

Δομές Δεδομένων σε C

Μάθημα 8:

Δένδρα - Σωροί

Δημήτρης Ψούνης



www.psounis.gr

Περιεχόμενα Μαθήματος

A. Θεωρία

1. Δένδρο - Σωρός

1. Ορισμός Δένδρου-Σωρού
2. Βασικές Πράξεις και Αναπαράσταση
3. Υλοποίηση σε C: Δηλώσεις
4. Υλοποίηση σε C: Αρχικοποίηση
5. Υλοποίηση σε C: Εισαγωγή Κόμβου
6. Υλοποίηση σε C: Διαγραφή Κόμβου
7. Ο αλγόριθμος ταξινόμησης HeapSort

B. Ασκήσεις

A. Θεωρία

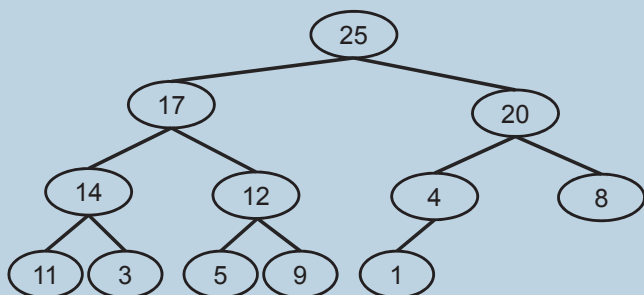
1. Δένδρο - Σωρός

1. Ορισμός Δένδρου - Σωρού

Το «**Δένδρο - Σωρός Μεγίστων**» (**heap**) είναι ένα πλήρες δυαδικό δένδρο στο οποίο:

- Κάθε κόμβος έχει τιμή μεγαλύτερη από τα παιδιά του.

Παράδειγμα Δένδρου - Σωρού:



Σωρός που αποθηκεύει αριθμούς

Παρατήρηση: Αντίστοιχα ορίζεται το δένδρο - σωρός ελαχίστων

A. Θεωρία

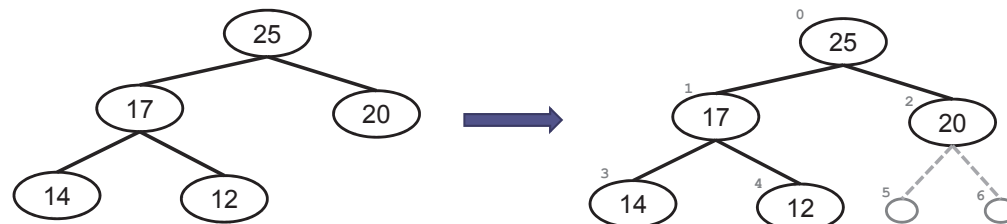
1. Δένδρο - Σωρός

2. Βασικές Πράξεις και Αναπαράσταση

Οι βασικές πράξεις σε ένα δένδρο-σωρό είναι:

- **Αρχικοποίηση** του σωρού (**HEAP_init**)
- **Εισαγωγή** ενός στοιχείου στο δένδρο (**HEAP_insert**)
- **Διαγραφή** της ρίζας του δένδρου (**HEAP_delete**)

Παρατήρηση: Επειδή το Δένδρο-Σωρός είναι Πλήρες προτιμάμε (λόγω ευκολίας πράξεων) την συνεχόμενη αναπαράσταση του δένδρου.



Σημαντικές παρατηρήσεις:

- Ο πατέρας του v είναι ο $(v - 1) \text{ DIV } 2$
- Το αριστερό παιδί του v είναι $2 * v + 1$
- Το δεξί παιδί του v είναι $2 * v + 2$

data =

0	1	2	3	4	5	6
25	17	20	14	12		

A. Θεωρία

1. Δένδρο - Σωρός

3. Υλοποίηση σε C: Δηλώσεις

Οι **δηλώσεις** σε C είναι οι ακόλουθες:

- Ο κόμβος του δένδρου είναι μία δομή (struct) με τα εξής στοιχεία:
 - Ένας πίνακας (data) με τα στοιχεία του δένδρου (σε τύπο δεδομένων που ορίζουμε).
 - Το πλήθος των στοιχείων του δένδρου (N)

```
/* heap.h : Dilwseis dendrou-swrou */

#define TRUE 1
#define FALSE 0
#define MAX_SIZE 31

typedef int elem;      /* typos dedomenwn dendrou*/

struct heap_tree{      /* Dendro-Swros */
    elem data[MAX_SIZE]; /* dedomena */
    int N;               /* plithos stoxeiwn */
};

typedef struct heap_tree HEAP; /* Sinwnimo tou swrou */
```

A. Θεωρία

1. Δένδρο - Σωρός

4. Υλοποίηση σε C: Αρχικοποίηση

Η αρχικοποίηση γίνεται θέτοντας το πλήθος των στοιχείων του δένδρου ίσα με το 0.

```
/* HEAP_init(): arxikopoiei to dendro */
void HEAP_init(HEAP *heap)
{
    heap->N=0;
}
```

Προσοχή:

- Πάντα προτού ξεκινάμε την χρήση του δένδρου θα πρέπει να καλούμε μία φορά αυτήν τη συνάρτηση!

