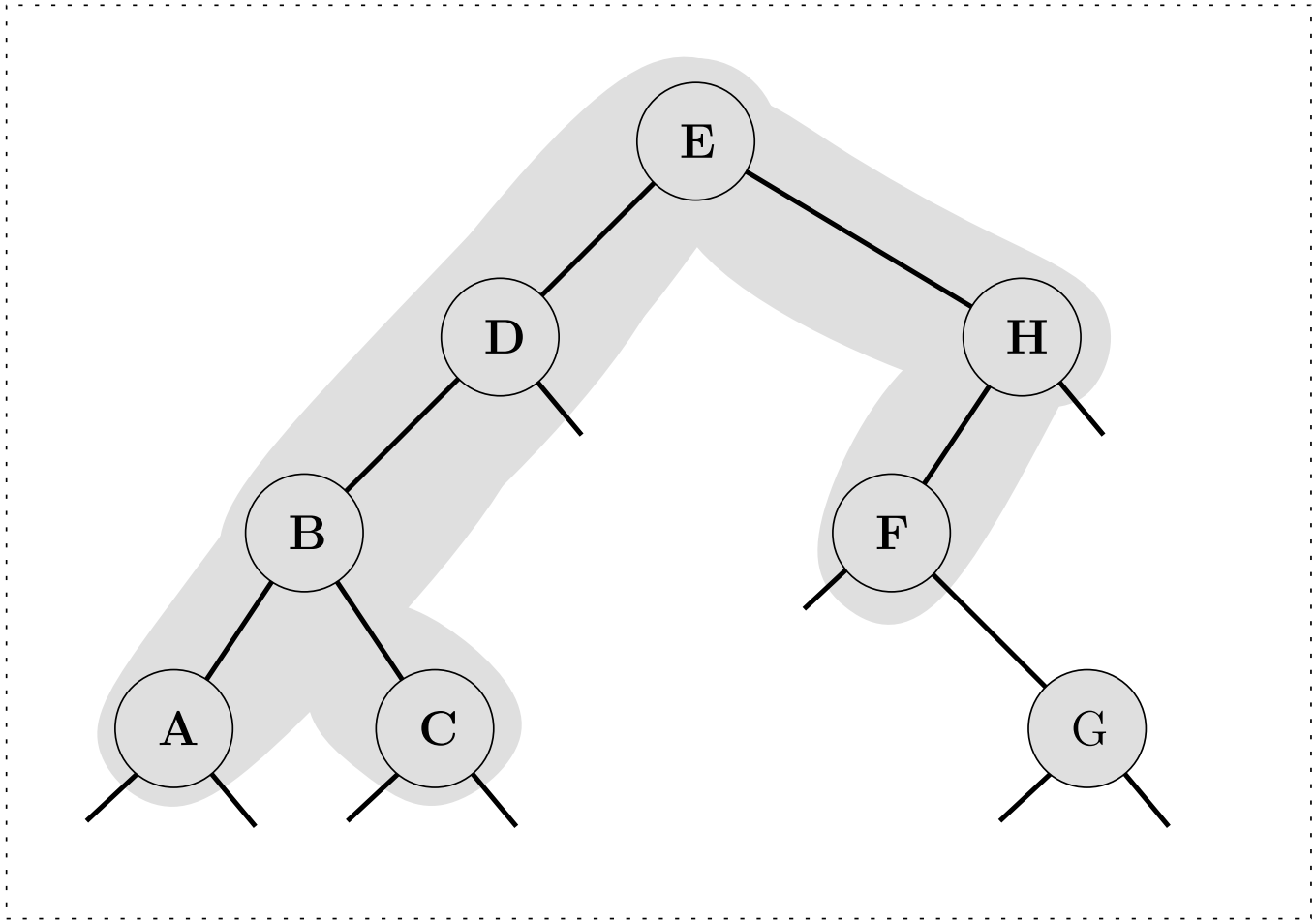
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

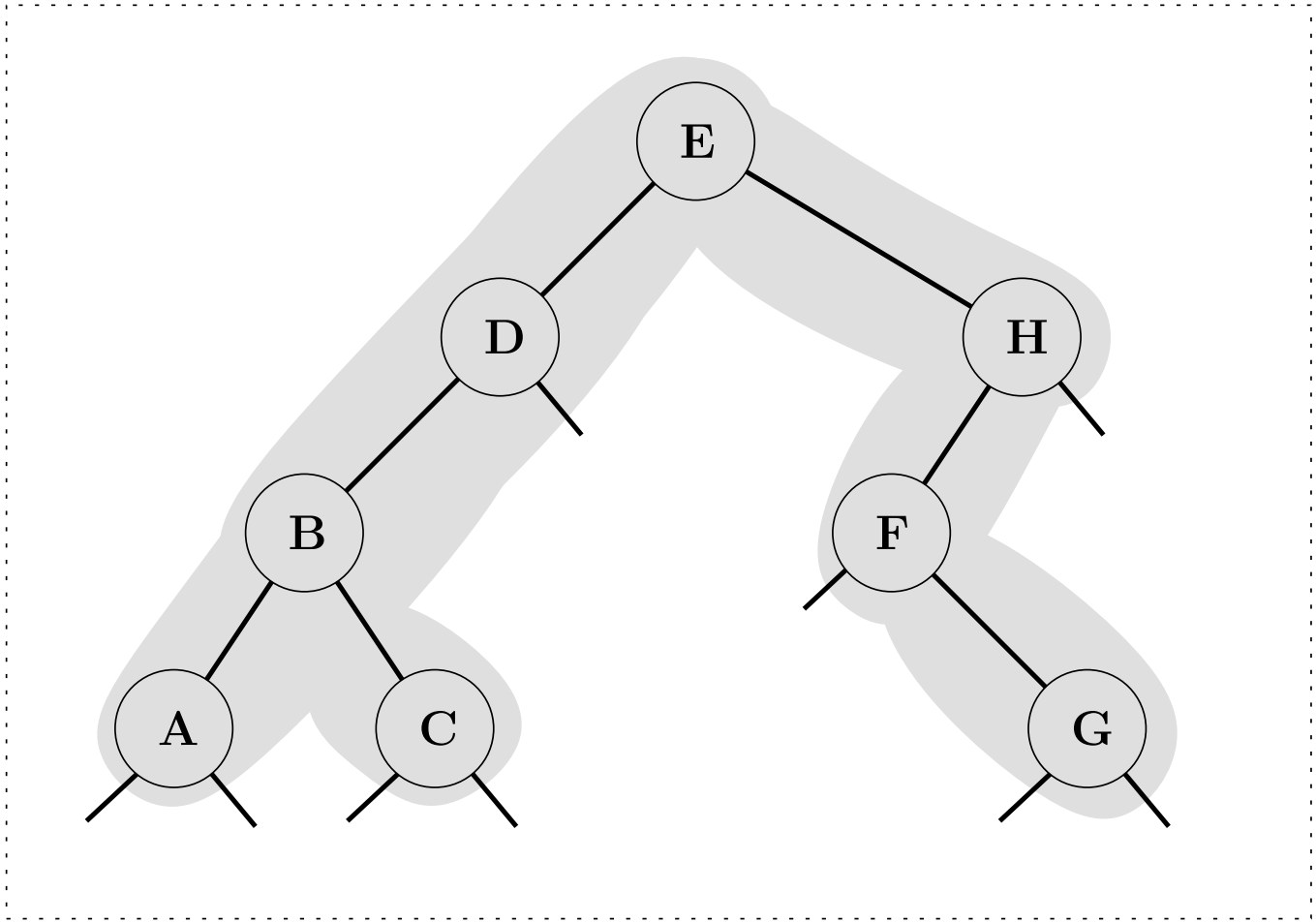
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο



preorder

κανόνας

Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο

```
1  void traverse( node t )
2  {
3      if ( t == NULL ) return;
4      visit( t );
5      traverse( t->left );
6      traverse( t->right );
7  }
```

preorder με στοίβα

κανόνας

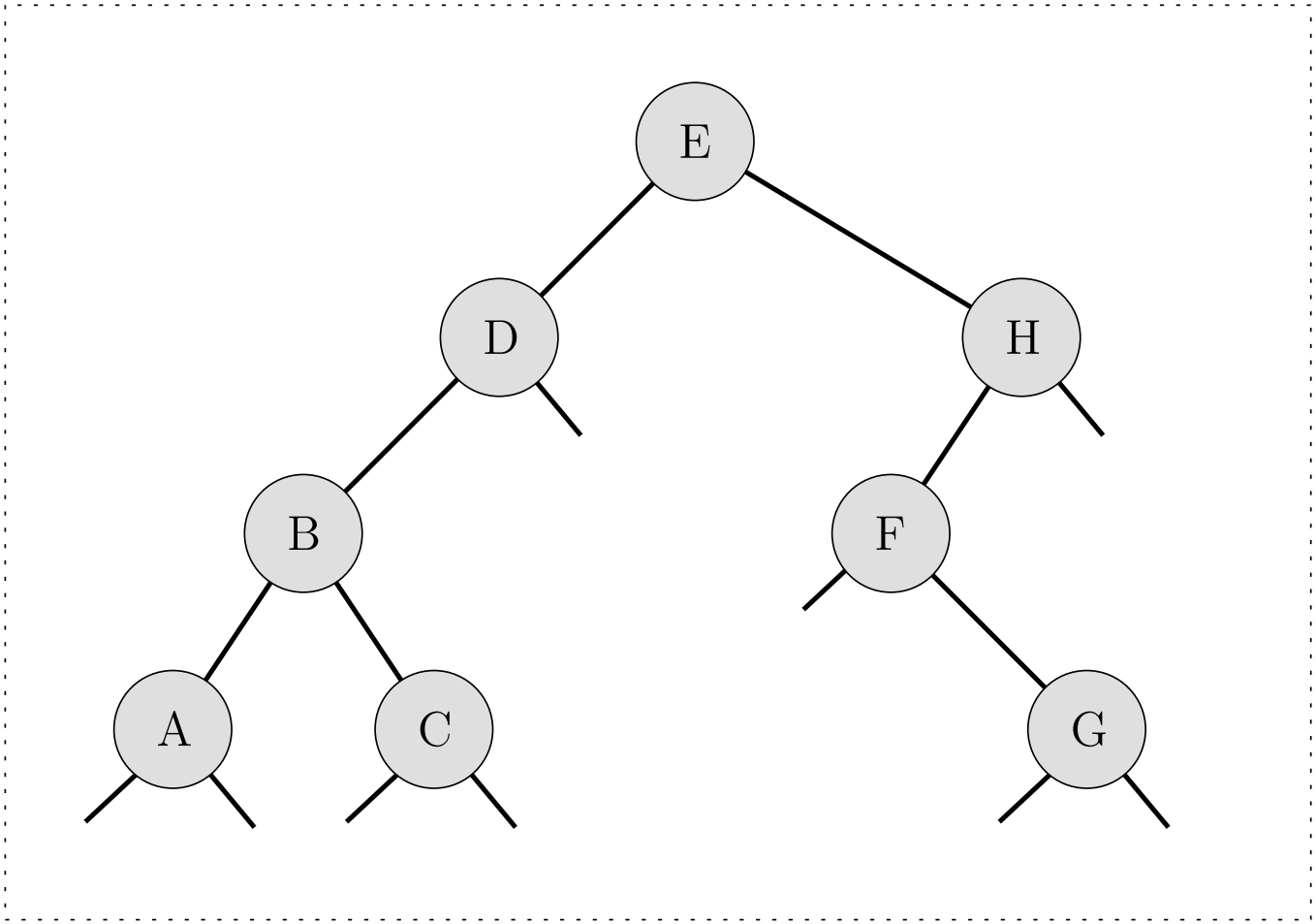
Επισκεπτόμαστε ένα κόμβο και μετά επισκεπτόμαστε το αριστερό και το δεξιό υποδέντρο

```
1  void traverse( node t )
2  {
3      node h;
4      initStack();
5      pushStack( t );
6
7      while( ! emptyStack() )
8      {
9          h = popStack();
10         visit( h );
11         if ( h->right != NULL ) pushStack( h->right );
12         if ( h->left  != NULL ) pushStack( h->left );
13     }
14 }
```

inorder

κανόνας

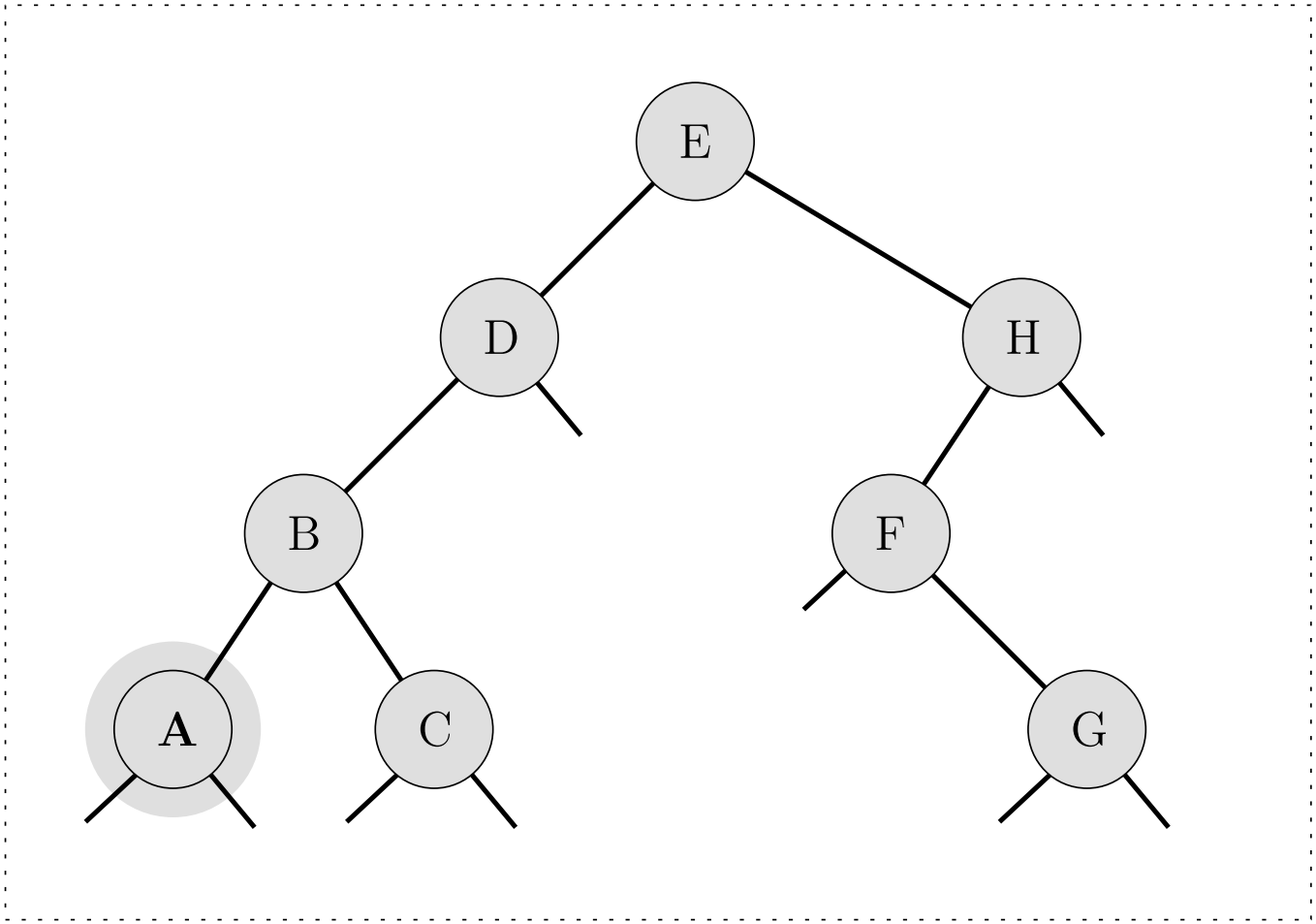
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

