소스 코드

*import* sys

args1 = sys.argv[1]

args2 = sys.argv[2]

file1 = args1

file2 = args2

*with* open(args1, "r", encoding='UTF8') *as* f:

    data1 = f.read()

*with* open(args2, "r", encoding='UTF8') *as* f:

    data2 = f.read()

wordSplit1 = data1.split()

wordCountFlag = len(wordSplit1)

syllableSplit1 = []

wordSplit2 = data2.split()

syllableSplit2 = []

*for* i in wordSplit1:

*if* (('.') in list(i)) or ((',') in list(i)) or (('"') in list(i)) or (("'") in list(i)):

*for* j in list(i):

*if* (j == '.') or (j == ',') or (j == '"') or (j == "'"):

*continue*

*else*:

                syllableSplit1 += j

*else*:

        syllableSplit1 += list(i)

*for* i in wordSplit2:

*if* (('.') in list(i)) or ((',') in list(i)) or (('"') in list(i)) or (("'") in list(i)):

*for* j in list(i):

*if* (j == '.') or (j == ',') or (j == '"') or (j == "'"):

*continue*

*else*:

                syllableSplit2 += j

*else*:

        syllableSplit2 += list(i)

syllableCountFlag = len(syllableSplit1)

wordDictSum = {}

syllableDictSum = {}

*for* i, word in enumerate(wordSplit1):

*if* wordSplit1[i] in wordDictSum:

        wordDictSum[word] += 1

*elif* not(wordSplit1[i] in wordDictSum):

        wordDictSum[word] = 1

*else*:

        print('error')

*for* i, syllable in enumerate(syllableSplit1):

*if* syllableSplit1[i] in syllableDictSum:

        syllableDictSum[syllable] += 1

*elif* not(syllableSplit1[i] in syllableDictSum):

        syllableDictSum[syllable] = 1

*else*:

        print('error')

*for* i, word in enumerate(wordSplit2):

*if* wordSplit2[i] in wordDictSum:

        wordDictSum[word] += 1

*elif* not(wordSplit2[i] in wordDictSum):

        wordDictSum[word] = 1

*else*:

        print('error')

*for* i, syllable in enumerate(syllableSplit2):

*if* syllableSplit2[i] in syllableDictSum:

        syllableDictSum[syllable] += 1

*elif* not(syllableSplit2[i] in syllableDictSum):

        syllableDictSum[syllable] = 1

*else*:

        print('error')

sameWords = 0

samesyllables = 0

*for* i, word in enumerate(wordSplit1):

*if* wordDictSum[word] > 1:

        sameWords += 1

*for* i, syllable in enumerate(syllableSplit1):

*if* syllableDictSum[syllable] > 1:

        samesyllables += 1

wordPers = (sameWords/wordCountFlag) \* 100

syllablePers = (samesyllables/syllableCountFlag) \* 100

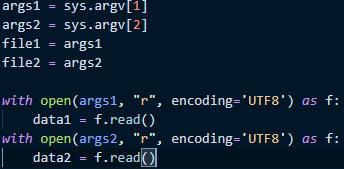
print(f'sentence 1 : {data1}')

print(f'sentence 2 : {data2}')

print(f'words similarity = {wordPers}%')

print(f'syllabic similarity = {syllablePers}%')

소스 코드 분석



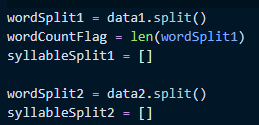
실행 방법을

python 20143109.py [text file 1] [text file 2]

로 실행한다.

text file1의 내용은 UTF-8로 인코딩 된 것이며 data1에 입력으로 들어간다.

text file2의 내용도 위와 마찬가지다.

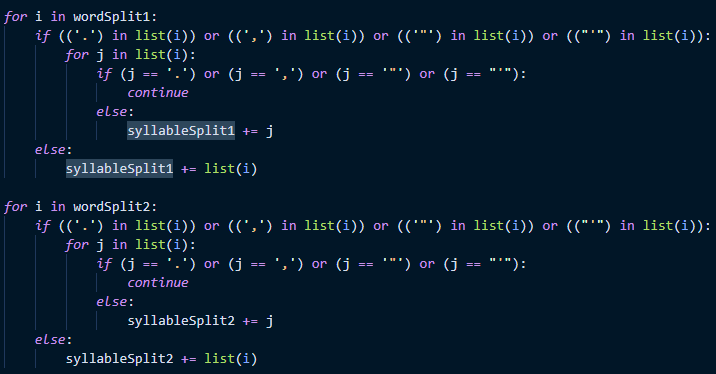


wordSplit 배열들에 split 함수를 사용하여 어절 단위로 구분을 하였다.

또한 어절의 개수를 세기 위한 wordCountFlag 변수를 설정하였다.

어절의 개수는 text file1을 기준으로 하였다.

syllableSplit 배열들은 음절을 담기 위한 변수다.



음절 단위로 나누기 위한 코드이다.

syllableSplit 배열들에 음절 단위로 나눈 값들이 들어간다.

