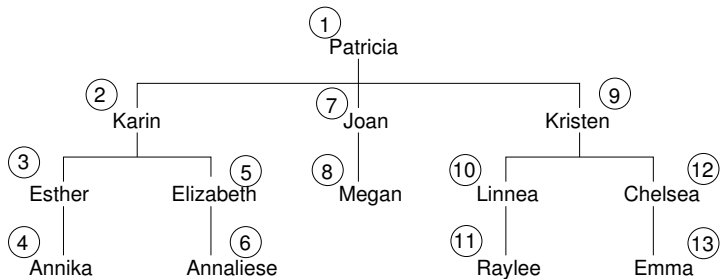
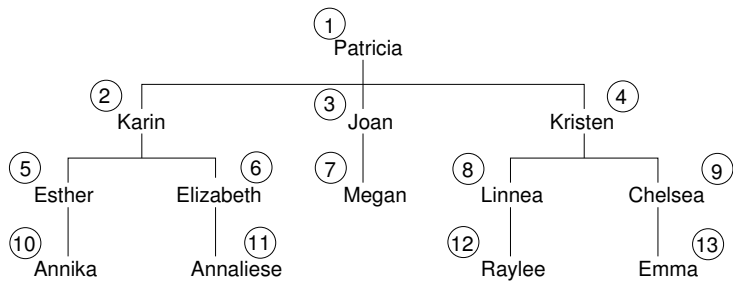


Chapter 3, Case Studies:

- ▶ Linear-time sorting algorithms (last Wednesday)
- ▶ Disjoint sets and array forests (last Friday)
- ▶ Priority queues (**Today**)
- ▶ N -sets and bit vectors (Wednesday)
- ▶ (Start graphs Friday)

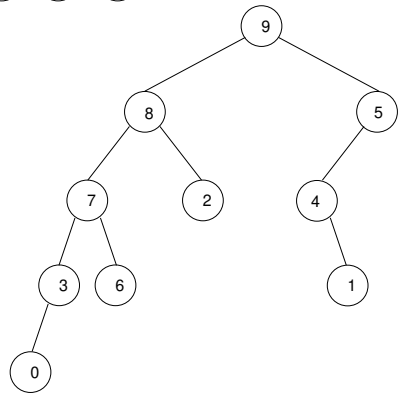
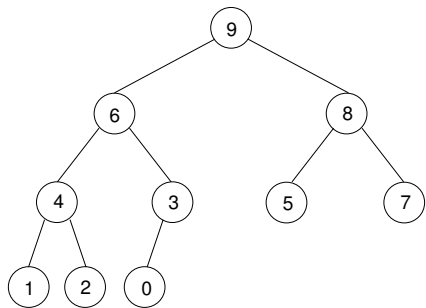
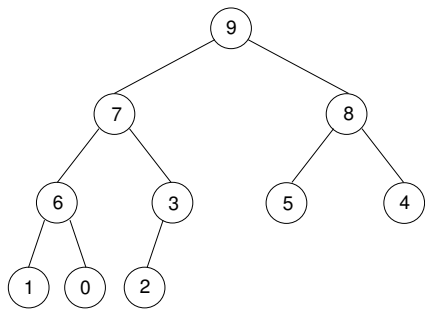
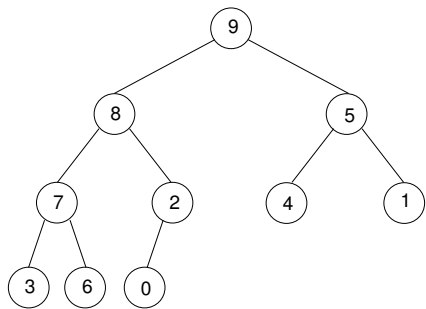
Today:

