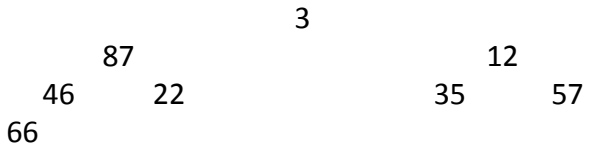
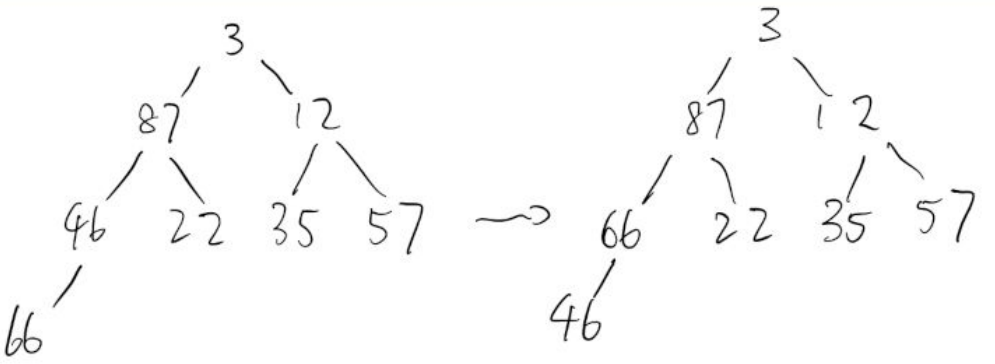
NAMES: _Hongyu Zeng, Greyson Davis, Dakota Krogmeier

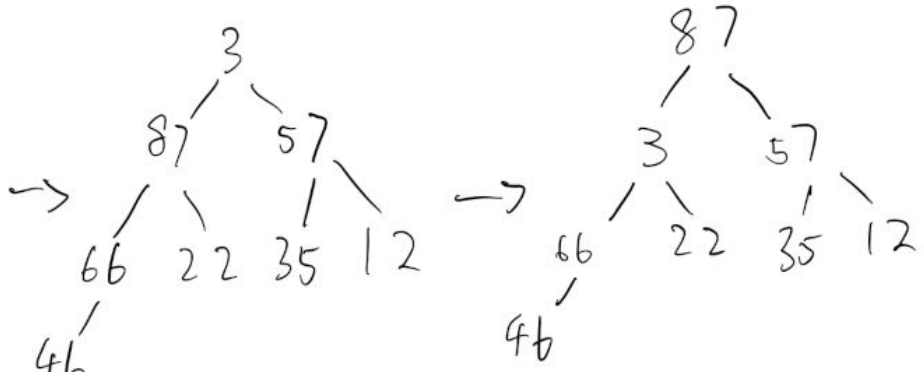
Worksheet 21: Work with the other students in your Zoom breakout room to complete this worksheet. You should only submit one paper for the group.

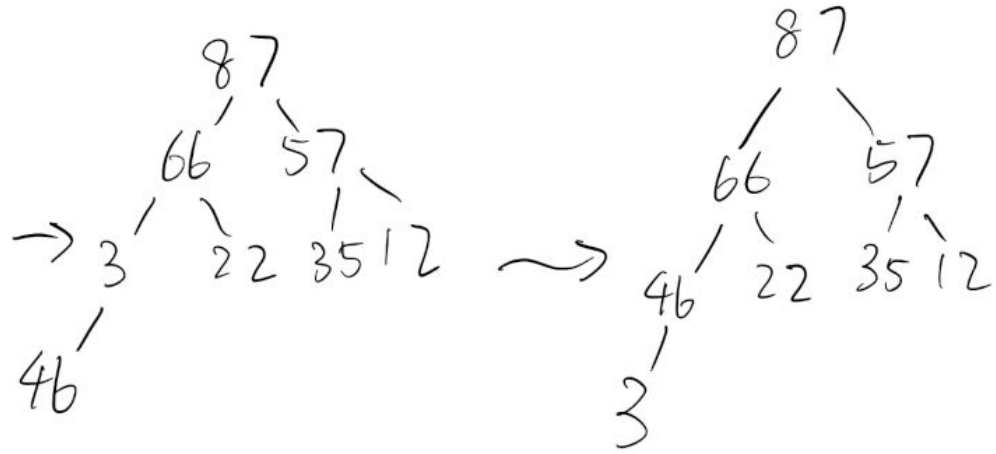
1. The following problem applies to implementing heap sort. Recall that the binary heap data structure is an array object that can be viewed as a nearly complete binary tree. Each node of the tree corresponds to an element of the array that stores the value of the node. The tree is completely filled on all levels except possibly the lowest, which is filled from the left as far as it can. The max–heap property is that for every node i (other than the root), the value of a node is not larger than the parent: A[parent(i)] ≥ A[i].
    a. Represent the array A = [3, 87, 12, 46, 22, 35, 57, 66] as a binary tree, where the tree is constructed so that A[0] is the root, A[1] and A[2] are the values of the nodes in level 1, and so on. This binary tree will NOT satisfy the max-heap property. (1 point)



    b. b. Illustrate the operation of building a max-heap on the binary tree representation of the array in (a). Show all steps as clearly and compactly as you can. (1 points)


3
87
12
46
22
35
57
66
3
87
12
66
22
35
57
46


3
87
57
66
22
35
12
46
87
3
57
66
22
35
12
46


87
66
57
3
22
35
12
46
87
66
57
46
22
35
12
3

2. Define the min-Heap property for every node (except the root) as follows: the value of each node is less than or equal to the value of its children, with the minimum-value element at the root. Develop an algorithm to modify a heap so that it has the min-heap property. (4 points)

KEY IDEA:

Start with the last leaf in the given index of i. Then compare the leaf to the parent in the i index and its sibling, if it has one. Take the smallest value of the parent and its siblings and move it to the parent position in the index of i and continue this throughout all the indexes of i of the tree to get the smallest value at the top.

Algorithm: Min-Heapify(A, i)

Input: an array, A, and an index i. Assume that the left and right binary trees have the min-heap property, but A[i] might be greater than its children

Output: An array, A, with A[i] in its "right place" in the tree relative to its children, so that the sub-tree with root vertex i has the min-heap property

PROCESS:

```
L ← left(i)
R ← right(i)
if A[L] < A[i] and L ≤ A.heapSize:
        smallest ← L
else:
        smallest ← i

if A[R] < A[smallest] AND R ≤ A.heapSize:
        smallest ← R

if smallest ≠ i:
        exchange A[i] AND A[smallest]
        Min-Heapify(A,smallest)
```