Δομές Δεδομένων σε C

Μάθημα 7:

Δυαδικά Δένδρα Αναζήτησης

Δημήτρης Ψούνης

Περιεχόμενα Μαθήματος

Α. Θεωρία

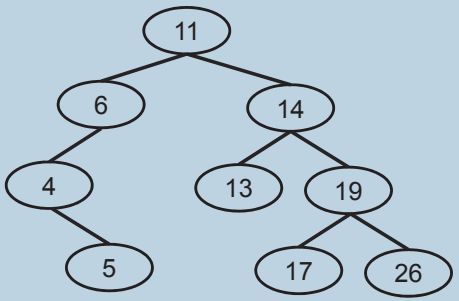
1. Δυαδικό Δένδρο Αναζήτησης

1. Ορισμός Δυαδικού Δένδρου Αναζήτησης

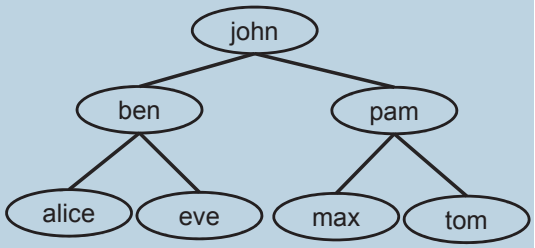
Το **«Δυαδικό Δένδρο Αναζήτησης» (ΔΔΑ)** είναι ένα δένδρο στο οποίο σε κάθε κόμβο έχει αποθηκευτεί μία τιμή v και επιπλέον:

- Στις κορυφές του αριστερού υποδένδρου, έχουν αποθηκευτεί τιμές μικρότερες της v.
- Στις κορυφές του δεξιού υποδένδρου έχουν αποθηκευτεί τιμές μεγαλύτερες της v.

Παραδείγματα Δυαδικών Δένδρων Αναζήτησης:



ΔΔΑ που αποθηκεύει αριθμούς



ΔΔΑ που αποθηκεύει συμβολοσειρές

Σημαντικό: Η ενδοδιατεταγμένη διαδρομή σε ένα ΔΔΑ επιστρέφει τους αριθμούς του ΔΔΑ σε αύξουσα σειρά

Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

2. Βασικές Πράξεις

Οι **βασικές πράξεις** σε ένα δυαδικό δένδρο αναζήτησης (επεκτείνοντας το δυαδικό δένδρο) είναι:

- **Εισαγωγή** ενός στοιχείου στο δένδρο (**insert_BST**)
- **Αναζήτηση** ενός στοιχείου στο δένδρο (**search_BST**)
- **Διαγραφή** ενός στοιχείου από το δένδρο (**delete_BST**)

Παρατήρηση: Επεκτείνουμε τον ορισμό του Δυαδικού Δένδρου, αφού η εισαγωγή και η διαγραφή κόμβων από το ΔΔΑ θα ακολουθούν κάποιο συγκεκριμένο αλγόριθμο, ώστε μετά από την εκτέλεση των πράξεων το δένδρο να εξακολουθεί να έχει την ιδιότητα του ΔΔΑ.

Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

3. Υλοποίηση σε C: Αναζήτηση σε ΔΔΑ

Η συνάρτηση «Αναζήτηση» ψάχνει για το στοιχείο Χ στο δυαδικό δένδρο αναζήτησης και επιστρέφει ΝΑΙ/ΟΧΙ ανάλογα με το αν το στοιχείο υπάρχει στο δένδρο:

Σκιαγράφηση αλγορίθμου:

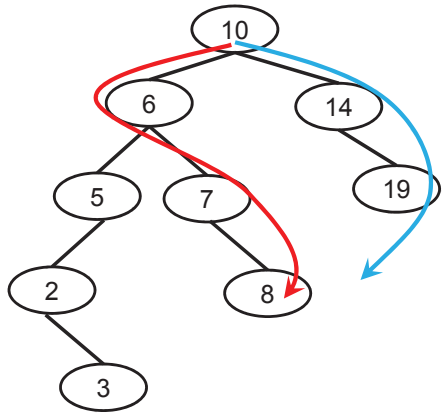
```
Θέτει Κ = ρίζα του δένδρου
Επανέλαβε όσο  Κ ≠ ΚΕΝΟ
   Αν (Χ = Κ)
        Επέστρεψε ΝΑΙ
   Αλλιώς αν (Χ > Κ)
        Θέσε Κ=δεξί παιδί της Κ.
   Αλλιώς αν (Χ < Κ)
        Θέσε Κ=αριστερό παιδί της Κ.
Τέλος-Επανάληψης
Επέστρεψε ΟΧΙ
```


Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

3. Υλοποίηση σε C: Αναζήτηση σε ΔΔΑ

- Παράδειγμα Αναζήτησης του δεδομένου 8 (κόκκινο χρώμα) και του 17 (μπλέ χρώμα)



- Αναζήτηση του 8: 10(αριστερά), 6(δεξιά), 7(δεξιά), 8 (βρέθηκε). Απάντηση: ΝΑΙ
- Αναζήτηση του 17: 10(δεξιά), 14 (δεξιά), 19(αριστερά). ΚΕΝΟ. Απάντηση: ΟΧΙ

Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

3. Υλοποίηση σε C: Αναζήτηση σε ΔΔΑ

```
/* TR_search_BST(): anazitisi tou x sto
        DDA me riza root  */
int TR_search_BST(TREE_PTR root, elem x)
{
   TREE_PTR current;

   current=root;

   while (current!=NULL)
   {
      if (x == current->data)
         return TRUE;
      else if (x < current->data)
         current=current->left;
      else // x > current->data
         current=current->right;
   }
   return FALSE;
}
```


Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

4. Υλοποίηση σε C: Εισαγωγή σε ΔΔΑ

Ο αλγόριθμος «Εισαγωγής σε ΔΔΑ» δεδομένου ενός ΔΔΑ Τ και ενός δεδομένου x:

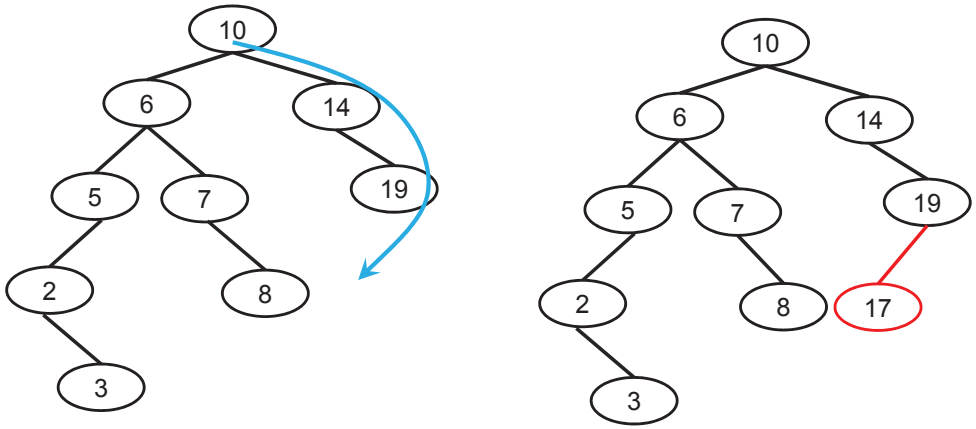
- **Αν το x υπάρχει στο δένδρο Τ επιστρέφει FALSE**
- **Αν το x δεν υπάρχει στο δένδρο Τ το εισάγει σε θέση που σέβεται το ΔΔΑ.**

Σκιαγράφηση αλγορίθμου:

Αν Τ είναι άδειο, τοποθέτησε το x ως ρίζα. Επέστρεψε TRUE
Θέσε Κ=ρίζα του δένδρου
Επανέλαβε:
  Αν x==Κ
    Επέστρεψε FALSE
  Αλλιώς αν x<Κ
    Αν Κ δεν έχει αριστερό παιδί:
      Κατασκεύασε αριστερό παιδί της Κ με δεδομένο x. Επέστρεψε TRUE
    Αλλιώς
      Θέσε Κ=αριστερό παιδί του Κ.
  Αλλιώς // x>Κ
    Αν Κ δεν έχει δεξί παιδί:
      Κατασκεύασε δεξί παιδί του Κ με δεδομένο x. Επέστρεψε TRUE
    Αλλιώς
      Θέσε Κ=δεξί παιδί του Κ.

Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
4. Υλοποίηση σε C: Εισαγωγή σε ΔΔΑ

- Παράδειγμα Εισαγωγής του δεδομένου 17



Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
4. Υλοποίηση σε C: Εισαγωγή σε ΔΔΑ

```
/* TR_insert_BST(): eisagwgi tou x
        sto DDA me riza root       */
int TR_insert_BST(TREE_PTR *root, elem x)
{
   TREE_PTR current;

   /* 1. Eisagwgi se adeio dentro */
   if (*root==NULL)
   {
      TR_insert_root(root, x);
      return TRUE;
   }

   /* 2. Anazitisi + Eisagwgi sto dendro */
   current=*root;
   while (1)
   {
      if (x == current->data)
         return FALSE;
```


Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
4. Υλοποίηση σε C: Εισαγωγή σε ΔΔΑ

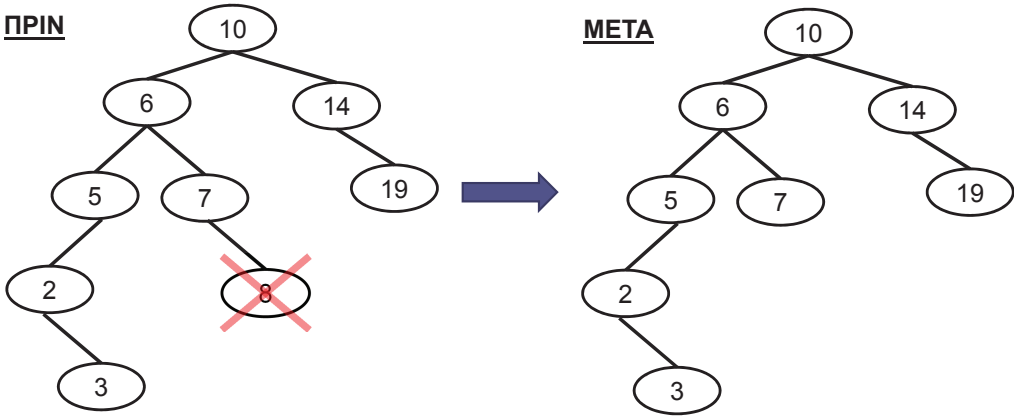
```
      else if (x < current->data)
      {
          if (current->left==NULL)
          {
             TR_insert_left(current,x);
             return TRUE;
          }
          else
             current=current->left;
      }
      else // x > current->data
      {
          if (current->right==NULL)
          {
             TR_insert_right(current,x);
             return TRUE;
          }
          else
             current=current->right;
      }
   }
}
```


Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

Ο αλγόριθμος «Διαγραφής σε ΔΔΑ» παίρνει ως όρισμα ένα δένδρο Τ και ένα δεδομένο x (που υπάρχει στο δένδρο) και ξεχωρίζει 3 περιπτώσεις:

- **Αν ο κόμβος του x δεν έχει παιδιά, τότε διαγράφουμε τον κόμβο.**

Παράδειγμα: Διαγραφή του «8»



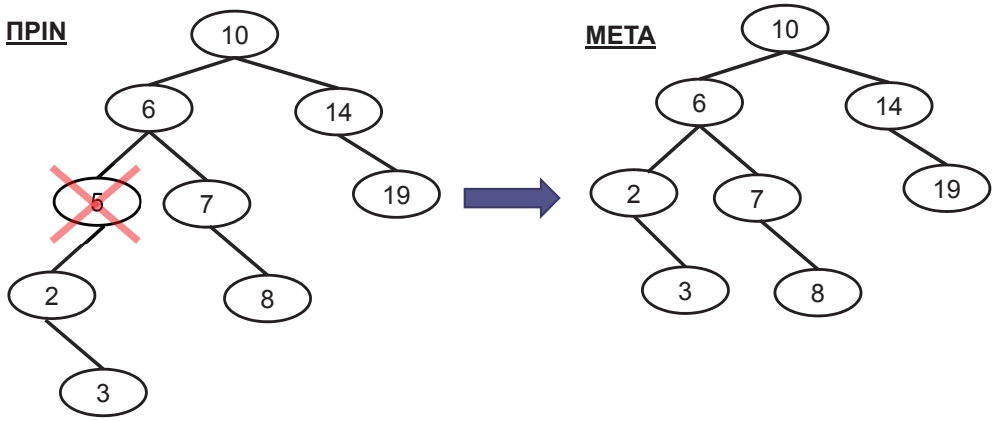
Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

