Δομές Δεδομένων

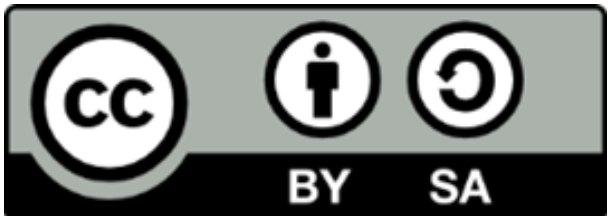
Ενότητα 10: Πλήρη Δυαδικά Δέντρα, Μέγιστα/Ελάχιστα Δέντρα & Εισαγωγή στο Σωρό- Ο ΑΤΔ Μέγιστος Σωρός

Καθηγήτρια Μαρία Σατρατζέμη

Τμήμα Εφαρμοσμένης Πληροφορικής

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.



Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Μακεδονίας» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σκοποί ενότητας

- Να εισάγει τους φοιτητές στις έννοιες: πλήρες δυαδικό δέντρο, γεμάτο δδ, μέγιστο/ελάχιστο δδ, σωρός.
- Να μελετηθεί ο ΑΤΔ Μέγιστος Σωρός και η υλοποίηση του με δομή καθώς και να δοθεί η υλοποίηση όλων των πράξεων.

Περιεχόμενα ενότητας

- Εισαγωγή (heap)
- Γεμάτο δυαδικό δέντρο
- Πλήρες δυαδικό δέντρο
- Μέγιστο/Ελάχιστο δέντρο
- Σωρός
- Εισαγωγή (max heap)
- Ορισμός του ΑΤΔ Μέγιστος Σωρός
- Υλοποίηση δομής
- Εισαγωγή στοιχείου στο σωρό
- Διαγραφή στοιχείου από μέγιστο σωρό
- Πακέτο για τον ΑΤΔ Σωρός

Πλήρη Δυαδικά Δέντρα, Μέγιστα/Ελάχιστα Δέντρα & Εισαγωγή στο Σωρό

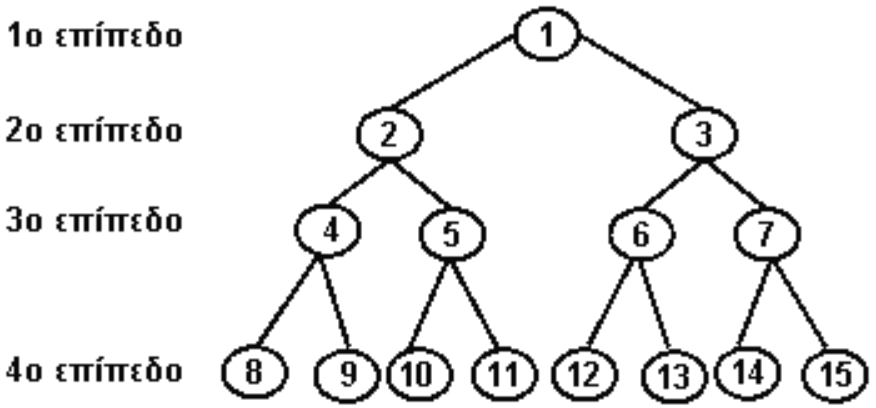
Εισαγωγή (heap)

- Ο ΑΤΔ του σωρού (heap) που θα παρουσιάσουμε σε αυτό το κεφάλαιο αποτελεί ειδική περίπτωση του πλήρους δυαδικού δέντρου.
- Για να είμαστε σε θέση όμως να περιγράψουμε τον ΑΤΔ του σωρού θα πρέπει πρώτα να δώσουμε μια αναλυτικότερη περιγραφή των εξής δομών:
  - γεμάτο δυαδικό δέντρο (full binary tree)
  - πλήρες δυαδικό δέντρο (complete binary tree)
  - μέγιστο/ελάχιστο δέντρο (max/min tree)

