Binary Trees

Lecture 14

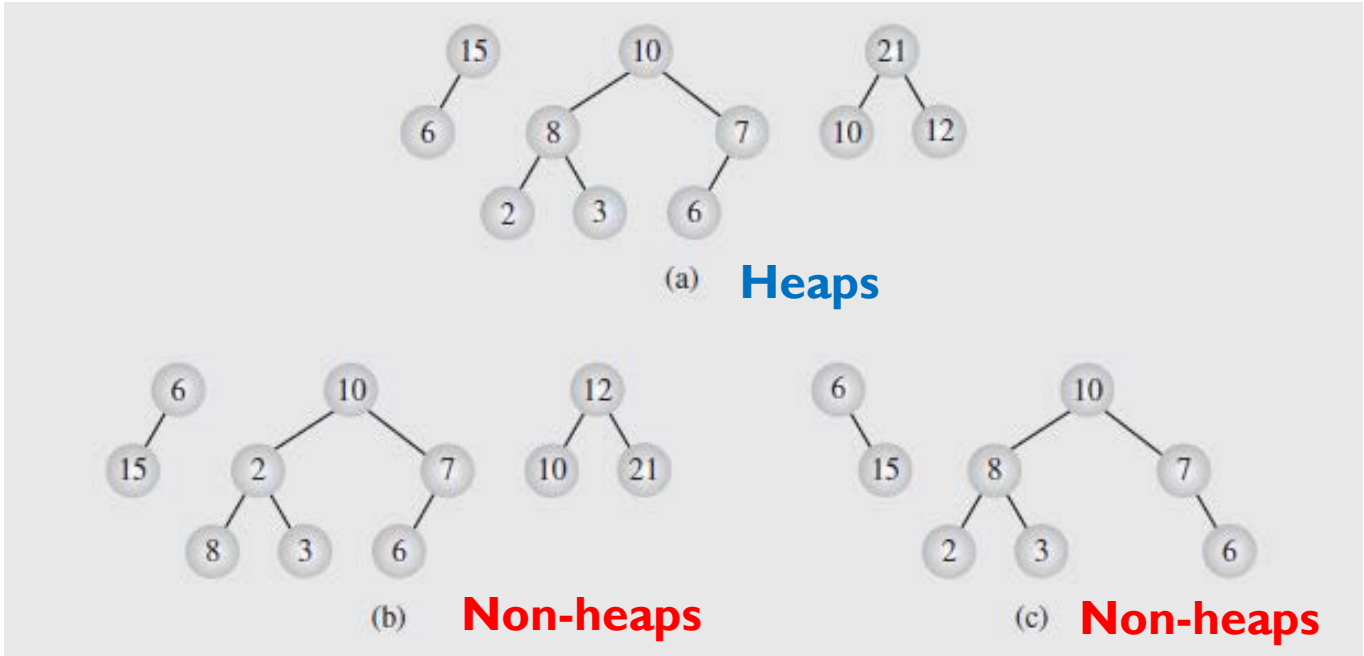
Instructor: Dr. Cong Pu, Ph.D.

cong.pu@okstate.edu

Adapted partially from Data Structures and Algorithms in Java, M.T. Goodrich, R.Tamassia and M. H. Goldwasser, Sixth Edition,Wiley; Data Structures and Algorithms in C++,Adam Drozdek, 4th Edition, Cengage Learning