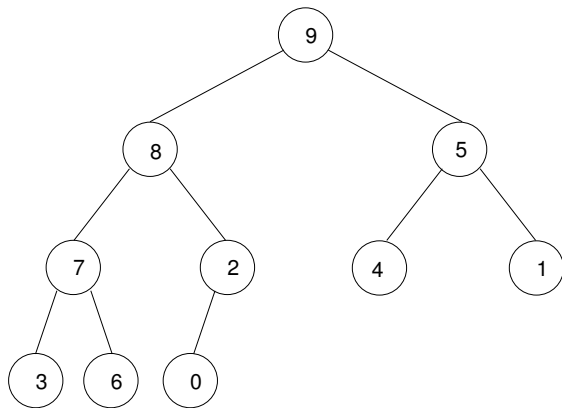
- ▶ Worklist algorithms
- ▶ Priority queue ADT (problem statement)
- ▶ Inefficient solutions
- ▶ Abstractions for the heap data structure
- ▶ Heap implementation details, part 1
- ▶ Excursion: heap sort
- ▶ Heap implementation details, part 2
- ▶ Analysis and optimization



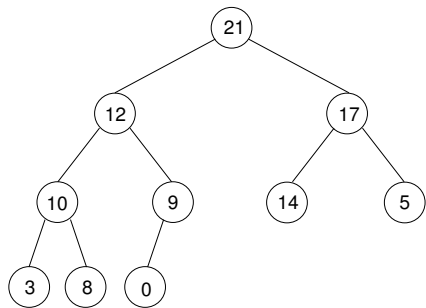
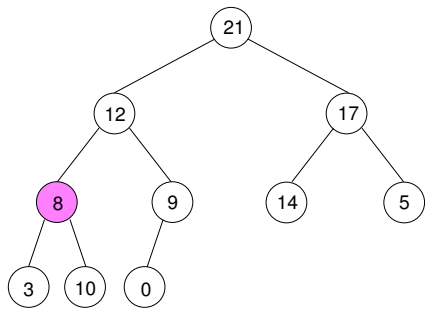
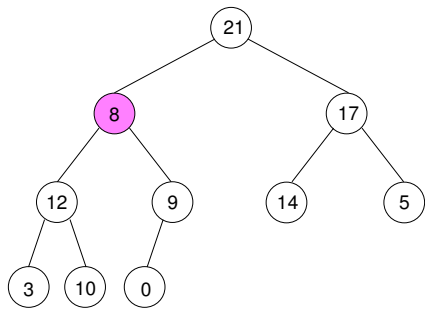
	ListPriorityQueue	SortedPriorityQueue
--	-------------------	---------------------

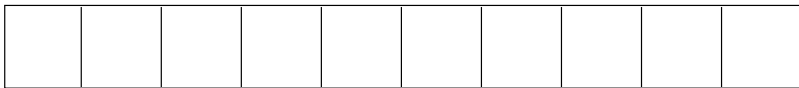
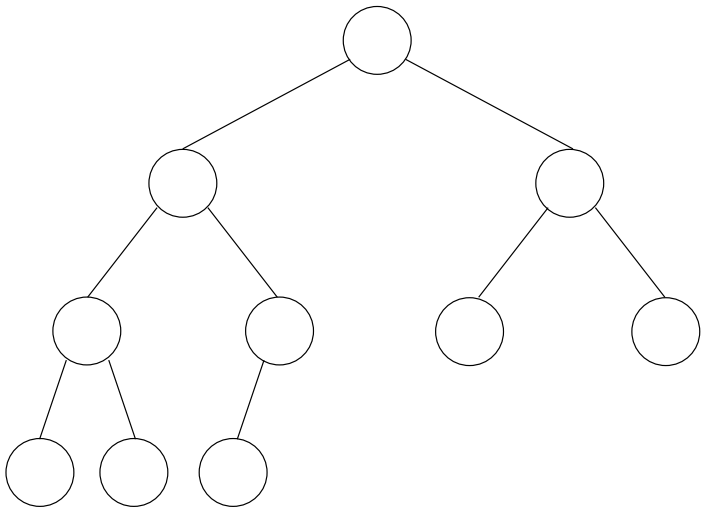
Initialize empty	$\Theta(1)$	$\Theta(1)$
Initialize populated	$\Theta(n)$	$\Theta(n^2)$
insert	$\Theta(1)$	$\Theta(n)$
max	$\Theta(n)$	$\Theta(1)$
extractMax	$\Theta(n)$	$\Theta(1)$
contains	$\Theta(n)$	$\Theta(n)$
increaseKey	$\Theta(1)$	$\Theta(n)$
decreaseKey	$\Theta(1)$	$\Theta(n)$



