Algorithms Final Project Report

Student Name	Student ID
Sohaib Mohiuddin	100593657

CRN: 43513

Algorithms Used and Metrics for Measuring Similarity

This program is a plagiarism checker that checks for similarities between an original file and a file that will compare with the original. All the files that this program will compare are defined in respective lists which store the names and directory of the original files and the non-original files.

The process of one iteration of the program starts with the program initializing the lists containing the names of the files and opening a csv file called Similarity_Report which will contain all the similarity results of all the files compared. The main function is then called the the first files in both lists with be passed as parameters into the main function. The main function opens both files and reads both files into string lists. The string lists are split into lists of sub-lists of size 5 which will be use in the similarity checker algorithm.

The similarity checker algorithm implements the use of four "for" loops. The first loop iterates through list 1 to get the sub-lists. A temporary array is initialized to empty which will hold the similarity of each sub-list. The second loop iterates through list 2 to get the sub-lists. Temporary counters are initialized to hold the similarity value of each word comparison. The third loop iterates through the sub-lists of list 1 to get the words. The fourth loop iterates through the sub-lists of list 2 to get the words. There are two "if" statements that follow; the first condition checks if each word in sub-list 2 exists in sub-list 1 and if it does, the second condition checks if the words in sub-list 1 and sub-list 2 are equal. If the words match, the tempcounter is incremented. Once the 5 words in sub-list 2 are compared to the 5 words in sub-list 1, the value stored in the temporary counter is divided by the length of the sub-list and appended to the temporary array. Once each sub-list in list 2 is compared with the first sub-list in list 1, the maximum value of all the values in the temporary array (which holds the similarity value of sub-list 1 compared with all the sub-lists in list 2) is taken and appended into a final list which will hold all the similarities between the sub-lists of both lists. Once every word in both lists have been compared, the final list will contain the similarity values of all the sub-lists. The average value of the final list is taken as the similarity between both files.

The similarity value of a comparison between two files is between 0 and 1. 0 being 0%, which means there is absolutely no similarity between both files. 1 being 100%, which means both files are exactly identical to each other.

The Time Complexity of this algorithm is: $O(n^4)$

The similarity checker algorithm used in the program:

```
# Similarity Checker Algorithm <- Start Here
for m in sep_one:
    temparr = []
    for n in sep_two:
        tempcount = 0
        tempcountmax = 0
        for i in m:
            for j in n:
```

```
            if j in m:
                if i == j:
                    tempcount += 1
        tempcountmax = tempcount / len(m)
        temparr.append(tempcountmax)
    k = max(temparr)
    finalarr.append(k)
similarity = Avg(finalarr)
if (similarity > 1):
    similarity = 1.0
# Similarity Checker Algorithm <- End Here
```