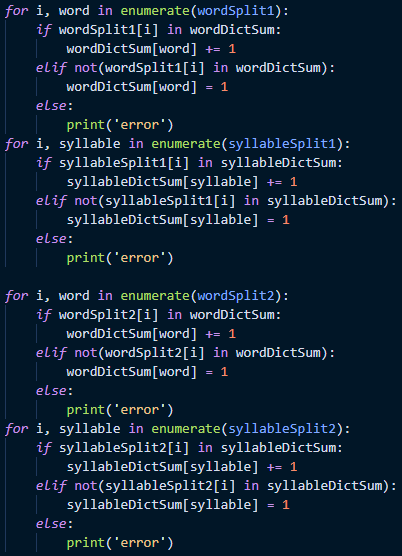
여기서 . ‘ “ , 같은 문장 부호들은 제외를 시켜주었다.



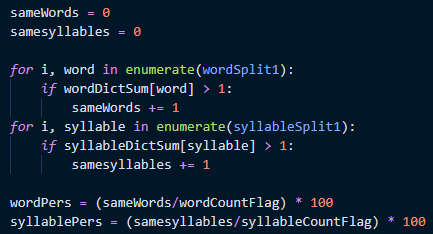
text file 1을 기준으로 음절 개수를 카운팅 하였다.



파이썬의 딕셔너리를 사용하여 동일한 어절 또는 음절을 찾을 것이다.



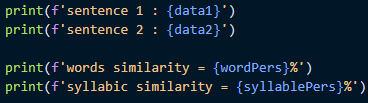
파이썬의 딕셔너리를 사용하여 동일한 어절 또는 음절이 있다면 1을 증가시켜주고 없다면 초기값을 1로 설정하였다.



같은 음절 또는 어절이 존재한다는 판단은 딕셔너리의 값이 1을 초과하면 두 문장에 동일한 어절 또는 음절이 존재한다고 판단하였다.

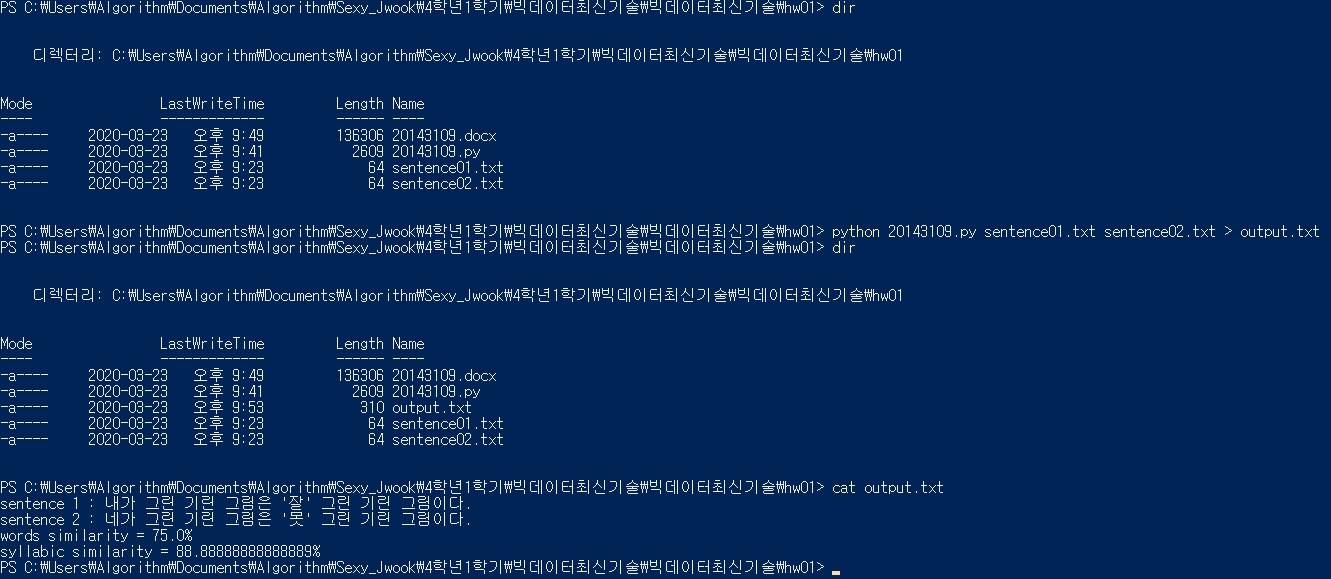
그래서 어절 또는 음절의 유사도는

(동일한 어절 또는 음절의 개수) / (1번 문장의 어절 또는 음절의 개수) \* 100 으로 유사도를 측정하였다.



이와 같이 출력을 한다.

실행 화면



문장 1: 내가 그린 기린 그림은 ‘잘’ 그린 기린 그림이다.

문장 1: 네가 그린 기린 그림은 ‘못’ 그린 기린 그림이다.

어절 유사도는 75.0%로 측정하였다.

음절 유사도는 88.88888888888889%로 측정하였다.