

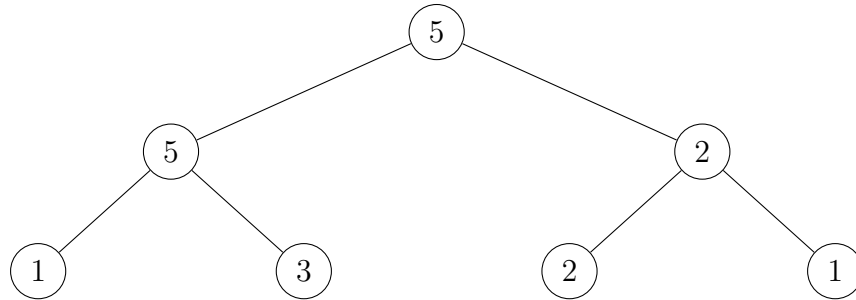
CS240 Tutorial 3

Graham Cooper

May 20th, 2015

Heaps

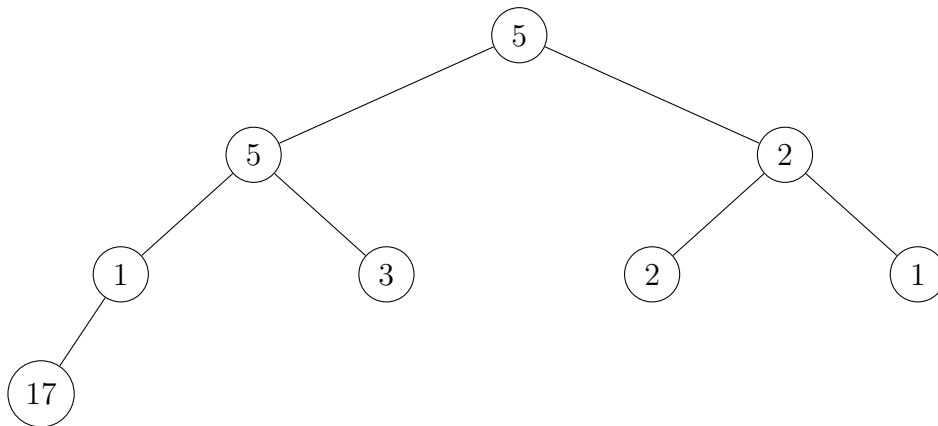
Consider the following heap (max-heap):

