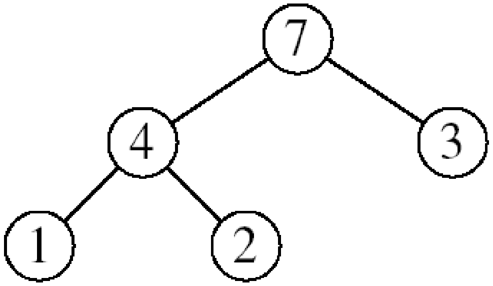
Heapsort

- Goal: sort an array using heap representations
- Procedure:
  - Build a max-heap from the array
  - Swap the root (the maximum element) with the last element in the array
  - “Discard” this last node by decreasing the heap size
  - Call Max-Heapfy on the new root
  - Repeat process until only one node remains



Heapsort running time

```
Heapsort (A)
  Build-Max-Heap (A)
  for i ← length[A] downto 2
     do exchange A[1] ↔ A[i]
        heap-size[A] ← heap-size[A]-1
        Max-Heapfy (A,1)
```

O(n) *O(n)*

n-1 times

O(1)

O(1)

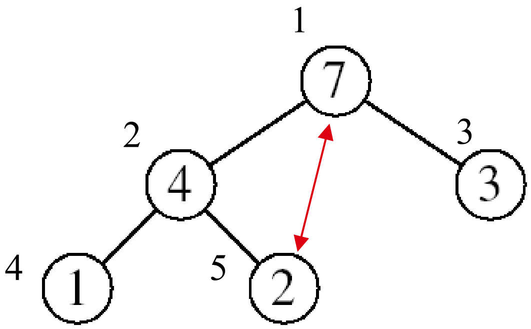
O(lgn) *O(lgn)*

n-1 times

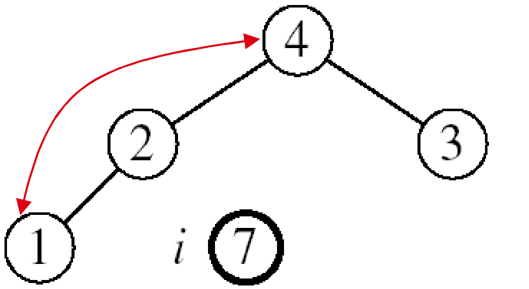
- We discard the previous root when applying Max-Heap (to the remaining heap)
- Running time is *O(n lg n)* + Build-Heap(*A*) time, which is *O(n)*

Example 1

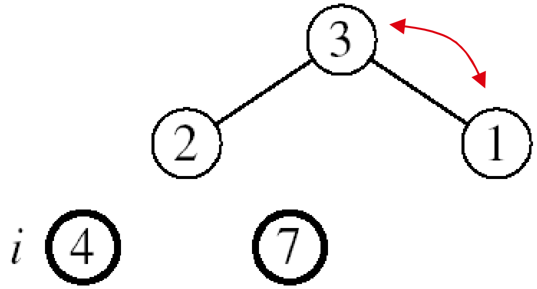
A:	7	4	3	1	2



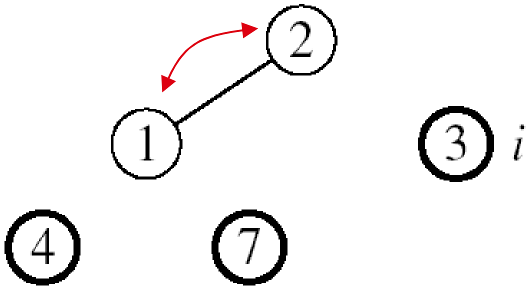
Max-Heapify (A, 4)



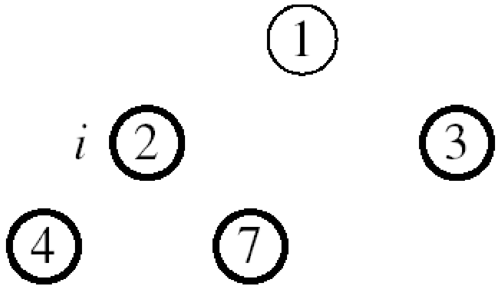
Max-Heapify (A, 3)



Max-Heapify (A, 2)



