Python programming & practices

ROAD DESIGN SUPPORTER

Final Report

Date : 2023. 12. 24

Name : SeongJun, Na

ID : 192002

1. Introduction

1) Background

Modern society has achieved tremendous growth through rapid industrialization and thr birth of the fourth industry. Technologies in various fields such as transportation, IT, medicine, and science are developing day by day. It makes our lives more convenient and safe. In particular, the development of transportation and roads made the movement of resources convenient and led to active exchanges between local residents. This is one of the factors that greatly contributed to the rapid development of the industry. In addition, since human's desire for convenience is endless, the development of autonomous driving through artificial intelligence is currently underway. As time goes by, the amount of data we have increases, and it is almost impossible for us humans to memorize all these contents. That's why we keep information and use it through searches. The road and transportation sectors, like other fields, are making endless progress. Recently, as artificial intelligence and autonomous driving have begun to be introduced, new standards for roads have been proposed. In addition, many factors such as the width of the road, the radius of the curve, and the design reference speed of the road are considered in the process of designing the road. In this situation, it is inefficient for us to memorize all those vast amounts of data Because we have Python. We need programs that can easily manage vast amounts of data used in the process of designing roads. I'm going to make this through Python.

2) Project goal

Data on road design standards are analyzed and made into data. This can be referenced through simple keyword input when necessary and helps designers understand through image search. In addition, it forms data on frequently sought content through the storage of search records and helps to use easy keywords easily. Lastly, for the easy use of useful information obtained from using the program, I produce a program that can send the contents I organized by e-mail. That is the goal of this project.

3) Differences from existing programs

Searching for documents and finding data through keywords is solved by our search engines. However, what I felt while studying civil engineering was that the specialized contents such as roads, traffic, and structures were only guidelines issued by government agencies. To utilize this, it is necessary to access the homepage, read the pdf, read major books and read standard guidelines directly. I had to find each one myself. I have never experienced a database management program focused on the road field. So I'm going to make it myself.

2. Functional Requirement

1) Design Criteria Search Engine

- Recommend highly relevant content when entering keywords on road design criteria.

2) Image Search Engine

- Improve understanding and convenience through image output such as tables and charts when entering keywords related to road design criteria.

3) Search History Retention Function

- Store search history to make it easy for users to see and use.

4) Data mail transfer function

- Provide the ability to mail user notes from the application

5) Recommendation of the searching keywords

- In order to provide guidelines to users, we provide recommended keywords by storing search keywords in the order of the highest search frequency.

3. Implementation

(1) main program

- input & output

part_1 importing modules

```
importreadlines
```

: import "readlines" module to read the file which we want to find the contents. Using this module is for using search engine which we made.

```
importsending_email
```

: import “sending_email” module to send the contents which we want to use. This module sends the emails with using python program.

```
importfinding_image
```

: import “finding_image” module to find the appropriate images for our keywords.

```
importkeyword_recommendation
```

: import “keyword_recommendation” module to show the Top 3 keywords. This module will provide guidelines for users to use the search function.

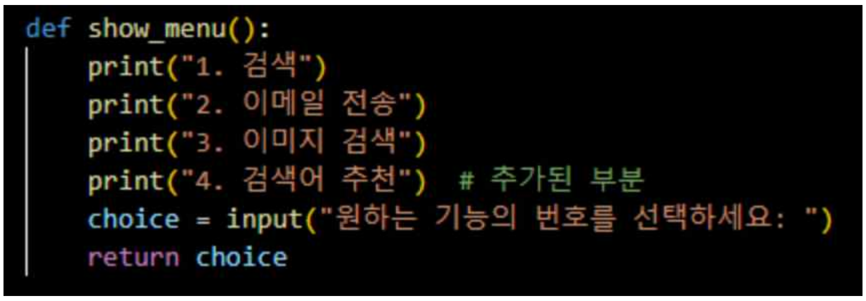
```
importtkinterastk
```

: import “tkinter” library and give the nickname “tk” to the library. This library is standard python library for GUI programming.