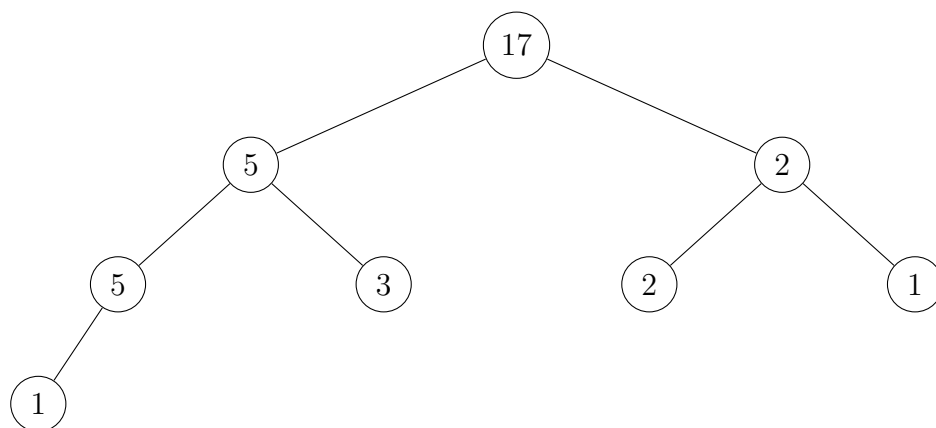
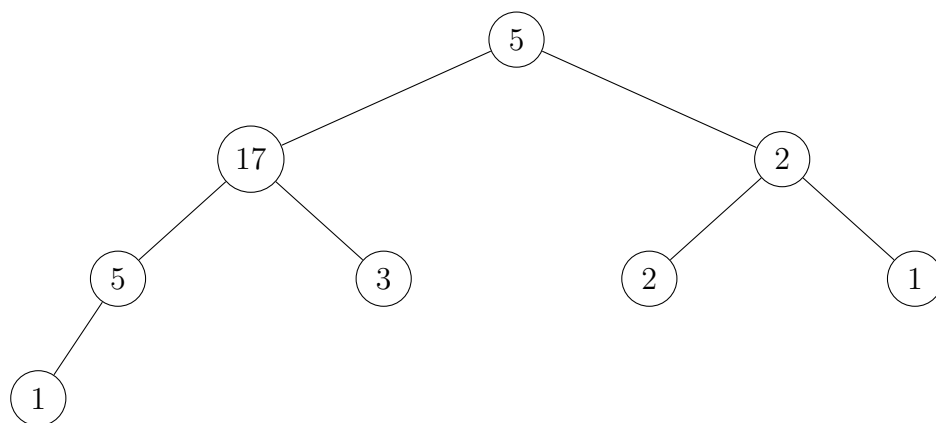
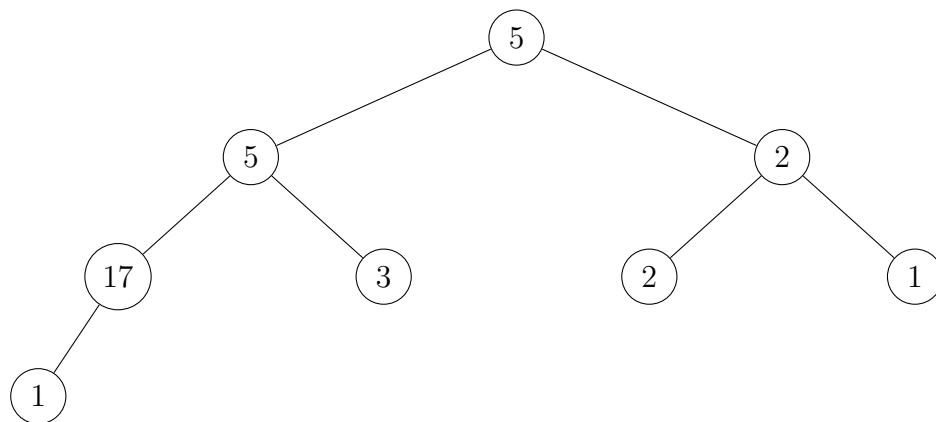


A: [5,5,2,1,3,2,1]