A. Θεωρία

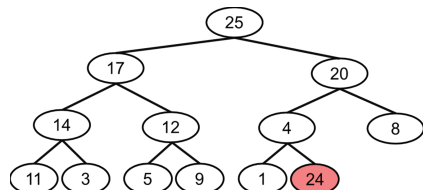
1. Δένδρο - Σωρός

5. Υλοποίηση σε C: Εισαγωγή Κόμβου

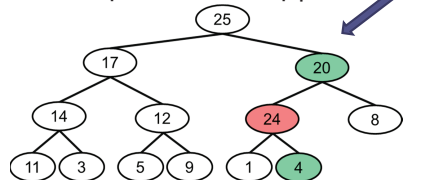
Η συνάρτηση «Εισαγωγή» εισάγει έναν νέο κόμβο:

- Θέτει τον κόμβο στην επόμενη θέση του πίνακα
- Ανταλλάσσει τη θέση με το γονέα του όσο έχει μεγαλύτερη τιμή από αυτόν.

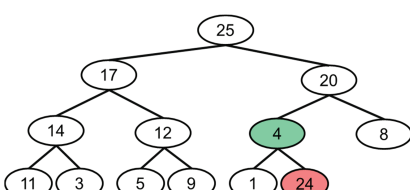
Παράδειγμα: Το 24 στην επόμενη θέση:



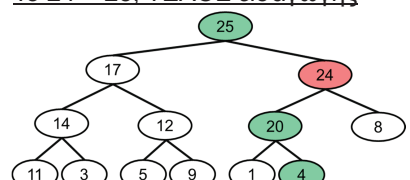
Το 24 > 20, γίνεται ανταλλαγή:



Το 24 > 4, γίνεται ανταλλαγή:



Το 24 < 25, ΤΕΛΟΣ εισαγωγής



A. Θεωρία

1. Δένδρο - Σωρός

5. Υλοποίηση σε C: Εισαγωγή Κόμβου

```
/* HEAP_insert(): Eisagei to stoixeio x
                   sto dentro-swros heap */
int HEAP_insert(HEAP *heap,elem x)
{
    int posParent, posCurrent;
    elem temp;

    /* An xwraei sto swro */
    if (heap->N == MAX_SIZE)
        return FALSE;

    /* 1. Eisagwgi tou neou komvou */
    heap->data[heap->N]=x;
    heap->N ++;
```

A. Θεωρία

1. Δένδρο - Σωρός

5. Υλοποίηση σε C: Εισαγωγή Κόμβου

```
/* 2. Antimetathesi me to gonea
   efoson vrei mikroteri timi */
posCurrent=heap->N - 1;
while (posCurrent>0)
{
    posParent=(posCurrent-1)/2;
    /* 2.1 Exei megaliteri timi apo gonea. Antimetathesi. */
    if (heap->data[posCurrent] > heap->data[posParent])
    {
        swap(&heap->data[posCurrent], &heap->data[posParent]);
        posCurrent=posParent;
    }
    /* 2.2 Pire tin oristiki tou thesi. Diakopi */
    else
        break;
}

return TRUE;
}
```