part_2 function_1 show_menu

```
defshow_menu():
```

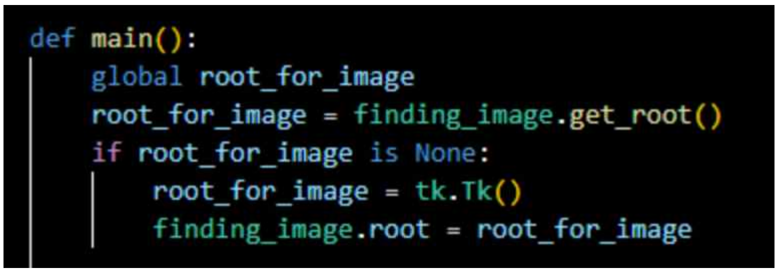
define the function which name is “show_menu”. It serves to show the user the menu and receive selections.



```
def show_menu():
    print("1. 검색")
    print("2. 이메일 전송")
    print("3. 이미지 검색")
    print("4. 검색어 추천")  # 추가된 부분
    choice = input("원하는 기능의 번호를 선택하세요: ")
    return choice
```

It is the step of introducing the functions provided by the program we have written and determining which functions the user will use.

part_3 function_2 main



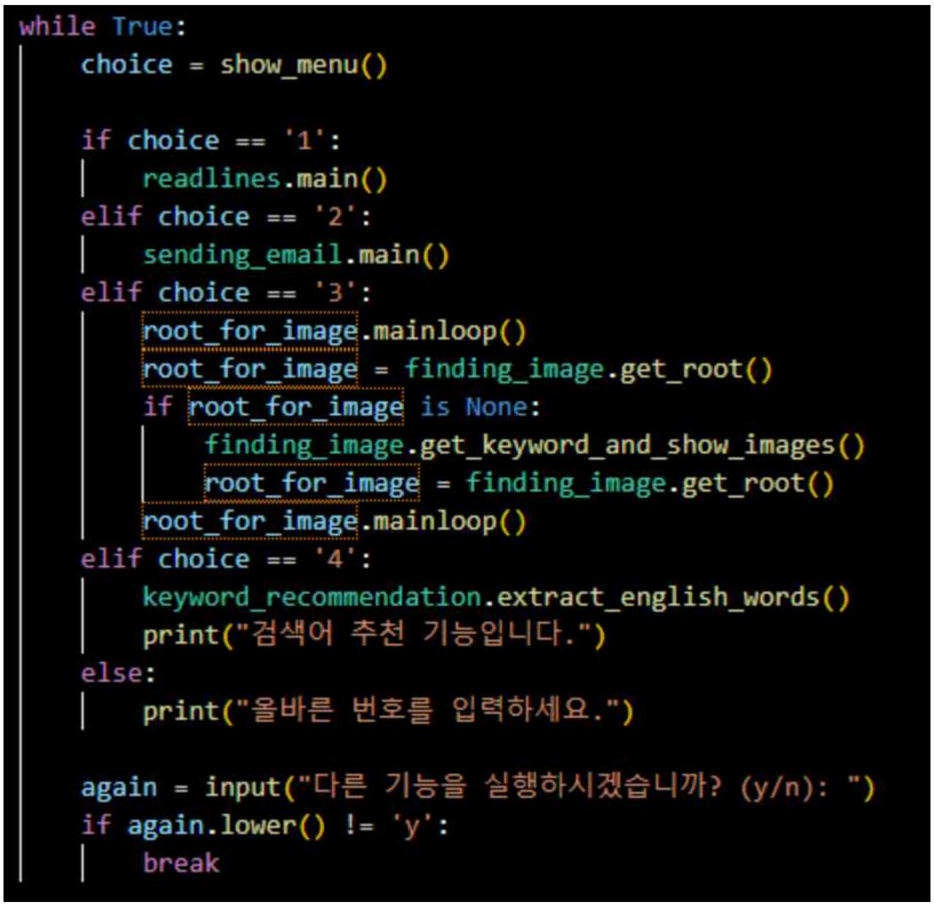
```python
def main():
    global root_for_image
    root_for_image = finding_image.get_root()
    if root_for_image is None:
        root_for_image = tk.Tk()
        finding_image.root = root_for_image
```

This part includes the main logic of our program. First, We declare a global variable. In the image search module, use the get_root function to initialize the root_for_image value.

