1) Build Max Heap (A)
A.heapsize = 1
for i = 2 to A.length
Max Heap Insert (A, A[i])

7 4 3 5 1

7

5 3

4 1

7 5 3 4 1

7

4 3

5 1

1 2 3 6 5 4 7

7 7 5 6 1 3 2 4

5 6

1 3 2 4

1 2 3 6 5 4 7

7 7 6 4 2 5 1 3

6 4

2 5 1 3

a) They do not result in the same heap
The array [1 2 3 6 5 4 7] results in [7 5 6 1 3 2 4] done this way, to [7 6 4 2 5 1 3] done the way in the textbook

b) The new Build Max Heap requires $\Theta(n \log n)$ time for it. worst case because the for loop runs n times & Max Heap Insert takes log n times because it can be called once for each level of the tree if the inserted node is larger than all other nodes in the tree. This means that a log n function is called n times, or $\Theta(n \log n)$ if the initial array is already sorted in increasing order