A. Θεωρία

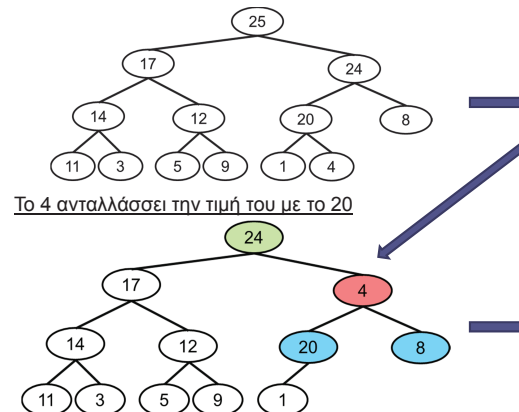
1. Δένδρο - Σωρός

6. Υλοποίηση σε C: Διαγραφή Κόμβου

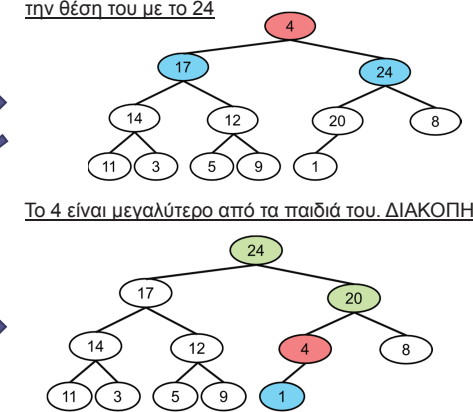
Η συνάρτηση «Διαγραφή» διαγράφει τη ρίζα (μέγιστο στοιχείο του δένδρου):

- Αποθηκεύει την τιμή της ρίζας. Θέτει το τελευταίο στοιχείο του πίνακα στη ρίζα.
- Επαναληπτικά, αν το στοιχείο έχει μικρότερη τιμή από κάποιο παιδί του, τότε ανταλλάσσει τη θέση με το μεγαλύτερο από τα δύο παιδιά του.

Παράδειγμα: Απομακρύνεται το 25.



Το 4 παίρνει τη θέση του 25: Ανταλλάσσει την θέση του με το 24



A. Θεωρία

1. Δένδρο - Σωρός

6. Υλοποίηση σε C: Διαγραφή Κόμβου

```
/* HEAP_delete(): Διαγραfei ti riza tou dendrou */
int HEAP_delete(HEAP *heap, elem *x)
{
    int posCurrent, posLeft, posRight, pos;
    elem temp;

    /* An o swros einai adeios */
    if (heap->N == 0)
        return FALSE;

    /* 1. Sigkratisi (epistrofi) tis rizas */
    *x=heap->data[0];

    /* 2. Topothetisi tou teleutaiou stoxeiou sti riza */
    heap->data[0]=heap->data[heap->N - 1];
    heap->N --;
}
```

A. Θεωρία

1. Δένδρο - Σωρός

6. Υλοποίηση σε C: Διαγραφή Κόμβου

```
/* 3. Antallagi me to megalitero tw n paidiwn toy */
posCurrent=0;
while (posCurrent<heap->N)
{
    posLeft=2*posCurrent+1;
    posRight=2*posCurrent+2;

    if (posLeft >= heap->N)
        posLeft=-1;
    if (posRight >= heap->N)
        posRight=-1;

    /* 3.1. Den exei paidia */
    if (posLeft== -1 && posRight== -1)
        break;
}
```

A. Θεωρία

1. Δένδρο - Σωρός

6. Υλοποίηση σε C: Διαγραφή Κόμβου

```
/* 3.2. Exei mono aristero paidi */
else if (posLeft!=-1 && posRight==1)
{
    if (heap->data[posCurrent] < heap->data[posLeft])
    {
        swap(&heap->data[posCurrent], &heap->data[posLeft]);
        posCurrent=posLeft;
    }
    else
        break;
}
```

A. Θεωρία

1. Δένδρο - Σωρός

6. Υλοποίηση σε C: Διαγραφή Κόμβου

```
/* 3.3. Exei dyo paidia */
else // posLeft!=-1 && posRight!=-1
{
    /*3.3.1 Eyresi tou megaliterou apo ta dyo paidia */
    if (heap->data[posLeft] < heap->data[posRight])
        pos=posRight;
    else
        pos=posLeft;

    /*3.3.2 Antimetathesi an einai mikrotero */
    if (heap->data[posCurrent] < heap->data[pos])
    {
        swap(&heap->data[posCurrent], &heap->data[pos]);
        posCurrent=pos;
    }
    else
        break;
}
```