In the if sentences, “root_for_image is None:” is checking that “root_for_image” is “None”.

”root_for_image = tk.Tk()” is Initializing root_for_image to the default TK object in tkinter.

part_4 function_2 main_while_loop



```python
while True:
    choice = show_menu()

    if choice == '1':
        readlines.main()
    elif choice == '2':
        sending_email.main()
    elif choice == '3':
        root_for_image.mainloop()
        root_for_image = finding_image.get_root()
        if root_for_image is None:
            finding_image.get_keyword_and_show_images()
            root_for_image = finding_image.get_root()
        root_for_image.mainloop()
    elif choice == '4':
        keyword_recommendation.extract_english_words()
        print("검색어 추천 기능입니다.")
    else:
        print("올바른 번호를 입력하세요.")

    again = input("다른 기능을 실행하시겠습니까? (y/n): ")
    if again.lower() != 'y':
        break
```

“while True” is starting infinite Loop.

Get the choice value from the function_1, “show_menu” and Use if statements to perform functions that match variable values.

· if choice value is ‘1’, perform the function “Design Criteria Search Engine” and “Search History Retention Function”.

· if choice value is ‘2’, perform the function “Data mail transfer function”

· if choice value is ‘3’, perform the function “Image Search Engine”

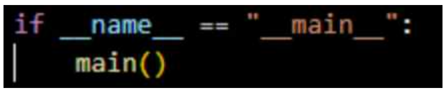
· if choice value is ‘4’ perform the function “Recommencation of the searching keywords”

· if we input the value which is not between 1~4, this program will show the message

“ 올바른 번호를 입력하세요. ”.

When we finished the work for one time, This program will show the message “다른 기능을 실행하시겠습니까? (y/n):”. If you input “y”, the function_1 will be performed again. If you input “n”, the program will be end.

part_5 call the main function



```
if __name__ == "__main__":
    main()
```

Call the main function when the script runs directly.

- Explanation

The main program serves as the backbone of the program we created this time and determines what functions it will perform based on the values you enter. There are four functions that this program can implement.

1. Design Criteria Search Engine and Search History Retention Function

2. Data mail transfer function

3. Image Search Engine

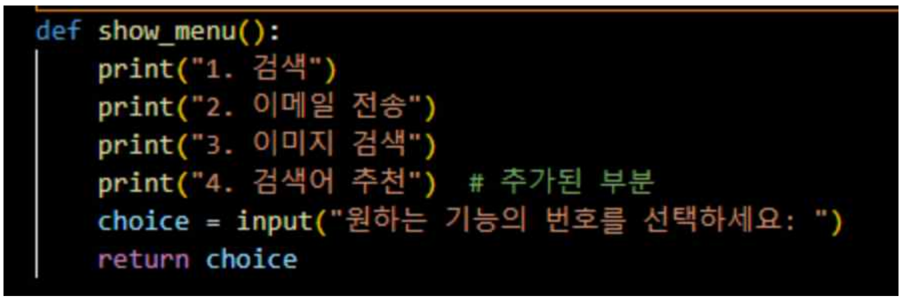
4. Recommendation of the searching keywords

- The contents which we have learned.

```
importreadlines
importsending_email
importfinding_image
importkeyword_recommendation
importtkinterastk
```

· module

This section contains what we learned about the module. A module is a collection of functions, variables, or classes and refers to a Python file that can be retrieved and used by other Python programs. This is used by using the import command. Import is a command that allows you to use Python modules you have already created and is used in the form of "import module_name".



```python
def show_menu():
    print("1. 검색")
    print("2. 이메일 전송")
    print("3. 이미지 검색")
    print("4. 검색어 추천")  # 추가된 부분
    choice = input("원하는 기능의 번호를 선택하세요: ")
    return choice
```

