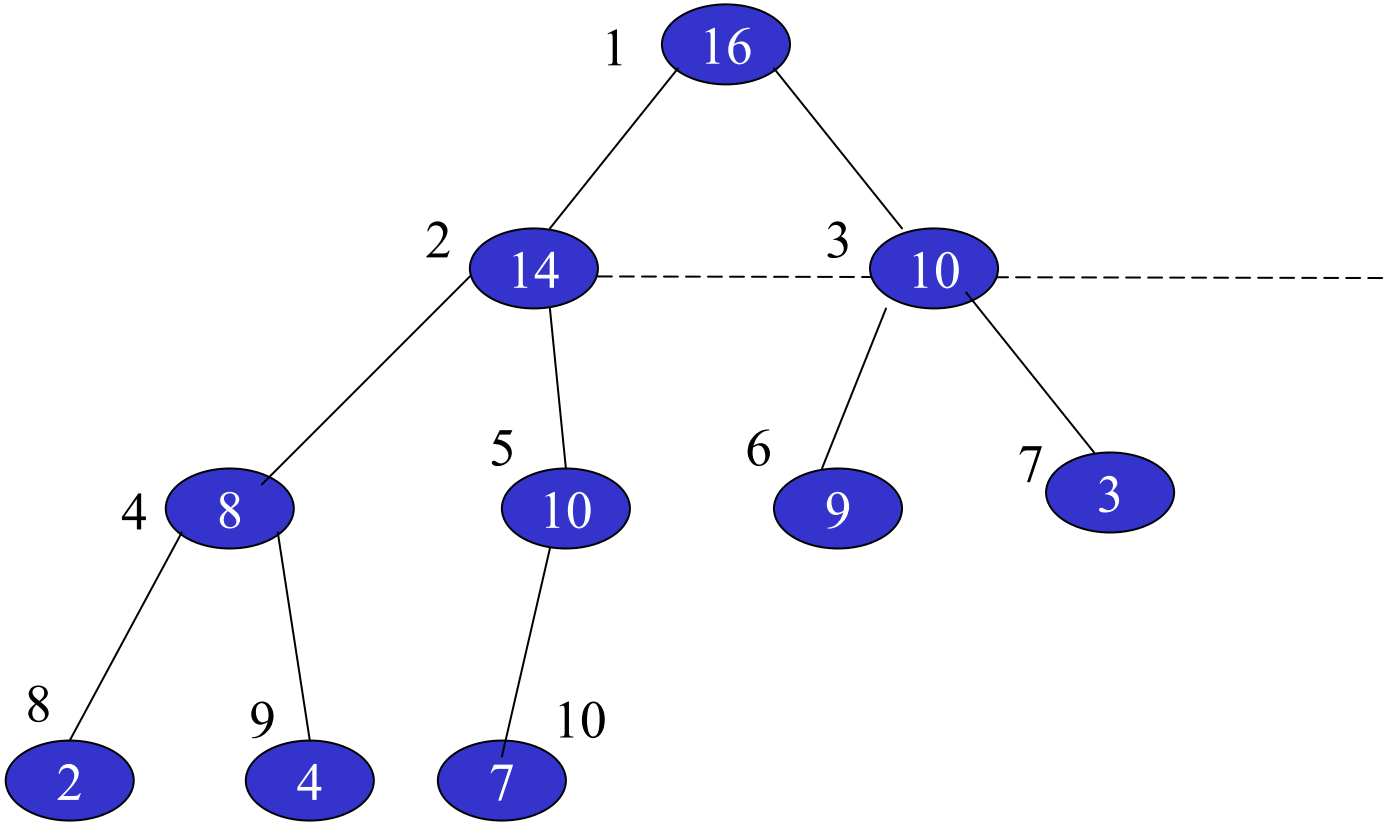
Εφαρμογές σωρού

Max heap: χρονοπρογραμματισμός

Min heap: simulation

Ταξινόμηση με τη βοήθεια σωρού

Εφαρμογές σωρού



14→ 13

14 → 6

Διατήρηση σωρού

```
MAX-HEAPIFY (A, i)
l  ←  left [i]
r  ←  right (i)
if  l  ≤  heap-size [A] and A [l] >A [i]
     then largest ← l
     else largest ← i
if r ≤ heap-size [A] and A [r]>A [largest]
     then largest ← r
if largest ≠ i
     then exchange A [i] ↔ A [largest]
          MAX-HEAPIFY [A, largest]
```

