**#------------------------------------------------------------**

**# Name : Susmitha Konintees**

**# Date : 11-03-2023**

**# Subject : INTRO TO PROG USING SCRIPTING CS-504-D**

**# Title : Algorithm Assignment**

**#-------------------------------------------------------------------**

**Merge sort Algorithm- Time Complexity Analysis.**

As per my understanding the Merge Sort Algorithm uses the concept of divide and conquer approach to sort the elements as it divides the input into two halves and sort the elements till the size of the array to 1, after that that merge the smaller array into large array in the form of sorted way till the original array is sorted the time complexity for the process of divide is O(log n) , where n is the number of elements of array as it is dividing into repeatedly two equal halves. The time complexity of a merging step is O(n), since in the worst case we need to compare all the elements and sort them. Combining both steps, we get the time complexity of the merger sort algorithm is O (n log n) making this as best algorithm with more efficient for the larger datasets. For different datasets the time complexity remains same as O (n log n) regardless of the initial input data order, which is not case for the quick sort and all other algorithms when it comes to smaller data sets since “n “may be small so the merge sort will be slower than other sorting algorithms because of the mainly due to overheard recursion and extra space for the merging. As the merge sort will be dividing into two halves due to the recursive nature making it to divide problems into subproblems, sort and merge is more efficient because it merges two sorted arrays in a linear time as key observation. The divide step makes the problem into smaller subproblems makes them as independently making the algorithm more efficiently. Overall the merge sort algorithm is a stable and efficient sorting algorithm with time complexity O (n log n) as its recursive nature and divide -and-conquer nature makes the algorithm more efficient for the large datasets making as the best algorithm in various applications that need the sorting. However for the smaller datasets its overhead nature makes this algorithm less efficient than other sorting algorithms.