· function

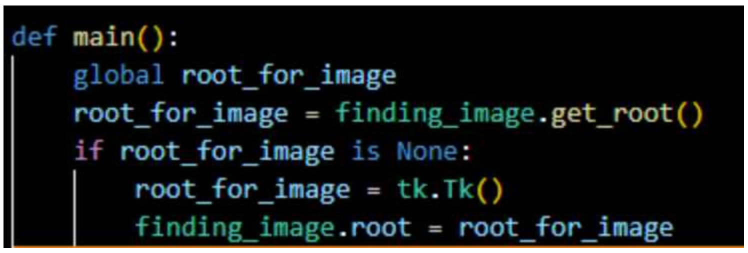
A function is a block of code that performs a specific task. Using a function can increase the reusability of the code and improve the readability of the code. function is defined “def_function’s name(parameter1,parameter2,...)”. After you define a function, you can call it using the function name and the required parameters (factors).

· print

"print" show the characters and the results on the computer's monitor.

· input

"input" get the input from the users and store it in the parameter with the type "character".



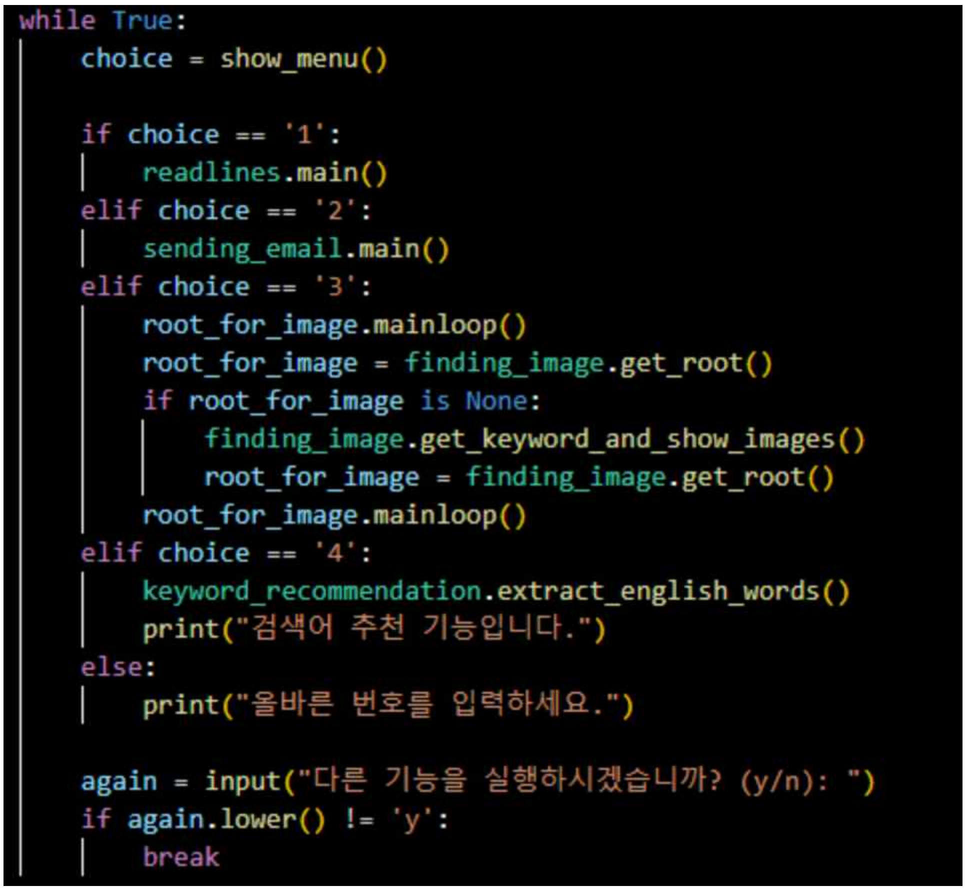
```python
def main():
    global root_for_image
    root_for_image = finding_image.get_root()
    if root_for_image is None:
        root_for_image = tk.Tk()
        finding_image.root = root_for_image
```

· global variables

Variables include regional and global variables. Regional variables can only be used within the unit of code block we are using, and become variables that do not exist outside the range. However, a global variable refers to a variable that has a wide range so that it can be used within the entire program. Depending on how you use it, you have to use it carefully because it greatly affects the results of the program.

