Data Structures and Algorithms- Lab 11

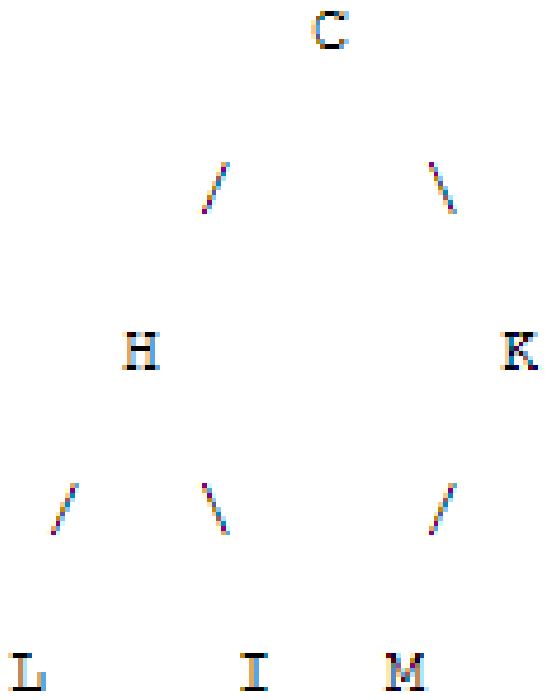
Iulia-Cristina Stanica
iulia.stanica@gmail.com

Roadmap
Heap

- Heap
- C++ implementation

Almost complete binary tree

- **Def:** An almost complete binary tree is a binary tree which has the following properties:
  - All the leaves are on the last level of the tree (or the last 2 levels)
  - All the leaves are situated at the leftmost positions
  - All levels are fully occupied (except for, eventually, the last level)