Γεμάτο δυαδικό δέντρο

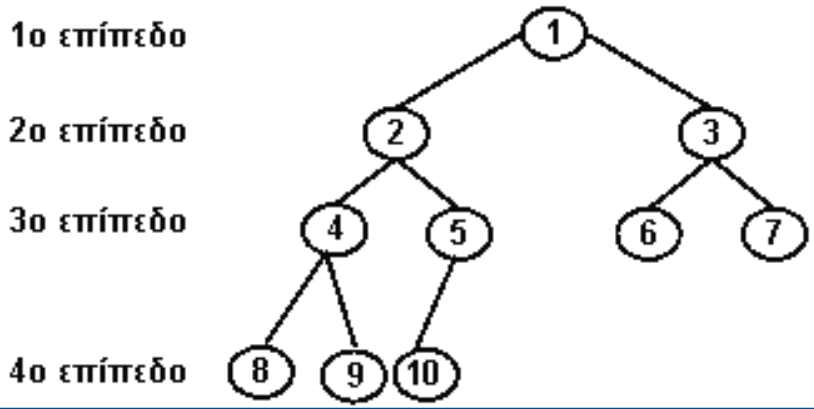
- Ένα γεμάτο δυαδικό δέντρο (full binary tree) ύψους k είναι ένα δυαδικό δέντρο με 2k - 1 κόμβους, όπου k>=0.
- Ο αριθμός των κόμβων (2k - 1) ενός γεμάτου δυαδικού δέντρου είναι ο μέγιστος αριθμός κόμβων που μπορεί να έχει ένα δυαδικό δέντρο ύψους k.
- Στο σχήμα που ακολουθεί φαίνεται ένα γεμάτο δυαδικό δέντρο ύψους 4.

  Οι κόμβοι αριθμούνται ξεκινώντας με τη ρίζα που βρίσκεται στο 1ο επίπεδο, στη συνέχεια αριθμούνται οι κόμβοι του 2ου επιπέδου από αριστερά προς τα δεξιά κ.ο.κ.



Πλήρες δυαδικό δέντρο -1-

- Ένα πλήρες δυαδικό δέντρο (complete trees), είναι ένα δέντρο στο οποίο κάθε επίπεδο είναι συμπληρωμένο πλήρως, εκτός ίσως από το τελευταίο, στο οποίο οι κόμβοι βρίσκονται στις πιο αριστερές θέσεις ή αλλιώς
- ένα δυαδικό δέντρο ύψους k με n κόμβους, ονομάζεται πλήρες αν οι κόμβοι του αντιστοιχούν στους κόμβους 1 έως n ενός γεμάτου δυαδικού δέντρου ύψους k.
- Στο παρακάτω σχήμα φαίνεται ένα πλήρες δυαδικό δέντρο με 10 κόμβους.



Πλήρες δυαδικό δέντρο -2-

- Ο τρόπος αρίθμησης των κόμβων του πλήρους δυαδικού δέντρου μας επιτρέπει να χρησιμοποιήσουμε ένα μονοδιάστατο πίνακα για την αποθήκευση των κόμβων του.
- Για την αναπαράσταση ενός πλήρους δυαδικού δένδρου λοιπόν, χρησιμοποιούμε ένα πίνακα στον οποίο τοποθετούμε τη ρίζα στην θέση 1, τα παιδιά της στις θέσεις 2 και 3, τους κόμβους του επόμενου επιπέδου στις θέσεις 4, 5, 6, 7 κ.τ.λ.
- Η αναπαράσταση αυτή είναι πολύ χρήσιμη γιατί επιτρέπει την εύκολη μετάβαση από οποιαδήποτε κόμβο στον πατέρα και στα παιδιά του.

Πλήρες δυαδικό δέντρο -3-

Στην περίπτωση αναπαράστασης ενός πλήρους δυαδικού δέντρου n κόμβων με ένα πίνακα, για κάθε κόμβο j:

1. ο πατέρας του βρίσκεται στη θέση j div 2, αν j ≠ 1.