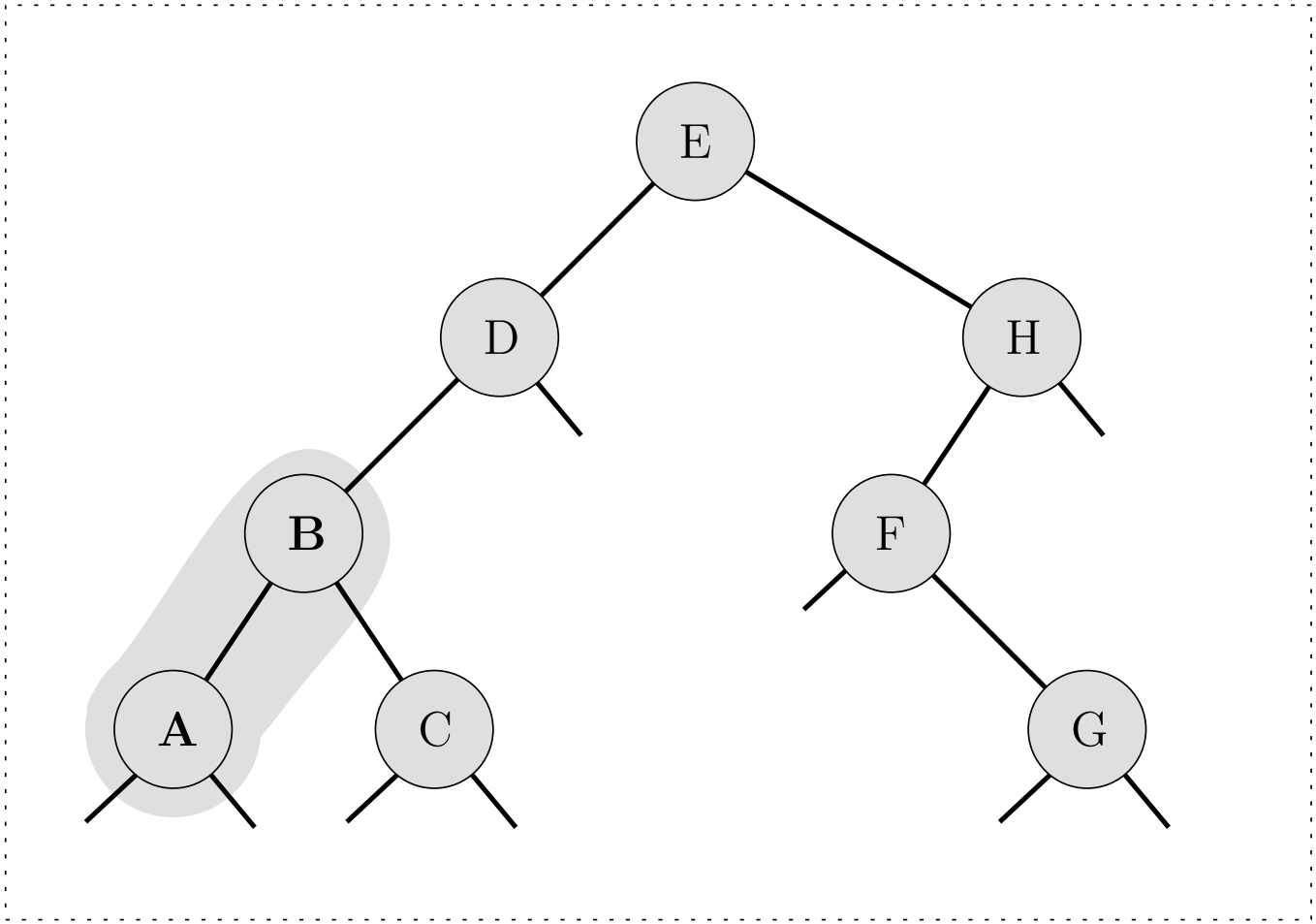
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

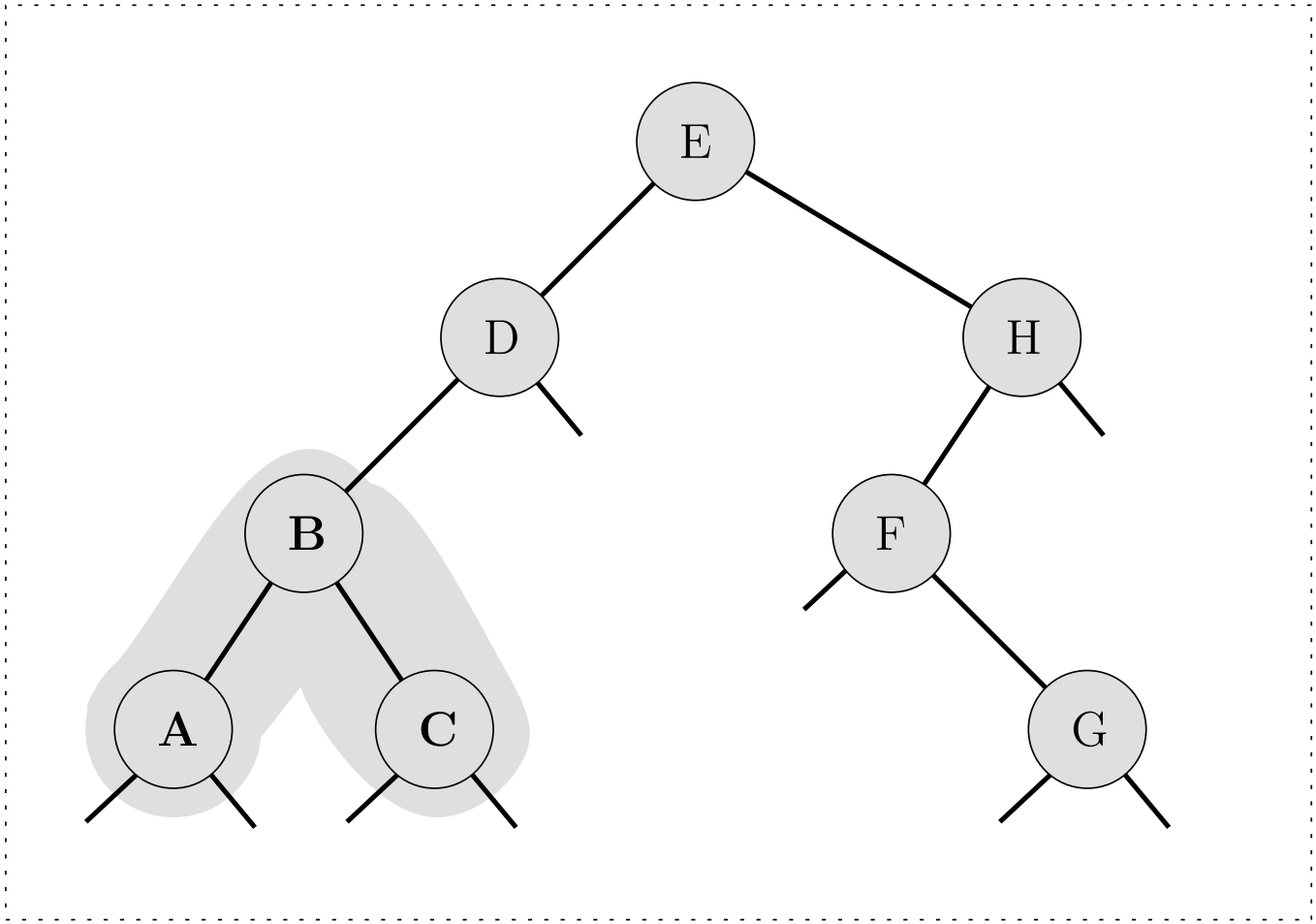
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

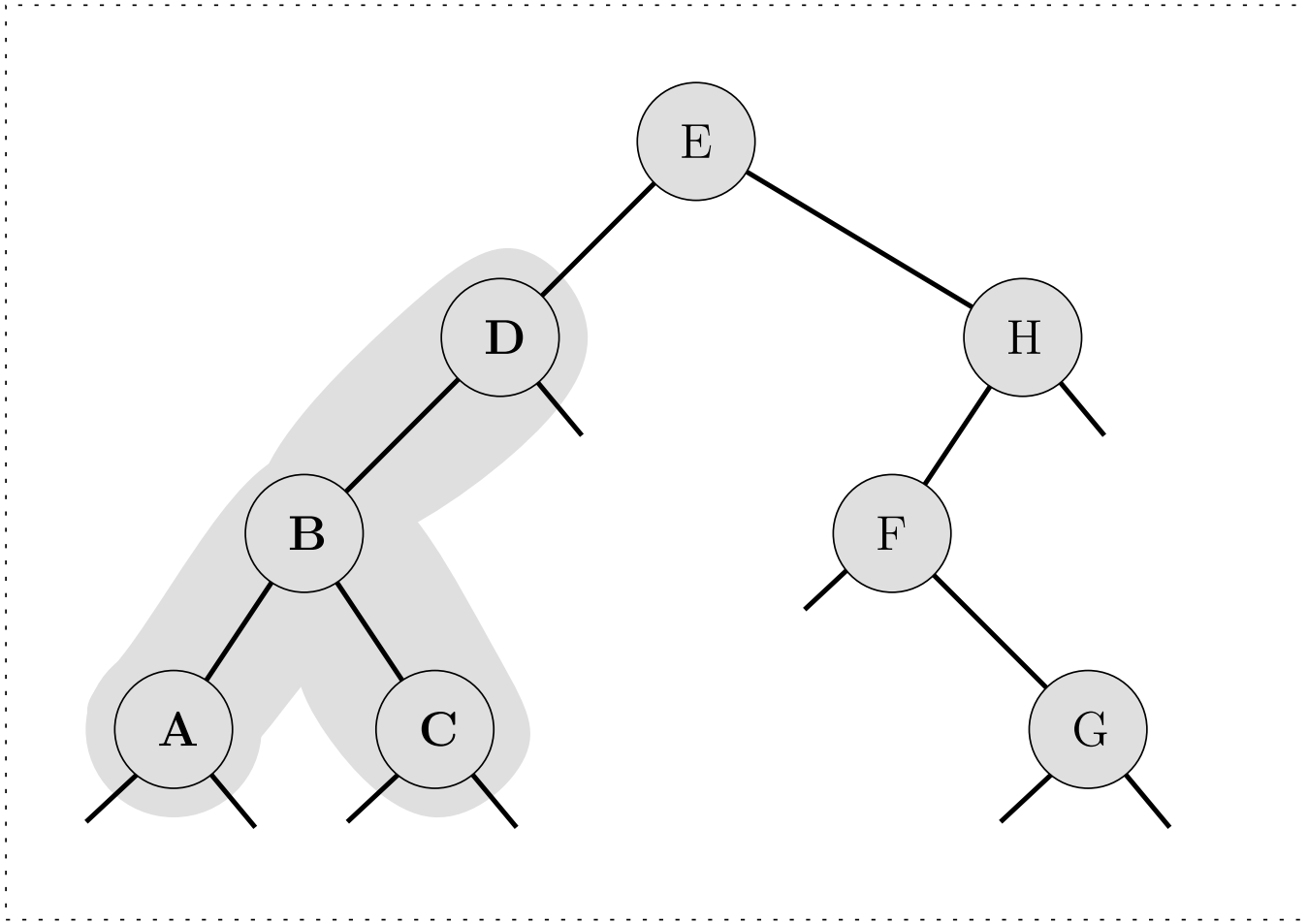
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

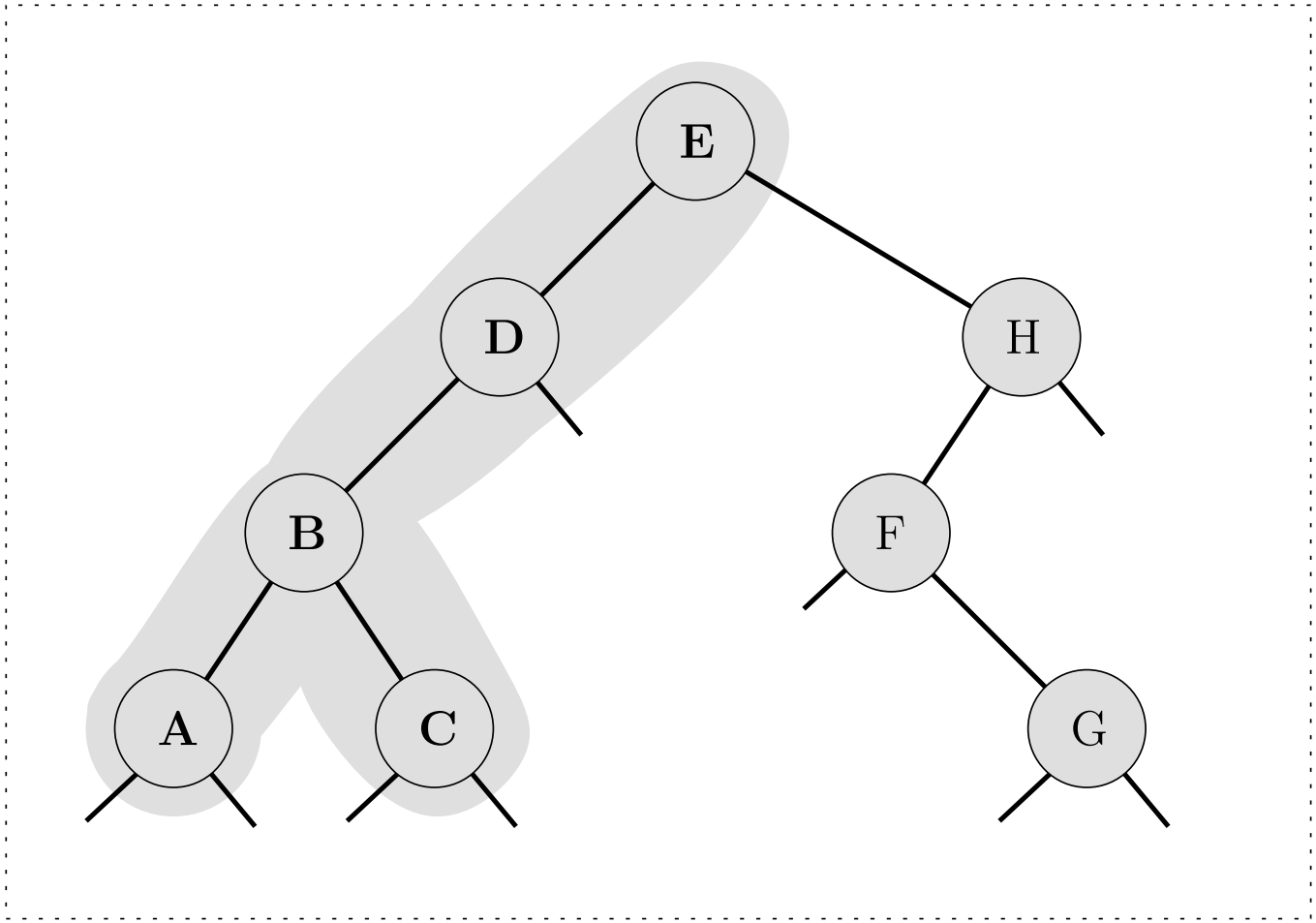
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

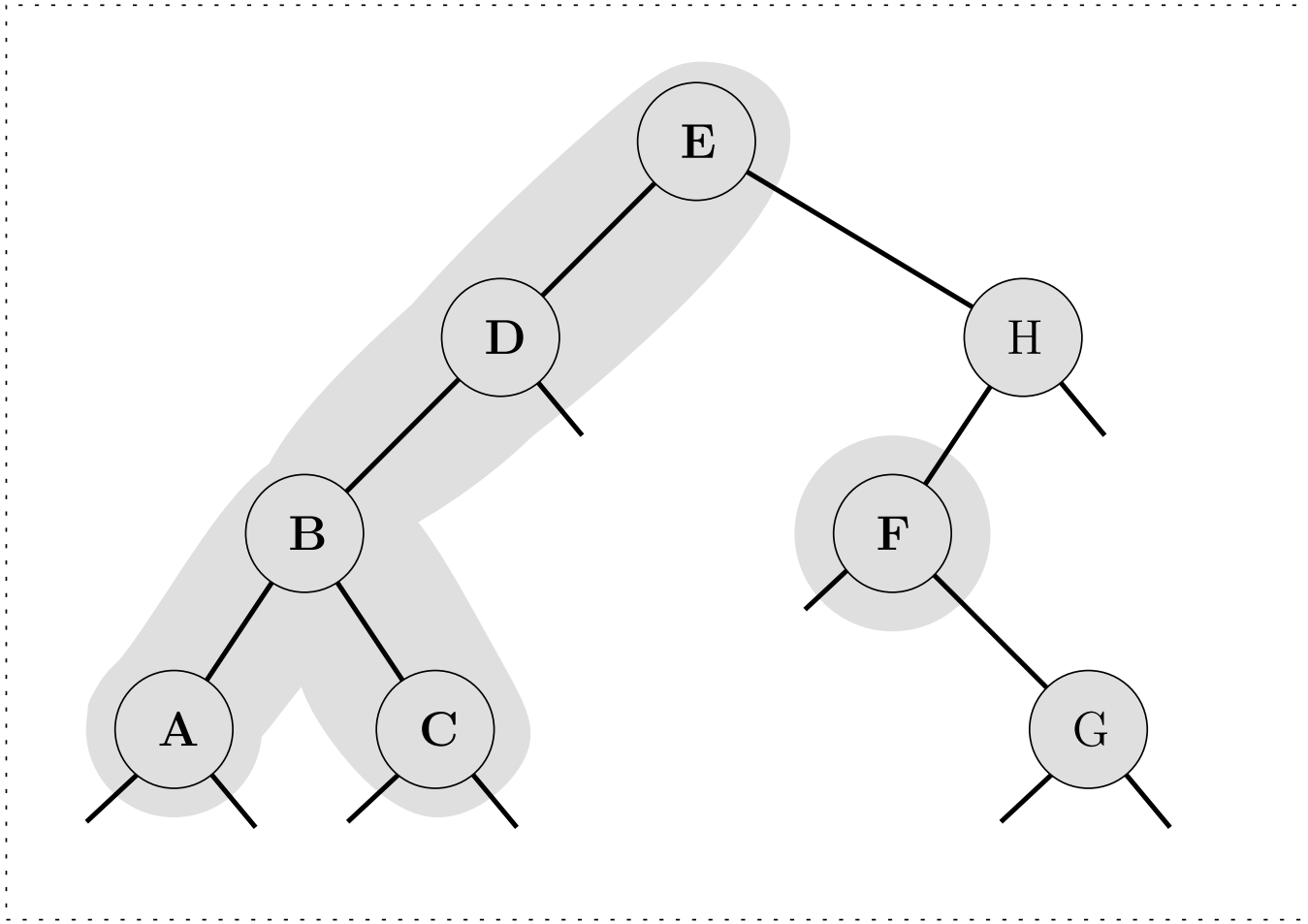
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