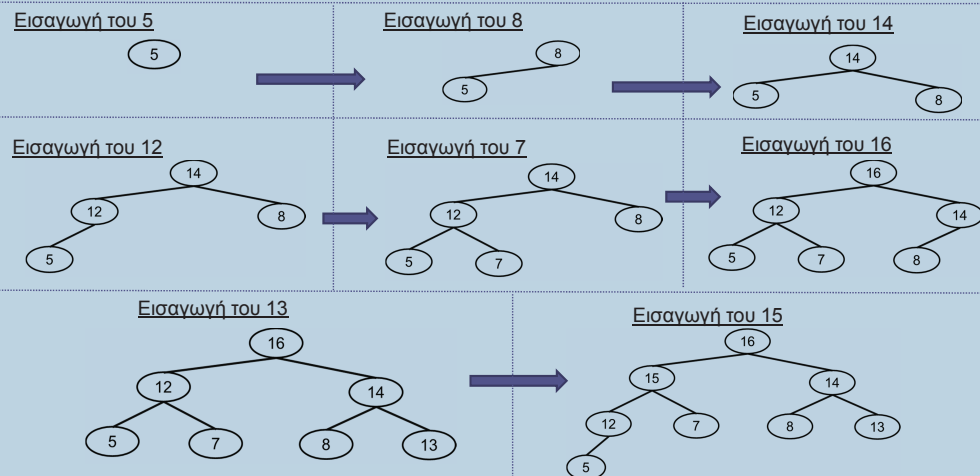
A. Θεωρία

1. Δένδρο - Σωρός

7. Ο αλγόριθμος ταξινόμησης HeapSort

Ο αλγόριθμος HeapSort χρησιμοποιεί τη δομή δεδομένων Σωρού, ώστε να ταξινομήσει μια ακολουθία δεδομένων: Αρχικά, τα δεδομένα εισάγονται στο σωρό.

Παράδειγμα: πίνακας [5, 8, 14, 12, 7, 16, 13, 15]

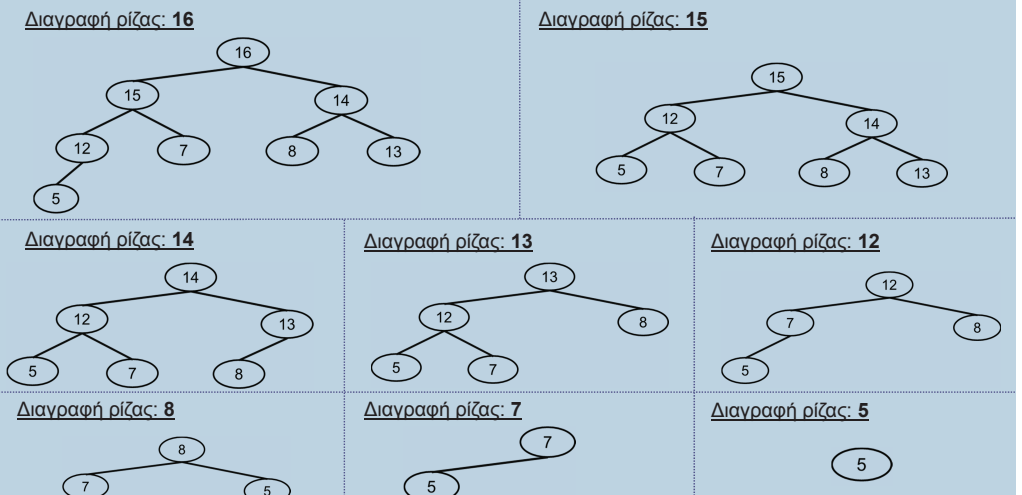


A. Θεωρία

1. Δένδρο - Σωρός

7. Ο αλγόριθμος ταξινόμησης HeapSort

Έπειτα με διαδοχικές διαγραφές διαγράφονται τα δεδομένα από το σωρό. Έτσι προκύπτει η ακολουθία των δεδομένων σε φθίνουσα σειρά.





B. Ασκήσεις

Εφαρμογή 1: Μελέτη Προγράμματος

- Μελετήστε το project heap.dev στο οποίο υλοποιούνται οι βασικές πράξεις του δένδρου-σωρού που μελετήσαμε στο μάθημα.



B. Ασκήσεις

Εφαρμογή 2: Δομές Δεδομένων «με το χέρι»

Δίνεται η ακολουθία δεδομένων «5 8 3 2 1 6 9 7»

1. Εισάγετε τα δεδομένα σε μία στοίβα.
2. Εξάγοντας τα δεδομένα από τη στοίβα, εισάγετέ τα σε ένα δυαδικό δένδρο αναζήτησης.



B. Ασκήσεις

Εφαρμογή 2: Δομές Δεδομένων «με το χέρι»

3. Δώστε την προδιατεταγμένη διαδρομή του δυαδικού δένδρου αναζήτησης.

4. Εισάγετε τα δεδομένα σε ένα σωρό ελαχίστων με σειρά την προδιατεταγμένη διαπέραση.



B. Ασκήσεις

Εφαρμογή 3: Δένδρο-Σωρός Ελαχίστων

- Τροποποιήστε το πρόγραμμα ώστε να υλοποιεί ένα δένδρο-σωρό ελαχίστων.