CPSC 319 Assignment 2

1. The sorting method I used to sort the array storing all the words from the input file is quicksort with the pivot selected from a random number generator. I used quicksort because it has an O(nlogn \* L) (L from the comapreTo function needed to traverse across the length of the string) and when compared with merge sort requires less memory to run as well as having a lower constant (from lectures). I used a random number generator to select my pivot because that guards from getting an O(n^2) running time from already sorted or reserve sorted arrays (or almost sorted arrays).

The sorting method I used to sort the letters of the words to determine if they are anagrams is an insertion sort method. I choose this because it is a simple sort method O(n^2) as word length will usually be small. It is easier to code than an O(nlogn) sort as well as needing less overhead and more efficient than bubble sort a similar O(n^2) method as well as having around the same constant as selection sort.

2. To estimate the O time, I calculated the O for each of the steps of my program.

For reading the input we have a 4n + 3 because of the while loop which calls four steps (two comparisons, read line and add word to list) this is O(n).

For the sorting method for anagram which is called once we have O(L^2) from the known properties of insertion sort.

For the quicksort algorithm we have O(nlogn \* L) from the known properties of the quicksort as well as the compare function.

Then, when I create the linked list from the sorted word list, we have O(n^2) as it is a nested loop configuration where the outer depends on n and the inner depends on the outer variable.

Therefore, the O(n,L) = O(n^2) + O(L^2) + O(nlogn \* L) as these are the terms that will asymptotically dominant the expression. We need O(nlogn \* L) as because it depends on both L and n it is unknown if it will be less than O(n^2) or O(L^2).

3. When our list contains only two words everything with regards to n becomes constant and the only thing that O depends on is L the word length. From the expression we derived O(n,L) = O(n^2) + O(L^2) + + O(nlogn \* L) we make n a constant. Therefore, it will only depend on L so our expression becomes O(L^2) + O(L). Then, asymptotically O(L^2) grows faster we have O(L^2).