1. Introduction

1. Project Purpose and Background: This project was conducted to use functions to write code to apply knowledge learned for seven weeks. The goal is to practice practical practice based on the lessons applied.

2. Goal: Develop a basic search engine to search for sentences similar to the user's case-insensitive queries

2. Requirements

1. User Requirements: The system must be able to search for sentences similar to your query.

2. Functional Requirements:
   1) pre-process and list sentences in search
   2) receive and pre-process English string input from user
   3) alculate similarity between English string and sentences in search (similarity is the same number of words)
   4) rank sentences based on similarity, output top 10 of the ranked sentences to user
   5) calculate similarity without case (shows user input case-insensitive input as it is)

3. Design and Implementation

1. Implementation Details:
   1) Implementing a function that can strip, lower, and split sentences
   2) Enter the string from the user, save it to the list, and apply it to the preprocessing function
   3) Implementing a function that computes similarities
   4) Implementing a sorting system based on similarity
   5) Number 10 rank output using the for statement
   6) Recall and output pre-saved variables for existing sentences entered by the user

1) Functional Implementation :

① Pre-processing function

```
# Preprocess function
def preprocess(sentence):
    preprocessed_sentence = set(sentence.strip().lower().split(" "))
    return preprocessed_sentence
```

- sentence = sentences entered by the user
- Return Value = Preprocessed Sentence
- Result = Stripe, lower, and split the sentences input by the user
- Explanation = Get input from the user, preprocess, and return the preprocessed sentence

② Indexing function

```
# Indexing function
def indexing(file_name):

    file_tokens_pairs = [] # make a list
    lines = open(file_name, "r", encoding="utf8").readlines()   # Read the sentences from the file and save them in a variable

    for line in lines:
        tokens = preprocess(line)     # Call the preprocess function, add it to the list, and return it
        file_tokens_pairs.append(tokens)
        #print(tokens)
    return file_tokens_pairs
```

- file_name = Gets a sentence from a file and indexes it
- Return value = Tokenize and import the words in the file
- Results = Access statements in the file and bring them into the list in token units and return them
- Explanation = Create a list, access a sentence in the file via a recurring statement, and save it to the list in units of tokens on each line of the sentence

③ Similarity calculation function

```
# Calculate similarity function
def calc_similarity(preprocessed_query, preprocessed_sentences):

    score_dict = {}
    query_token_set = set(preprocessed_query)


    # Take the sets, calculate the following values, find the similarity, and return them to score_dict
    for i, sentence_tokens in enumerate(preprocessed_sentences):
        all_tokens = query_token_set | sentence_tokens
        same_tokens = query_token_set & sentence_tokens
        similarity = len(same_tokens) / len(all_tokens)
        score_dict[i] = similarity

    return score_dict
```

- preprocessed_query = Preprocessed values entered by the user
- preprocessed_sentences = Preprocessed values imported from the file
- Return value = Dictionary value, a set of preprocessed values entered by the user and preprocessed values taken from the file
- Results = Preprocessed values entered by the user and preprocessed values taken from the file compute to obtain similarity of similarity
- Explanation = Repeat calculating each token of the preprocessed value entered by the user and the preprocessed value taken from the file to obtain the similarity

4. Testing

1) Test Results by Function :

① Code to be entered by user

② Accessed and tokenized code for a sentence in a file

③ When there is no similar sentence

```
영어 쿼리를 입력하세요.hello
There is no similar sentence.
```

```
{'generally', 'farm', 'us', 'fruit', 'and', 'helping', 'all', 'be', 'picking',
"you'll", 'work.', 'usual', 'the', 'do'}
{'with', 'middle', 'and', 'ages,', 'were', 'very', 'not', 'clean,', 'filled',
'cities', 'garbage.', 'in', 'streets', 'the'}
{'sooner', 'will', 'society', 'strings,', 'hiding', 'apron', 'may', 'with', 'b
ehind', 'their', 'they', 'moment', 'up', 'later', 'yet', 'the', 'or', 'but',
'progressive', 'world.', 'catch', 'be', 'for'}
{'do', 'what', 'know', 'cow', 'minister.', 'you', 'said', 'answered?"', 'the'}
{'seem', 'and', 'italy', 'very', 'poland', 'countries.', 'like', 'different',
'may'}
{'in', 'stayed', 'smith', 'and', 'whole', 'oxford.', 'day', 'mr.', 'i', 'the'}
{'a', 'sight', 'gave', 'an', 'of', 'traffic', 'red', 'signal', 'him', 'idea.',
'the'}
{'used', 'instead.', 'so', 'pumpkins', 'they'}
{'particular', 'offer', 'me', 'a', 'might', 'not', 'state', 'much', 'of', 'mon
ey.', 'affairs:', 'occasion', '2.', 'they'}
{'information', 'hope', "you'll", 'this.', "i'm", 'skills,', 'include', 'i',
```

④ The part that shows the case-free value entered by the user

```
영어 쿼리를 입력하세요.hellO mY naMe is mIKE
rank    Index   score   sentence
Input sentence :  hellO mY naMe is mIKE
```

2) Final result :

```
영어 쿼리를 입력하세요.hellO mY naMe is mIKE
rank    Index   score   sentence
Input sentence :  hellO mY naMe is mIKE
1       679     0.5     name is mike. my
2       526     0.2857142857142857      bob is brother. my
3       538     0.2857142857142857      is hobby traveling. my
4       453     0.25    sketching my is mother them.
5       241     0.2222222222222222      running my father is with so-ra.
6       336     0.2222222222222222      my the family is park. at
7       212     0.2     for waiting my me. is sister betty
8       505     0.18181818181818182     annie five years little my is old. sister
9       610     0.15384615384615385     i voice my yell, "lunch is ready!" and would raise
10      190     0.14285714285714285     is it sunday.
```

5. Results and Conclusion

1. Results: Successfully developed a search engine.
2. Conclusion: It was difficult to call the file and access the sentence, but I understand a little.