Διατήρηση σωρού: πολυπλοκότητα

Πολυπλοκότητα

$$T(n) \leq T(2n/3) + \Theta(1)$$

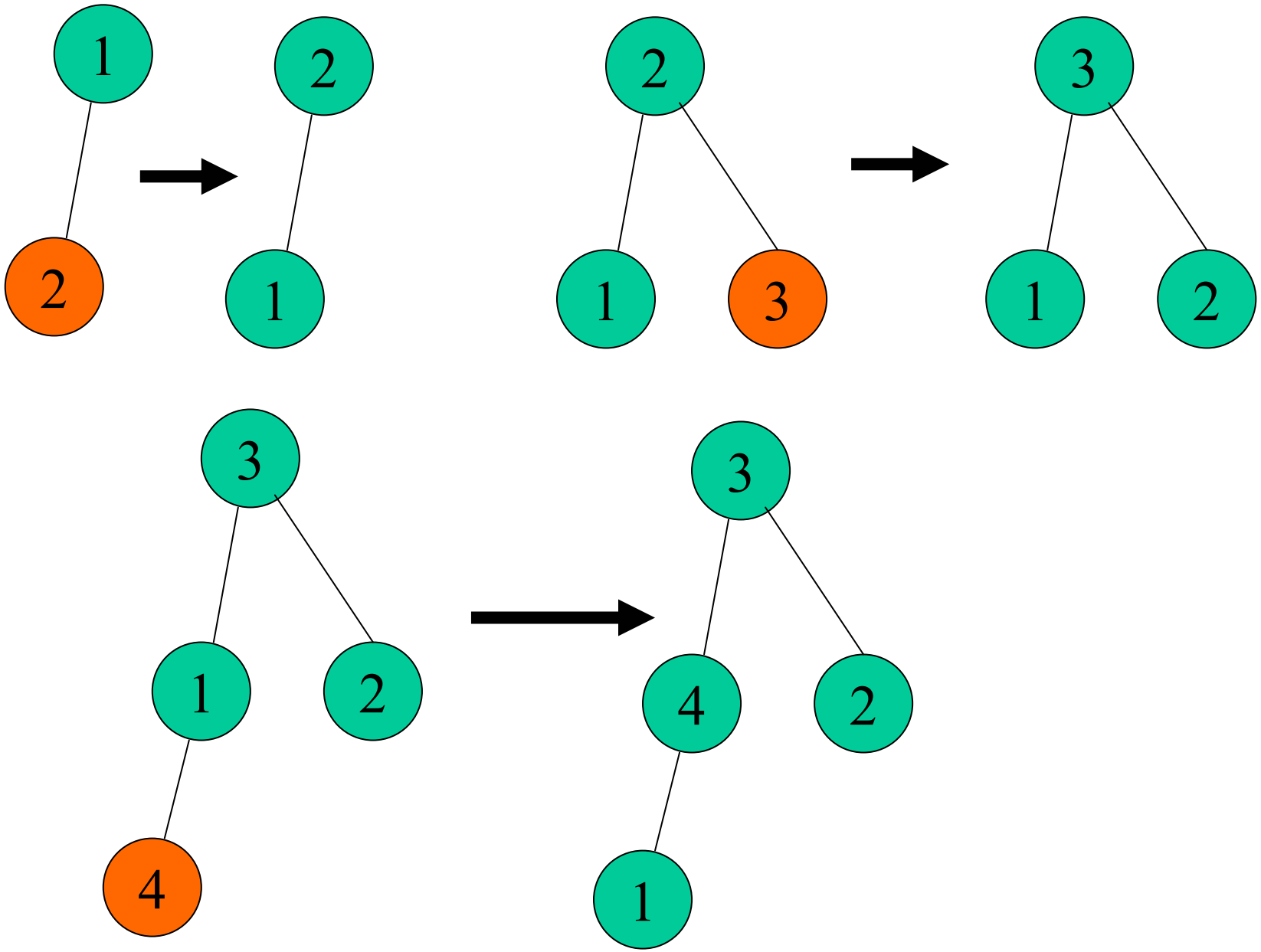
$$T(n) = O(h) = O(\log n),\ h = \text{ύψος}$$

Input: τα στοιχεία φθάνουν το ένα μετά το άλλο: a[i],
$1 \leq i \leq n$ **(τα στοιχεία δεν είναι όλα διαθέσιμα)**

Output: Δομή σωρού

Ακολουθία στοιχείων:

1 2 3 4 . . .


