Homework #5

1. (10 points) **Exercise 6.1-5 (page 154):**

Ans:

Yes. For an array with sorted order, we have *A[i]* <= *A[j]* for *i*<*j*, and satisfies that *A*[PARENT(i)] <= *A[i]* since PARENT(*i*) < *i*.

2. (10 points) **Exercise 6.1-6 (page 154):**

Ans:

No. In the heap, the "6" is the parent of "5" and "7", which violates the max-heap property A[PARENT(i)] >= A[i].

3. (10 points) **Problem 6.2-4 (page 156) :**

Ans:

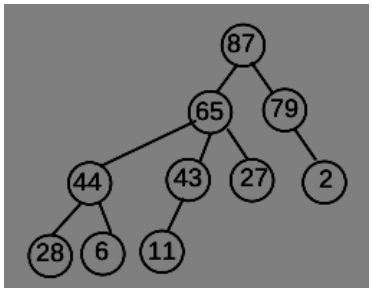
There is no effect, because the element *A[i]* for *i>A.heapsize/2* are leaves.

4. (10 points) **Problem 6.3-2 (page 159) :**

Ans:

Because the procedure BUILD-MAX-HEAP is working in a bottom-up manner by using MAX-HEAPIFY which compares the node and its children. The procedure assumes that the sub-trees under the current node are already max-heaps. Because the nodes A[i] with i>are leaves, so the loop starts from and finally ends at the top node of the entire heap (i=1).

5. (10 points) **Does the tree on the right represent a MAX HEAP?**



Ans:

No, the tree is not a MAX HEAP. The (binary) heap in this chapter is an array object that we can view as a nearly complete binary tree, but the node with value 65 has three children.

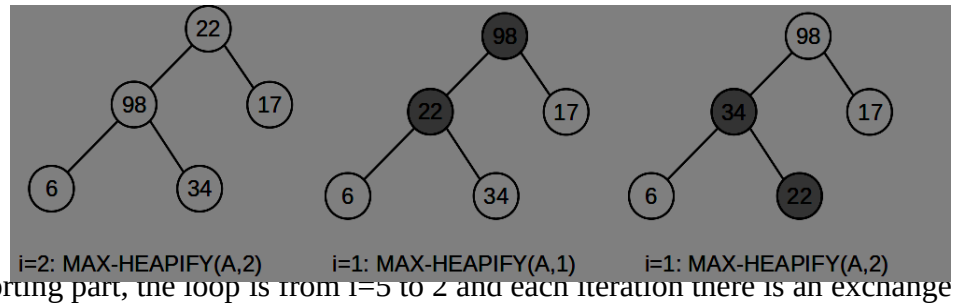
6. (20 points) **Given the array A = < 22, 98, 17, 6, 34 > :**

(a) Illustrate the operation of **HeapSort** using Figure 6.4 on p. 161 as a model. Sort the numbers into non-decreasing order.

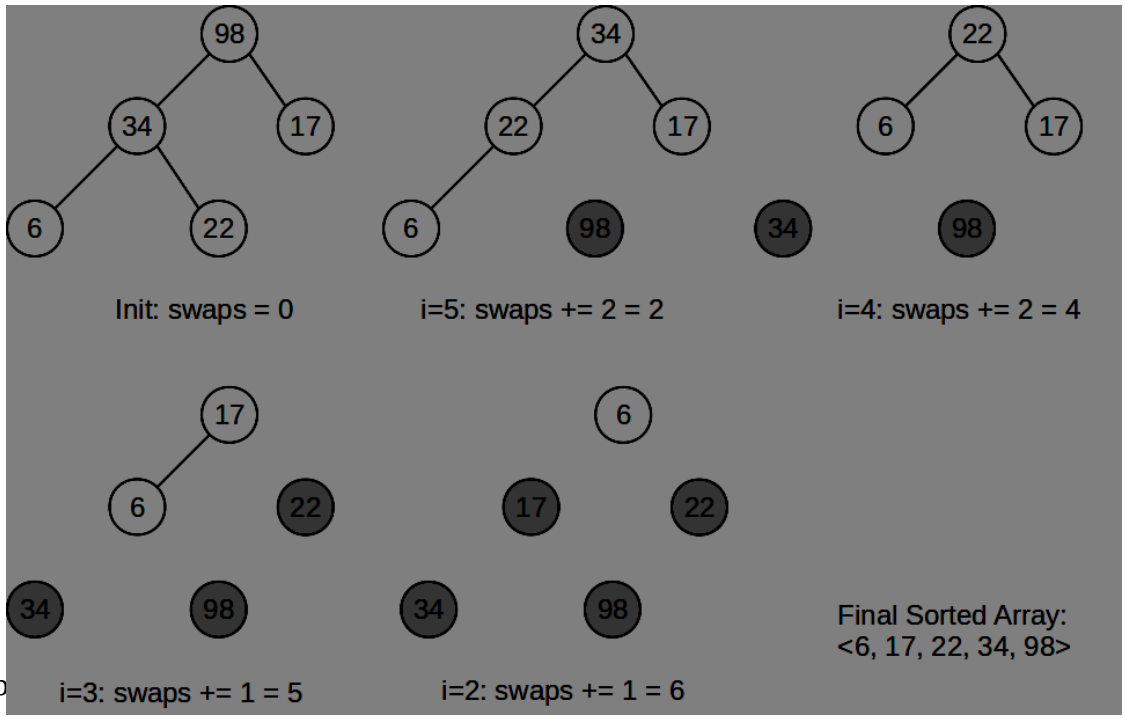
(b) A swap is an exchange of two elements in the array: A[i] ↔ A[j] . How many swaps are performed by **HeapSort** to sort the array **A**? Note: Include the swaps used to build the heap.

Ans:

(a) A.length = 5, so the BUILD-MAX-HEAP will use MAX-HEAPIFY from i=2 to 1. For iteration i=2, MAX-HEAPIFY does nothing, and there is two exchange at the iteration i=1 where 22 and 98 are swapped and then 22 and 34 are changed.



For sorting part, the loop is from i=5 to 2 and each iteration there is an exchange between A[1] and A[i] and maybe another exchange for MAX-HEAPIFY.



(b