· calling the function

"root_for_image = finding_image.get_root()" is calling the function in the modules. When fetching a function in a module, use it in the form of a module name.Function name (parameter 1, parameter 2, ...).



```python
while True:
    choice = show_menu()

    if choice == '1':
        readlines.main()
    elif choice == '2':
        sending_email.main()
    elif choice == '3':
        root_for_image.mainloop()
        root_for_image = finding_image.get_root()
        if root_for_image is None:
            finding_image.get_keyword_and_show_images()
            root_for_image = finding_image.get_root()
        root_for_image.mainloop()
    elif choice == '4':
        keyword_recommendation.extract_english_words()
        print("검색어 추천 기능입니다.")
    else:
        print("올바른 번호를 입력하세요.")

    again = input("다른 기능을 실행하시겠습니까? (y/n): ")
    if again.lower() != 'y':
        break
```

· while loop and if sentences

The while iteration executes a particular block of code repeatedly while the condition is True. When the condition is false, it terminates the iteration.

How it works (while iteration)

1. Check conditions: First, check the conditions of the while keyword.

2.Code block execution: If the condition is True, run the corresponding code block.

3.Recheck condition: Check condition again after code block execution is complete.

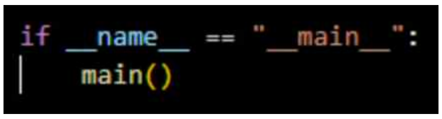
4.Repeat: If the condition is still True, repeat the process above.

5.Exit: When the condition is false, exit the while iteration and proceed to the next code.

If statements execute a particular block of code only if the given condition is True; if the condition is False, that block of code will not be executed.

if and else: You can run different blocks of code when the condition is true and false.

if and elif and else: Multiple conditions can be examined continuously.

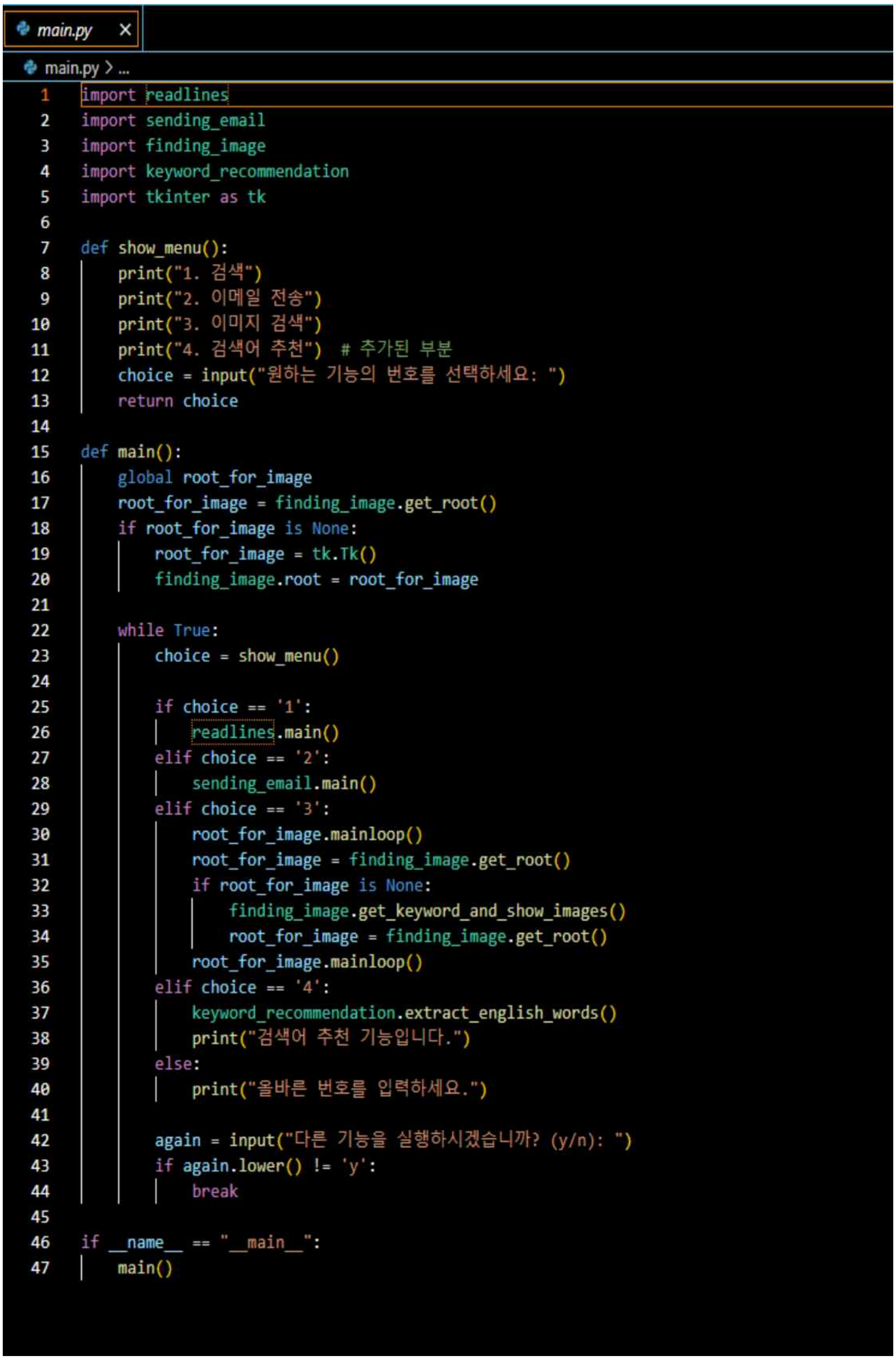


```
if __name__ == "__main__":
    main()
```