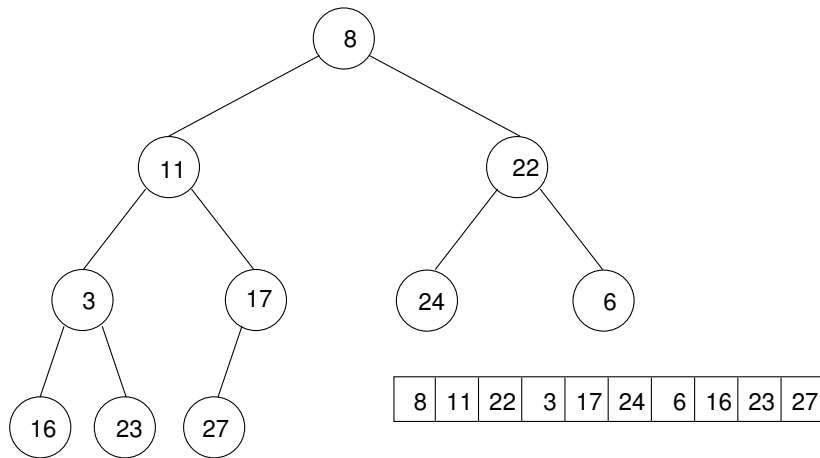


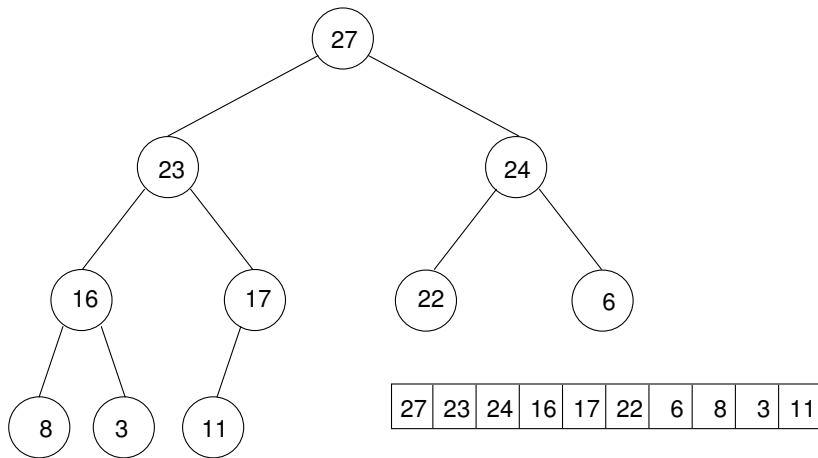
9	8	5	7	2	4	1	3	6	0
---	---	---	---	---	---	---	---	---	---

