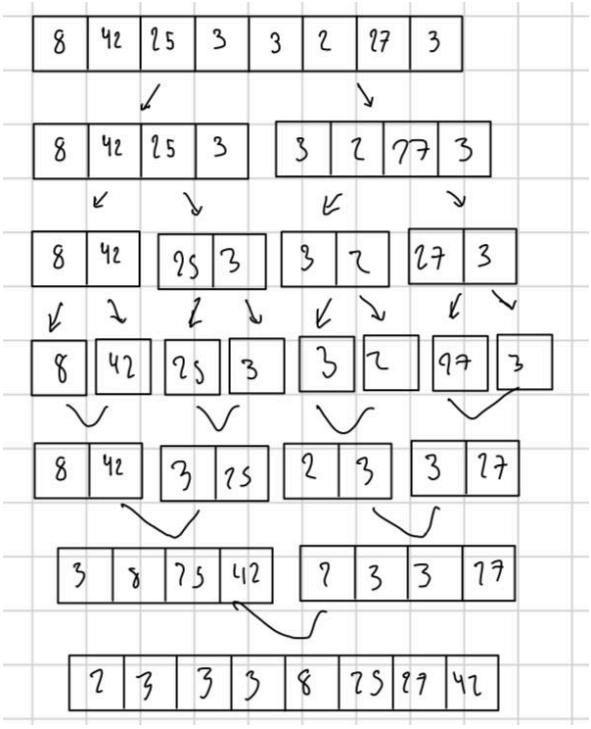
ENSF338 Exercise 1

Question 2: Argue that the overall algorithm has a worst-case complexity of O(nlogn)

Merge_sort follows a divide and conquer approach. It divides the array into two halves recursively until each subarray contains only one element. This step has a time complexity of O(log n) because the array is divided in half each time. After dividing the array into individual elements, the merge operation takes place. In the worst-case scenario, every element in the array needs to be compared and merged. This merging step has a time complexity of O(n) because each element in the array needs to be processed once.

Since the divide step takes O(log n) time and the merge step takes O(n) time, the overall worst-case time complexity of merge sort is the product of these two steps, which is O(n log n).

Question 3:



Question 4:

Yes, the number of steps is consistent with the complexity analysis of O(n log n). As discussed before, merge_sort is a O(log n) function and the merge function is a O(n) function. So for each of the log(n) branches, we do n merges for all elements. Which has a complexity of O(n log n) which is consistent with the complexity analysis done before.