· It is a conventional method for executing a particular block of code when Python scripts are executed directly.

"__name__" is the name of the present module. When module is performed, "__main__" string will be printed.

- **Code screenshot**


main.py

main.py > ...

```python
import readlines
import sending_email
import finding_image
import keyword_recommendation
import tkinter as tk

def show_menu():
    print("1. 검색")
    print("2. 이메일 전송")
    print("3. 이미지 검색")
    print("4. 검색어 추천")  # 추가된 부분
    choice = input("원하는 기능의 번호를 선택하세요: ")
    return choice

def main():
    global root_for_image
    root_for_image = finding_image.get_root()
    if root_for_image is None:
        root_for_image = tk.Tk()
        finding_image.root = root_for_image

    while True:
        choice = show_menu()

        if choice == '1':
            readlines.main()
        elif choice == '2':
            sending_email.main()
        elif choice == '3':
            root_for_image.mainloop()
            root_for_image = finding_image.get_root()
            if root_for_image is None:
                finding_image.get_keyword_and_show_images()
                root_for_image = finding_image.get_root()
            root_for_image.mainloop()
        elif choice == '4':
            keyword_recommendation.extract_english_words()
            print("검색어 추천 기능입니다.")
        else:
            print("올바른 번호를 입력하세요.")

        again = input("다른 기능을 실행하시겠습니까? (y/n): ")
        if again.lower() != 'y':
            break

if __name__ == "__main__":
    main()
```

(2) Data mail transfer function

- input & output

part_1 import modules

```
importsmtplib
```

· Module required to create and use SMTP client sessions for sending emails.

```
fromemail.mime.textimportMIMEText
```

· Used to generate multipurpose Internet Mail Extensions (MIME) format messages for emails.

```
fromemail.mime.multipartimportMIMEMultipart
```

· email.mime.multipart module

The module implements a multipart subtype of Multipurpose Internet Mail Extensions (MIME) messages that allows email messages to be divided into different parts and contain different formats and attachments. MIME defines the standard format of email messages and supports multi-part messages, text messages, attachments and more.