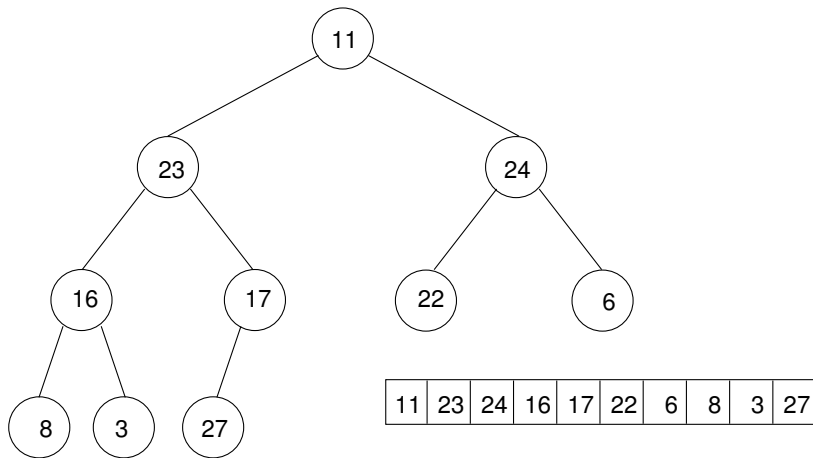


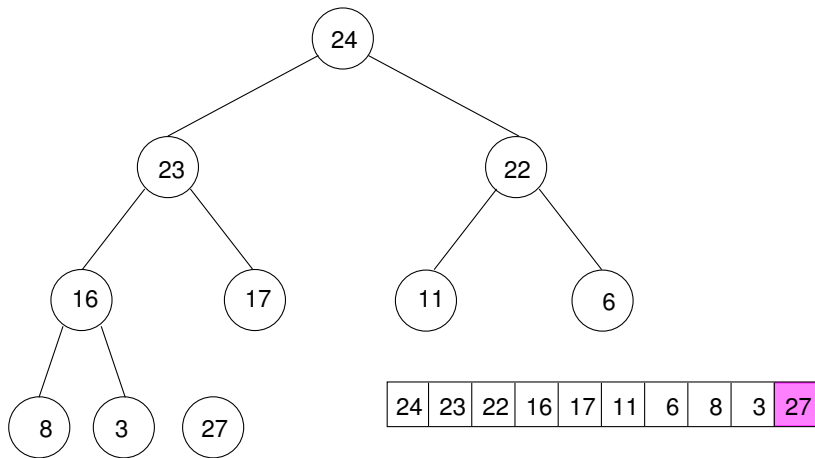


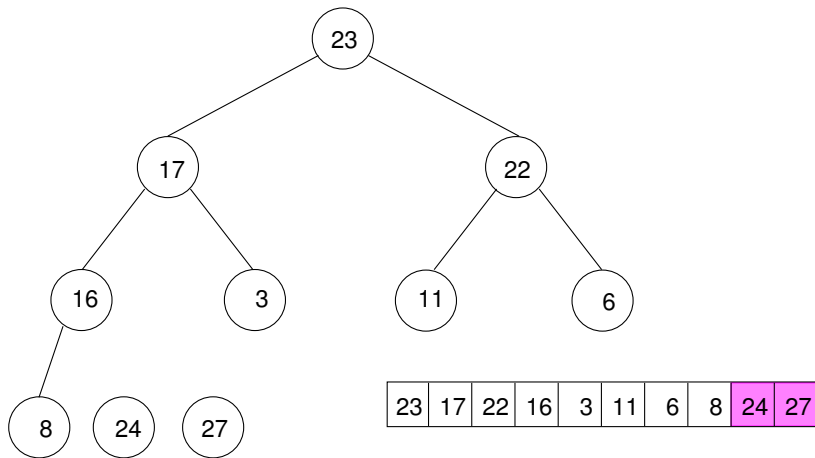
$$\begin{aligned}\sum_{i=0}^{h-1} 2^i (h-1-i) &= (h-1) \sum_{i=0}^{h-1} 2^i - \sum_{i=0}^{h-1} i 2^i \\ &= (h-1)(2^h - 1) - 2 - (h-2)2^h \\ &= h2^h - 2^h - h + 1 - 2 - h2^h + 2 \cdot 2^h \\ &= 2^h - h - 1 \\ &= 2^{\lg(n+1)} - \lg(n+1) - 1 \\ &= n + 1 - \lg(n+1) - 1 \\ &= n - \lg(n+1)\end{aligned}$$

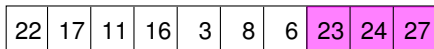
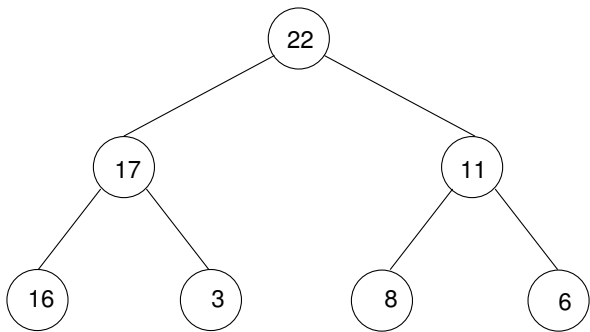