   Αν j = 1 τότε πρόκειται για τη ρίζα, η οποία φυσικά δεν έχει πατέρα.

2. το αριστερό παιδί του βρίσκεται στη θέσης 2*j, αν 2*j > n.

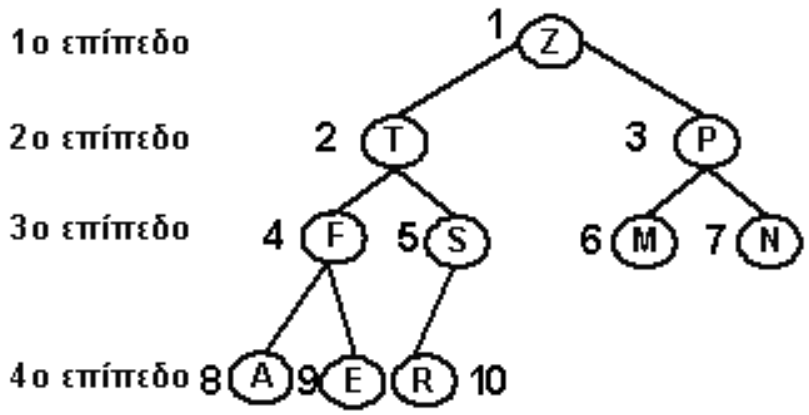
   Αν 2*j > n τότε ο κόμβος j δεν έχει αριστερό παιδί.

3. το δεξί παιδί του βρίσκεται στη θέση 2*j+1, αν 2*j +1 > n.

   Αν 2*j +1 > n τότε ο κόμβος j δεν έχει δεξί παιδί..

Πλήρες δυαδικό δέντρο -4-

- Αναπαράσταση ενός πλήρους δυαδικού δέντρου σχηματικά και με πίνακα:



1	2	3	4	5	6	7	8	9	10
Z	T	P	F	S	M	N	A	E	R

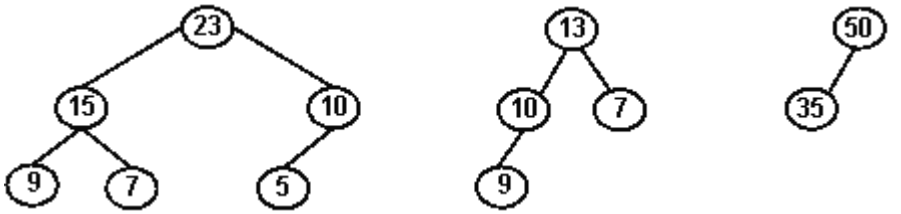
- Με έντονα γράμματα φαίνεται η αρίθμηση των κόμβων, η οποία συμπίπτει και με τις θέσεις του πίνακα που αποθηκεύονται οι κόμβοι.

Μέγιστο/Ελάχιστο δέντρο

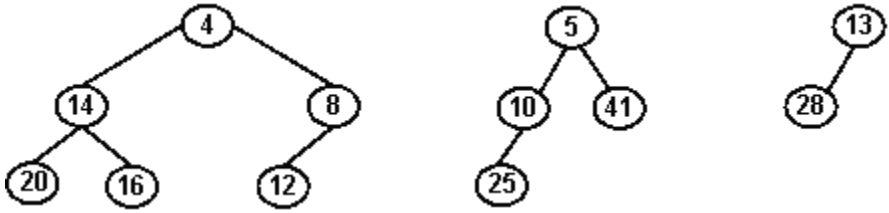
- Ένα μέγιστο δέντρο (max/min tree), είναι ένα δέντρο στο οποίο η τιμή του πεδίου κλειδιού κάθε κόμβου δεν είναι μικρότερη από τις τιμές των πεδίων κλειδιών των παιδιών του, εφόσον υπάρχουν.
- Ένα ελάχιστο δέντρο (min tree), είναι ένα δέντρο στο οποίο η τιμή του πεδίου κλειδιού κάθε κόμβου δεν είναι μεγαλύτερη από τις τιμές των πεδίων κλειδιών των παιδιών του, εφόσον υπάρχουν.
- Από τον ορισμό του μέγιστου και του ελάχιστου δέντρου, είναι προφανές ότι η τιμή του πεδίου κλειδιού της ρίζας ενός μέγιστου δέντρου είναι η μεγαλύτερη τιμή στο δέντρο, ενώ σε ένα ελάχιστο δέντρο είναι η μικρότερη τιμή.

