# Introduction

## what is it about?

My report is about calculating Minimum Edit Distance (also known as Levenshtein distance), which is a string metric that calculates the degree of difference between two strings. We can think of the Levenshtein distance as the minimum number of times required to edit a single character (such as modification, insertion, deletion) when modifying from one string to another.

## What did I achieve?

My program uses the method of dynamic programming to realize the calculation of the shortest edit distance between strings.

1. Enter two words and calculate the shortest edit distance between them.

2. Input two sentences and calculate the shortest edit distance between them.

3. Import a text, each row contains a word and a symbol, 'R', or 'H',

indicating the correct Reference and the Hypothesis, respectively. My program will compare each hypothesis to the reference, and appends the minimum edit distance to the corresponding hypothesis in the output file, as shown in the following diagram.

4. Import a text, each row contains a word and a symbol, 'R', or 'H',

indicating the correct Reference and the Hypothesis, respectively. My program can calculate the shortest edit distance between the reference and the hypothesis and output it in a custom text.

# Description of my project

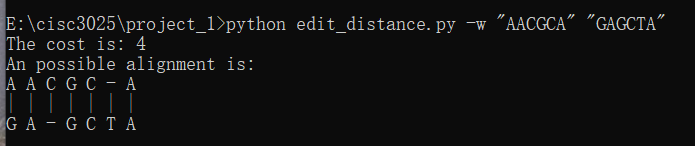
### 2.1. word\_edit\_distance(word1,word2)

The first method: word\_edit\_distance()

Usage: Calculate the minimum edit distance between two words, and give a possible alignment

Implementation method: I defined two matrices, one matrix is a distance matrix, which stores the distance, and the other matrix records how to get the previous distance matrix for backtracking. In my definition, 2 in the second matrix represents substitute, 5 represents retain, 3 represents insert, 4 represents delete, the combination of the two matrices can not only calculate the shortest edit distance, but also calculate the possible alignment

Result: As shown below:



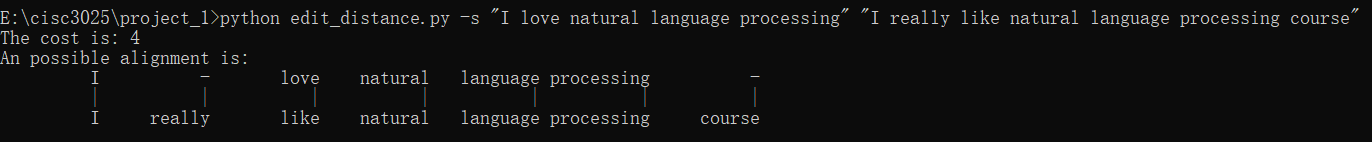
### 2.2. sentence\_edit\_distance(sentence1,sentence2)

The second method: sentence\_edit\_distance()

Usefulness: Find the minimum edit distance between two sentences and give a possible alignment.

Implementation method: Like method one, except that the single character contained in the list in method one is changed into a single word, and the others are basically the same.

Result: as shown below:



### 2.3. batch\_word(inputfile,outputfile)

The third method output\_alignment():

Uses: Enter a text, each row contains a word and a symbol, 'R', or 'H', indicating the correct Reference and the Hypothesis, respectively. My program compares each hypothesis to the reference, and appends the minimum edit distance to the corresponding hypothesis in the output file, as shown in the following diagram.

Implementation method: use the readlines() method to read a line in the text each time, if it starts with R, add the following characters to the string list str1, if it is H, add it to the string str2, and then execute the method word\_edit\_distance (str1, str2), and write the output to outputfile

Results :as shown blow:

图形用户界面, 文本

描述已自动生成

### 2.4. batch\_sentence(inputfile,outputfile)

The third method output\_alignment():

Uses: Enter a text, each row contains a word and a symbol, 'R', or 'H', indicating the correct Reference and the Hypothesis, respectively. My program compares each hypothesis to the reference, and appends the minimum edit distance to The corresponding hypothesis in the output file, as shown in the following diagram.

Implementation method: Similar to method 3, use the readlines() method to read a line in the text each time. If it starts with R, add all the following characters to the string list str1, and if it is H, add all the characters to the string str2, then execute the method sentence\_edit\_distance(str1,str2), and write the output to outputfile

Result:

文本

描述已自动生成

# Conclusion

By completing this project by myself, not only have a deeper understanding of calculating the shortest distance between strings, but also gained a lot of scattered knowledge points in the process of programming with python, such as reading the file line by line, and the blank part behind Will be read, so you need to use the rstrip() function to remove the blank part. For this project, maybe my meeting is not different, maybe there will be (because I don't know what other people's projects are like), but for me, I was better before I wrote this project, this is for me For the biggest difference.

For the future courses, I am looking forward to it. Learning theory may be very boring, but it must be interesting to realize it by hand.