Heapsort (7 points)

1. [2 points] What is the running time of heapsort on an array A of length n that is already sorted in increasing order? What about decreasing order?

Solution: If it's already sorted in increasing order, doing the build max heap-max-heap call on line 1 will take $\Theta(n \lg(n))$ time. There will be n iterations of the for loop, each taking $\Theta(\lg(n))$ time because the element that was at position i was the smallest and so will have $\lfloor \lg(n) \rfloor$ steps when doing max-heapify on line 5. So it will be $\Theta(n \lg(n))$ time.

If it's already sorted in decreasing order, then the call on line one will only take $\Theta(n)$ time, since it was already a heap to begin with, but it will still take $n \lg(n)$ peel off the elements from the heap and re-heapify.

2. We can build a heap by repeatedly calling MAX-HEAP-INSERT to insert the elements into the heap. Consider the following variation on the BUILD-MAX-HEAP procedure:

BUILD-MAX-HEAP$'$ (A)
1. $A.heap\text{-}size = 1$
2. for $i = 2$ to $A.length$
3. MAX-HEAP-INSERT$(A, A[i])$

(a) [2 points] Do the procedures BUILD-MAX-HEAP and BUILD-MAX-HEAP$'$ always create the same heap when run the same input array? Prove that they do, or provide a counterexample.

Solution: They do not. Consider the array $A = \langle 3, 2, 1, 4, 5 \rangle$. If we run BUILD-MAX-HEAP, we get $\langle 5, 4, 1, 3, 2 \rangle$. However, if we run BUILD-MAX-HEAP$'$, we will get $\langle 5, 4, 1, 2, 3 \rangle$ instead.

(b) [3 points] Show that in the worst case, BUILD-MAX-HEAP$'$ requires $\Theta(n \lg n)$ time to build an n-element heap.

Solution: An upper bound of $O(n \lg n)$ time follows immediately from there being $n - 1$ calls to MAX-HEAP-INSERT, each taking $O(\lg n)$ time. For a lower bound of $\Omega(n \lg n)$, consider the case in which the input array is given in strictly increasing order. Each call to MAX-HEAP-INSERT causes HEAP-INCREASE-KEY to go all the way up to the root. Since the depth of node i is $\lfloor \lg i \rfloor$, the total time is

$$\begin{aligned}
\sum_{i=1}^{n} \Theta(\lfloor gi \rfloor) &\geq \sum_{i=\Gamma n/2 \rceil}^{n} \Theta(\lfloor \lg \lceil n/2 \rceil \rfloor) \\
&\geq \sum_{i=\lceil n/2 \rceil}^{n} \Theta(\lfloor \lg(n/2) \rfloor) \\
&= \sum_{i=\lceil n/2 \rceil}^{n} \Theta(\lfloor \lg n - 1 \rfloor) \\
&\geq n/2 \cdot \Theta(\lg n) \\
&= \Omega(n \lg n)
\end{aligned}$$

In the worst case, therefore, BUILD-MAX-HEAP′ requires $\Theta(n \lg n)$ time to build an n-element heap.