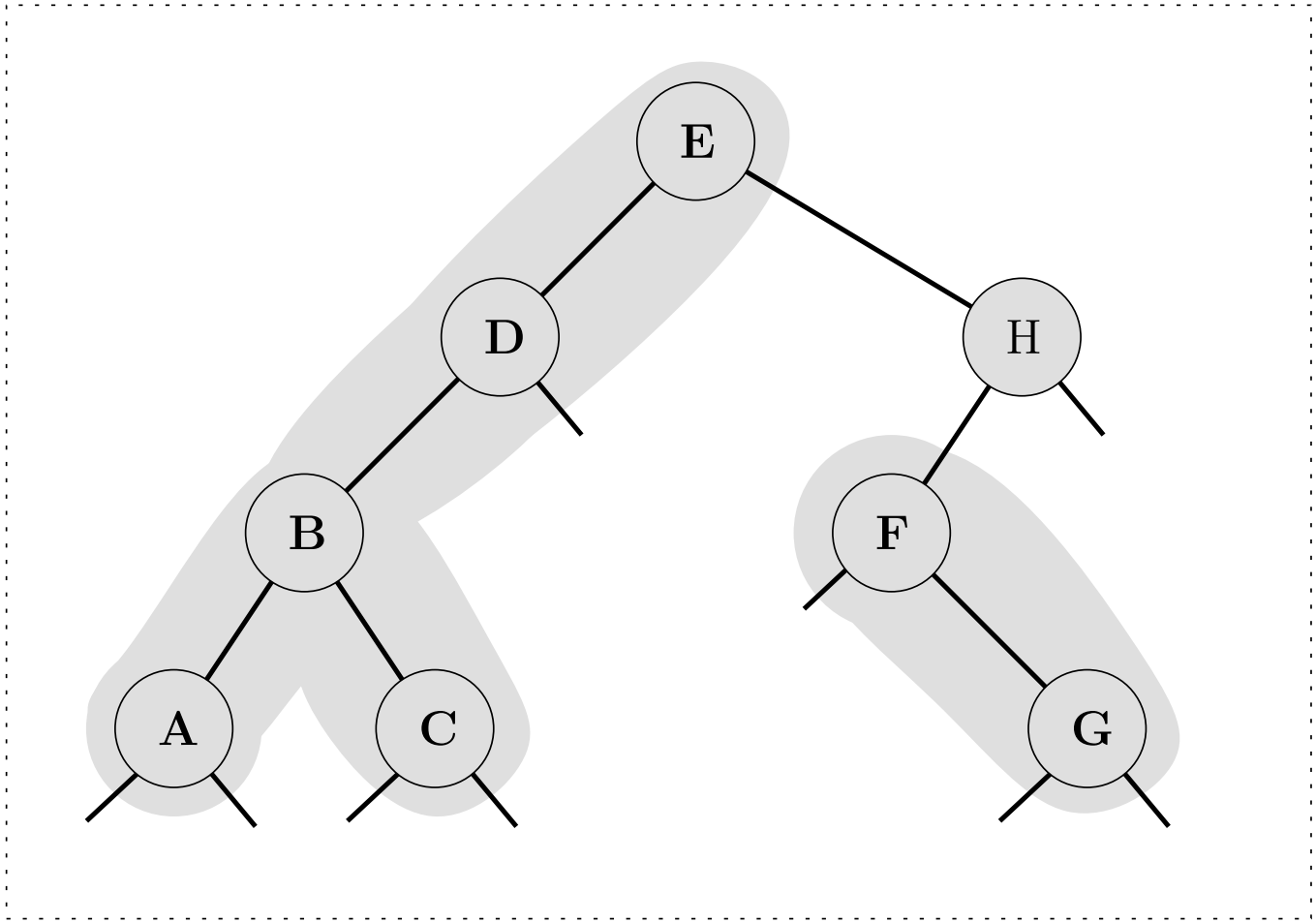
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

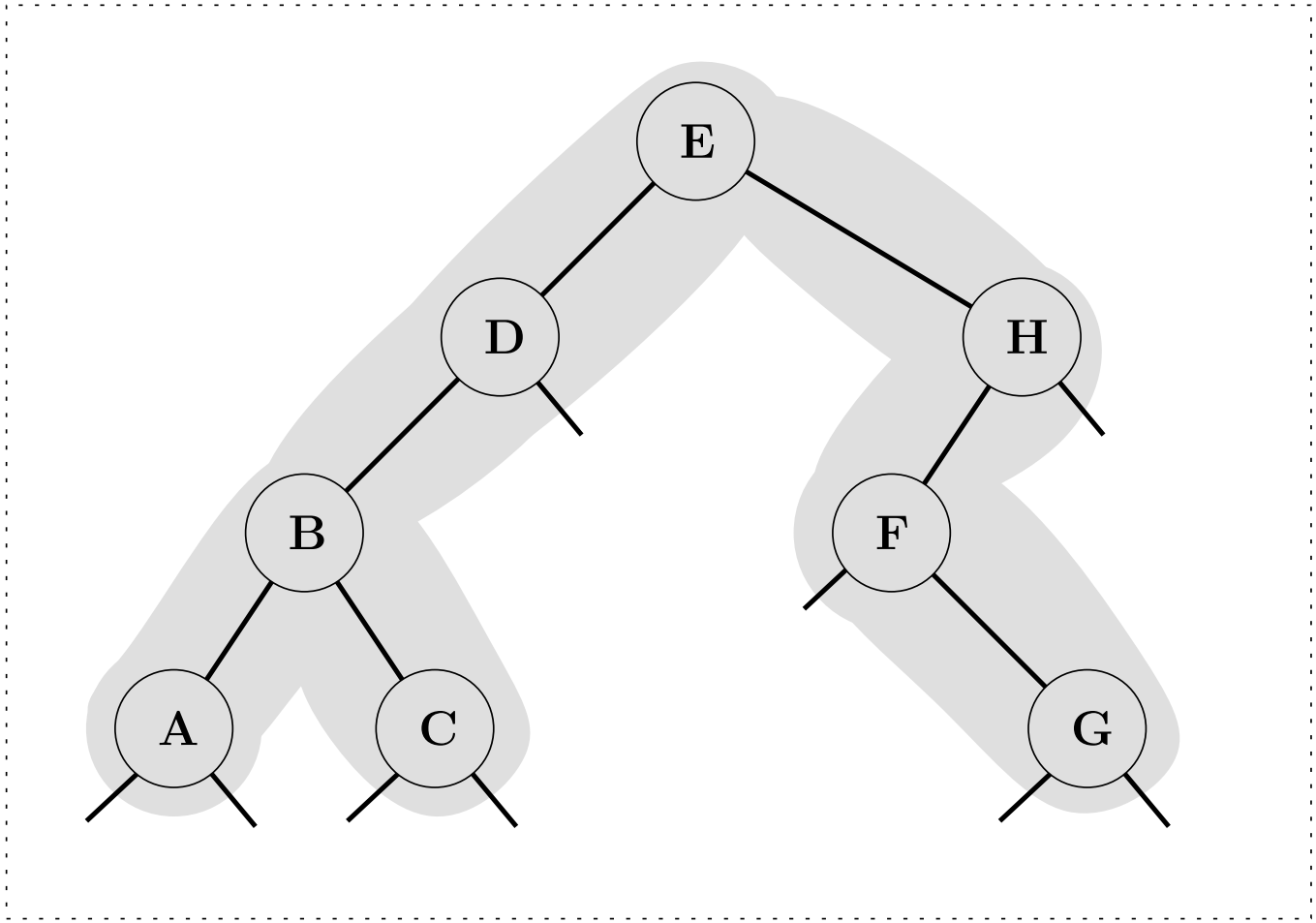
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο



inorder

κανόνας

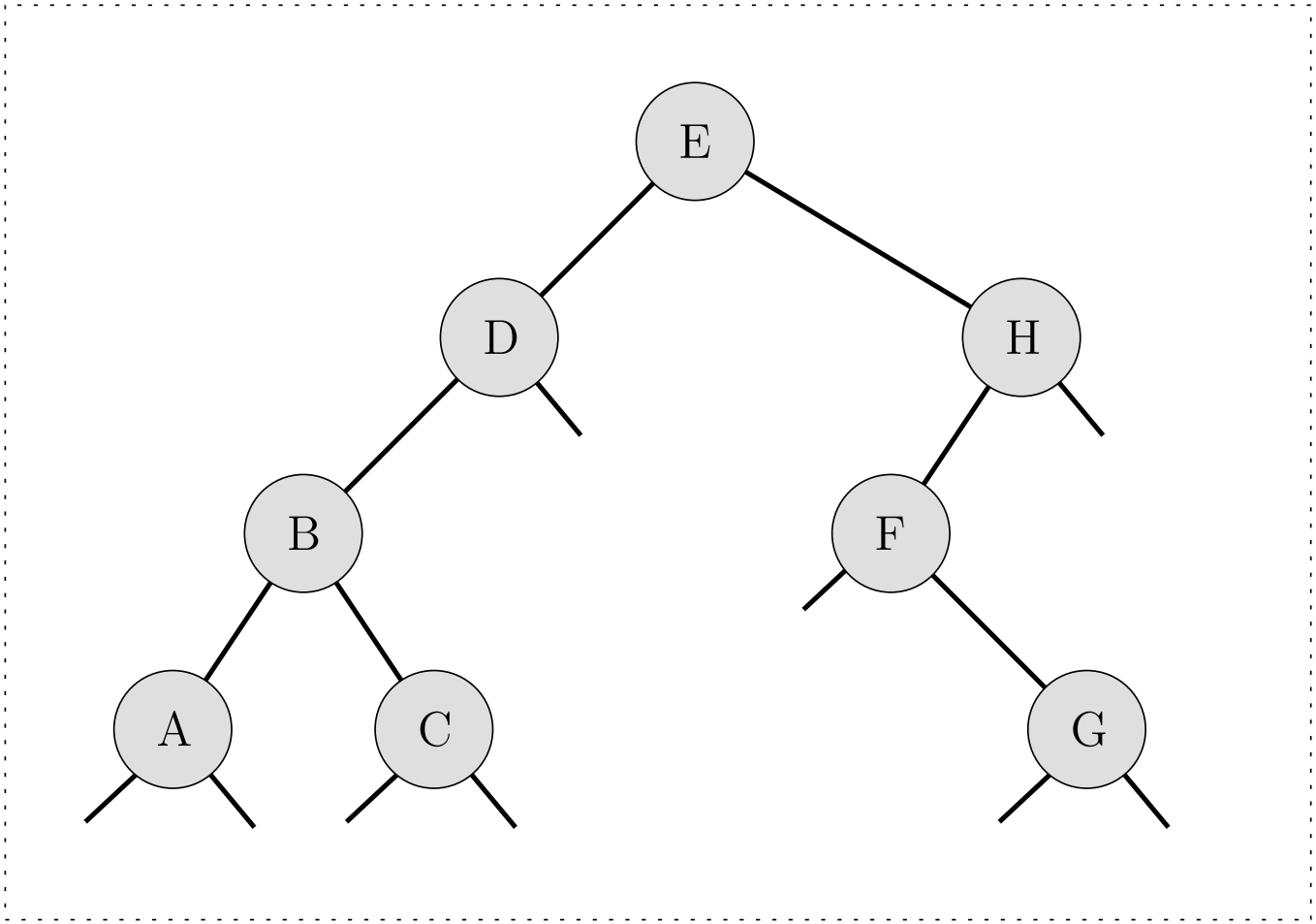
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά τον κόμβο και μετά το δεξιό υποδέντρο

```
1  void traverse( node t )
2  {
3      if ( t == NULL ) return;
4      traverse( t->left );
5      visit( t );
6      traverse( t->right );
7  }
```

postorder

κανόνας

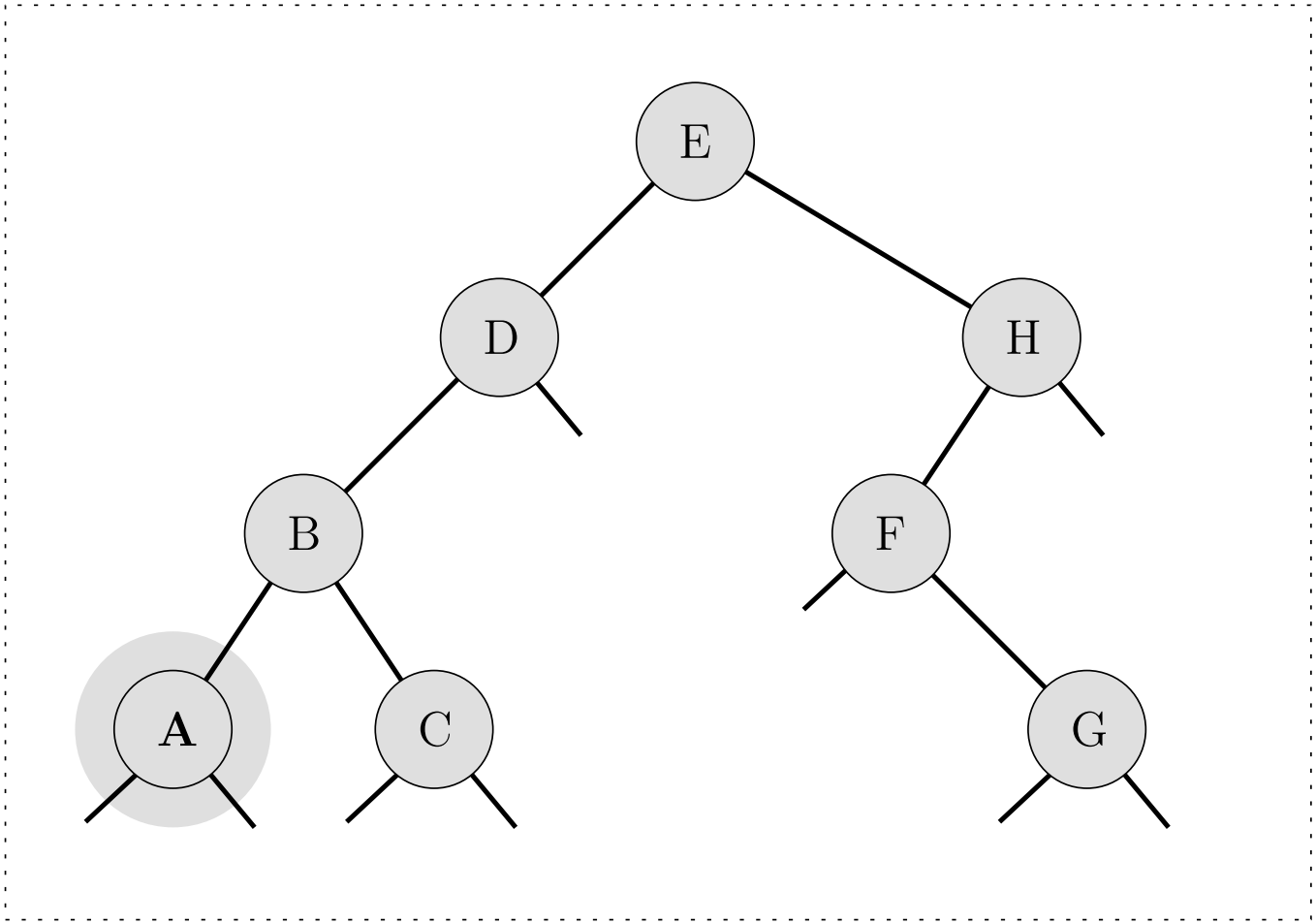
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

