Java Report

For sorting my non-zero elements, I have decided to use a merge sort. Merge sort is a sorting algorithm that takes an array as the input and recursively divides it into halves until the size of the arrays becomes 1 – Each step this takes O(1) times. When the size of the arrays is 1, they start merging back together in the correct order (ordered subarrays) until the whole original array has been merged back into one – Each step takes O(n) for n elements. Time complexity of merge sort is O(n Log (n)) in both worst-case and best-case scenarios. This is because the entire input must be iterated through completely and this happens O(log(n)) times, then n elements are iterated log(n) times, which gives us O(n log(n)).

For the Add method, I used a while loop to count the number of elements that have been added to the new ArrayList. This means it is iterated through every element from each matrix once, except from where they intersect. For this, it will add the values of the elements from the same position in both matrices at the same time, removing one from the overall number of elements required to add. The worst case for this method is when the two matrices don’t intersect at all and must iterate through all the elements from both matrices – this gives us worst case O(N1 + N2) where N1 is number of elements in one matrix and N2 is the number of elements in the others.