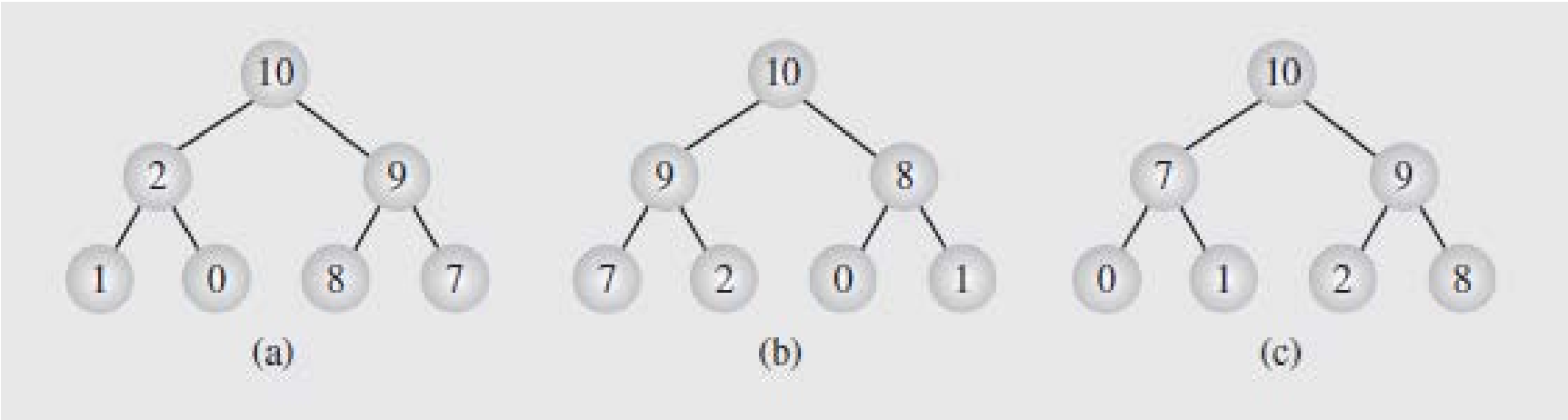
Heaps

- A ***heap,*** a special type of *binary tree*
  - the value of each node is **greater than or equal** to the values stored in its **children**
  - the tree is ***perfectly balanced***, and the leaves in the last level are **leftmost** in the tree



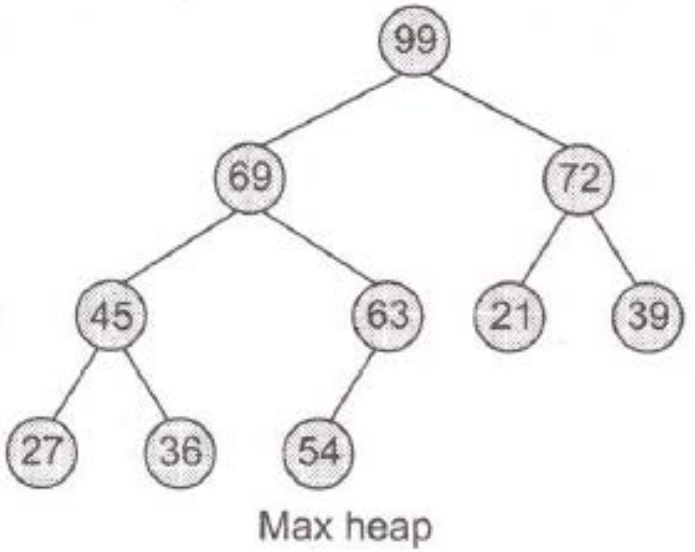
Heaps (cont.)

- Different heaps constructed with the same elements

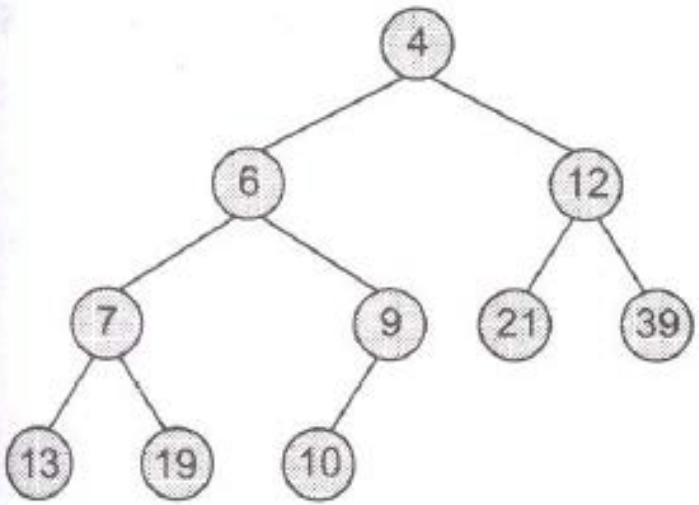


Heaps (cont.)

- A ***max heap***;
  - the value of each node is **greater than or equal** to the values stored in its **children**
  - the root of a max heap, the **largest** element
- A ***min heap***
  - the value of each node is **less than or equal** to the values stored in its **children**
  - the root of a min heap, the **smallest** element