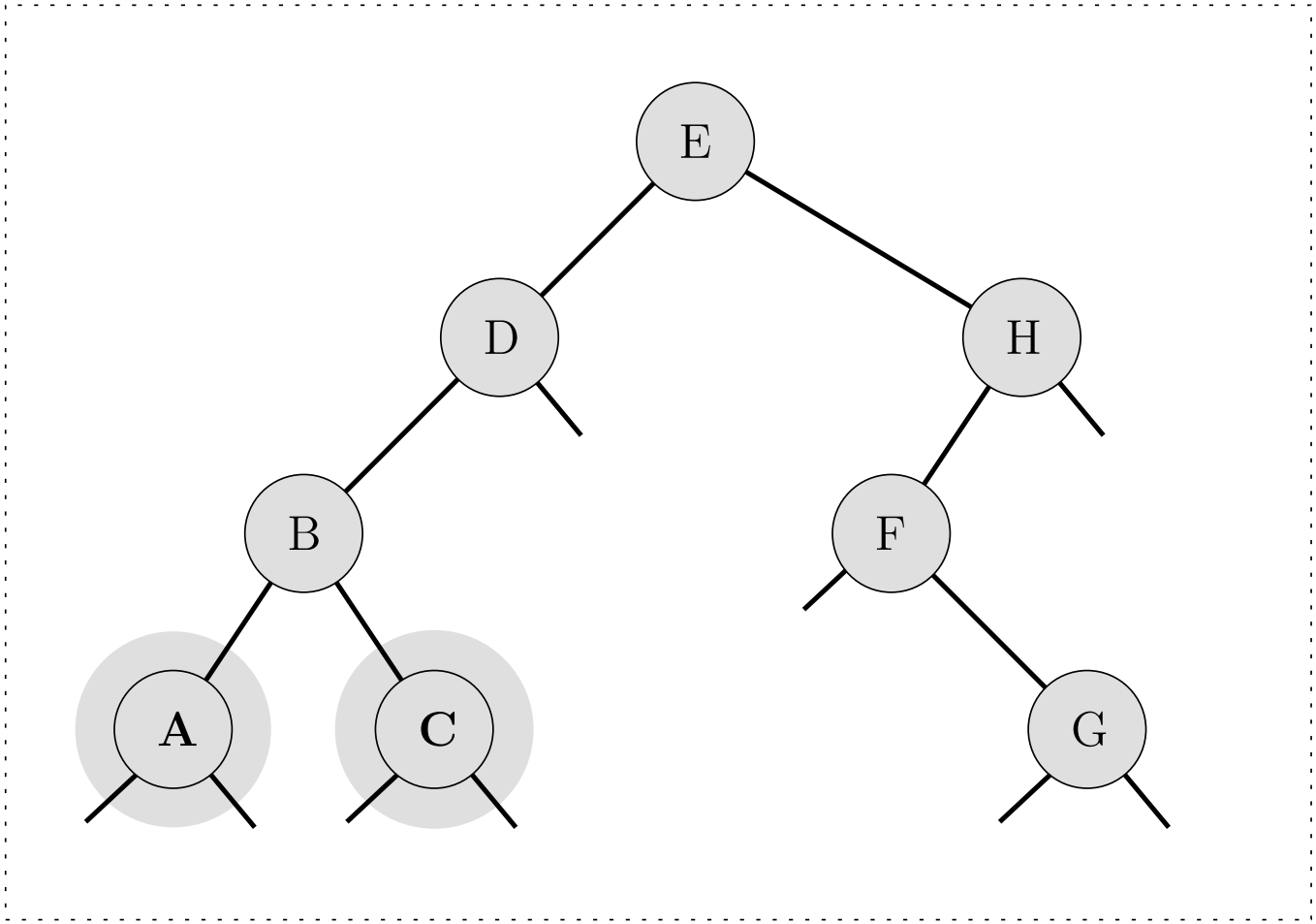
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

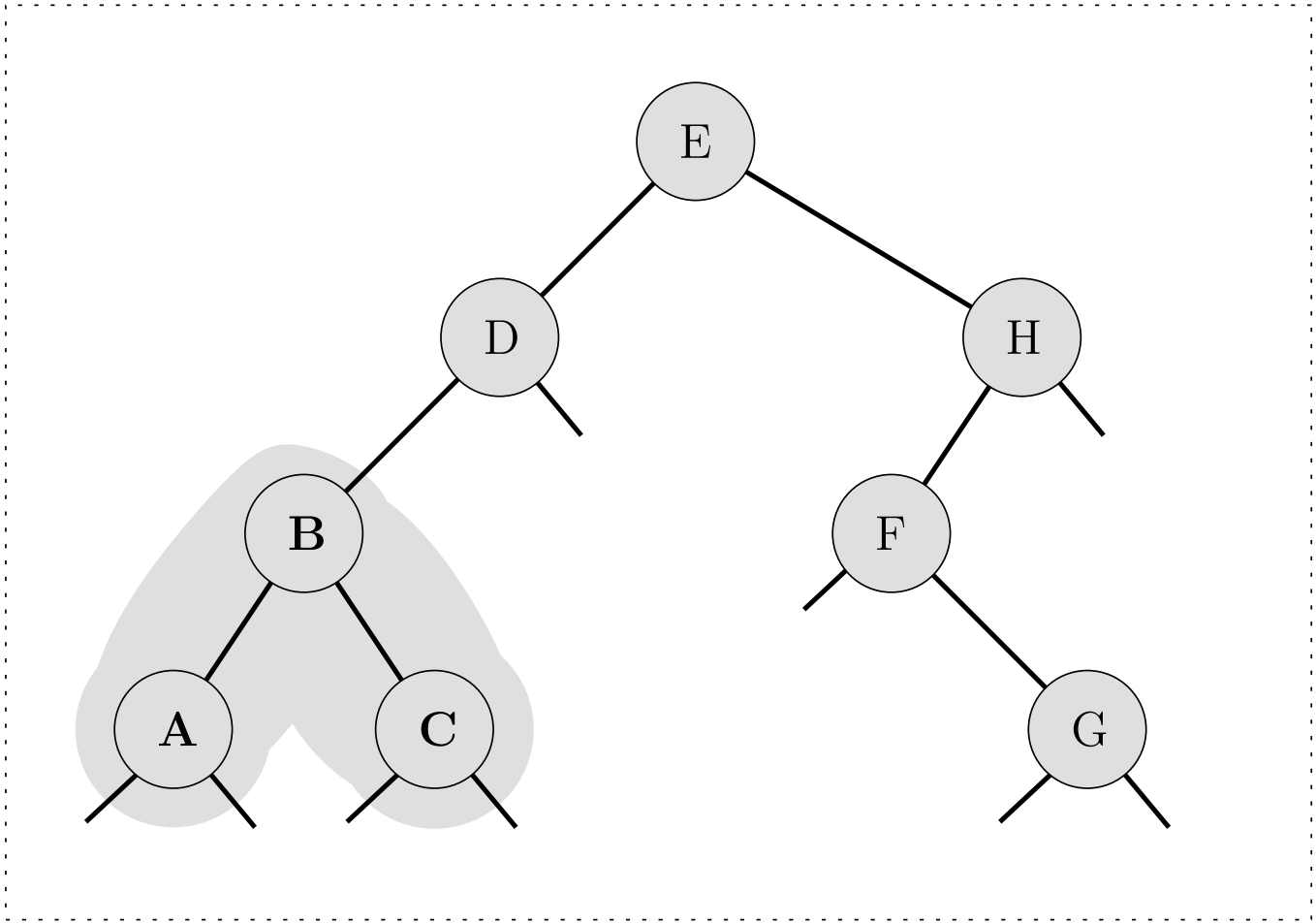
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

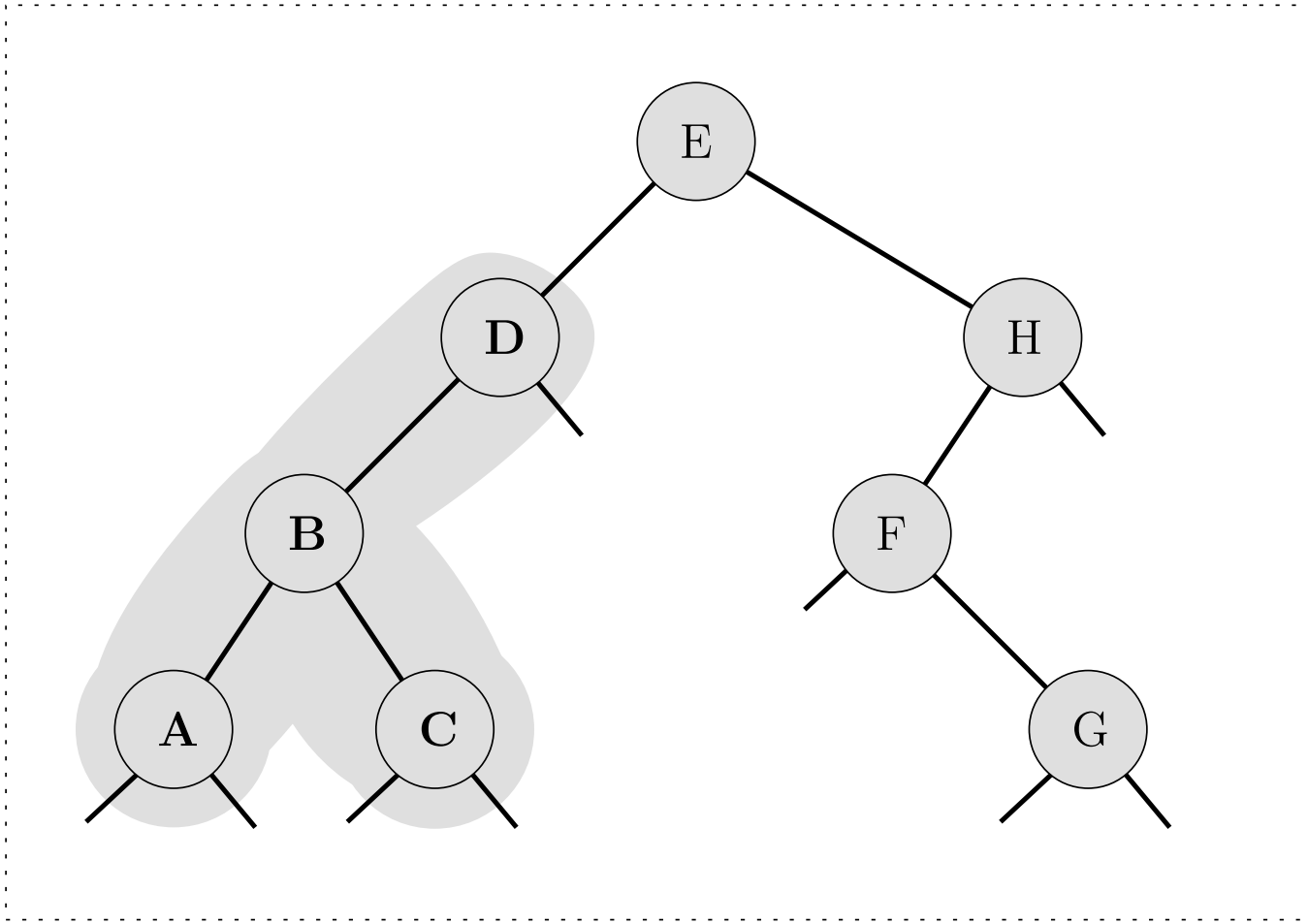
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

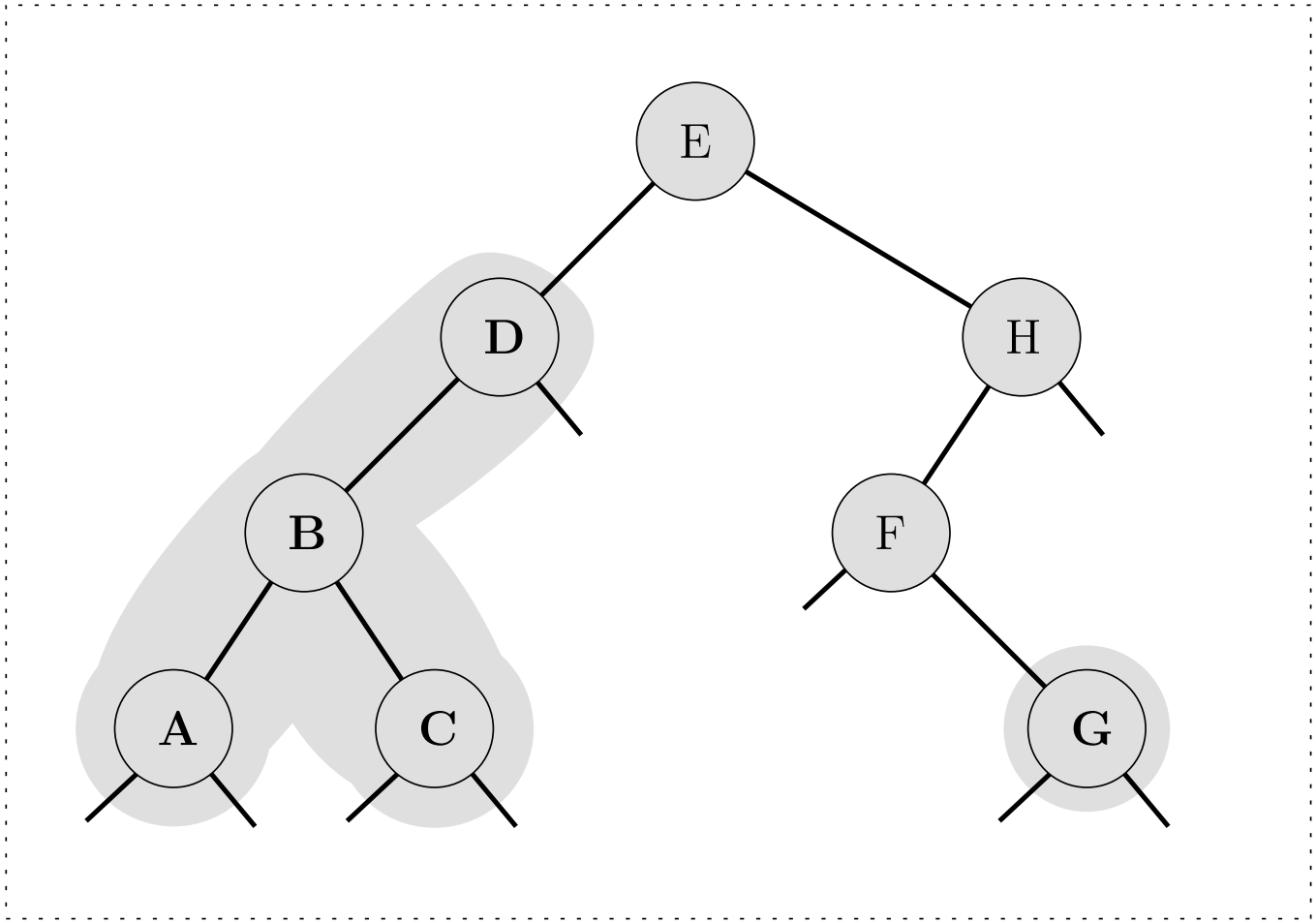
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

