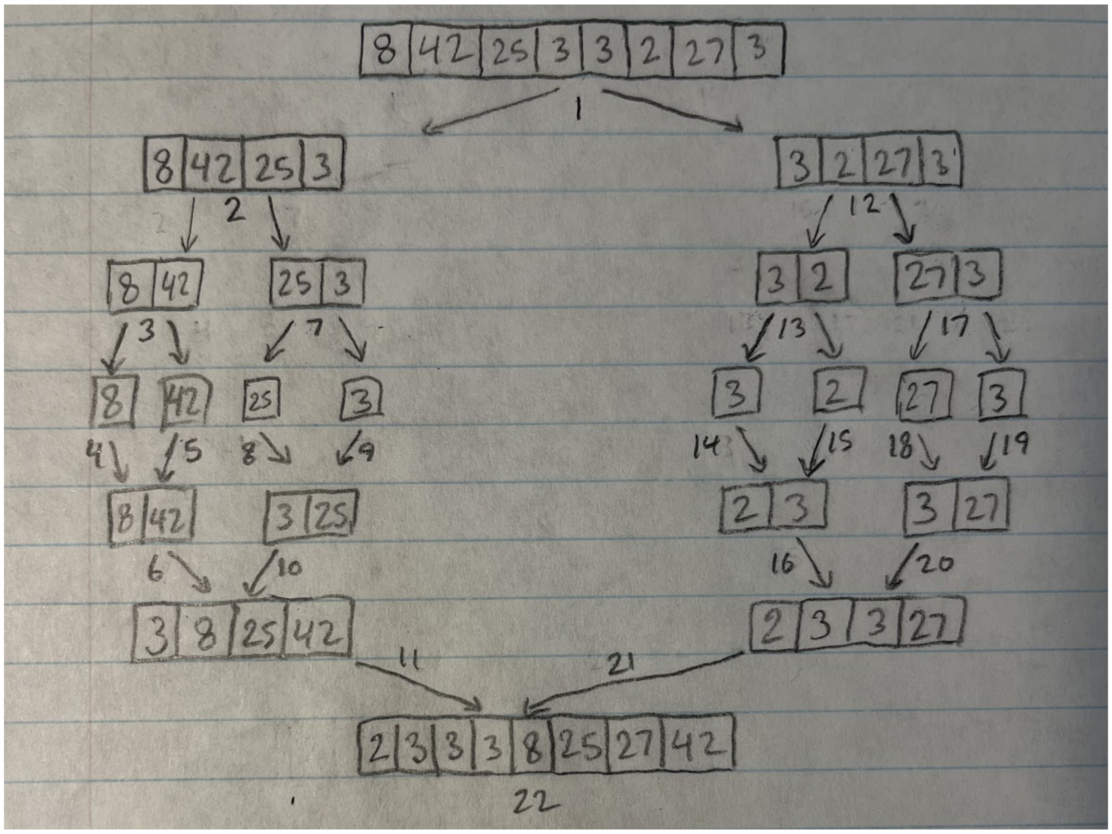
1.2

Within the merge() function there are 3 while loops. The first while loop compares the elements from the left and right subarrays, so it is run n times for each element. The second and third while loops make sure the arrays keep merging even if one of the subarrays is empty, so they are each run for n/2 times. Therefore the worst-case complexity of the merge function is O(n). Due to dividing each array into halves, the time complexity of the mergesort() function is O(log n), so the overall algorithm has a worst-case complexity of O(n log n).

1.3



Keeps dividing the leftmost array into arrays until each array has one value, then merges them. This process is repeated from the leftmost array to the rightmost array. Each array division counts as one operation and each merge counts as 2 operations.

1.4

Yes, because O((8)log(8)) = O(24) which is approximately 22.