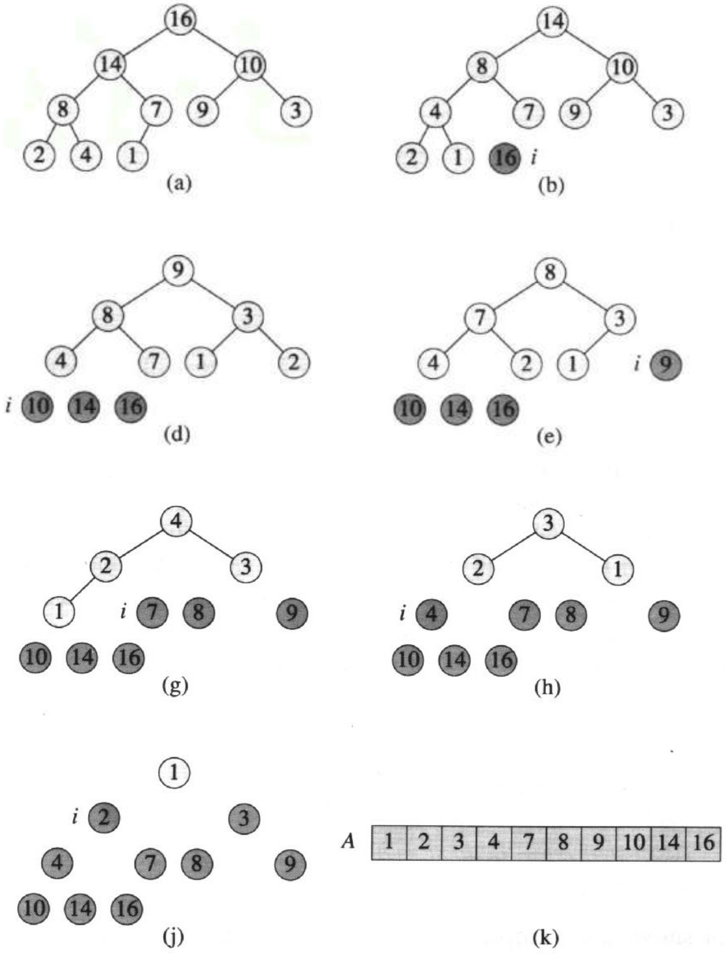
Max-Heapify (A, 1)



A:	1	2	3	4	7

Example 2

A:									
16	14	10	9	8	7	4	3	2	1



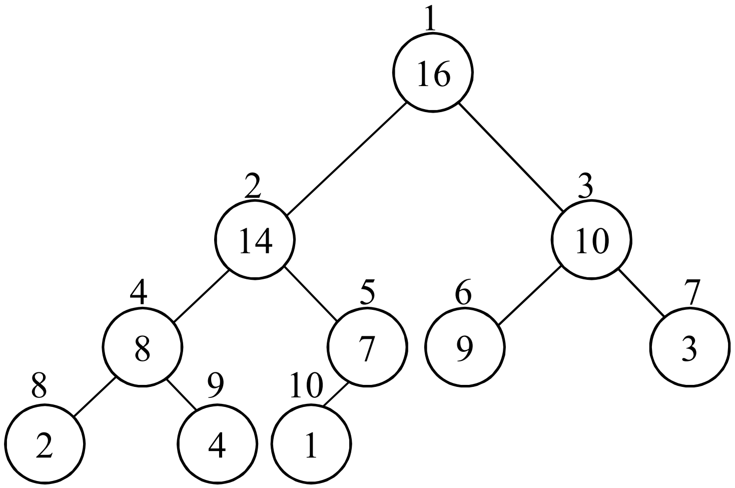
A									
1	2	3	4	7	8	9	10	14	16

Summary

- Heapsort uses a heap data structure to improve selection sort and make the running time asymptotically optimal
- Running time is *O(n log n)*
  - Like merge sort, but unlike selection, insertion, or bubble sorts
- Sorts in place
  - Like insertion, selection or bubble sorts, but unlike merge sort

Exercise

- Assuming the data in a max-heap are distinct, what are the possible locations of the second-largest element?



Exercise

1. Given a max heap B of height h
   a) What is the maximum number of nodes in B?
   b) What is the maximum number of leaves?
   c) What is the maximum number of internal nodes?

Exercise

- Demonstrate, step by step, the operation of Build-Heap on the array

  A=[5, 3, 17, 10, 84, 19, 6, 22, 9]

Exercise

- Let A be a heap of size n. Give the most efficient algorithm for the following tasks:
  (a) Find the sum of all elements
  (b) Find the sum of the largest lgn elements

Next Week

- Hashing

Acknowledgement

- A large part of this material were adapted from
  - Simonas Šaltenis, Algorithms and Data Structures, Aalborg University, Denmark
  - Mary Wootters, Design and Analysis of Algorithms, Stanford University, USA
  - George Bebis, Analysis of Algorithms CS 477/677, University of Nevada, Reno
  - David A. Plaisted, Information Comp 550-001, University of North Carolina at Chapel Hill

Questions