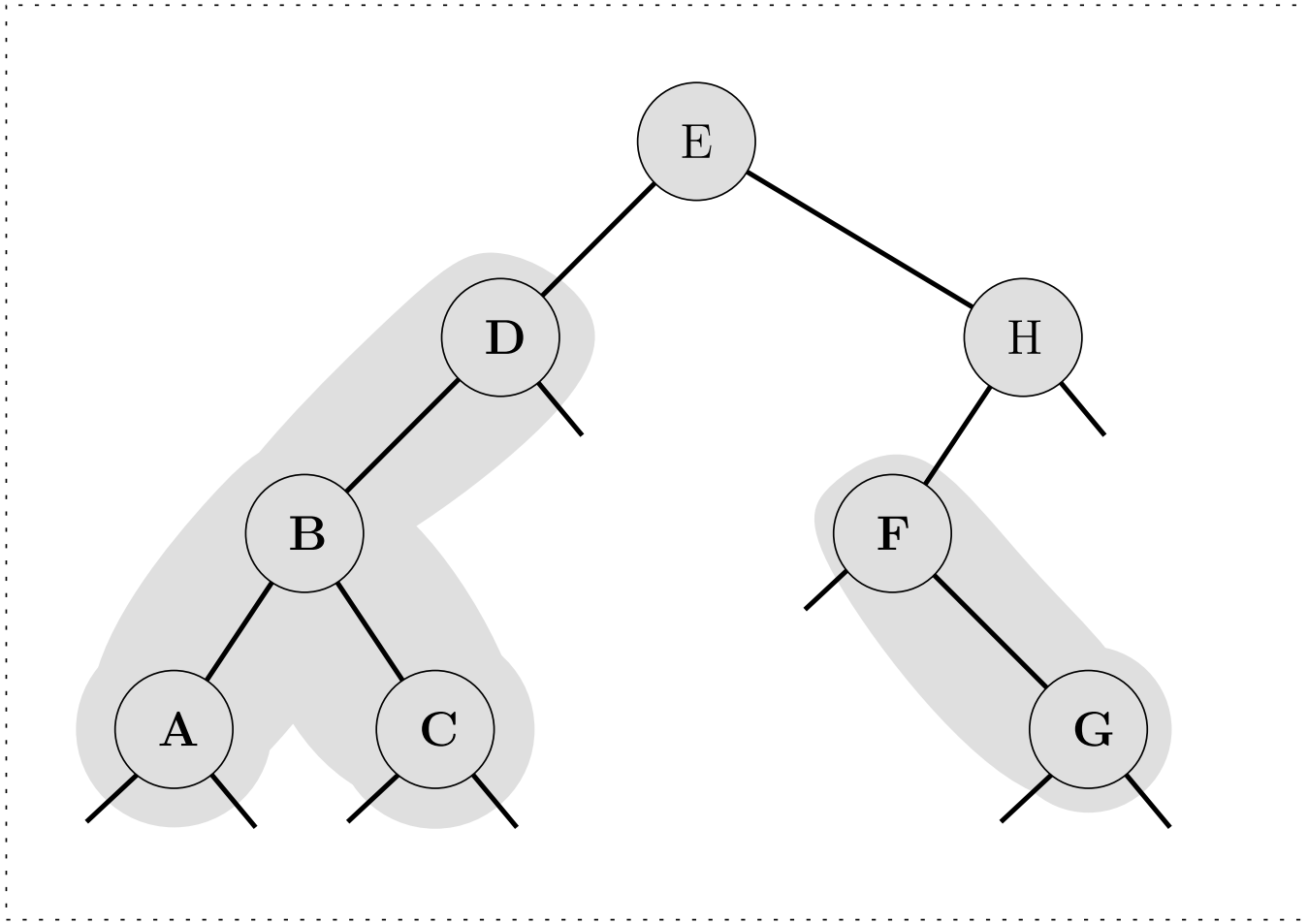
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

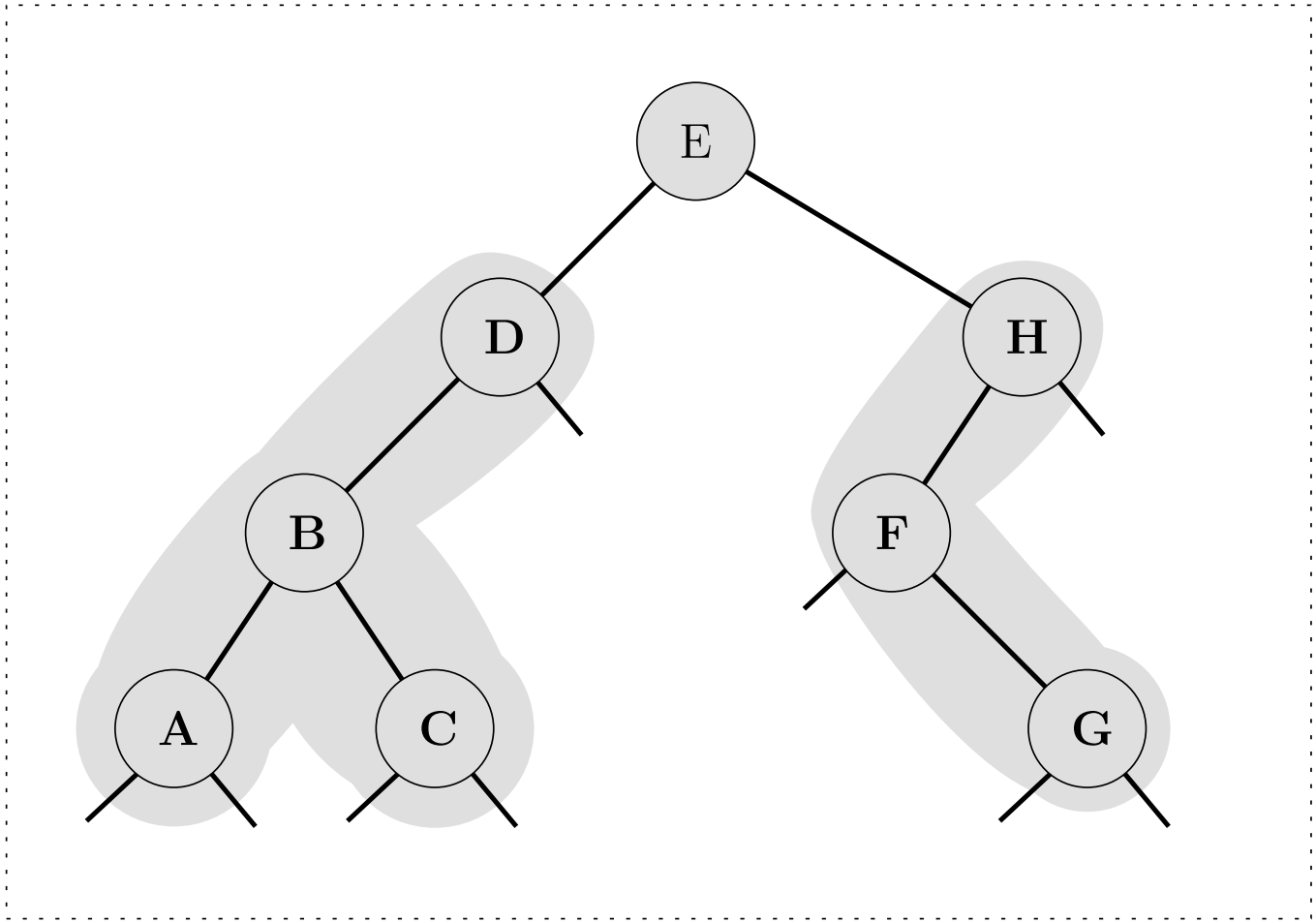
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

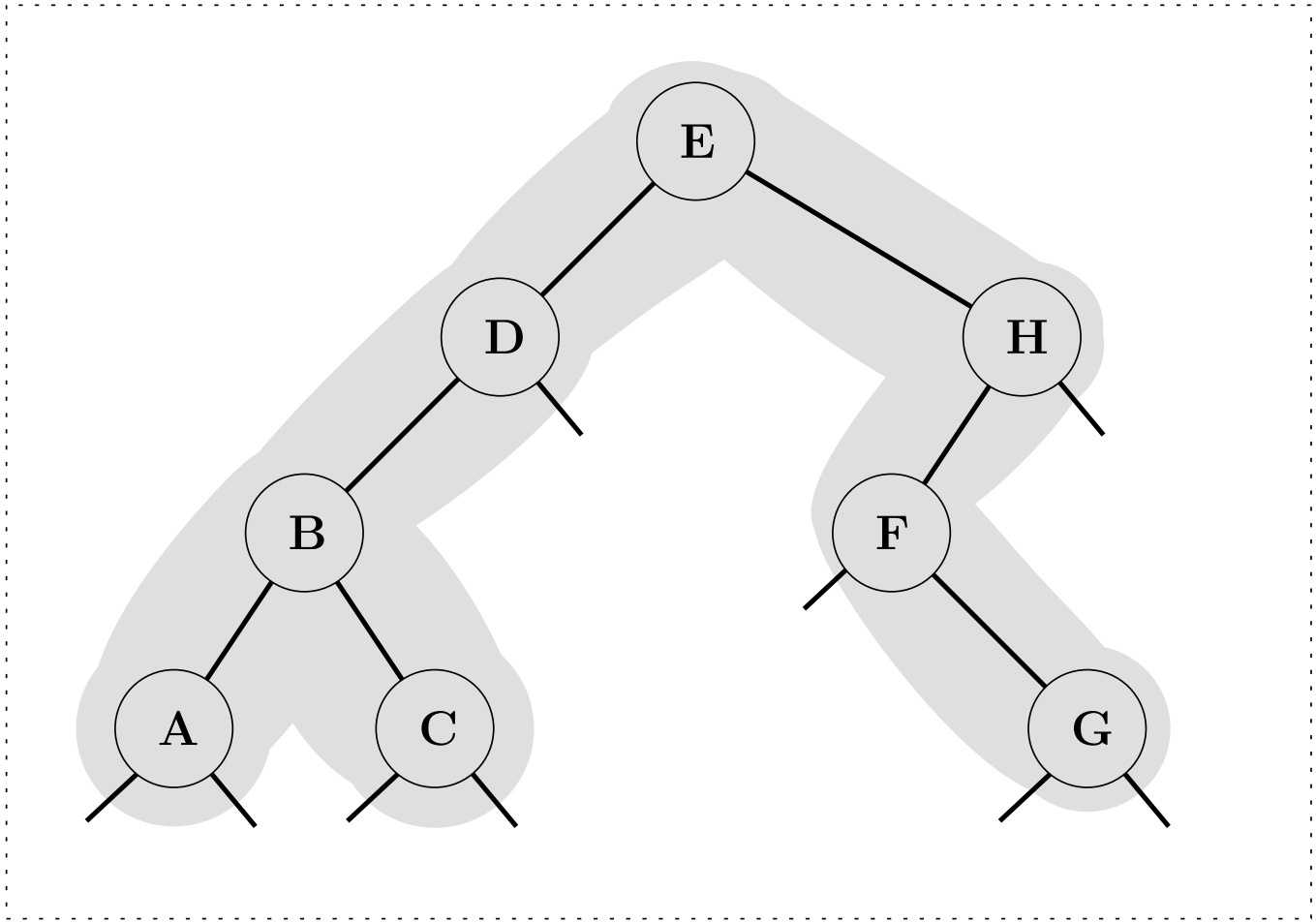
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο



postorder

κανόνας

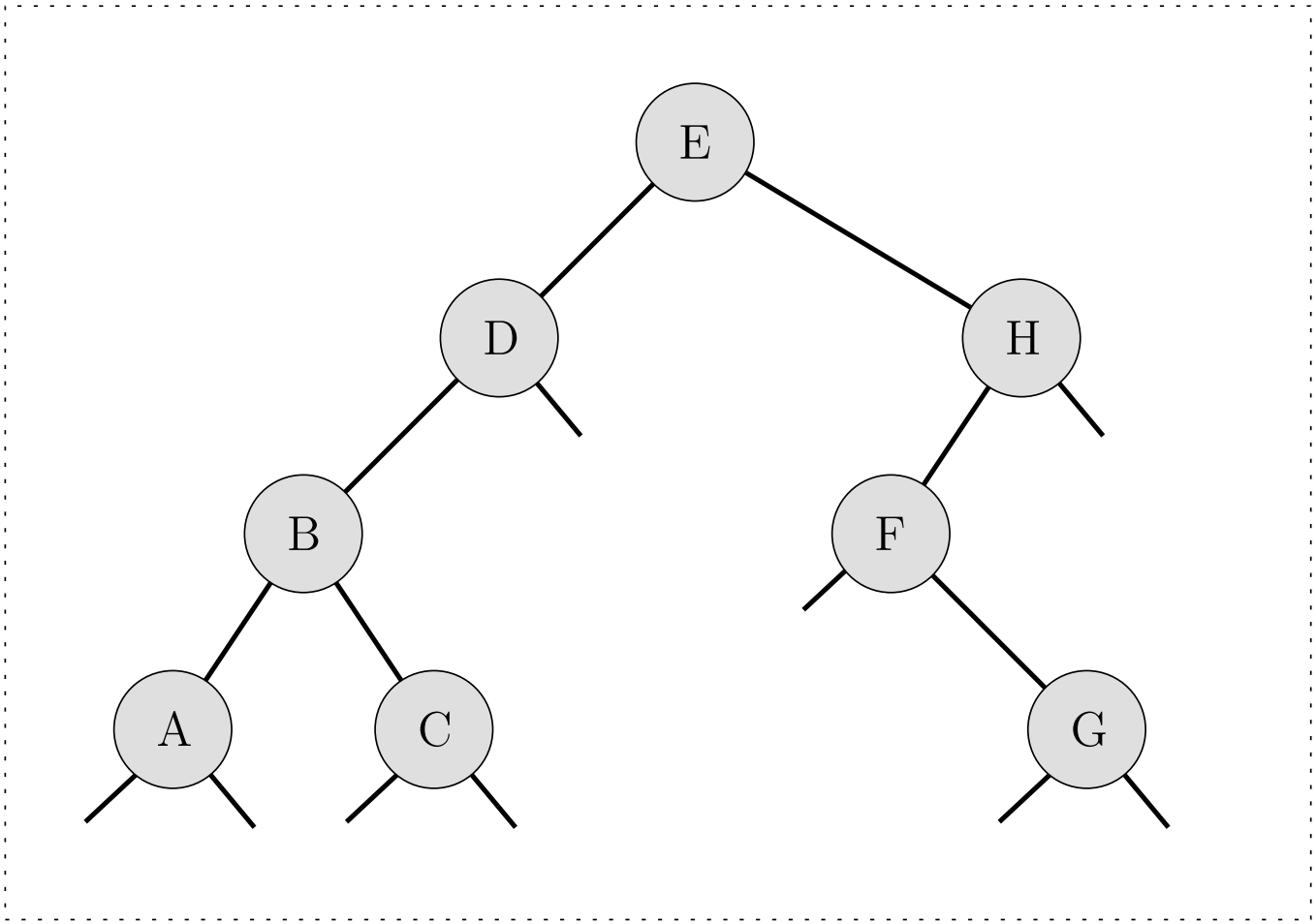
Επισκεπτόμαστε πρώτα το αριστερό υποδέντρο, μετά το δεξιό υποδέντρο και τέλος τον κόμβο

```
1  void traverse( node t )
2  {
3      if ( t == NULL ) return;
4      traverse( t->left );
5      traverse( t->right );
6      visit( t );
7  }
```

level-order

κανόνας

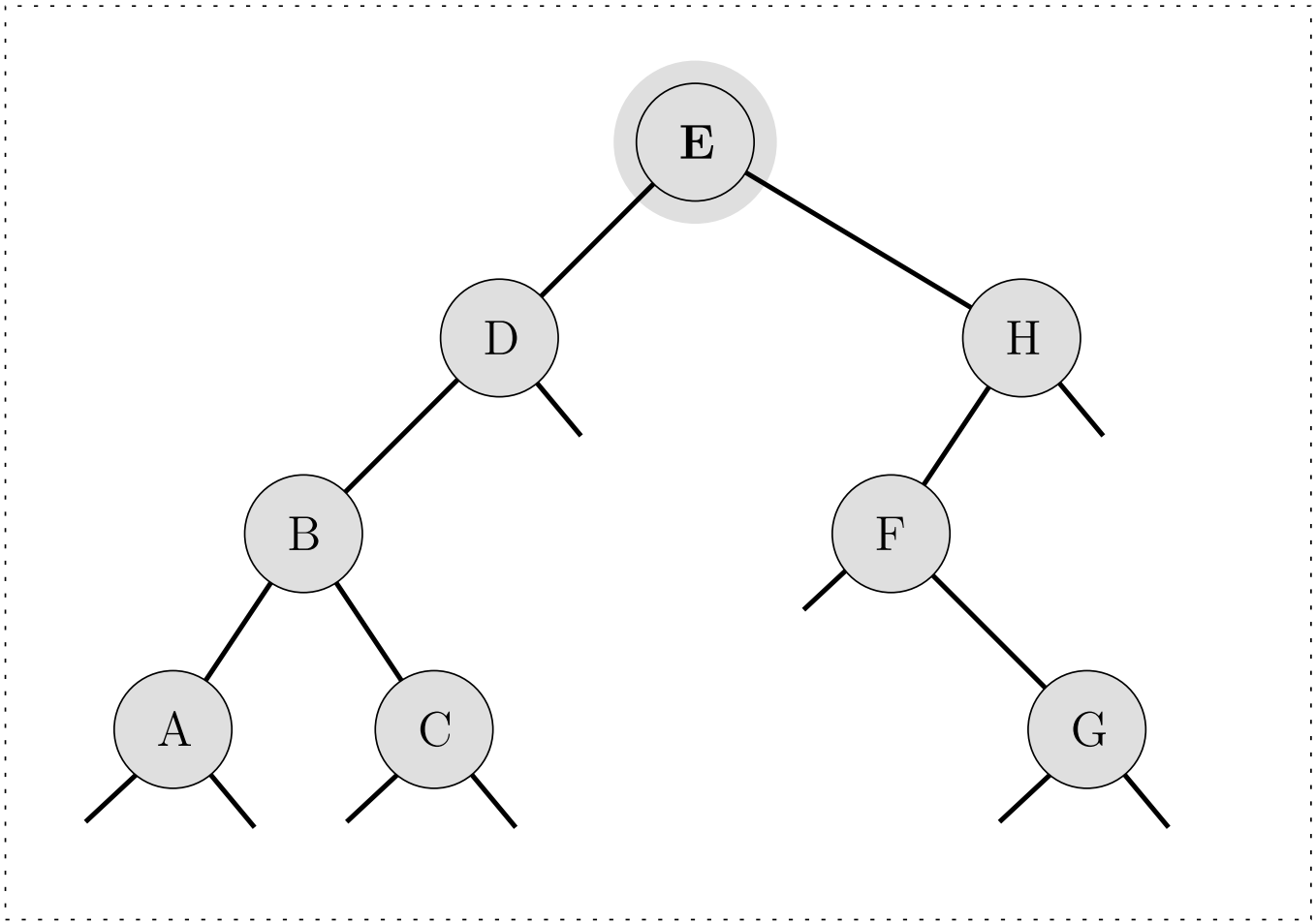
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