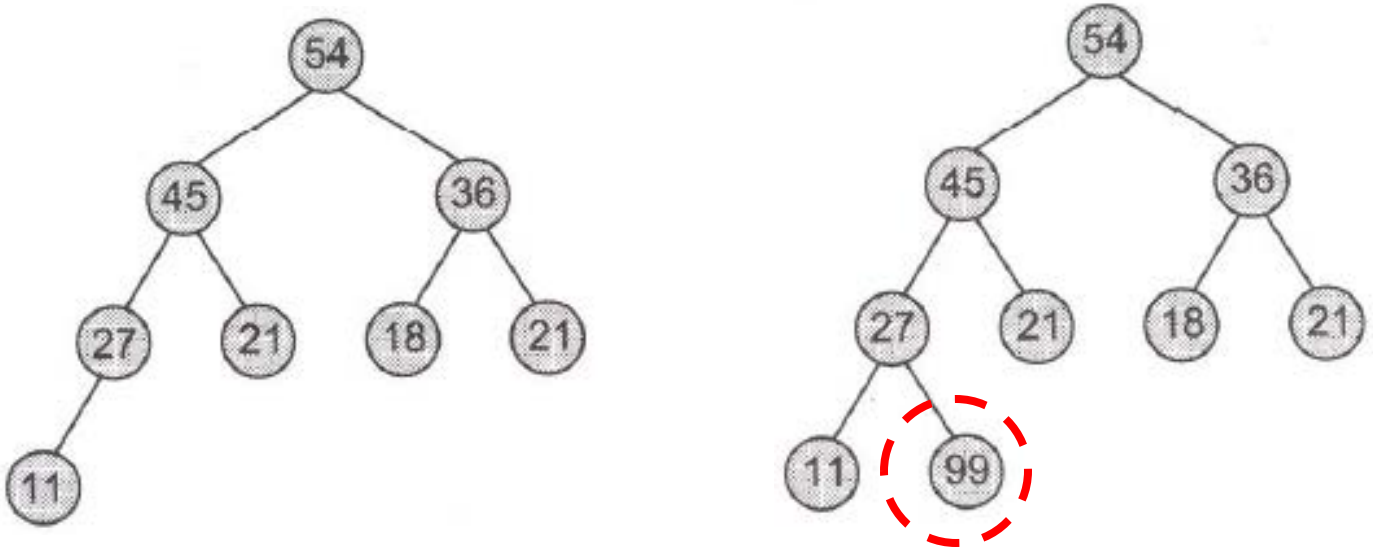
Max heap



Min heap

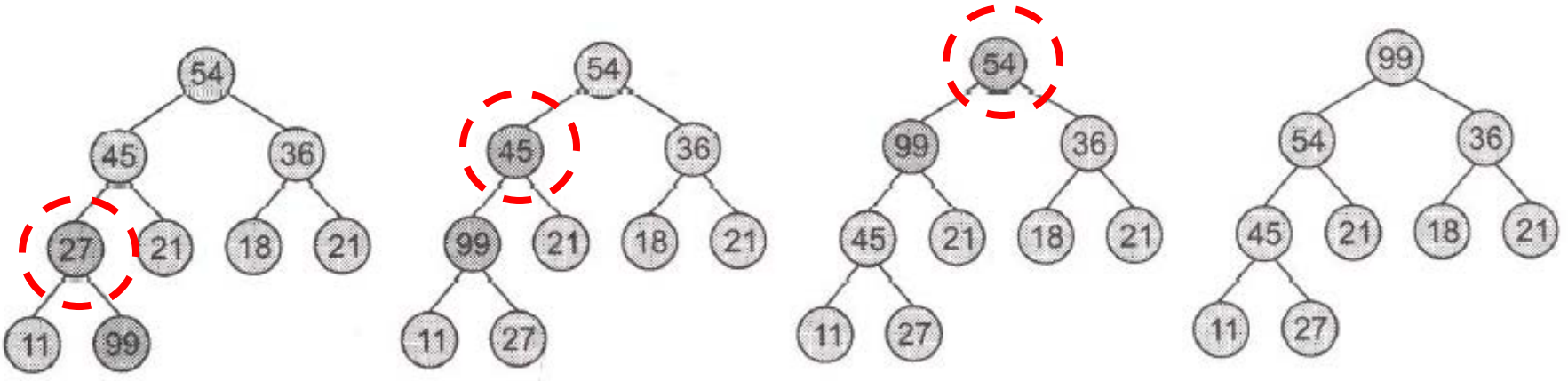
Heaps (cont.)

