2. The worst-case complexity of the overall mergesort algorithm using this merge() function is O(n log n).
This is because the divide step takes O(log n) time as we repeatedly halve the array until reaching arrays of size
1, and the merge step takes O(n) time as we merge the divided arrays.

3.
Initial list: [8, 42, 25, 3, 3, 2, 27, 3]
[8, 42, 25, 3] and [3, 2, 27, 3]
[8, 42] [25, 3]
[8] [42]
[8, 42]

[25, 3]
[25] [3]
[3, 25]
[8, 42] and [3, 25] to get [3, 8, 25, 42]

[3, 2, 27, 3]
[3, 2] [27, 3]
[3] [2]
[2, 3]
[27] [3]
[3, 27]
[2, 3] and [3, 27] to get [2, 3, 3, 27]

[3, 8, 25, 42] and [2, 3, 3, 27] to get [2, 3, 3, 3, 8, 25, 27, 42]

4.The number of steps taken is consistent with our complexity analysis. Each division step halved the size of
the array until we reached arrays of size 1, which took O(log n) steps, and each merge step combined the divided
arrays, taking O(n) steps. Thus, the overall time complexity matches our analysis of O(n log n).