1
2
2
1
2
1
3
1
3
2
3
1
2
4
3
4
2
1

```
ConstructHeap;

  nheap := 0;
  for i := 1 to n do
    insert(a[i]);
end; ****
```

Πολυπλοκότητα

$$n! \leq n^n$$

$$n! \geq (n/2)^{n/2}$$

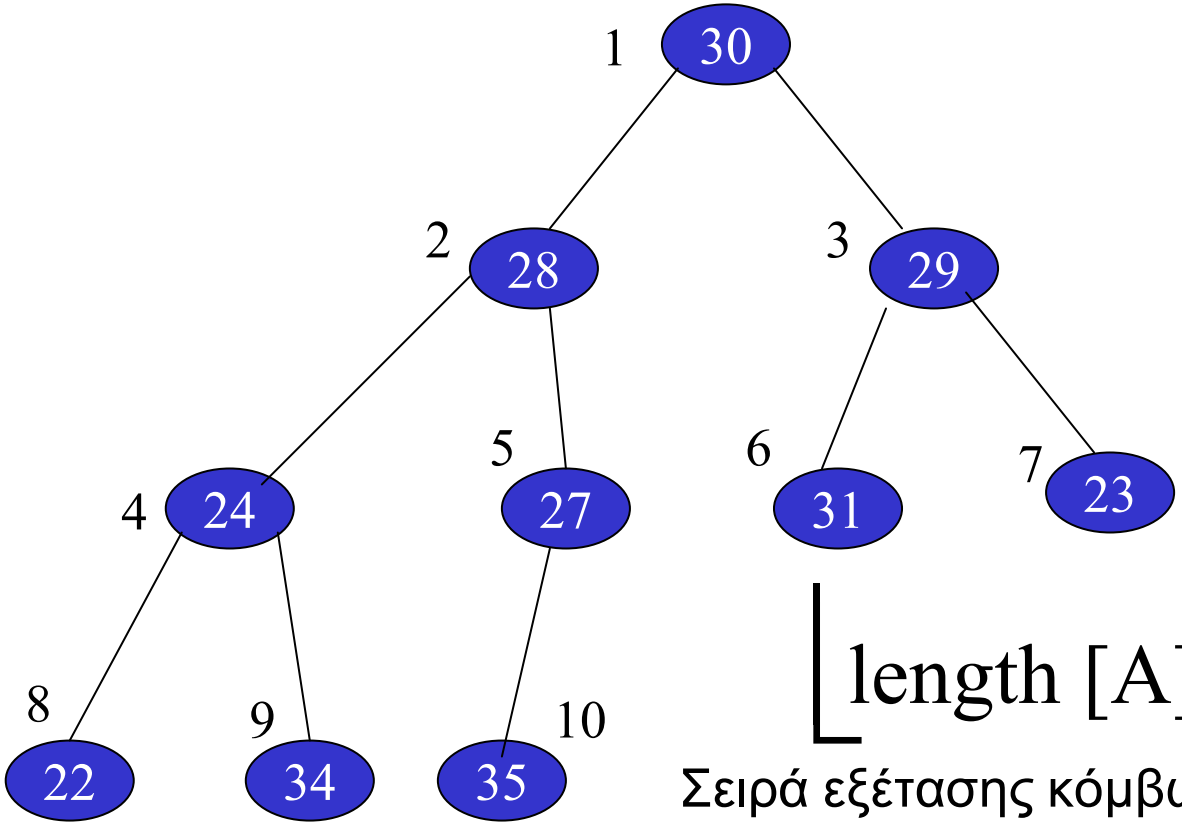
Κατασκευή σωρού σε O(n)

Τα στοιχεία είναι όλα διαθέσιμα στο πίνακα A

BUILD-MAX-HEAP (A)

heap-size [A] ← length [A]
for i ← ⌊length [A]/2⌋ downto 1 do
    MAX-HEAPIFY (A, i)