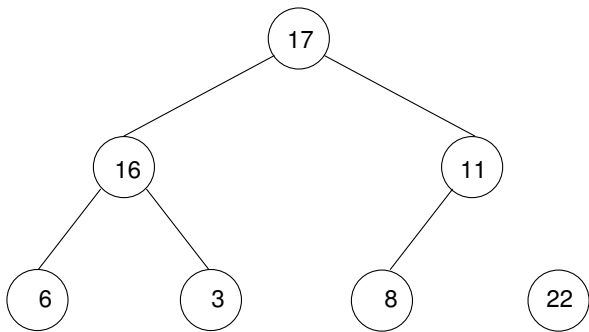




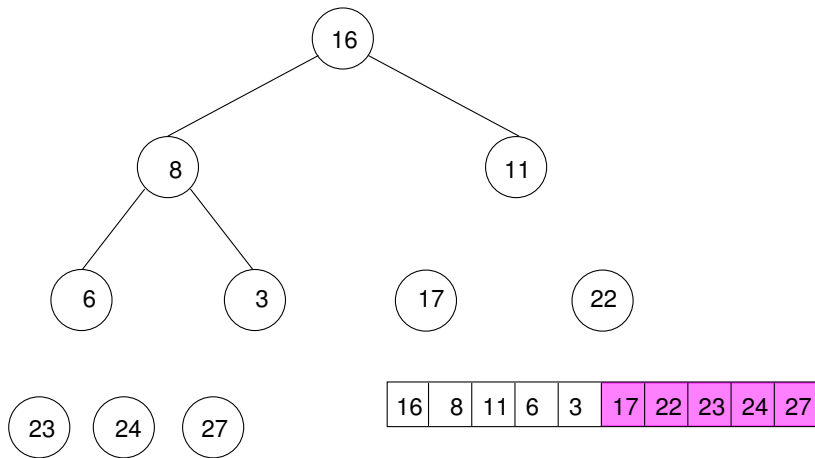


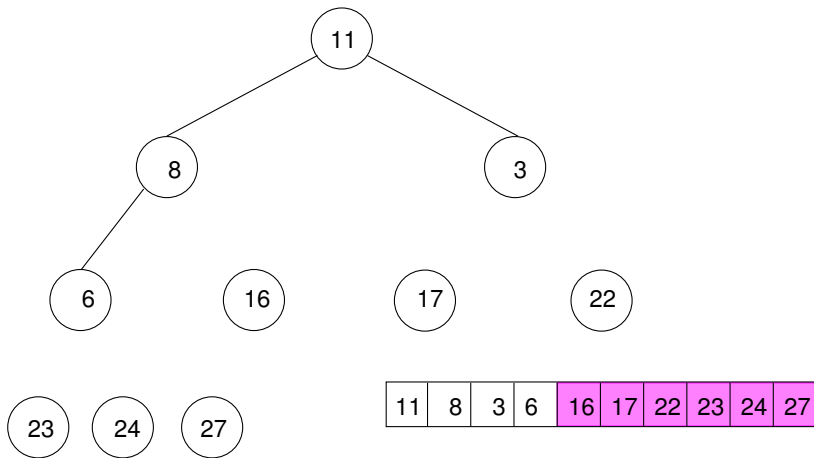


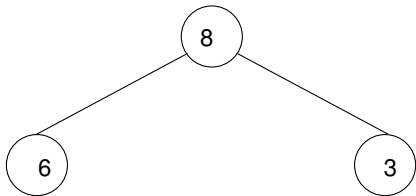




17	16	11	6	3	8	22	23	24	27
----	----	----	---	---	---	----	----	----	----







11

16

17

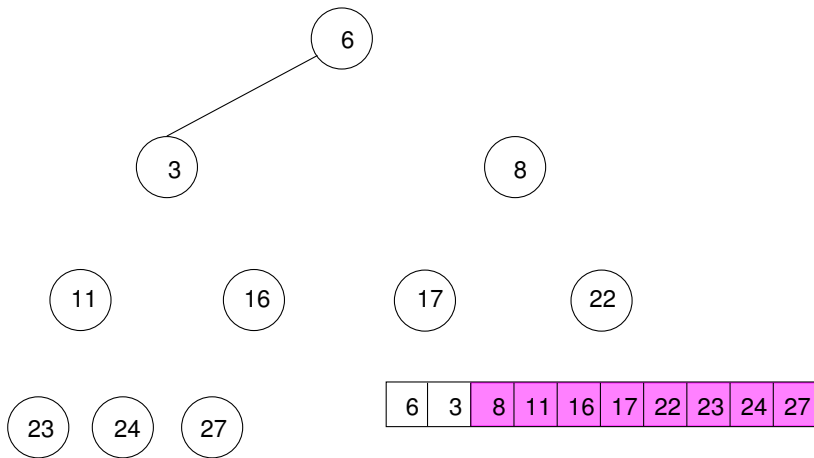
22

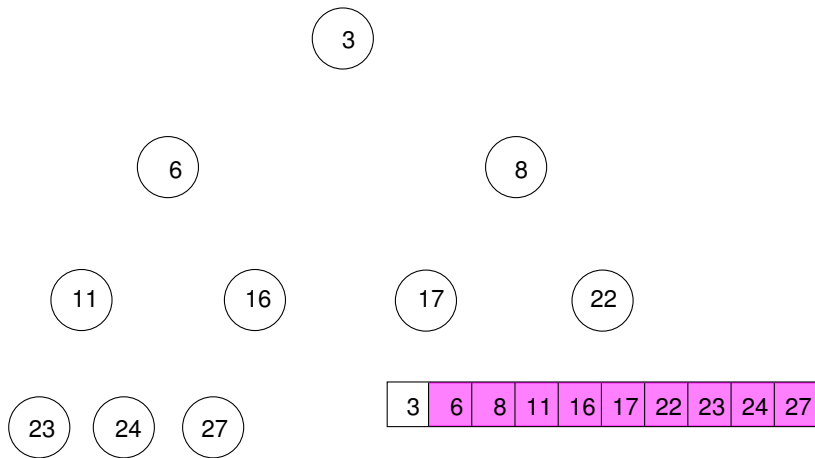
23

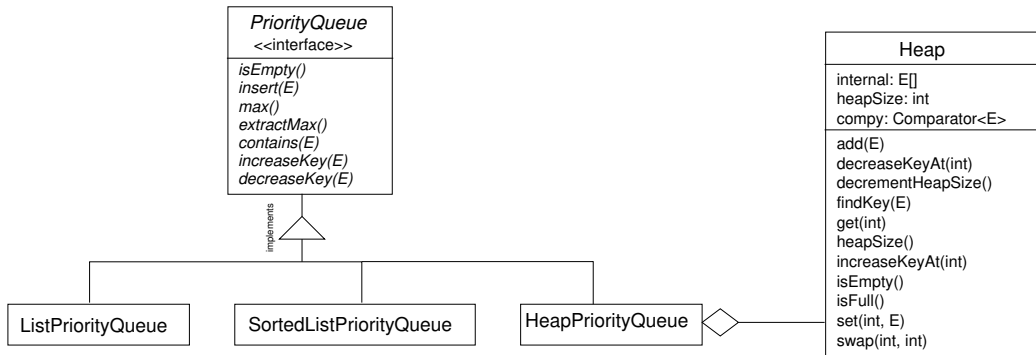
24

27

8	6	3	11	16	17	22	23	24	27
---	---	---	----	----	----	----	----	----	----







	ListPriorityQueue	SortedPriorityQueue	HeapPriorityQueue
Initialize empty	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$
Initialize populated	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n)$
insert	$\Theta(1)$	$\Theta(n)$	$\Theta(\lg n)$
max	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$
extractMax	$\Theta(n)$	$\Theta(1)$	$\Theta(\lg n)$
contains	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$
increaseKey	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$
decreaseKey	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$

Coming up: (all end-of-day)

*Do **linear sorting** project (suggested by today)*

*Do **heaps and priority queue** project (suggested by Mon, Feb 14)*

*Due **Today:***

Read Section 3.3 (heaps and priority queues)

*Due **Wed, Feb 9:***

Take heap/pq quiz

*Due **Thurs, Feb 10:***

Read Section 3.4

Do Exercises 3.(27 & 28).

Take N-sets quiz