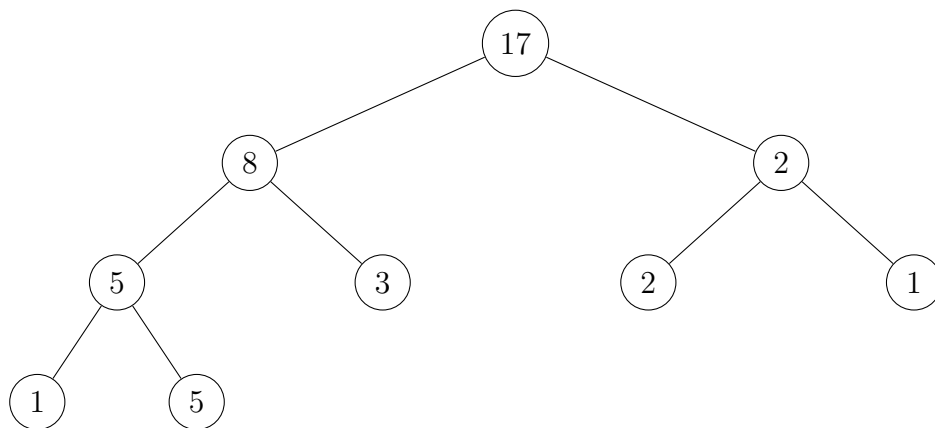
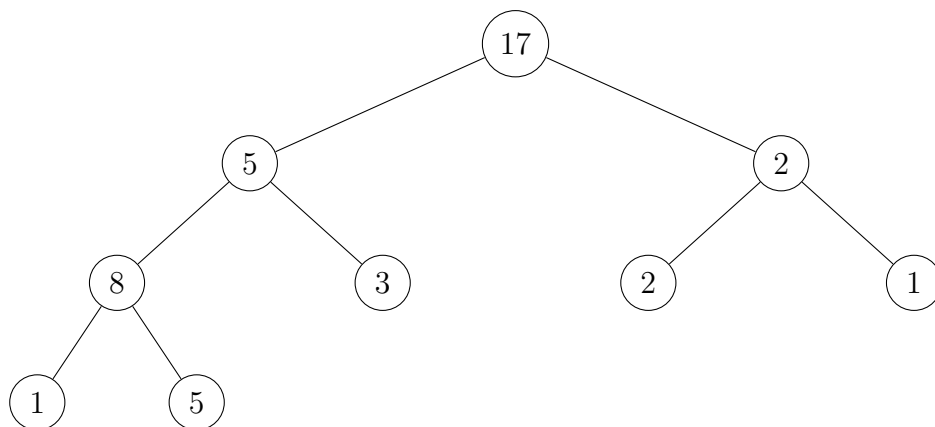
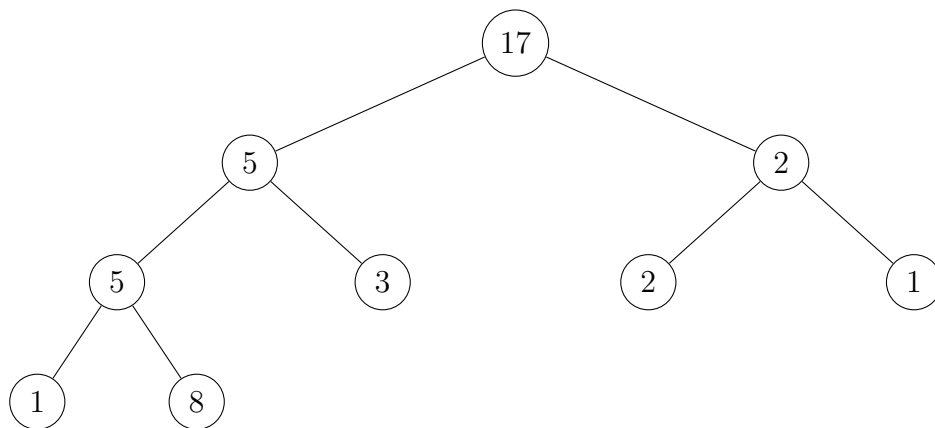
Insert

Suppose we insert 17 and 8 into the heap





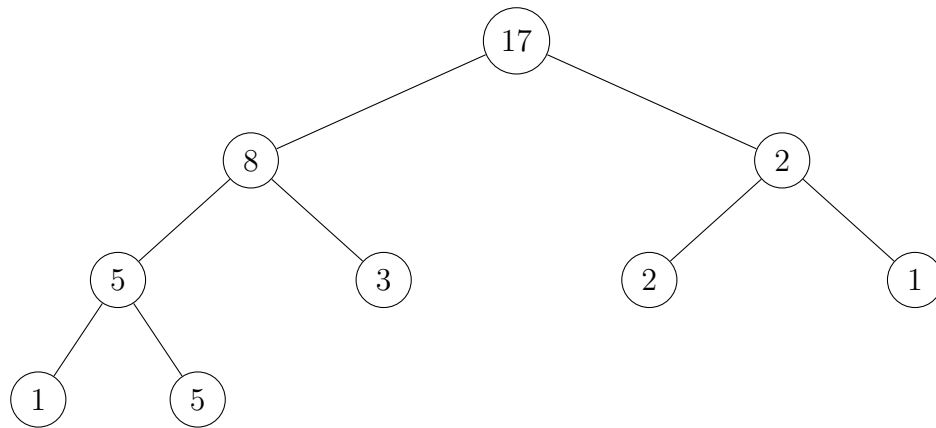
For 8:



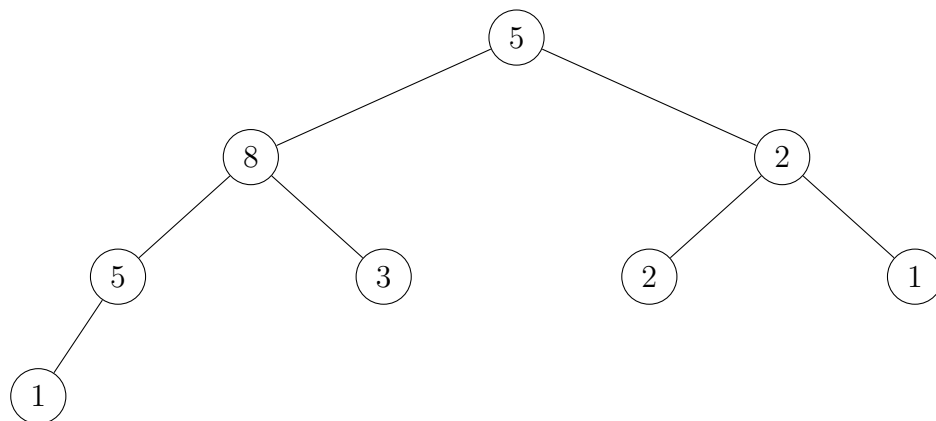
Do not swap with 17.

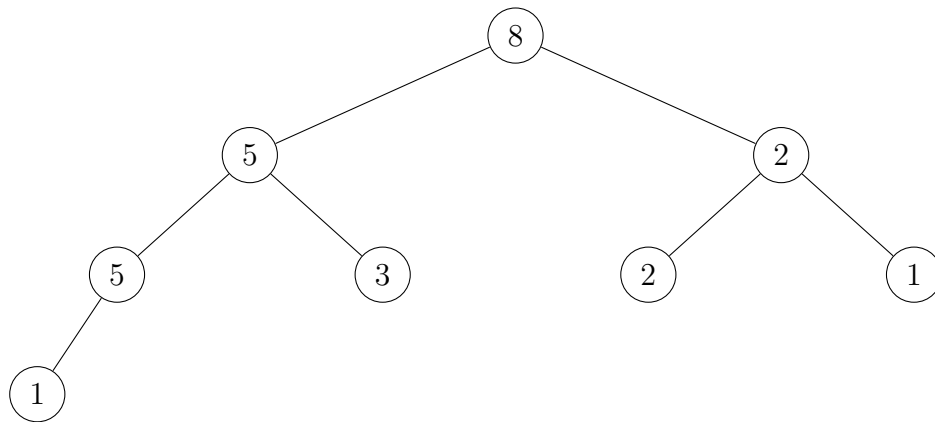
Delete-max

Suppose now we want to delete-max:



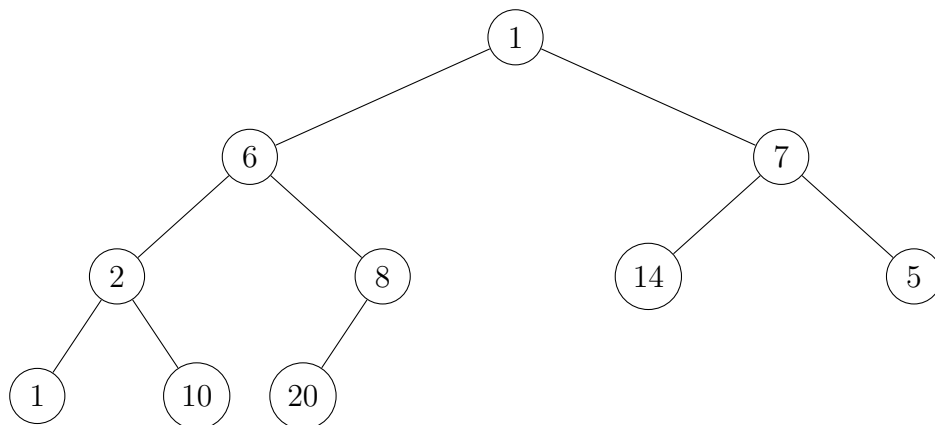
- 1) Swap root with the right most leaf, (on bottom level)
- 2) remove the largest element
- 3) Bubble down the new root

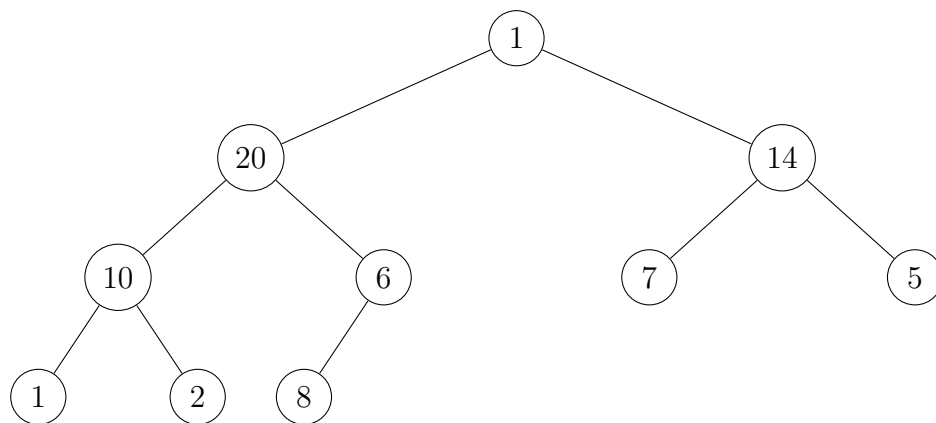
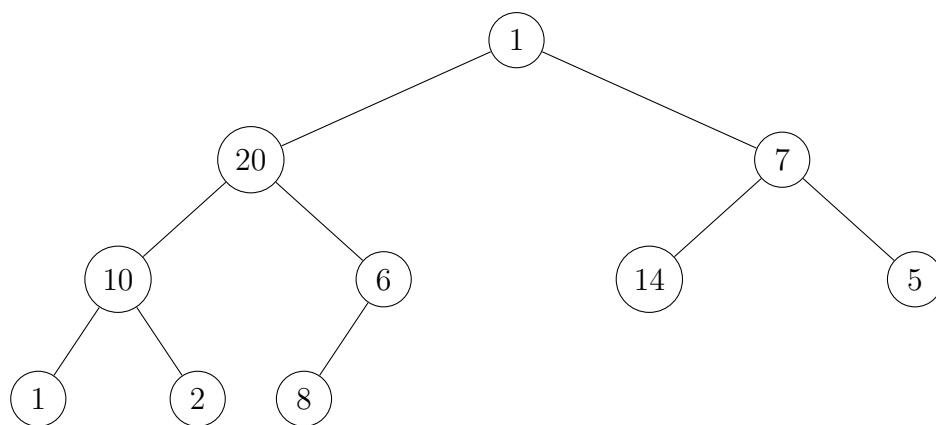
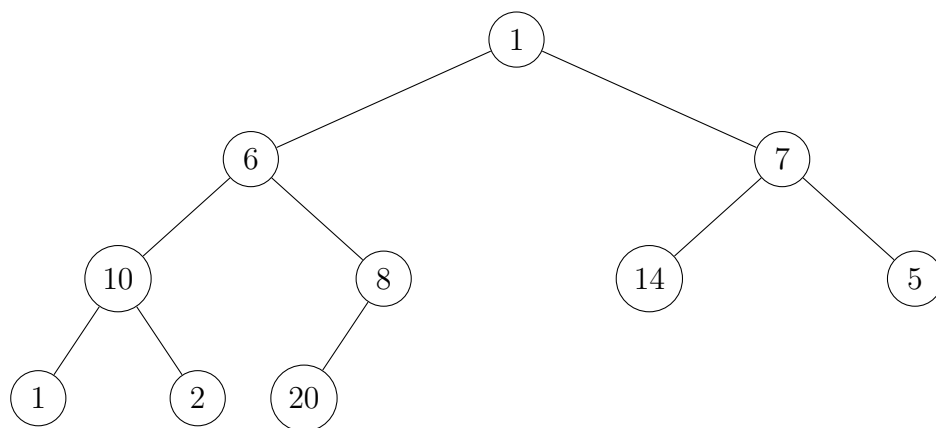


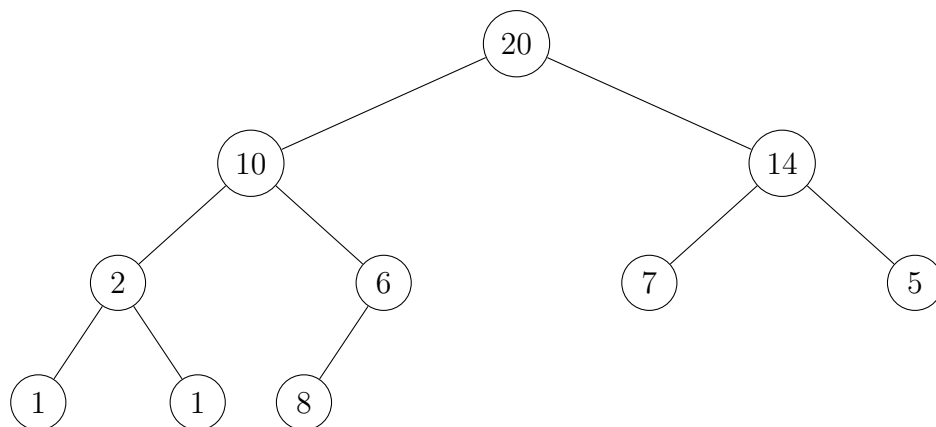
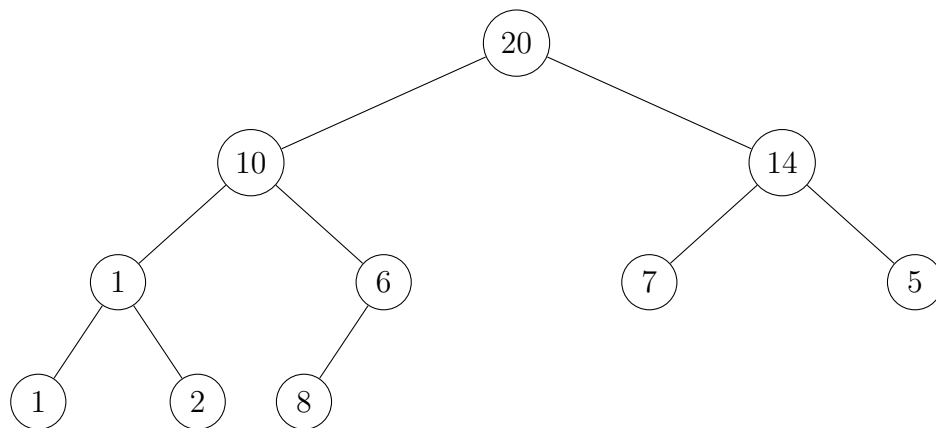


0.1 Heapify:

1. $n \leftarrow \text{size}(A) - 1$
2. for $i = \text{floor}(n/2)$ down to 1
3. $\text{bubble-down}(A, i)$







Overall we bubbled down 2, 6, 7, 6 and then 1

Heapsort

Use heapsort to sort the following array:

A: [2520, 1982, 34]800, 34000, 322,159,2845, 9]

1. $H \leftarrow \text{Heapify}(A)$
2. call $\text{delete-max}(H)$ n times and store the result in A.

Stack

Q: How can we simulate a stack using a priority queue (heap).

A stack should support push and pop

```
Stack {  
  - max-heap H  
  - priority P  
}  
  
Push(e) {  
  H.heapinsert(p,e);  
  p = p + 1;  
}  
  
Pop() {  
  H.deletemax();  
  p = p-1;  
}
```

Q: Given K sorted lists, where the combination of all k lists has n elements, combine them into 1 sorted list $O(n \log k)$ time. Hint: use priority queues.

First idea (use heapsort)

- use heapify $O(n)$
- using delete-max n times gives $O(n \log(n))$ time which is a bit too much

1	7	12	19		
2	9	14	18	21	23
3	8	16			
5	10	17	70	72	

Look the the first collumn to create a min-heap. Then delete-min and add on the next element onto the min-heap.

Note: Always have k items in our heap (height $\log k$) and repeat n times for a total running time of $O(n \log k)$ time