Κατασκευή σωρού σε O(n)



$\lfloor \text{length}[A]/2 \rfloor$

Σειρά εξέτασης κόμβων: 5,4,3,2,1

Κατασκευή σωρού σε O(n)

Πολυπλοκότητα:

Το πολύ $\left\lceil \frac{n}{2^{h+1}} \right\rceil$ κόμβοι ύψους h

$$T(n) = \sum_{h=1}^{\lfloor \log n \rfloor} \left\lceil \frac{n}{2^{h+1}} \right\rceil O(h) = \ldots O(n)$$

- Υπενθύμιση: $$\sum_{h=0}^{\infty} \frac{h}{2^h} = \frac{\frac{1}{2}}{\left(1-\frac{1}{2}\right)^2} = 2$$

Υπενθύμιση

Για $x \neq 1$ $$\sum_{k=0}^{n} x^k = 1 + x + x^2 + \ldots + x^n = \frac{x^{n+1} - 1}{x - 1}$$

Για $|x| < 1$ $$\sum_{k=0}^{\infty} x^k = \frac{1}{1 - x}$$

$$\sum_{k=0}^{\infty} k x^k = \frac{x}{(1 - x)^2}$$

Αλγοριθμος ταξινόμησης με σωρό

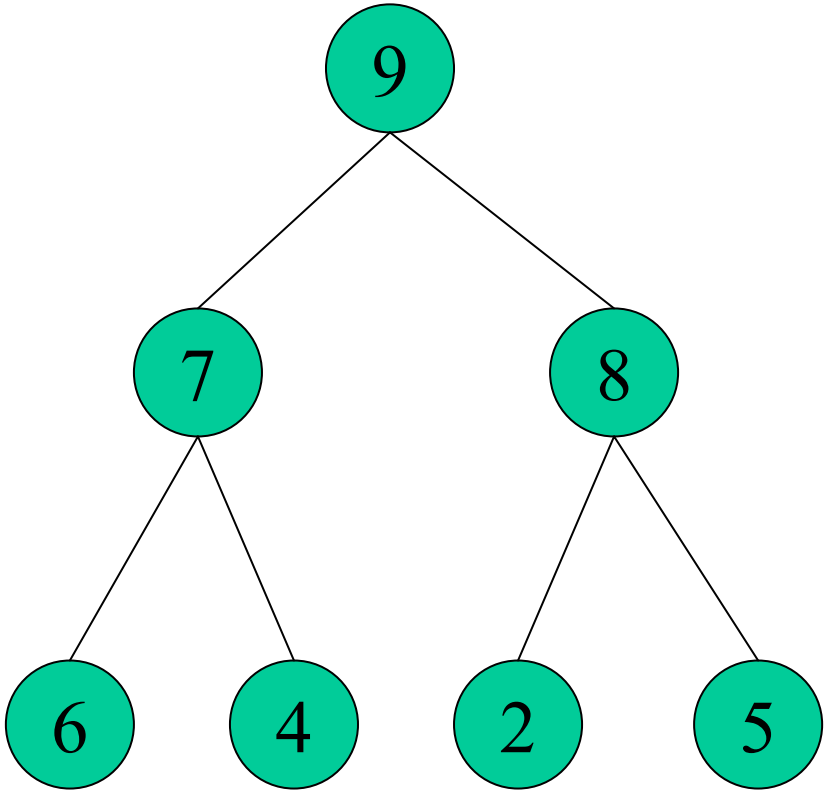
Input: πίνακας a[i], $1 \leq i \leq n$
Output: πίνακας $\mathbf{a[i_j]}$, $\mathbf{a[i_j] \leq a[i_{j+1}]}$