Σωρός -1-

- Ο σωρός αποτελεί ειδική περίπτωση ενός πλήρους δυαδικού δέντρου και διακρίνεται σε μέγιστο και ελάχιστο σωρό:
- Ο μέγιστος σωρός (max heap) είναι ένα πλήρες δυαδικό δένδρο, το οποίο είναι ταυτόχρονα και ένα μέγιστο δέντρο.
- Αντίστοιχα, ο ελάχιστος σωρός (min heap) είναι ένα πλήρες δυαδικό δένδρο που είναι ταυτόχρονα και ένα ελάχιστο δέντρο.



παραδείγματα μέγιστων σωρών



παραδείγματα ελάχιστων σωρών

Σωρός -2-

- Στις περισσότερες εφαρμογές, όπως για παράδειγμα στις ουρές προτεραιότητας (priority queues) που υλοποιούνται με τον ΑΤΔ του σωρού, χρησιμοποιούνται μέγιστοι σωροί.
- Γι' αυτό και όταν αναφερόμαστε στη δομή του σωρού, χωρίς να προσδιορίζουμε αν πρόκειται για μέγιστο ή ελάχιστο σωρό, εννοούμε μέγιστο σωρό.
- Πολλές φορές μάλιστα χρησιμοποιείται ο παρακάτω ορισμός για το σωρό.
- Ο σωρός είναι ένα πλήρες δυαδικό δέντρο, του οποίου κάθε κόμβος ικανοποιεί την ιδιότητα προτεραιότητας, σύμφωνα με την οποία το κλειδί σε κάθε κόμβο πρέπει να είναι μεγαλύτερο ή ίσο από τα κλειδιά των παιδιών του (αν έχει).

Ο ΑΤΔ Μέγιστος Σωρός

Εισαγωγή (max heap)

- Ο ΑΤΔ του μέγιστου σωρού (max heap) περιλαμβάνει τις ακόλουθες βασικές λειτουργίες:
  - δημιουργία ενός κενού σωρού
  - εισαγωγή στοιχείου στο σωρό
  - διαγραφή του μεγαλύτερου στοιχείου από το σωρό

Ορισμός του ΑΤΔ Μέγιστος Σωρός -1-

Συλλογή στοιχείων δεδομένων:

Ένα πλήρες δυαδικό δέντρο με n>0 στοιχεία οργανωμένα έτσι ώστε η τιμή σε κάθε κόμβο να είναι τουλάχιστο τόσο μεγάλη όσο εκείνη των παιδιών της.

Βασικές λειτουργίες:

- **Δημιουργία κενού σωρού (CreateMaxHeap):**

  Λειτουργία: Δημιουργεί ένα κενό σωρό.

  Επιστρέφει: Ένα κενό σωρό.

- **Έλεγχος άδειου σωρού (EmptyHeap):**

  Δέχεται: Ένα σωρό.

  Λειτουργία: Ελέγχει αν ο σωρός είναι άδειος.

  Επιστρέφει: TRUE, αν ο σωρός είναι άδειος, FALSE διαφορετικά.

Ορισμός του ΑΤΔ Μέγιστος Σωρός -2-

Εισαγωγή στοιχείου (InsertMaxHeap):

Δέχεται: Ένα σωρό και ένα στοιχείο δεδομένων.

