Report: A pdf file describing your solution and program output should be produced. This file should contain:

1. **A high‐level description of the overall solution strategy.**

My program reads in the file line by line and removes any punctuation, with the removePunct() function. I then use a getline() to separate by a space and insert the lower-case word into a new Word object. I then add the word object to an array if it unique and increment the lastIndex counter otherwise always increase the totalWord counter. The word object is made up of a count and word. I then close the file and sort the array using quicksort first by the count and then portioned by the count number alphabetically. Then once this is complete, we reverse the array to get the descending order. Then prints the required result.

1. **A list of all of the data structures used, where they are used and the reasons for their choice.**

Class Word – Used to store the count and word together in an object. This is so we can place it in an array as a packaged pair

Class UniqueWords – Contains an array of unique words, the current end index of the array and a counter for the total amount of word that have been tried to be put in the array. The counters are used to iterate through the array.

Word words[] – an array of unique word objects which contain a count and a unique word. This is so we can access either count or the word when iterating through the array.

1. **A list of any standard algorithms used, where they are used and why they are used.**

**QuickSort was used twice in the sort() function once on the count to sort the array and then again when portioned to each number of the count to order in alphabetical order.**

**The array then used swap() in the sort function recursively to switch element to desc order.**

1. **The output produced by your program on the provided “sample-long.txt” file.**

**Enter Filename:**

**sample-long.txt**

**Total Words Attempted: 3441, Total Unique Words: 954**

**First Fifteen---------**

**the:131**

**of:94**

**to:91**

**and:91**

**you:66**

**a:64**

**he:59**

**i:49**

**was:48**

**mr:46**

**his:45**

**in:43**

**with:42**

**that:42**

**not:40**

**Last Fifteen------------**

**:0**

**1:1**

**2:1**

**3:1**

**abuse:1**

**accomplished:1**

**account:1**

**acknowledged:1**

**acquaintances:1**

**act:1**

**added:1**

**addressed:1**

**adjusting:1**

**admiration:1**

**admire:1**

**admitted:1**