**Introduction**

This program is an implementation of Merge Sort. The sequence to sort is written in *input.txt*(with the sequence’s size as the first line), while the output sorted sequence would be saved in*output.txt*.

**Concept**

1. Read Input Sequence
   1. Read the first line of *input.txt*to get the sequence size *n*.
   2. Read *n*consecutive numbers from the file and save them into vector *vec*.
2. Do Merge Sort

A recursive function is used to divide and conquer the sorting issue.

* 1. *MergeSort* function (Divide)
     1. When there are more than one element in the sequence to sort, the sequence is divided into two halves.
     2. Recursively process each of the half sequence with *MergeSort*.
     3. After the halves are both sorted, call *Merge* to combine them.
  2. *Merge* function (Conquer)
     1. Copy the first and second half of the sequence to sort to two vectors, *v1*and *v2*, respectively. (In-place)
     2. Add “∞” to the end of both vectors so as to get rid of range checking in the following process.
     3. In the for-loop, we compare *v1*and*v2* element by element, and then choose the smaller one to overwrite the original sequence’s corresponding element.  
        Example: Assume the sequence to sort is *vec[5]* to *vec[8]*.  
        If *v1[0]* is smaller *v2[0]*, then *vec[5]* = *v1[0]*. Next, *v1[1]*and *v2[0]* is compared, and the smaller one replaces *vec[6]*, and so on.

After all comparisons are done, we get a sorted *vec[5]* to *vec[8]*

subsequence, which will further continue merging with another subsequence.

By recursively calling these two functions, we will occasionally obtain a complete sorted sequence *vec*.

1. Write Output Sequence
   1. Read every element in *vec* and write it into *output.txt*.