Group 4

Biruk, Dante, and Elena

Sorting Information

Original score:

- Dataset One - 6252, 6361 and 6487 milliseconds
- Dataset Two - 11049, 11063, and 11939 milliseconds

Our Score

- Dataset One - 4035, 4082, and 4026 milliseconds
- Dataset Two - 4023, 4039, and 4060 milliseconds

Sorting Information

DESCRIPTION OF ALGORITHM

- Our algorithm uses mergesort with a modified comparator.
- We modified three things in the comparator: The numberOfOnes() function to use Integer.numBinaryOnes() function (which is O(1) instead of the original O(n)), a return statement that would run if the number of ones are equal, and the numberOfRepeatedSubstrings() function.
- We tried an optimized quicksort and an alternative numberOfRepeatedSubstrings() function that utilized suffix trees but they were either less efficient or incorrect.
- Sources: numberOfRepeatedSubstring() function is written by Ashutosh Singh (https://iq.opengenus.org/longest-repeating-non-overlapping-substring/).

Number of Repeated Substrings

DESCRIPTION OF FUNCTION

- Keep a record of the LRS length initialized to 0 and a two dimensional array storing the length of the repeated substring at each i and j.
- Iterate through the string for each letter at index i, and for each of those letters, read ahead the letters at index j.
- If the character at index i is equal to the character at index j and they are non overlapping, then increment the two dimensional array at that respective i and j to be the previous iterations value of i and j incremented by 1.
- If this new value is greater than the currently greatest repeated string length that we are keeping track of, then set it to this new value.
- Return the length of the greatest repeated string length.

Correctness Issues and worst case efficiency

No correctness issues were reported.

Our algorithm has a best case efficiency of **nlog(n)** and a worst case efficiency of $m^2 n \log(n)$ where m is the number of binary digits and n is the number of elements to sort.

The average case efficiency of our algorithm is $\frac{(m^2+1)}{2} n \log(n)$