CS 61B Spring 2018

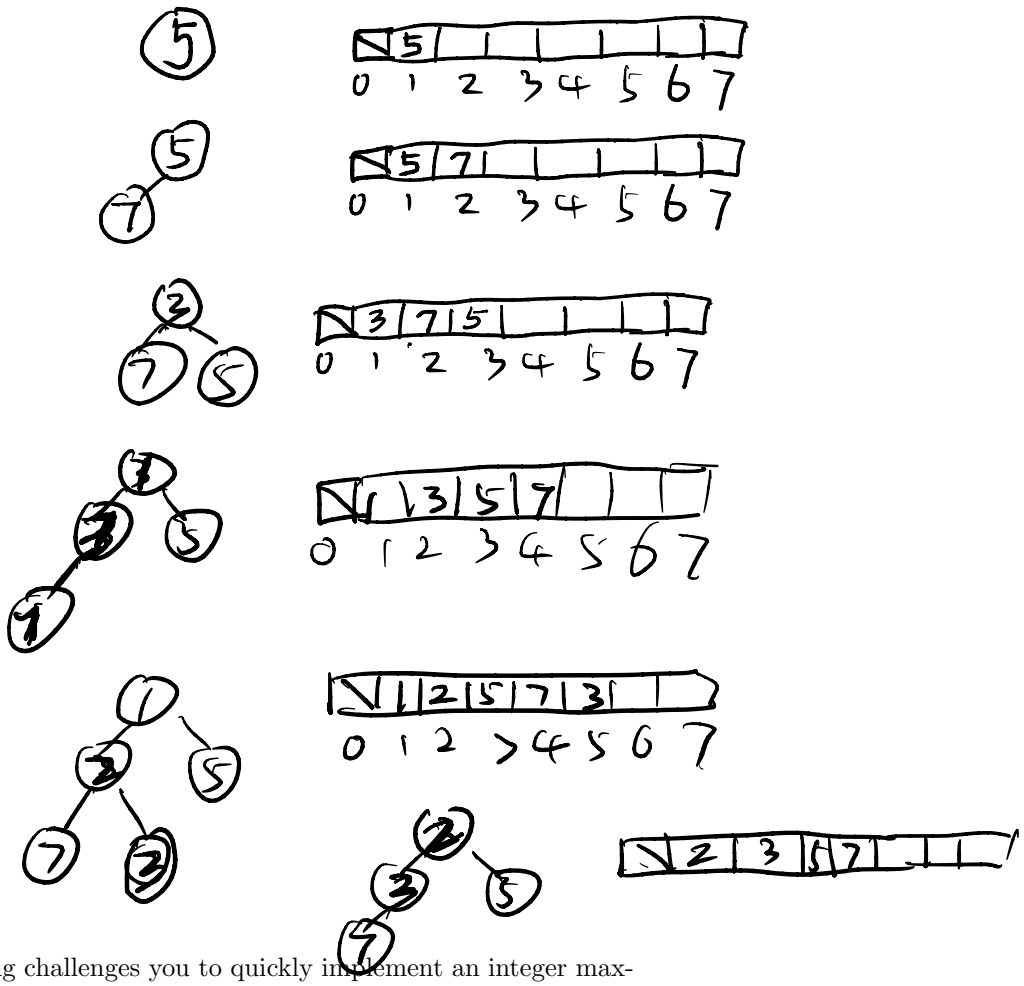
Heaps, Traversals & Trees

Discussion 10: March 20, 2018

1 Heaps of Fun

1.1 Assume that we have a binary min-heap (smallest value on top) data structure called `Heap` that stores integers, and has properly implemented `insert` and `removeMin` methods. Draw the heap and its corresponding array representation after each of the operations below:

```
1 Heap h = new Heap();
2 h.insert(5);
3 h.insert(7);
4 h.insert(3);
5 h.insert(1);
6 h.insert(2);
7 h.removeMin();
8 h.removeMin();
```

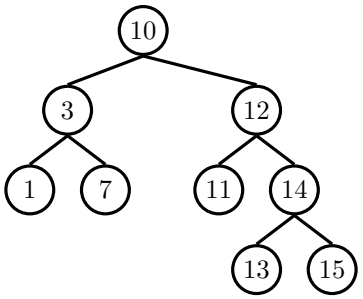


1.2 Your friend Sahil Finn-Garng challenges you to quickly implement an integer max-heap data structure. "Hah! I'll just use my min-heap implementation as a template to write MaxHeap.java," you think to yourself. Unfortunately, two Destroyer Penguins manage to delete your `MinHeap.java` file. You notice that you still have `MinHeap.class`. Can you still complete the challenge before time runs out?

Hint: You can still use methods from `MinHeap`.

```
public MaxHeap {
    Heap h;
    public MaxHeap() {
        h = new MinHeap();
    }
    void insert(int x) { h.insert(-x); }
    public int removeMax() {
        return -h.removeMin();
    }
}
```

2 Tree Traversals



2.1 Write the pre-order, in-order, post-order, and level-order traversals of the above binary search tree.

pre-order: 10-3-1-7-12-11-14-13-15

in-order: 1-3-7-10-11-12-13-14-15

post-order: 1-7-3-11-13-15-14-12-10

level-order: 10-3-12-1-7-11-14-13-15

3 Quadtrees

NE NW SW SE

3.1 Draw the quadtree built by inserting the following nodes with the given coordinates.

```
insert A (2, 3);
insert B (-1, 1);
insert C (3, 2);
insert D (0, 0);
insert E (4, 4);
insert F (-3, 2);
```