one_similiarity

```
import math
import string
import sys



def read_file(filename):


try:
with open(filename, 'r') as f:
data = f.read()
return data


except IOError:
print("Error opening or reading input file: ", filename)
sys.exit()



translation_table = str.maketrans(string.punctuation+string.ascii_uppercase,"
"*len(string.punctuation)+string.ascii_lowercase)



def get_words_from_line_list(text):


text = text.translate(translation_table)
word_list = text.split()


return word_list
```

```
def count_frequency(word_list):

D = {}

for new_word in word_list:

if new_word in D:
D[new_word] = D[new_word] + 1

else:
D[new_word] = 1

return D



def word_frequencies_for_file(filename):

line_list = read_file(filename)
word_list = get_words_from_line_list(line_list)
freq_mapping = count_frequency(word_list)

print("File", filename, ":", )
print(len(line_list), "lines, ", )
print(len(word_list), "words, ", )
print(len(freq_mapping), "distinct words")

return freq_mapping
```

```
def dotProduct(D1, D2):
Sum = 0.0


for key in D1:


if key in D2:
Sum += (D1[key] * D2[key])


return Sum




def vector_angle(D1, D2):
numerator = dotProduct(D1, D2)
denominator = math.sqrt(dotProduct(D1, D1)*dotProduct(D2, D2))


return math.acos(numerator / denominator)




def documentSimilarity(filename_1, filename_2):




sorted_word_list_1 = word_frequencies_for_file(filename_1)
sorted_word_list_2 = word_frequencies_for_file(filename_2)
distance = vector_angle(sorted_word_list_1, sorted_word_list_2)
```

```
print("The distance between the documents is: % 0.6f (radians)"% distance)



documentSimilarity('sample1.txt', 'sample2.txt')



#OUTPUT


# File sample1.txt :
# 598 lines,
# 113 words,
# 66 distinct words
# File sample2.txt :
# 779 lines,
# 154 words,
# 89 distinct words
# The distance between the documents is:  0.618456 (radians)
```