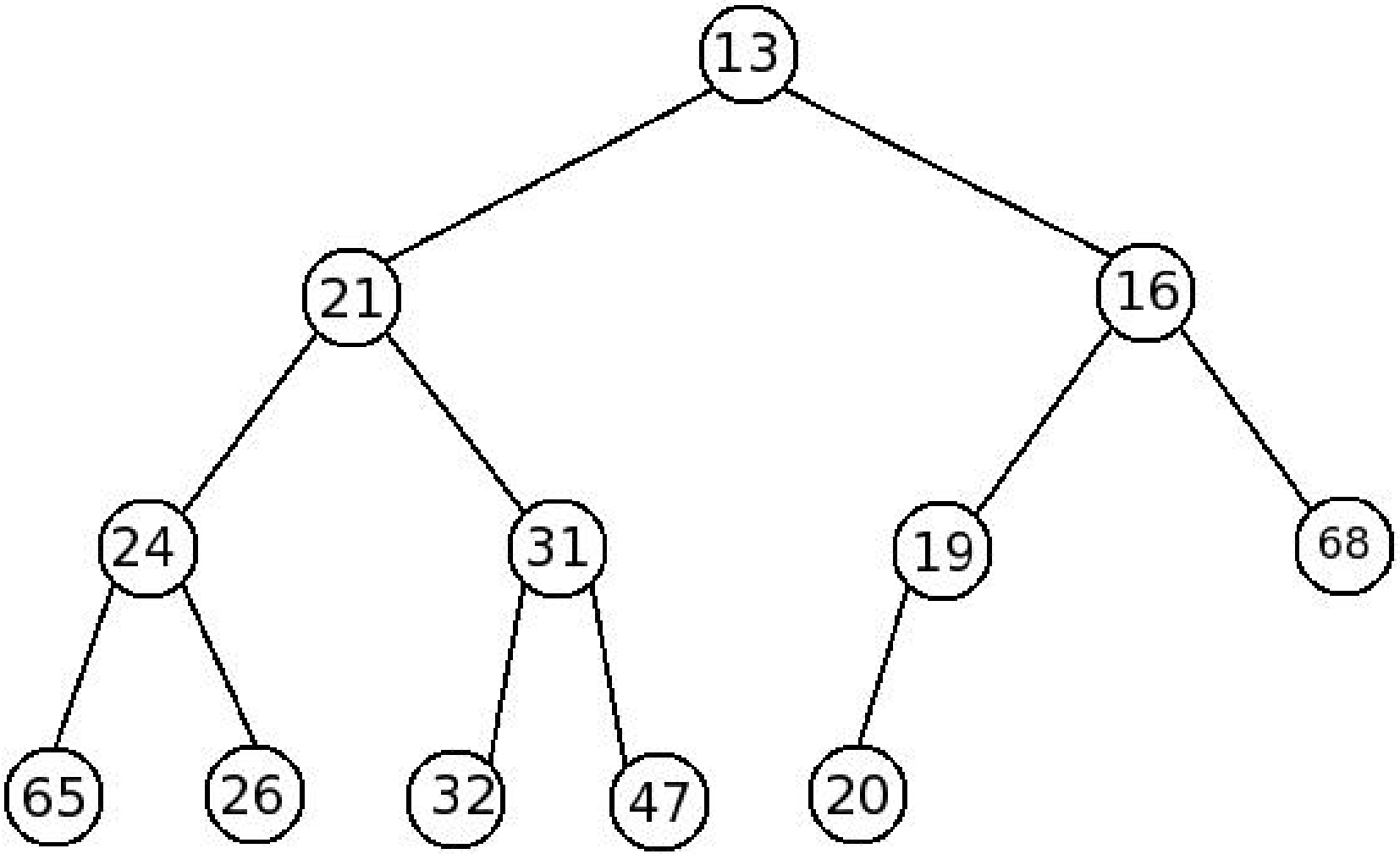


Heap

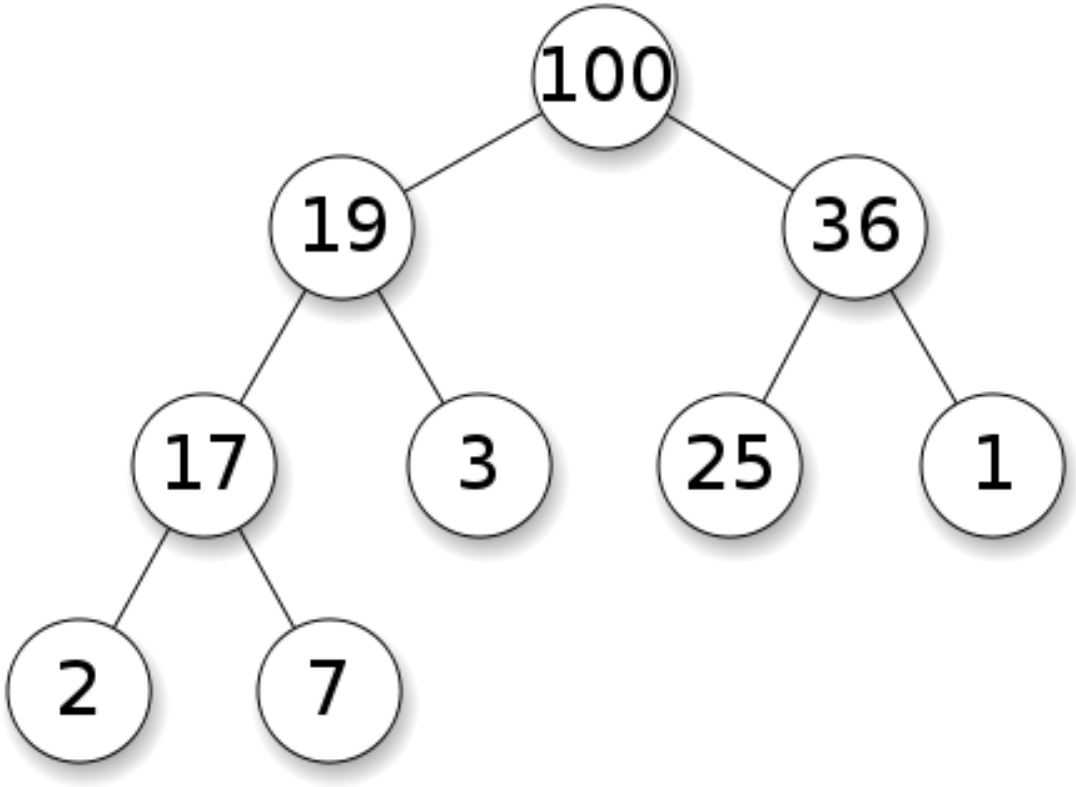
- **Def**: A min heap is an almost complete binary tree where the value of each parent node is less or equal to the values of the children.