Λειτουργία: Εισάγει το στοιχείο στο σωρό, αν ο σωρός δεν είναι γεμάτος.

Επιστρέφει: Τον τροποποιημένο σωρό.

Διαγραφή του μεγαλύτερου στοιχείου (DeleteMaxHeap):

Δέχεται: Ένα σωρό.

Λειτουργία: Ανακτά και διαγράφει το μεγαλύτερο στοιχείο του σωρού.

Επιστρέφει: Το μεγαλύτερο στοιχείο του σωρού και τον τροποποιημένο σωρό.

Ορισμός του ΑΤΔ Μέγιστος Σωρός -3-

Η υλοποίηση θα γίνει στατικά (με πίνακα)

Έλεγχος γεμάτου σωρού (FullHeap):

Δέχεται: Ένα σωρό.

Λειτουργία: Ελέγχει αν ο σωρός είναι γεμάτος.

Επιστρέφει: TRUE, αν ο σωρός είναι γεμάτος, FALSE διαφορετικά.

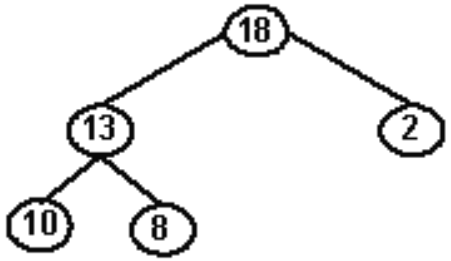
Υλοποίηση δομής

Μια τέτοια δομή μπορεί να υλοποιηθεί με μια εγγραφή (struct), όπως φαίνεται παρακάτω:

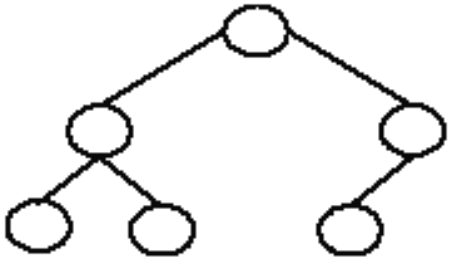
```
 #define  MaxElements 10          /*μέγιστο πλήθος στοιχείων του σωρού*/
typedef int  HeapElementType;  /*ο τύπος των στοιχείων του σωρού*/
typedef  struct {
    HeapElementType key;
    // int Data;                /*οποιοσδήποτε τύπος δεδομένων για τα
                                 παρελκόμενα στοιχεία του κόμβου */
} HeapNode;

typedef struct {
       int Size;
       HeapNode Element[MaxElements+1];
} HeapType;
```

Εισαγωγή στοιχείου στο σωρό -1-

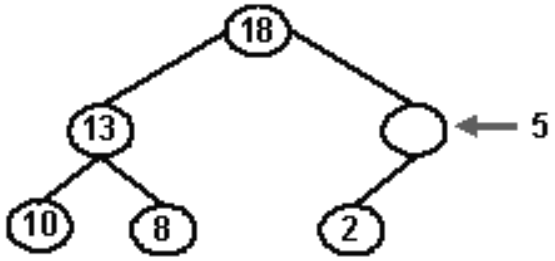


- Στο σχήμα που ακολουθεί φαίνεται ένας μέγιστος σωρός με 5 στοιχεία:
- Αν στο σωρό αυτό εισάγουμε ένα στοιχείο, τότε ο σωρός θα έχει τη δομή που φαίνεται παρακάτω, αφού ο μέγιστος σωρός είναι ένα πλήρες δυαδικό δέντρο.
- Εισαγωγή στοιχείου με τιμή κλειδιού 1:
- το στοιχείο μπορεί να εισαχθεί στο μέγιστο σωρό ως αριστερό παιδί του κόμβου με τιμή κλειδιού 2, οπότε θα προκύψει ο σωρός με τη δομή που φαίνεται παραπάνω