- Insert a new element, 99, in **max heap**
  - **leftmost** in the heap



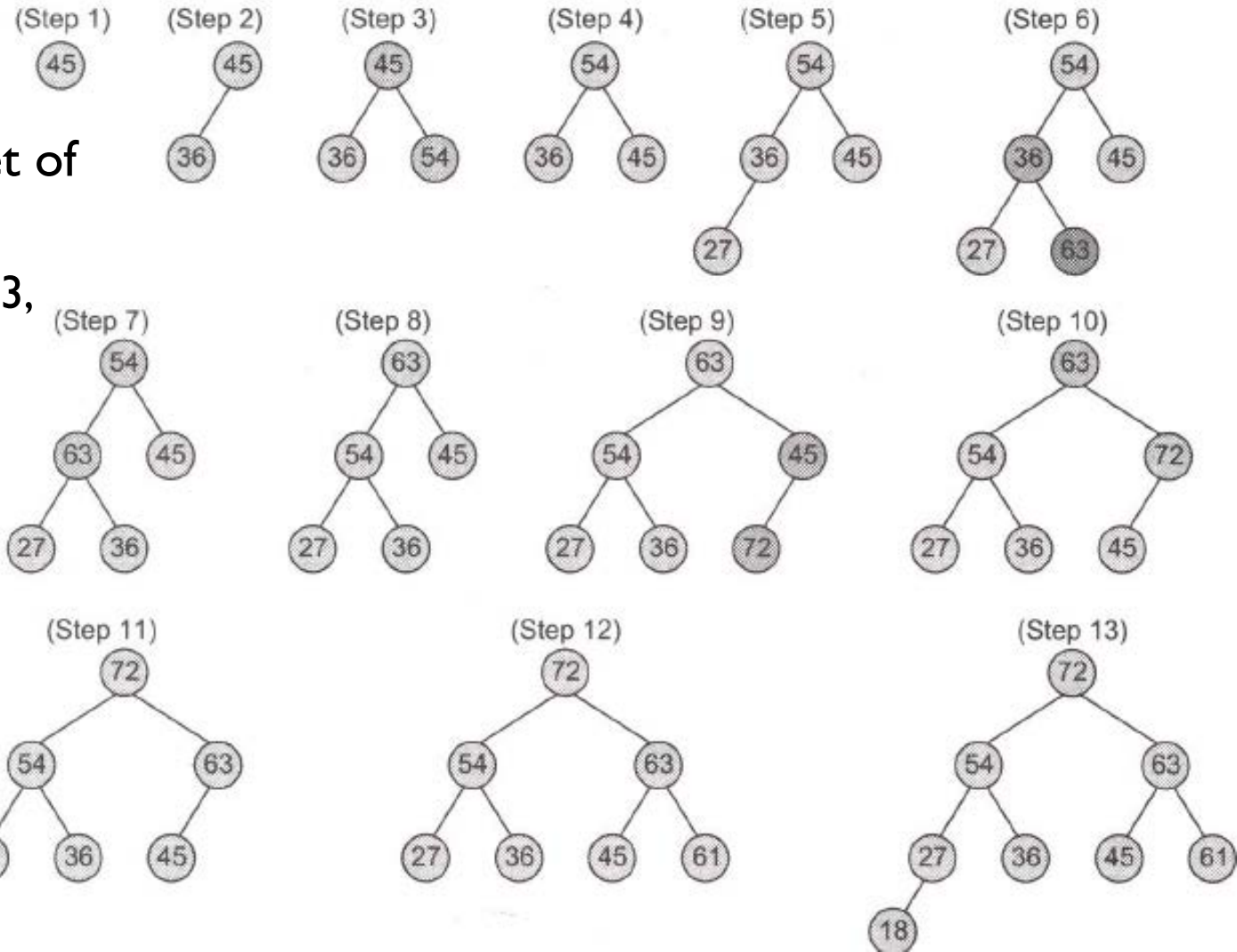
Heaps (cont.)

- Insert a new element, 99, in **max heap** (cont.)
  - heapify
    - the process of creating a heap data structure from a binary tree. It is used to create a Min-Heap or a Max-Heap



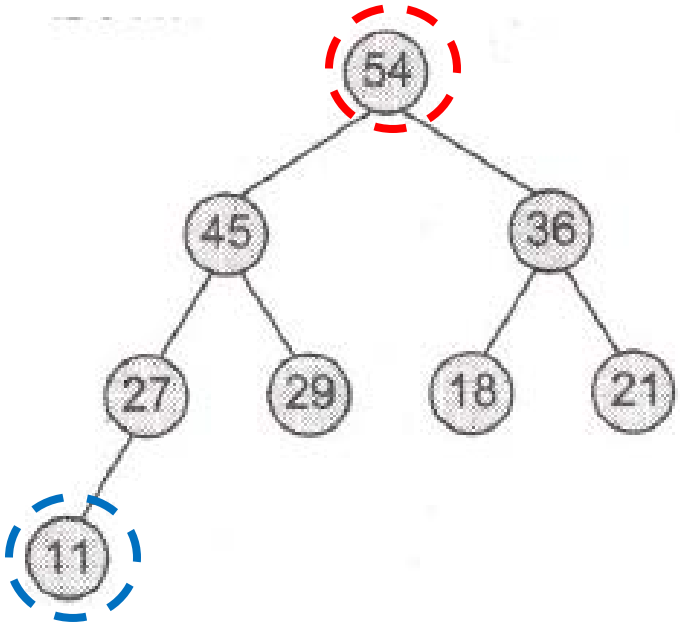
Heaps (cont.)

- Build a max heap from the given set of numbers
  - 45, 36, 54, 27, 63, 72, 61, 18



Heaps (cont.)

- Delete a node in **max heap**
  - always deleted from the root of the heap
  - **replace** the root node with the last node



Heaps (cont.)

- Delete a node in **max heap** (cont.)
  - always deleted from the root of the heap
  - **replace** the root node with the last node