- **Αν ο κόμβος του x έχει ένα παιδί, τότε διαγράφουμε τον κόμβο και το παιδί του τον αντικαθιστά.**

Παράδειγμα: Διαγραφή του «5»



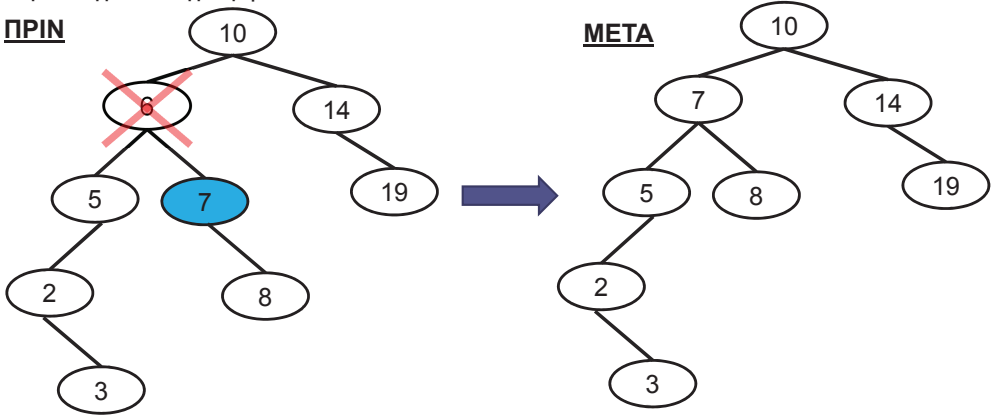
Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

- **Αν ο κόμβος του x έχει δύο παιδιά, τότε: Βρίσκουμε τον επόμενο στην ενδοδιατεταγμένη διαδρομή και αντικαθιστούμε τον x με αυτόν.**
  - **Περίπτωση 1: Το δεξί παιδί του x δεν έχει αριστερό παιδί**
    - Τότε ο x αντικαθίσταται από το δεξί παιδί του (που διατηρεί το δεξί υποδένδρο του).

Παράδειγμα: Διαγραφή του «6»



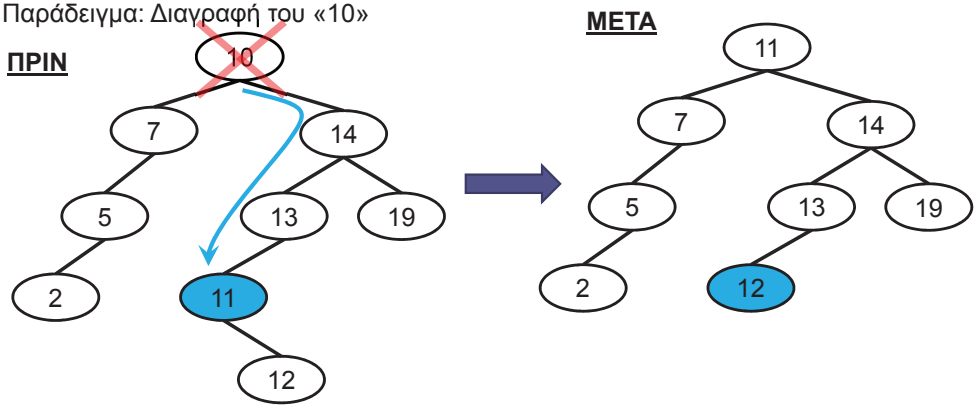
Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