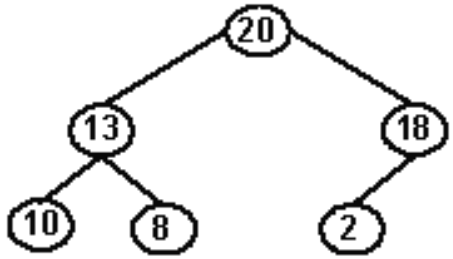
Εισαγωγή στοιχείου στο σωρό -2-

- Εισαγωγή στοιχείου με τιμή κλειδιού 5:
- στην περίπτωση αυτή το στοιχείο δεν μπορεί να εισαχθεί ως αριστερό παιδί του 2, γιατί τότε το πλήρες δυαδικό δέντρο που θα προκύψει δεν θα είναι ταυτόχρονα και μέγιστο δέντρο.
- Η επόμενη κίνησή μας είναι να μετακινήσουμε το 2 προς τα κάτω και συγκεκριμένα στη θέση του αριστερού παιδιού του (όπως φαίνεται στο Σχήμα) και να ελέγξουμε αν η εισαγωγή του 5 στην παλιά θέση του 2 οδηγεί ή όχι σε μέγιστο σωρό.
- Εφόσον, η τιμή κλειδιού του γονέα, η οποία είναι το 18, είναι τουλάχιστον τόσο μεγάλη όσο η τιμή του κλειδιού (5) του στοιχείου που εισάγεται, το στοιχείο μπορεί να εισαχθεί στη συγκεκριμένη θέση.



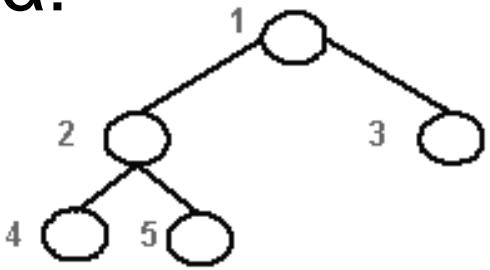
Εισαγωγή στοιχείου στο σωρό -3-

- Εισαγωγή στοιχείου με τιμή κλειδιού 20:
- αρχικά το 2 μετακινείται στη θέση του αριστερού παιδιού του, όπως φαίνεται στο Σχήμα.
- Στη συνέχεια ελέγχουμε αν το 20 μπορεί να εισαχθεί στην παλιά θέση του 2.
- Είναι προφανές ότι το 20 δεν μπορεί να εισαχθεί στη συγκεκριμένη θέση, αφού ο γονέας του κόμβου που βρίσκεται στη θέση αυτή έχει τιμή 18, η οποία είναι μικρότερη του 20.
- Συνεπώς, το 18 μετακινείται στη θέση του δεξιού παιδιού του και το νέο στοιχείο με τιμή κλειδιού 20 εισάγεται στη ρίζα του σωρού.

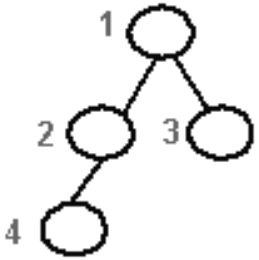


Διαγραφή στοιχείου από μέγιστο σωρό -1-

- Στο παρακάτω Σχήμα φαίνεται ένας μέγιστος σωρός με 5 στοιχεία:

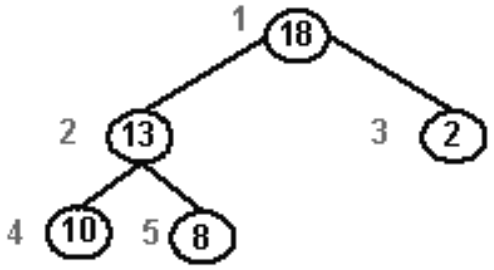


- Αν από το σωρό αυτό διαγράψουμε το μεγαλύτερο στοιχείο, τότε ο σωρός θα έχει τη δομή που φαίνεται παρακάτω, αφού ο μέγιστος σωρός είναι ένα πλήρες δυαδικό δέντρο:

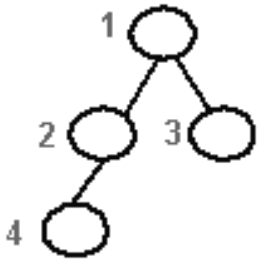


Διαγραφή στοιχείου από μέγιστο σωρό -2-

- Διαγραφή του μεγαλύτερου στοιχείου με τιμή κλειδιού 18:
- αν διαγράψουμε από τον μέγιστο σωρό που φαίνεται στο Σχήμα

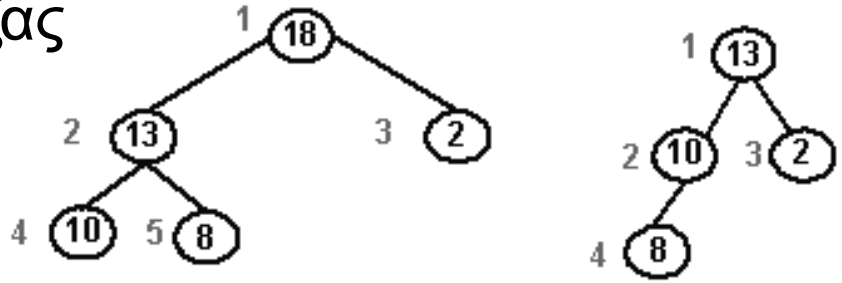


- το μεγαλύτερο στοιχείο με τιμή κλειδιού 18 τότε ο σωρός που θα προκύψει θα πρέπει να έχει τη δομή που φαίνεται στο Σχήμα



Διαγραφή στοιχείου από μέγιστο σωρό -3-

- Η πρώτη μας κίνηση λοιπόν είναι να διαγράψουμε τον κόμβο (με τιμή κλειδιού 8) που βρίσκεται στη θέση 5 του σωρού και να τον εισάγουμε στη ρίζα, η οποία είναι άδεια μετά τη διαγραφή του κόμβου με τιμή κλειδιού 18.
- Στη συνέχεια ελέγχουμε αν το πλήρες δυαδικό δέντρο που προκύπτει είναι ταυτόχρονα και μέγιστο δέντρο. Στο συγκεκριμένο παράδειγμα αυτό δεν συμβαίνει, αφού το στοιχείο της ρίζας με τιμή κλειδιού 8 δεν είναι μεγαλύτερο από τα παιδιά της που έχουν τιμές κλειδιού 13 και 2.
- Άρα, το στοιχείο με τιμή κλειδιού 13 που βρίσκεται στη θέση 2 μετακινείται προς τα πάνω και συγκεκριμένα στη ρίζα, ενώ το στοιχείο της ρίζας εισάγεται στον κενό πλέον κόμβο της 2ης θέσης.



Πακέτο για τον ΑΤΔ Σωρός -1-

```
//filename : HeapADT.h

#define  MaxElements 10          //το μέγιστο πλήθος των στοιχείων του σωρού

typedef int  HeapElementType;    //ο τύπος δεδομένων κάθε στοιχείου του σωρού

typedef  struct {
    HeapElementType key;
    //int Data;
} HeapNode;

typedef struct {
        int Size;
        HeapNode
Element[MaxElements+1];
} HeapType;

typedef enum {
    FALSE, TRUE
} boolean;
```

Πακέτο για τον ΑΤΔ Σωρός -2-

```
void CreateMaxHeap(HeapType *Heap);
boolean FullHeap(HeapType Heap);
boolean EmptyHeap(HeapType Heap);
void InsertMaxHeap(HeapType *Heap,
                   HeapNode Item);
void DeleteMaxHeap(HeapType *Heap,
                   HeapNode *Item);
```

