1. Solution: $O(N \log N)$

2. Solution: $O(N)$

3. Solution: $O(N \log N)$ (Because LinkedList get() is $O(N)$)

4. Solution: $O(N^2)$ (Creates a zig-zaging binary tree of height $N$)

5. Solution:
   A heap allows us to perform `enqueue` operations in $O(\log N)$, faster than with a linked list where `enqueue` would be $O(N)$.
   
   Using a linked list allows us to implement a fair priority queue, where, in the event of a tie, the most recently added elements are behind the least-recently added elements. (This is why we couldn’t use the Java PriorityQueue on the Huffman Assignment)

6. Solution: $O(N^4)$

7. Solution: $O(N^2)$