Βήμα 1: Διαμόρφωση πίνακα σε σωρό

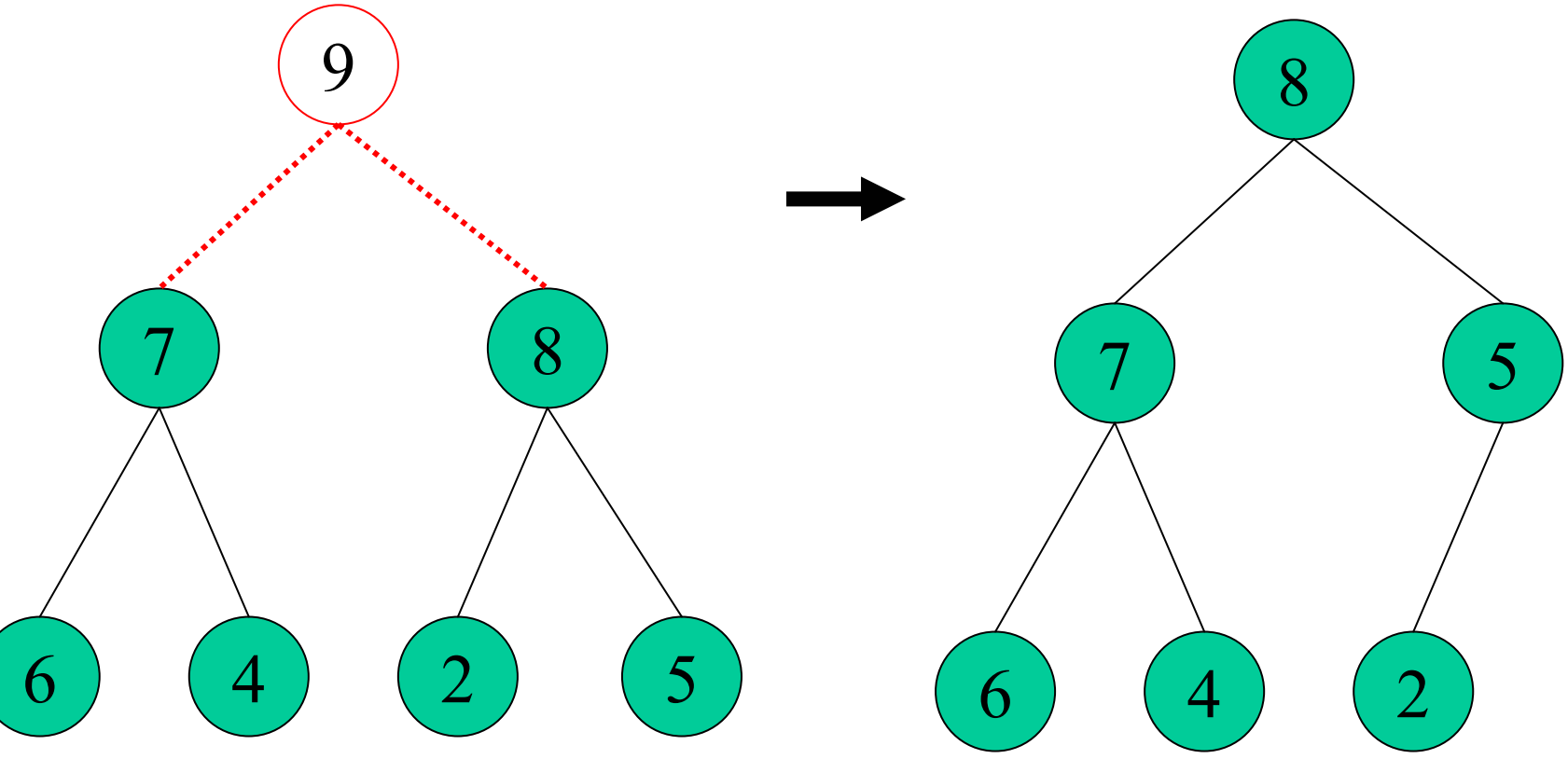
Βήμα 2: Επανέλαβε

- Πάρε μεγαλύτερο στοιχείο
- Αφαίρεσέ το από το σωρό
- Τοποθέτησέ το δεξιά στον πίνακα

6 9 2 7 4 5 8



9 7 8 6 4 2 5



```
HeapSort()

  nheap = 0;
  for i = 0 το n-1
    Insert(a[i])****

  for i = n-1 το 0
    v = Maximum()
    Delete()
    a[i] = v
```

Πολυπλοκότητα

Βήμα 1: $\Theta(n \log n)$

Βήμα 2: $\Theta(n \log n)$

$$\sum_{k=1}^{n} \log k = \log\left(\prod_{k=1}^{n} k\right) = \log(n!)$$