Pseudocode of Revised Lavenshtein Algorithm

stringCompare (string 1, string 2) {

\*deconstruct string inputs by words\*

Example:

String1 = “I am a string”

Deconstruct (String1)

String1 = [“I”, “am”, “a”, “string”]

\*initialization of variables\*

Note:

matrix[[],[]] is a 2D array to record String1 and String2 difference

for (loop until end of string1 and string2) {

if (string1[] == string2[]) { // check if strings are equal

\*record to matrix\*

} else { // if not, check if string2[] exists in other index of string1

for (loop through remaining string1) {

If (string1[] == string2[]) { // if there is another match

for (loop back from past indexes of string1) {

// mark passed indexes of string1 as deleted

\*record to matrix as deleted\*

}

// mark the other match

\*record to matrix string2[] as another match\*

// check if string2 is undefined, this happens when string1 is longer than string 2

} else if (we are past end index of string2) {

\*record to matrix remaining string1 as deleted\*

}

}

// if no other match then record string2[] as an inserted word

If (no other match is found) {

\*record to matrix as insert

// check if string2 is already on the last index then match all remaining string 1 as deleted words

} else if (we are at end index of string2) {

\*record to matrix as deleted\*

// if there was a match increment string1 pointer

} else

\*add to index pointer of string1\*

}

}

\*print matrix to html\*  
}