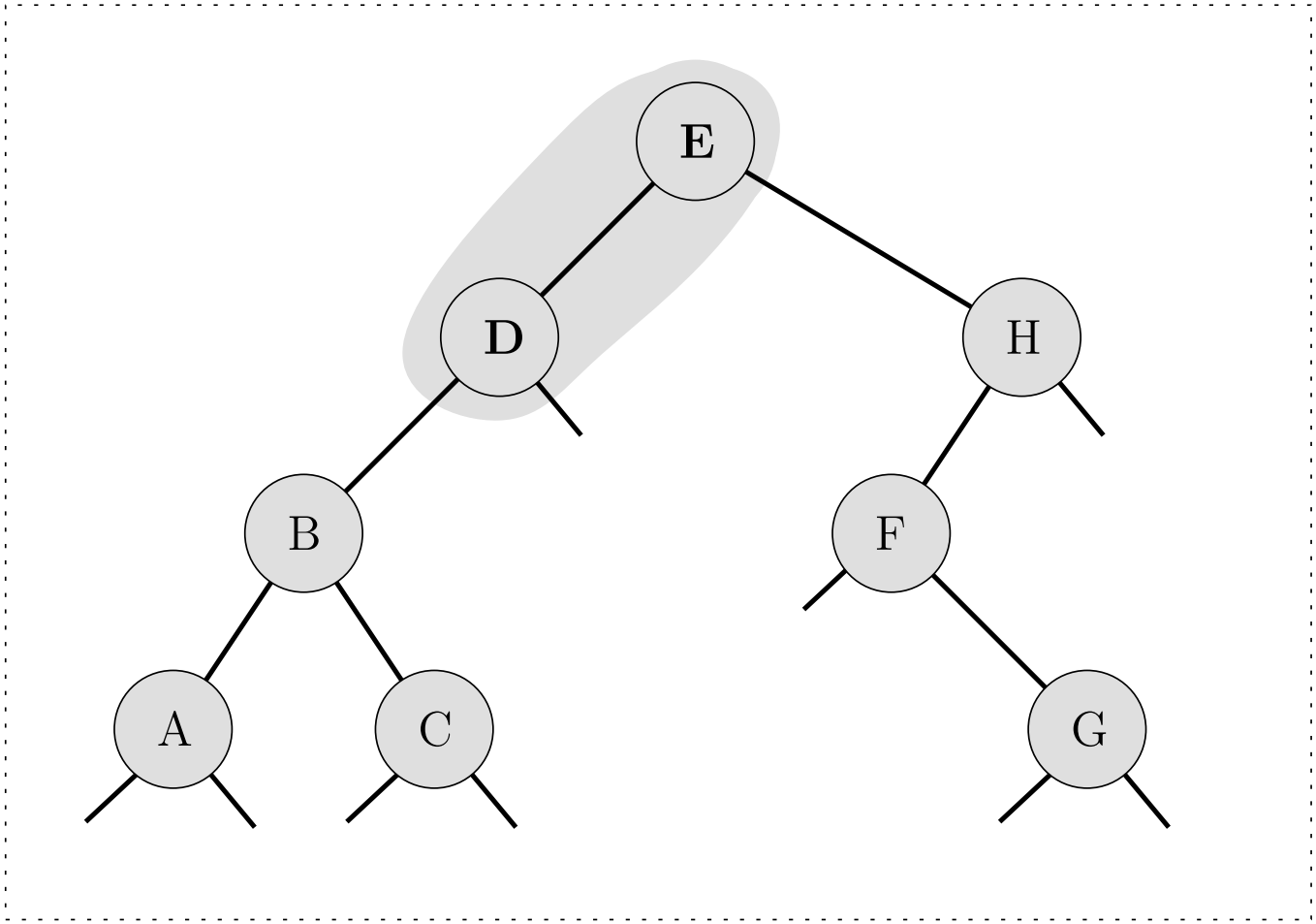
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

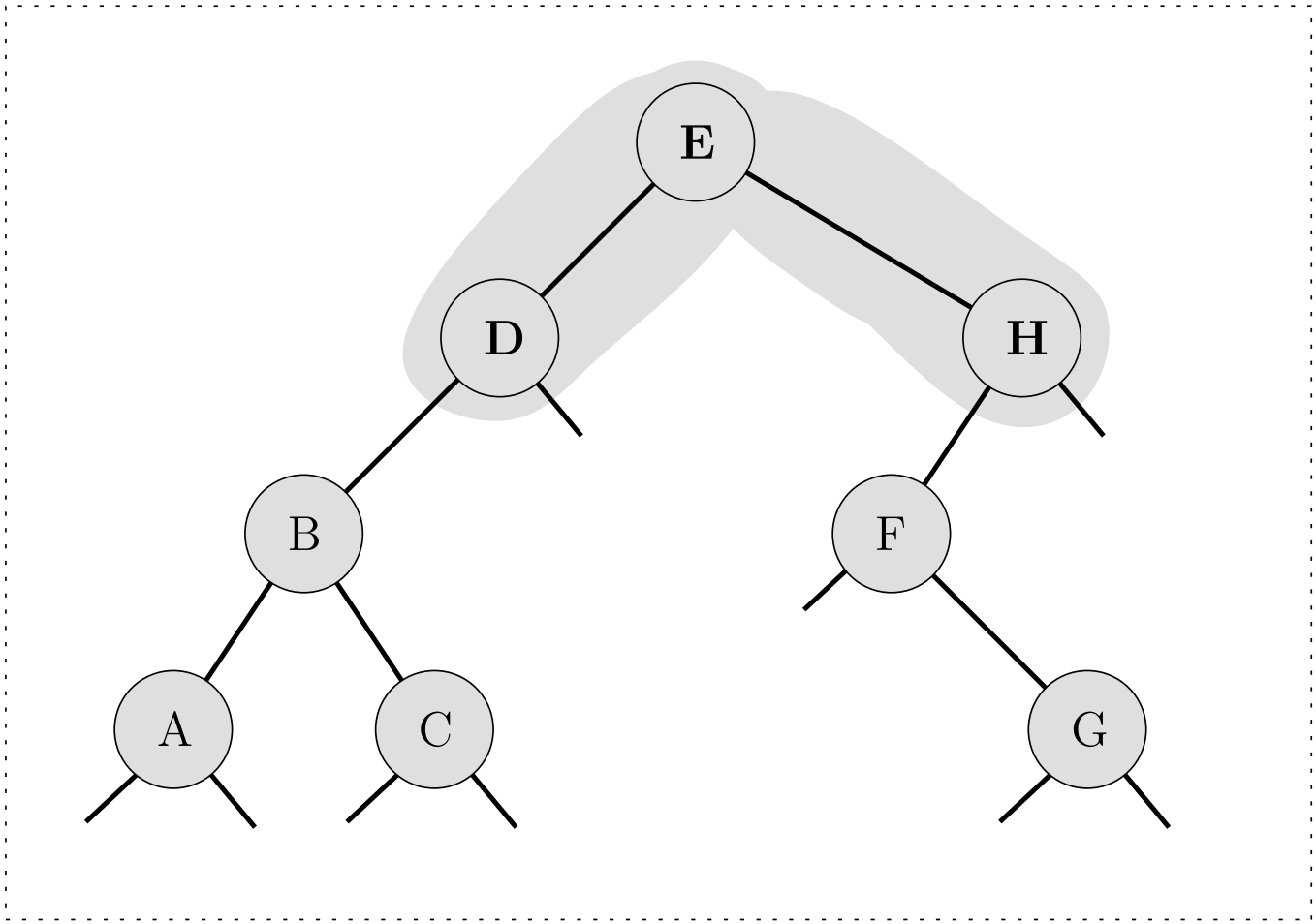
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

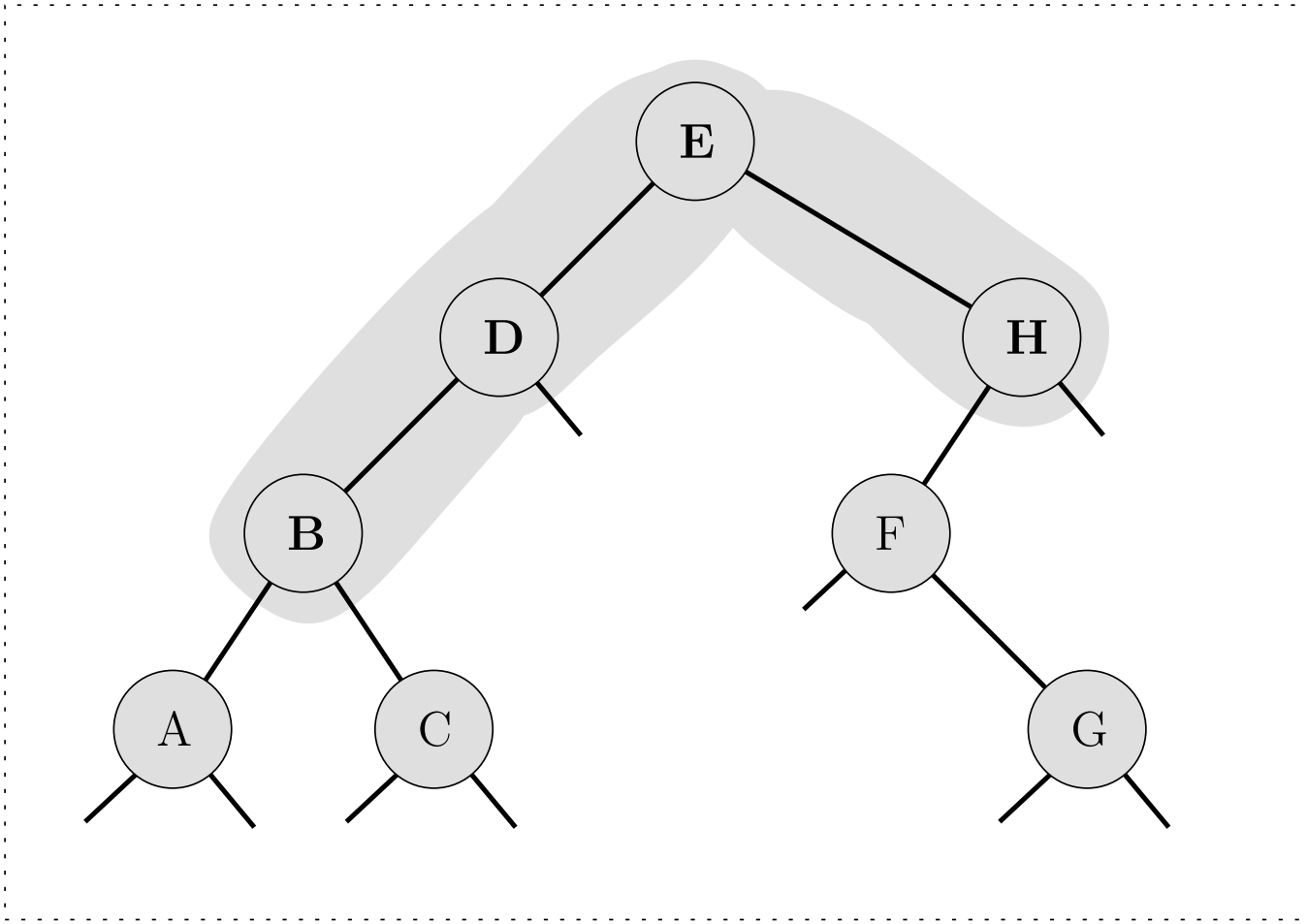
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

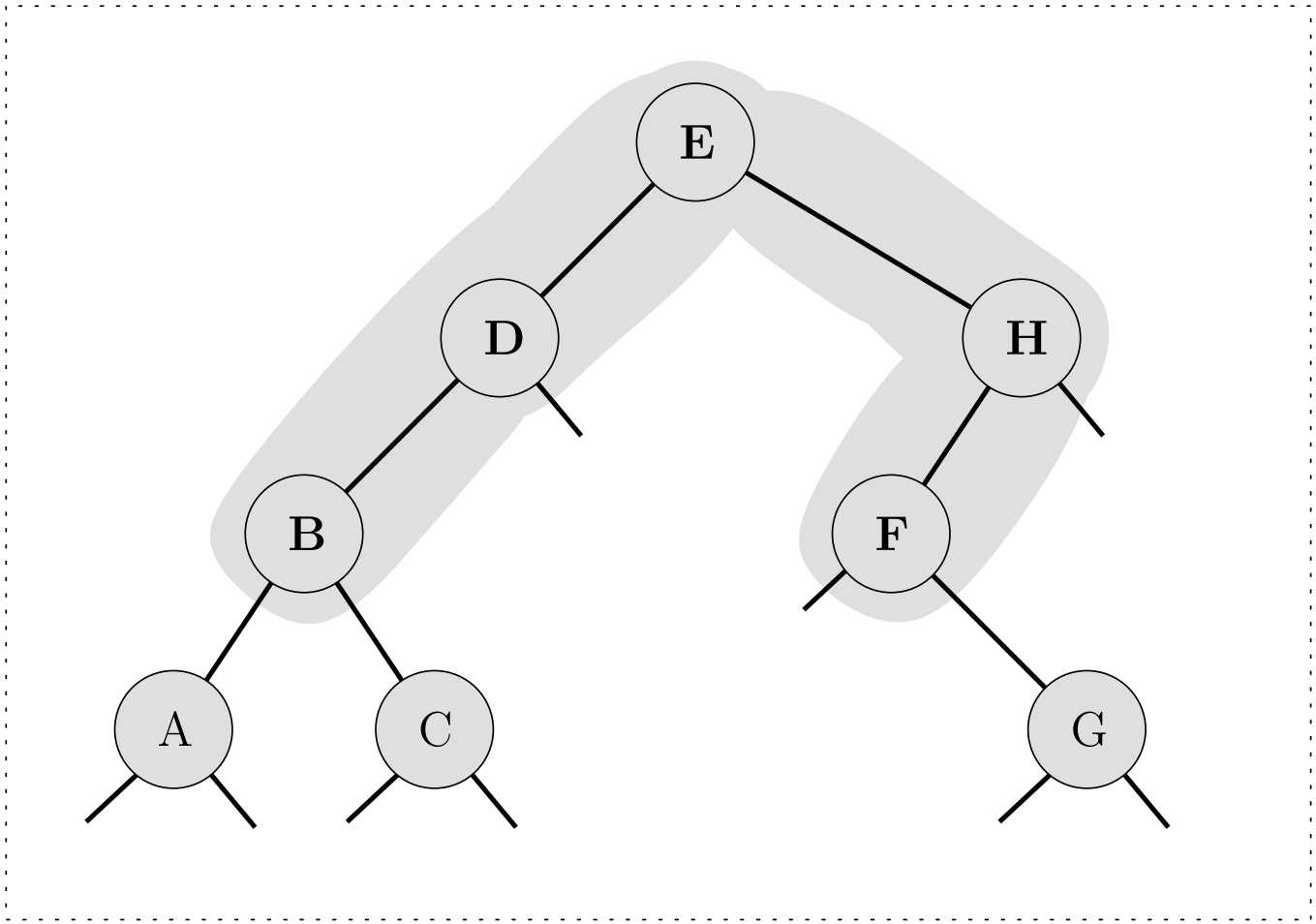
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

