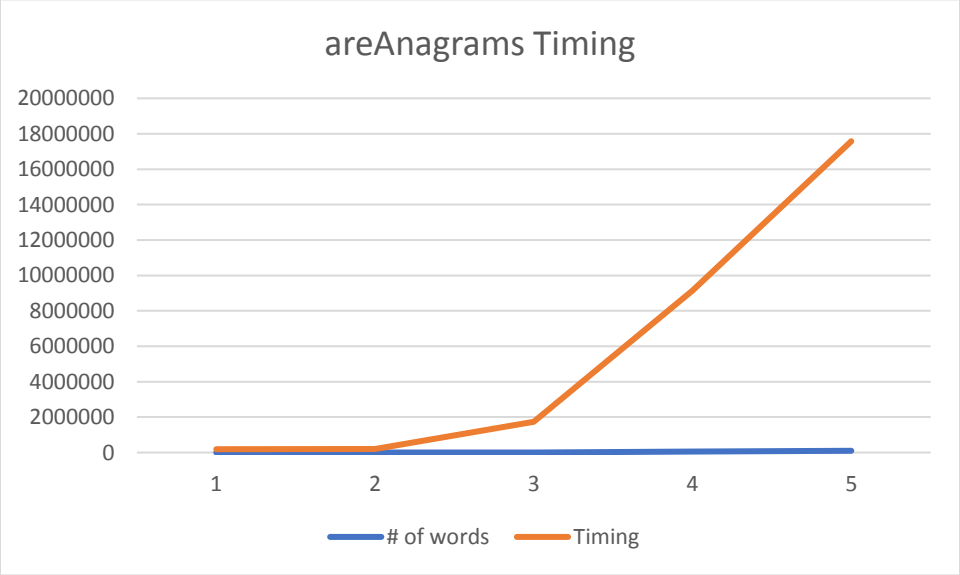
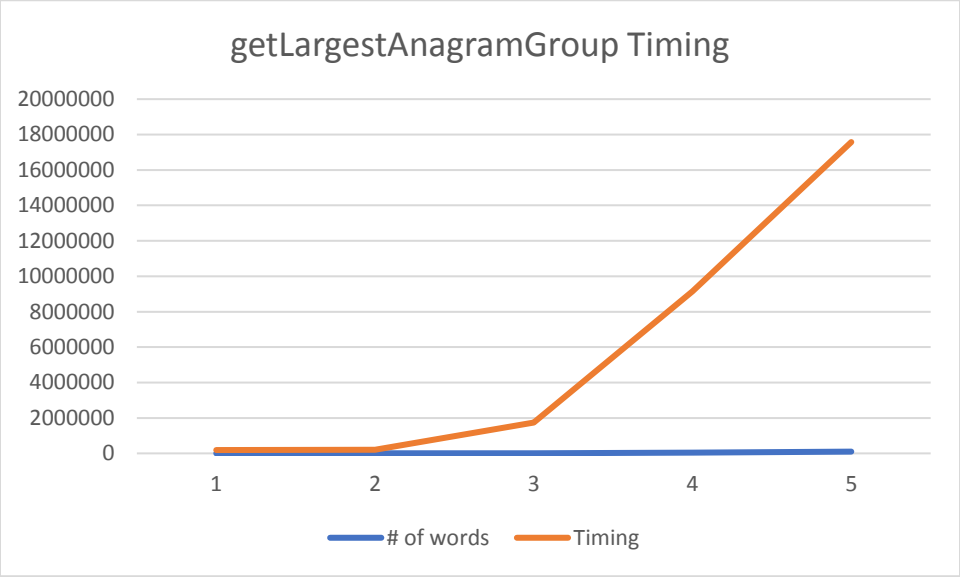
Brian Burton
CS 2420
02/11/17

Analysis Document

The Big-O behavior for the areAnagrams method would have to be N. The algorithm for this method is to sort the words, then compare them to see if they are the same. In order to do that you have to look through every letter of each word or N times. When we analyzed the run time performance of the areAnagrams method, we got the graph below. All of the timing given is in Nano seconds.



The Big-O behavior for the getLargestAnagramGroup method would also be N squared. The reason for this is because not only do we have to sort every word, but we also have to go back over the list adding up each anagram and putting them into sets. When I ran the same test for the getLargestAnagramGroup method I found the results graphed below. All timing given is in Nano seconds.



I was unable to implement the insertionSort timing method but I was able to time the getLargestAnagramGroup method using Java's ArrayList. The same rules apply to the last timing test. The Big-O behavior for the getLargestAnagramGroup method is N squared. The reason for this is because not only do we have to sort every word, but we also have to go back over the list adding up each anagram and putting them into sets. The results are graphed below. All timing is given in Nano seconds.