- Observations:
  - The min value is in the root.
  - A path from a leaf to the root gives a sequence of numbers in decreasing order.

Min heap: lowest at top



MAX Heap



Operations

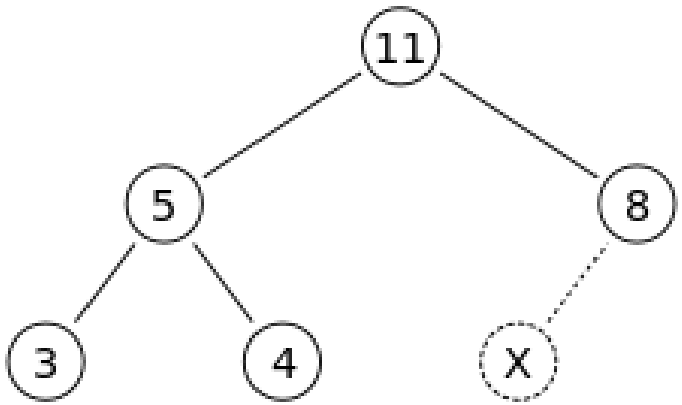
- Insertion
- Removal

1. Insertion

- Steps:
  - 1. Add the element on the last level, leftmost possible;
  - 2. Compare the added element with its parent; if the order is right, stop;
  - 3. If not, change the element with its parent and return to step 2.

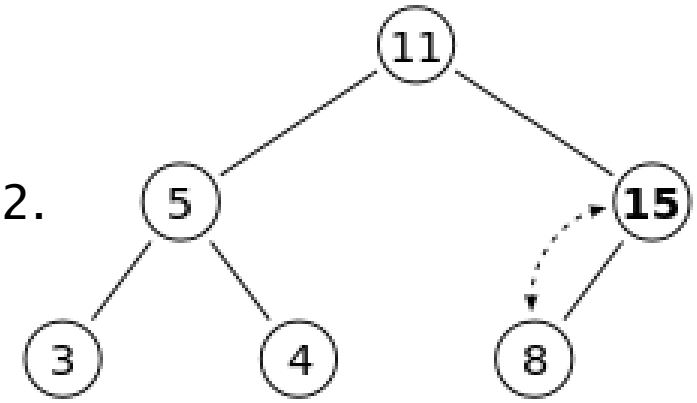
Example (max heap):

1.



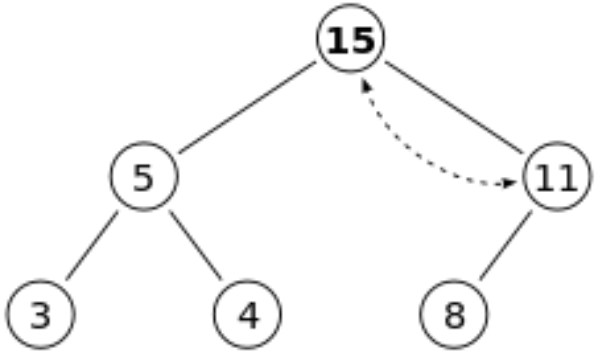
We add 15 (instead of X)

2.



We compare 15 with its parent and swap them

3.



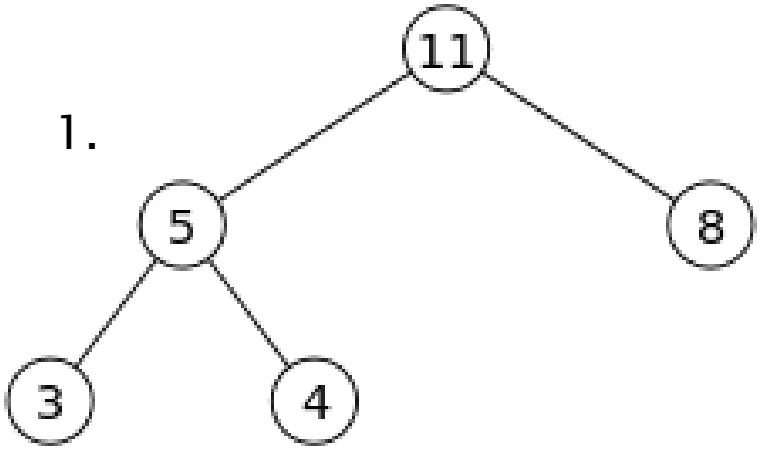
Same operation between 15 and the root

2. Removal

- We delete always the root of the heap.
- Steps:
  - 1. Temporarily replace the root with the last element from the last level.
  - 2. Compare the new root with its children; if the order is right, stop
  - 3. If not, change the element with its biggest child and return to step 2.