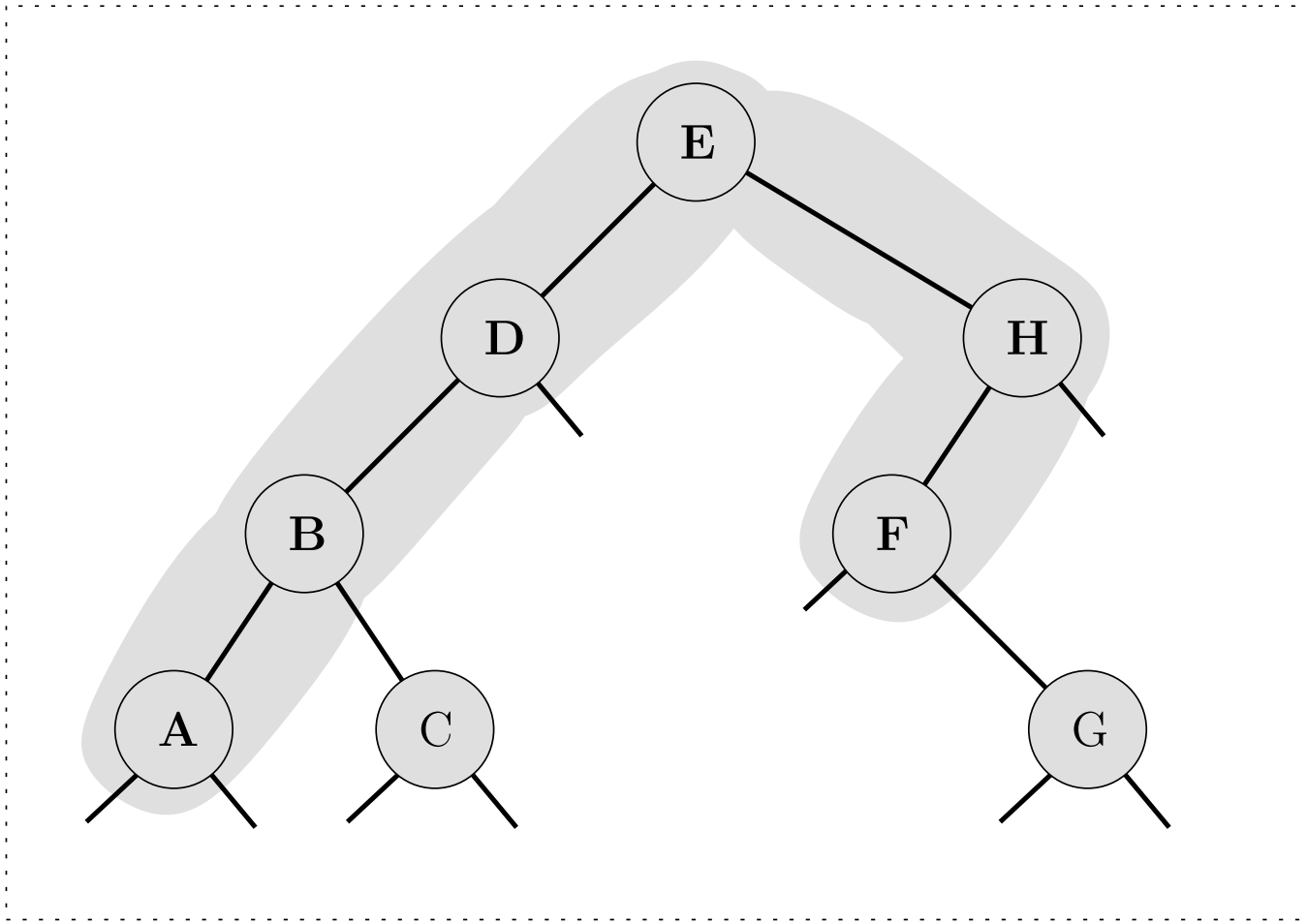
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

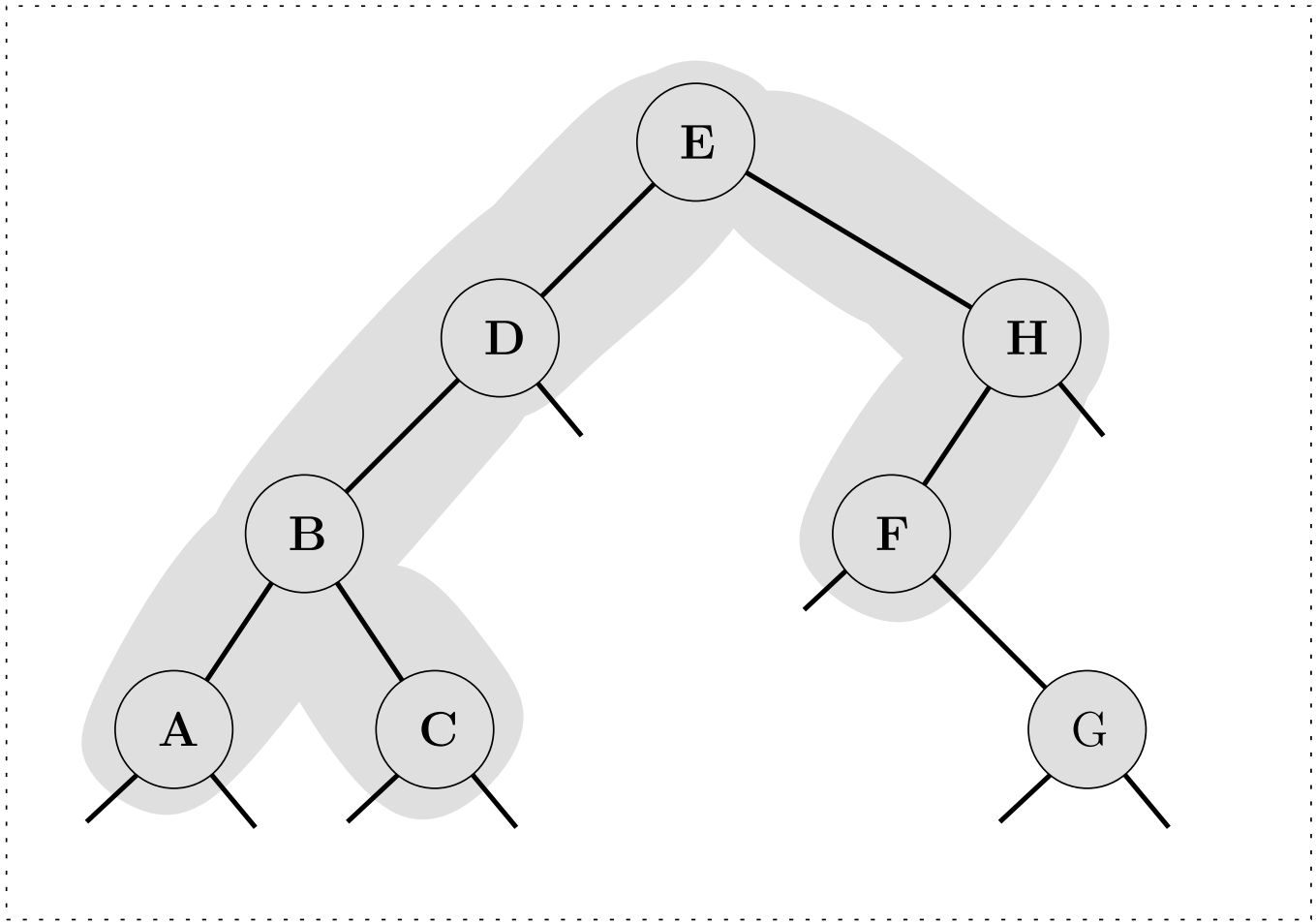
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

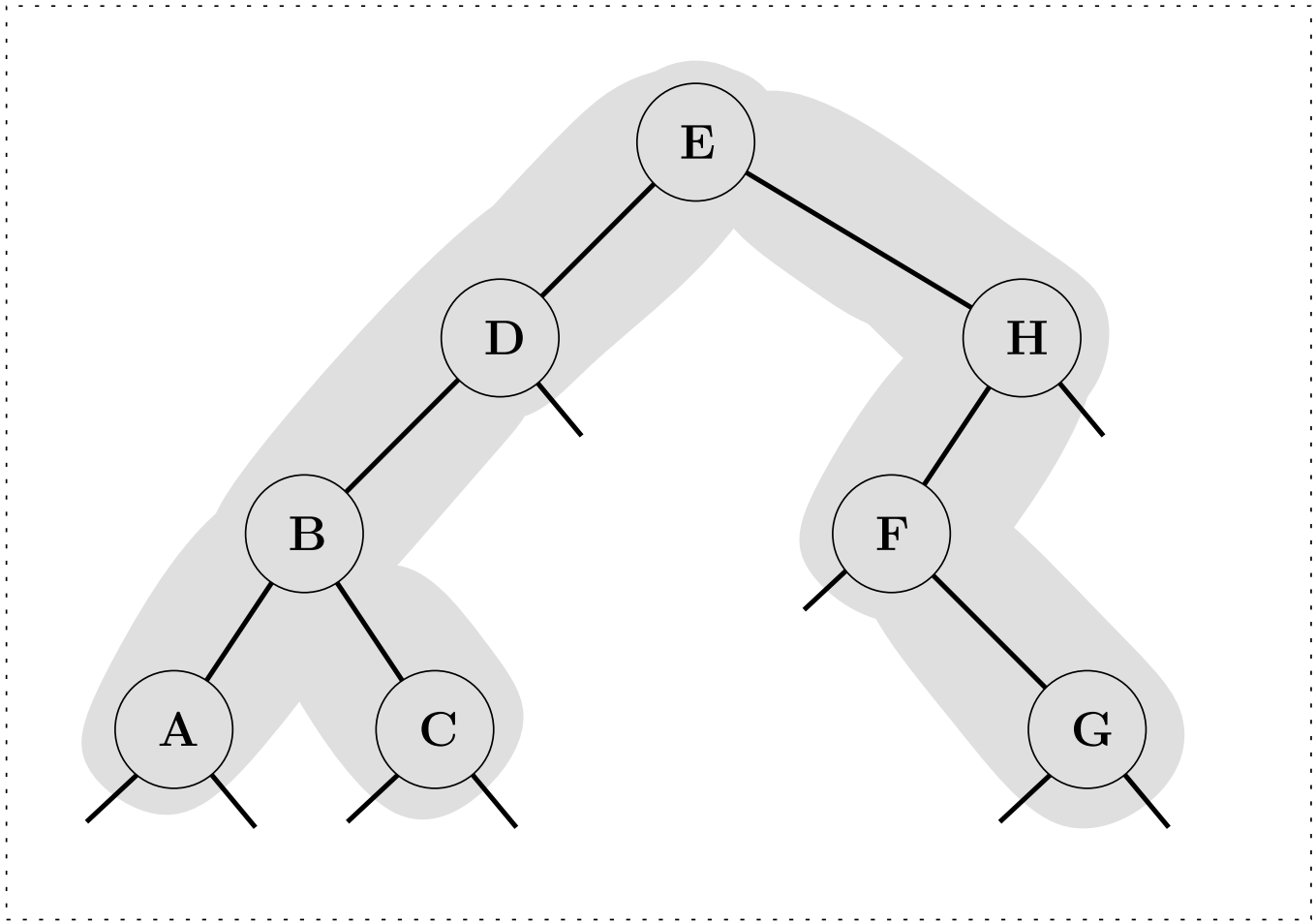
Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά



level-order

κανόνας

Επισκεπτόμαστε τους κόμβους ανά επίπεδο, από αριστερά προς τα δεξιά

```
1  void traverse( node t )
2  {
3      node h;
4      initQueue();
5      putQueue( t );
6
7      while( ! emptyQueue() )
8      {
9          h = getQueue();
10         visit( h );
11         if ( h->left != NULL ) putQueue( h->left );
12         if ( h->right != NULL ) putQueue( h->right );
13     }
14 }
```

Εφαρμογή Δέντρων: κώδικες Huffman

Για την αναπαράσταση χαρακτήρων στον υπολογιστή χρησιμοποιούμε κάποια κωδικοποίηση:

- κώδικας ASCII (7-bits)
- κώδικας Extended-ASCII (8-bits)
- Unicode (16-bits)

Για να αναπαραστήσουμε λοιπόν σε Extended-ASCII ένα κείμενο με 1000 χαρακτήρες χρειαζόμαστε 8000 bits.

Εφαρμογή Δέντρων: κώδικες Huffman

Για την αναπαράσταση χαρακτήρων στον υπολογιστή χρησιμοποιούμε κάποια κωδικοποίηση:

- κώδικας ASCII (7-bits)
- κώδικας Extended-ASCII (8-bits)
- Unicode (16-bits)

Για να αναπαραστήσουμε λοιπόν σε Extended-ASCII ένα κείμενο με 1000 χαρακτήρες χρειαζόμαστε 8000 bits.

Η αποκωδικοποίηση είναι εύκολη επειδή γνωρίζουμε πως κάθε 8 bits έχουμε διαβάσει ένα χαρακτήρα. π.χ

$$\begin{aligned} 01100011 \quad 01100001 \quad 01110010 &= \\ (99)_{10} \quad (97)_{10} \quad (114)_{10} &= \\ \textit{car} & \end{aligned}$$

κώδικας Huffman

Ανακαλύφθηκε το 1952 από τον τότε φοιτητή του MIT David A. Huffman.

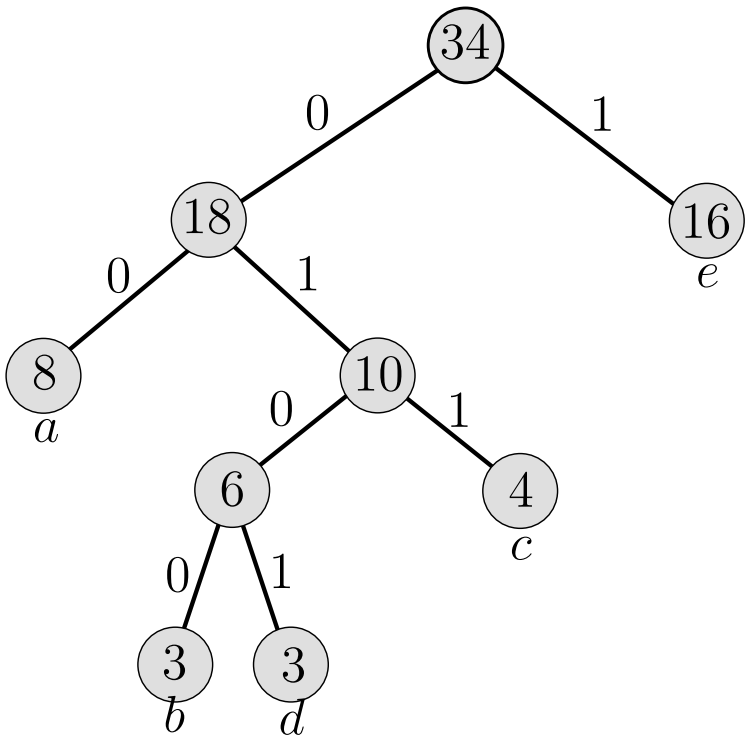
μεταβαλλόμενου μεγέθους κωδικοί

Ένας χαρακτήρας που έχει υψηλή πιθανότητα εμφάνισης σε ένα κείμενο (π.χ 'a', 'e', ...) πρέπει να χρησιμοποιεί όσο το δυνατό λιγότερα bits, ενώ χαρακτήρες με χαμηλότερες πιθανότητες εμφάνισης (π.χ. 'z') μπορούν να χρησιμοποιούν περισσότερα bits.

Η λογική αυτή οδηγεί σε προγράμματα συμπίεσης, κωδικοποίησης αρχείων, κ.τ.λ.

Δέντρο κωδικοποίησης

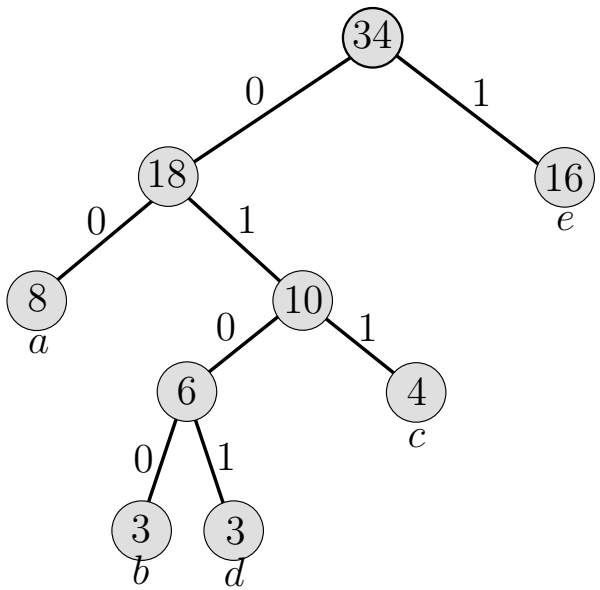
Δεδομένου πως ξέρουμε την συχνότητα εμφάνισης των γραμμάτων του αγγλικού αλφαβήτου μπορούμε να κατασκευάσουμε το εξής δέντρο.



