9-1:

(a) We can sort the n elements using Merge Sort in time $\Theta(n \log n)$. We can then output the first i elements of the sorted list in time $\Theta(i)$ for a total run time of $\Theta(n \log n)$.

(b) The initial building of the max-priority queue takes $\Theta(n)$ time. The $j^{th}$ call to EXTRACT-MAX takes $\Theta(\log(n - j))$ time. The total run time for all i EXTRACTS is $\Theta(\log n + \log(n - 1) + \cdots \log(n - i + 1)) = O(i \log n)$ (Is this $\Theta$ as well?). The total run time in the worst case is thus $O(n + i \log n)$.

(c) Finding the $i^{th}$ largest number can be done in $\Theta(n)$ time. Partitioning can be done in $\Theta(n)$, and finally sorting the i largest can be done in $\Theta(i \log i)$ with Merge Sort, for a total run time of $\Theta(n + i \log i)$ in the worst case. This yields the best asymptotic bound among the algorithms for a,b, and c.