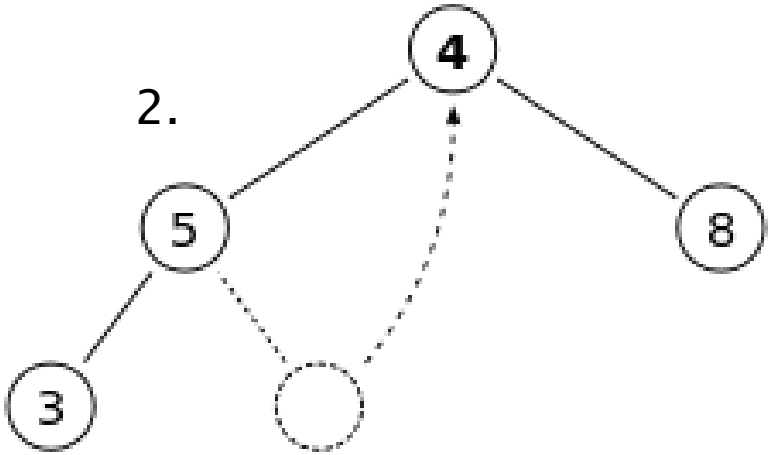
Example (max heap):

1.



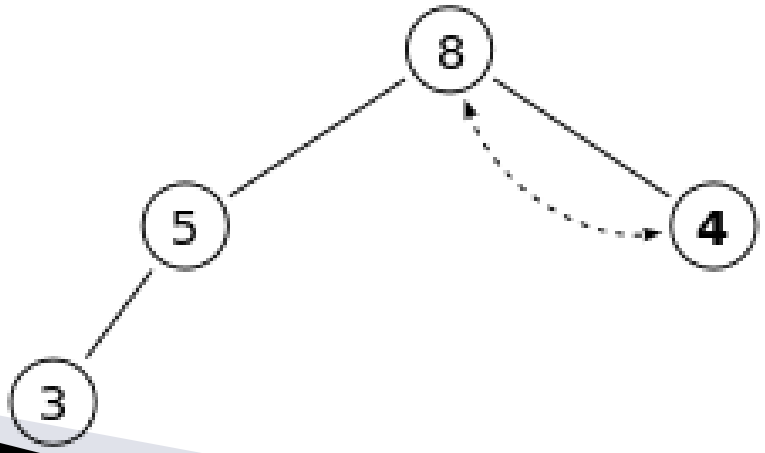
We delete the root (11)

2.



We put the last children in its place

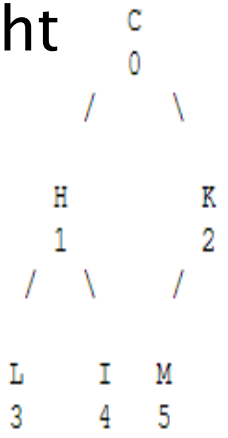
3.



We replace it with its biggest child

Implementation of Heap

- as a binary tree
- as an array
  - Define the numbers of the nodes based on the levels, from top to bottom, left to right
  - Save the values in an array
  - If CI is the current index:
    - Parent(CI) = (CI – 1) / 2
    - RightChild(CI) = 2 * (CI + 1)
    - LeftChild(CI) = 2 * CI + 1



C	H	K	L	I	M
0	1	2	3	4	5

CI pour H =1=> RightChild pour H (CI)=2*(1+1)=4 => I

Ex. 1

- Add to the Heap class the functions which link the parent with its children (formulas slide 12).

Attention!
In the Heap class, the array starts at index 1, not 0! You should change the formulas accordingly.
Each function has this structure:

```
int Parent (int CI)
{
        //operations based on CI
}
```

Ex. 2

- Add a function to display the heap. Add also the function to display the heap by levels. Test your implementation.

Homework

- We call a K-ary heap a heap where each node has maximum K children (not only 2). Change the implementation to get a Heap of order 3. Test your implementation.