- **Αν ο κόμβος του x έχει δύο παιδιά, τότε: Βρίσκουμε τον επόμενο στην ενδοδιατεταγμένη διαδρομή (y) και αντικαθιστούμε τον x με αυτόν.**
  - **Περίπτωση 2: Το δεξί παιδί του x έχει αριστερό παιδί**
    - **Ο y παίρνει τη θέση του x**
    - **Ο κόμβος του y αντικαθίσταται από το δεξί υποδένδρο του y**

Παράδειγμα: Διαγραφή του «10»



Α. Θεωρία

1. Δυαδικό Δένδρο Αναζήτησης

5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

```
/* TR_delete_BST(): diagrafi tou x
         apo to DDA me riza root        */
int TR_delete_BST(TREE_PTR *root, elem x)
{
   TREE_PTR current, parent, nextOrdered;
   int p; /* 1. deksi paidi, 2. aristero paidi tou current */
   int temp;

   /* 1. Anazitisi tou komvou */
   parent=NULL;
   current=*root;
   while (current!=NULL)
   {
      if (x == current->data)
         break;
      else if (x < current->data)
      {
         parent=current;
         p=1;
         current=current->left;
      }
```

Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

```
    else // x > current->data
    {
       parent=current;
       p=2;
       current=current->right;
    }
 }
 if (current==NULL)
    return FALSE;
 /* 2.1 An den exei paidia */
 if (current->left==NULL && current->right==NULL)
 {
    free(current);

    if (parent==NULL)
       *root=NULL;
    else
       if (p==1)
          parent->left=NULL;
       else
          parent->right=NULL;
       return TRUE;
 }
```


Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

```
 /* 2.2 Exei mono aristero paidi */
 else if (current->left!=NULL && current->right==NULL)
 {
    if (parent==NULL)
       *root=current->left;
    else
       if (p==1)
          parent->left=current->left;
       else
          parent->right=current->left;
    free(current);
    return TRUE;
 }
 /* 2.3 Exei mono deksi paidi */
 else if (current->left==NULL && current->right!=NULL)
 {
    if (parent==NULL)
       *root=current->right;
    else
       if (p==1)
          parent->left=current->right;
       else
          parent->right=current->right;
```


Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

```
       free(current);
              return TRUE;
       }
       /* 2.4 Exei aristero kai deksi paidi */
       else
       {
              /* 2.4.1 Vriskei ton epomeno stin endodiatetagmeni */
              p=1;
              nextOrdered=current->right;

              while (nextOrdered->left!=NULL)
              {
                     parent=nextOrdered;
                     nextOrdered=nextOrdered->left;
                     p=2;
              }

              /*2.4.2 Antallassei times me ton komvo pou diagrafetai */
              current->data=nextOrdered->data;
```


Α. Θεωρία
1. Δυαδικό Δένδρο Αναζήτησης
5. Υλοποίηση σε C: Διαγραφή σε ΔΔΑ

```
              /*2.4.3 Diagrafei ton komvo */
              if (p==1) /*2.4.3.1 O epomenos einai to deksi paidi */
              {
                     current->right=nextOrdered->right;
                     free(nextOrdered);
              }
              else
              {
                     parent->left=nextOrdered->right;
                     free(nextOrdered);
              }
       }
}
```

Β. Ασκήσεις
Εφαρμογή 1: Μελέτη Προγράμματος

- Μελετήστε το project tree.dev στο οποίο υλοποιούνται οι βασικές πράξεις των δυαδικών δένδρων αναζήτησης που μελετήσαμε στο μάθημα.

Β. Ασκήσεις
Εφαρμογή 2: Εκτέλεση «με το χέρι»

1. Εισάγετε τα ακόλουθα δεδομένα σε ένα ΔΔΑ: «10 6 3 9 12 5 4 2 1 8».
2. Διαγράψτε τον κόμβο 12
3. Εισάγετε τον κόμβο 7

Β. Ασκήσεις
Εφαρμογή 2: Εκτέλεση «με το χέρι»

4. Διαγράψτε τον κόμβο 3
5. Δώστε την ενδοδιατεταγμένη διαπέραση του δένδρου
6. Κατασκεύαστε πλήρες δυαδικό δένδρο αναζήτησης που να περιέχει τα περιεχόμενα του δένδρου .