Πακέτο για τον ΑΤΔ Σωρός -3-

```
void CreateMaxHeap(HeapType *Heap)
/*Λειτουργία:     Δημιουργεί ένα κενό σωρό.
  Επιστρέφει:     Ένα κενό σωρό.*/
{
    (*Heap).Size = 0
}

boolean FullHeap(HeapType  Heap)
/*Δέχεται:          Ένα σωρό.
  Λειτουργία:     Ελέγχει αν ο σωρός είναι γεμάτος.
  Επιστρέφει:     TRUE αν ο σωρός είναι γεμάτος, FALSE διαφορετικά.*/
{
    return (Heap.Size == MaxElements)
}
```

Πακέτο για τον ΑΤΔ Σωρός -4-

```
boolean EmptyHeap(HeapType  Heap)
/*Δέχεται:  Ένα σωρό Heap.
  Λειτουργία: Ελέγχει αν ο σωρός είναι κενός.
  Επιστρέφει: TRUE αν ο σωρός είναι κενός,
   FALSE διαφορετικά.*/
{
   return (Heap.Size == 0)
}
```

Πακέτο για τον ΑΤΔ Σωρός -5-

void InsertMaxHeap(HeapType *Heap, HeapNode Item)

/*Δέχεται: Ένα σωρό Heap και ένα στοιχείο δεδομένου Item .

Λειτουργία: Εισάγει το στοιχείο Item στο σωρό, αν ο σωρός δεν είναι γεμάτος.

Επιστρέφει: Τον τροποποιημένο σωρό.

Έξοδος: Μήνυμα γεμάτου σωρού αν ο σωρός είναι γεμάτος.*/

Πακέτο για τον ΑΤΔ Σωρός -6-

```
{
    int hole;
    if (!FullHeap(*Heap)) {
       (*Heap).Size++; hole=(*Heap).Size;
       while (hole>1 &&
                    Item.key > Heap->Element[hole/2].key) {
                    (*Heap).Element[hole]=
                              (*Heap).Element[hole/2];
                hole=hole/2;
        }
          (*Heap).Element[hole]=Item;
  }
```

Πακέτο για τον ΑΤΔ Σωρός -7-

```
void DeleteMaxHeap(HeapType *Heap, HeapNode *Item)
/*Δέχεται:  Ένα σωρό Heap.
  Λειτουργία: Ανακτά και διαγράφει το μεγαλύτερο στοιχείο του σωρού.
  Επιστρέφει: Το μεγαλύτερο στοιχείο Item (αυτό που θα διαγραφεί) του σωρού και τον τροποποιημένο σωρό.*/
{
    int parent, child;     // child δείχνει το μεγαλύτερο παιδί του κόμβου parent
     HeapNode last;     //last ο τελευταίος κόμβος
     boolean done;
```

Πακέτο για τον ΑΤΔ Σωρός
-8-

```
if (! EmptyHeap(*Heap)) {
        done = FALSE;
        (*Item) = (*Heap).Element[1];
            //item το στοιχείο που θα διαγραφεί
        last = (*Heap).Element[(*Heap).Size];
       (*Heap).Size --;
       parent = 1;
       child = 2;
```

Πακέτο για τον ΑΤΔ Σωρός -9-

```
while (child<=(*Heap).Size  && !done) {
      if (child < (*Heap).Size)
            if (*Heap).Element[child].key < (*Heap).Element[child +1].key)
                  child ++;
                  if (last.key >= (*Heap).Element[child].key)  done = TRUE;
                 else {
                        (*Heap).Element[parent] = (*Heap).Element[child];
                                                //μετακινούμε το παιδί πάνω
                         parent = child;        //συνεχίζουμε με τα παιδιά του
                         child = 2* child;
                        }
            }
            (*Heap).Element[parent] = last;
      }
      else  printf("Empty heap...\n");
}
```

Τέλος Ενότητας

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης