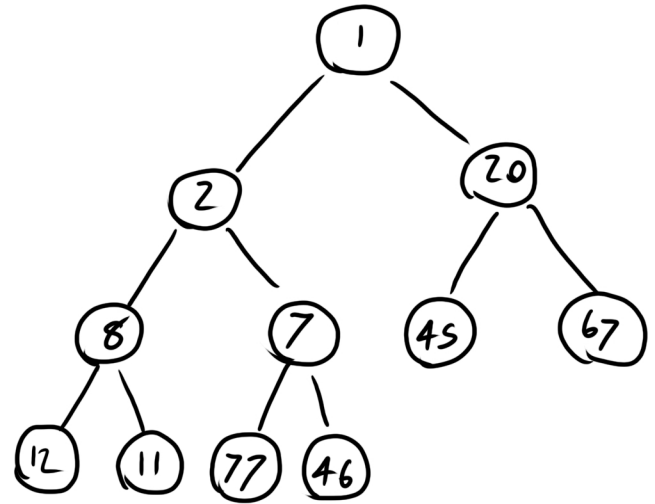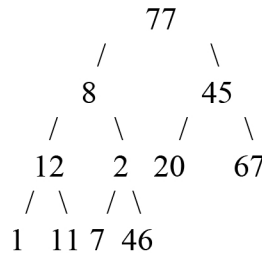6) Run the whole Heapify function on the following random values: (note that our target is to build minheap)
(this is the function that builds a heap in O(n) time)

Kauan Lima

```
              77
            /     \
          8        45
        /  \     /   \
      12    2  20     67
     / \   / \
    1  11 7  46
```


You don't have to write how to do heapsort. But, make sure you know the steps of heapsort

7) Explain each step shown in the code below, for the percolateDown function:

void percolateDown(struct heapStruct *h, int index) {

    int min; //declare min

    if ((2*index+1) <= h->size) { // if right child exists
        // find the smaller child
        min = minimum(h->heaparray[2*index], 2*index, h->heaparray[2*index+1], 2*index+1);
        // if current index is larger than min child
        if (h->heaparray[index] > h->heaparray[min]) {
            swap(h, index, min); // swap the positions
            percolateDown(h, min); // call percolateDown again with min as index
        }
    }
    else if (h->size == 2*index) { // if only left child
        if (h->heaparray[index] > h->heaparray[2*index]) // compare self with left
            swap(h, index, 2*index); // swap if self is larger
    }
}

(Note: Please reference heap.c *without looking at this function*, if necessary.)