γράμμα	συχνότητα
a	8
b	3
c	4
d	3
e	16

Δέντρο κωδικοποίησης

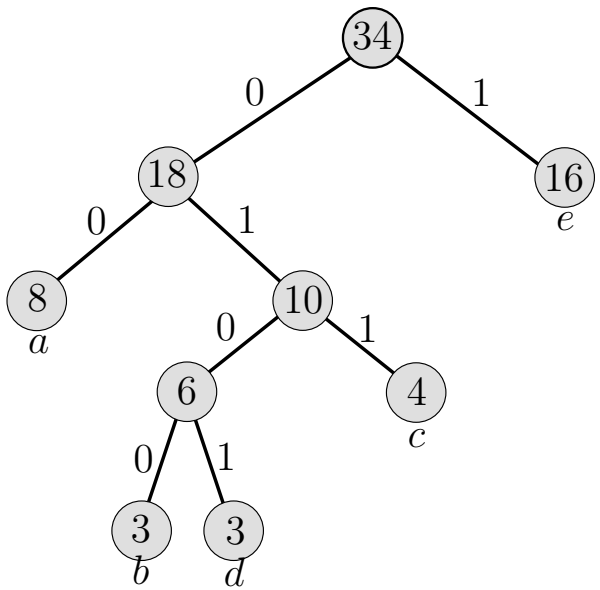
Μονοπάτια από τη ρίζα στα φύλλα του δέντρου αντιστοιχούν σε σειρές από 0 και 1 ανάλογα με το εάν πάμε προς τα αριστερά (0) ή προς τα δεξιά (1).



γράμμα	συχνότητα	κωδικοποίηση
a	8	00
b	3	0100
c	4	011
d	3	0101
e	16	1

Δέντρο κωδικοποίησης

Για να λειτουργεί μία τέτοια κωδικοποίηση δεν πρέπει κανένας κωδικός να είναι πρόθεμα άλλου κωδικού (δηλαδή να περιέχετε ως αρχικό τμήμα). Επειδή οι χαρακτήρες βρίσκονται στα φύλλα αυτό είναι εξ' ορισμού.



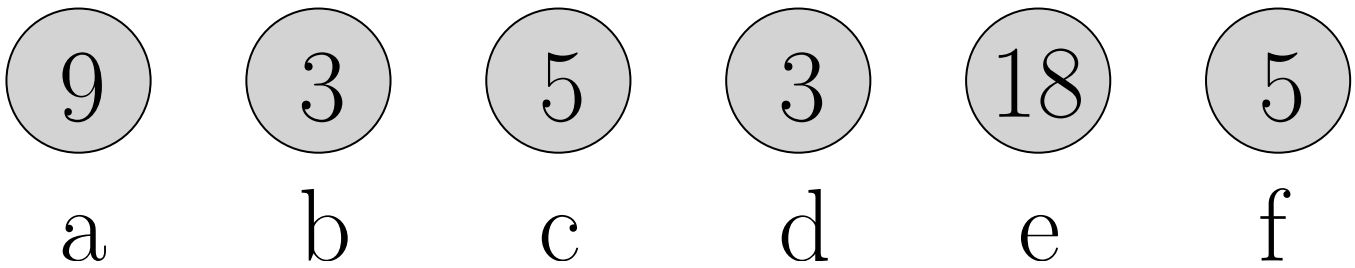
γράμμα	συχνότητα	κωδικοποίηση
a	8	00
b	3	0100
c	4	011
d	3	0101
e	16	1

Δημιουργία Βέλτιστου Δέντρου

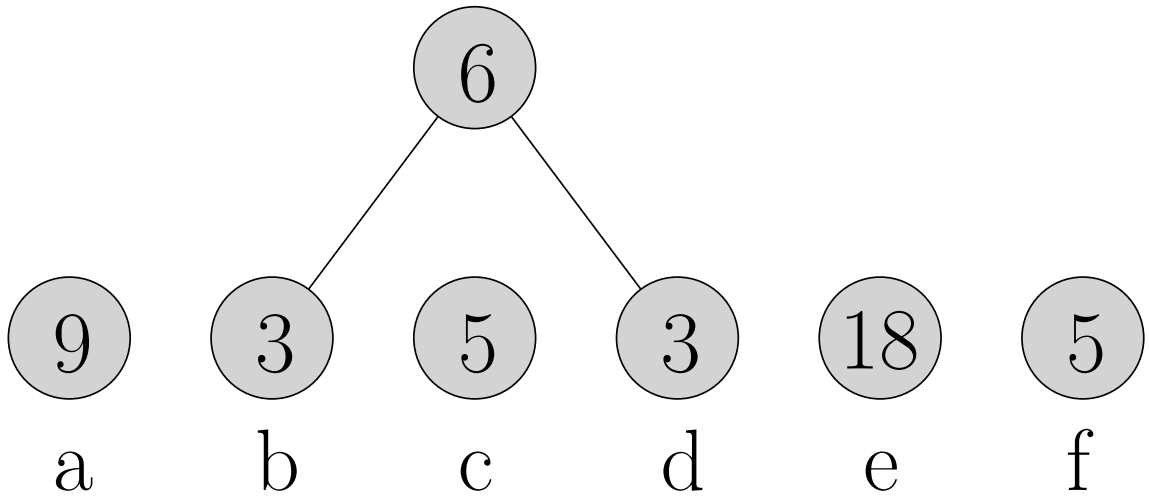
Ο Huffman σχεδίασε και απόδειξε πως η διαδικασία που ακολουθεί υπολογίζει ένα βέλτιστο δέντρο.

1. για κάθε χαρακτήρα φτιάξε ένα δέντρο (ενός κόμβου) με βάρος στην ρίζα την συχνότητα του χαρακτήρα και πληροφορία τον χαρακτήρα
2. μέχρι να μείνει ένα δέντρο κάνε τα εξής:
   - διάλεξε τα δύο δέντρα T_1 και T_2 με τα μικρότερα βάρη ρίζας
   - συγχώνευσε τα σε ένα νέο δυαδικό δέντρο που έχει αριστερό παιδί ρίζας το T_1 και δεξί παιδί ρίζας το T_2
   - δώσε βάρος στην ρίζα του T ίσο με το άθροισμα των βαρών των ριζών των T_1 και T_2

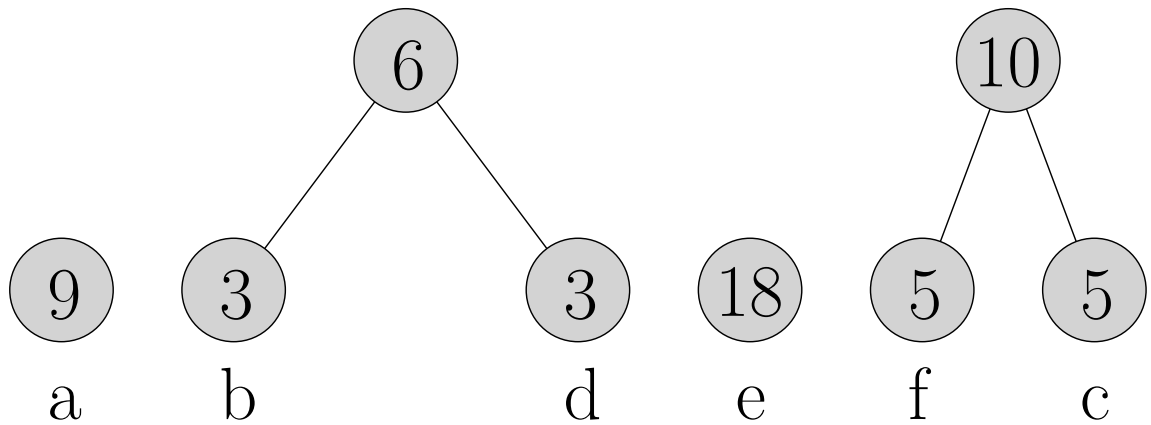
Δημιουργία Βέλτιστου Δέντρου



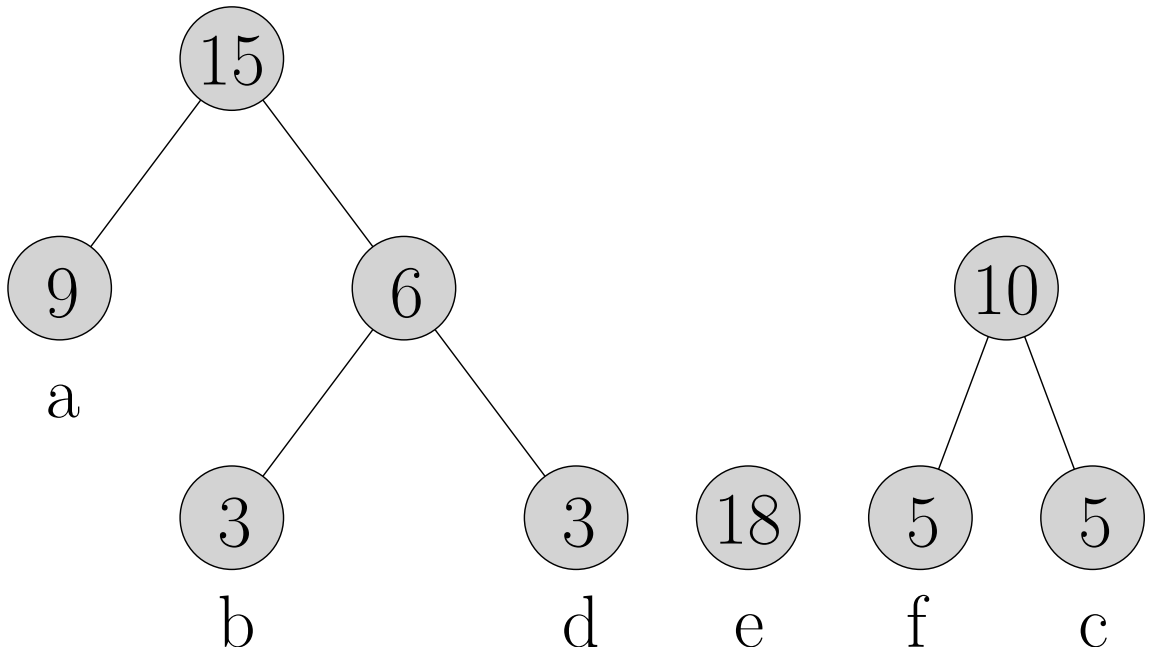
Δημιουργία Βέλτιστου Δέντρου



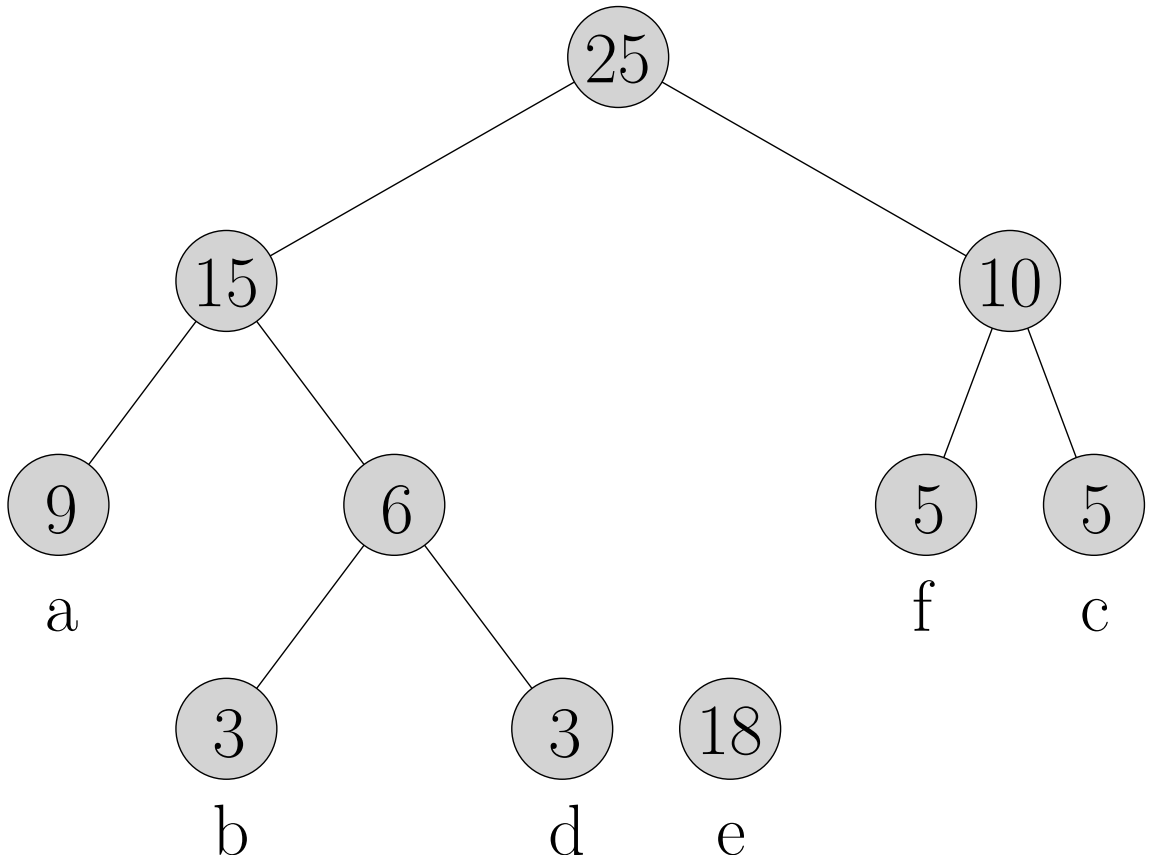
Δημιουργία Βέλτιστου Δέντρου



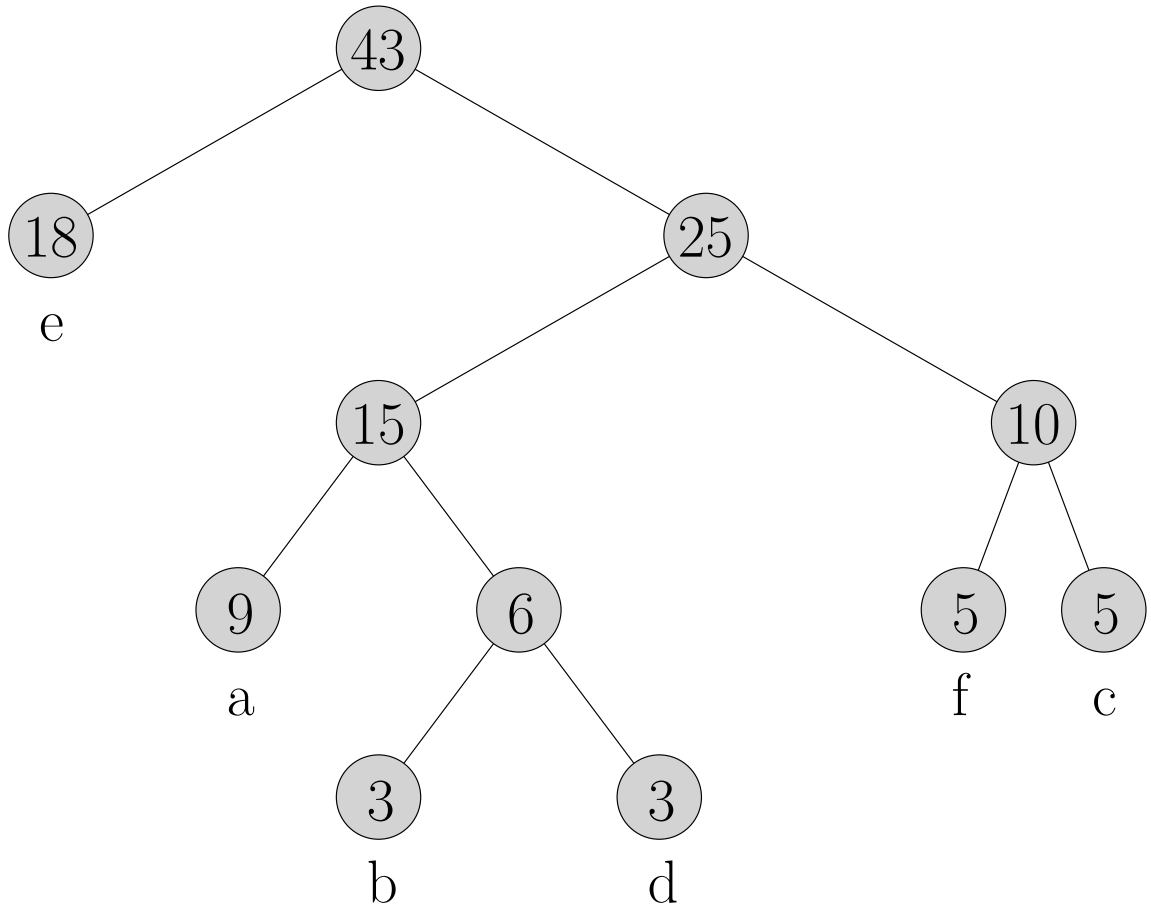
Δημιουργία Βέλτιστου Δέντρου



Δημιουργία Βέλτιστου Δέντρου



Δημιουργία Βέλτιστου Δέντρου



Δημιουργία Βέλτιστου Δέντρου