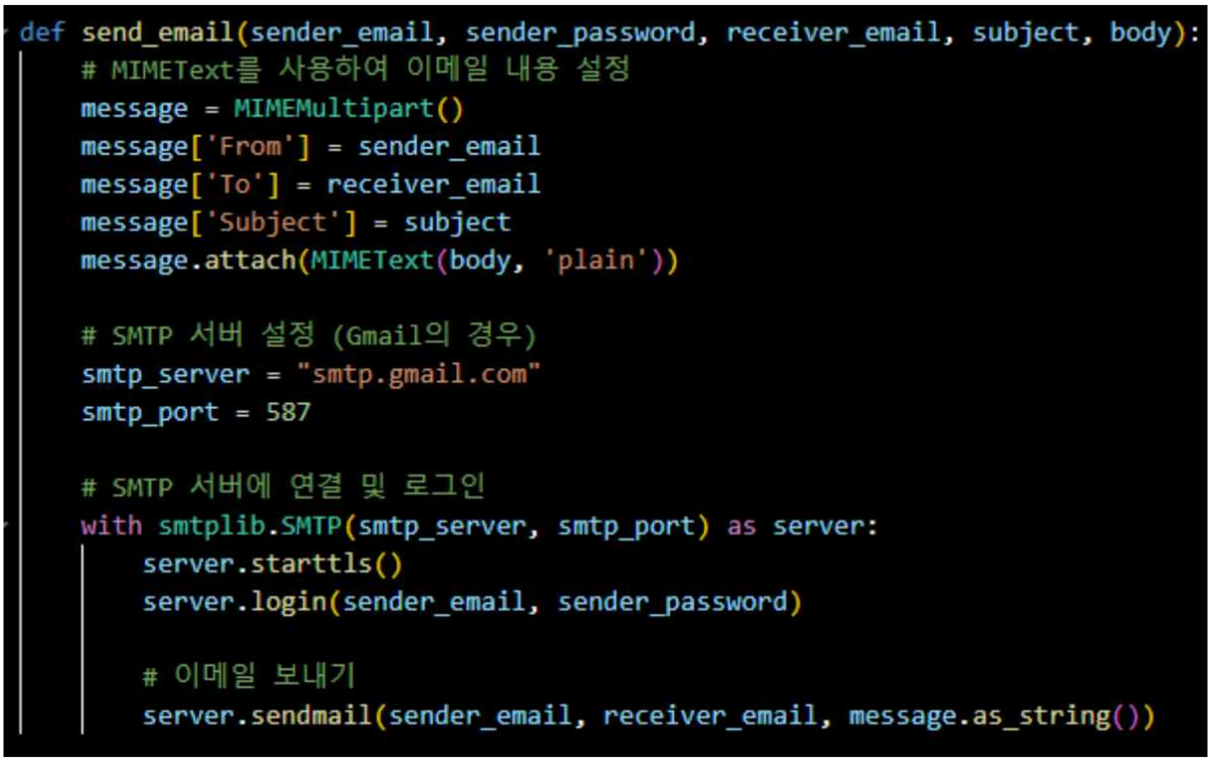
· MIMEMultipart class

MIME multipart is a class for generating multipart types of MIME messages. This class allows you to organzie email messages into multiple parts (for example, body text, attachments, etc). After you create a MIMEMultipart object, you can add different types of MIME parts using the attach() method.

```
importgetpass
```

· This module receives input without displaying text when entering a password.

part_2 function_1 send_email



```python
def send_email(sender_email, sender_password, receiver_email, subject, body):
    # MIMEText를 사용하여 이메일 내용 설정
    message = MIMEMultipart()
    message['From'] = sender_email
    message['To'] = receiver_email
    message['Subject'] = subject
    message.attach(MIMEText(body, 'plain'))

    # SMTP 서버 설정 (Gmail의 경우)
    smtp_server = "smtp.gmail.com"
    smtp_port = 587

    # SMTP 서버에 연결 및 로그인
    with smtplib.SMTP(smtp_server, smtp_port) as server:
        server.starttls()
        server.login(sender_email, sender_password)

        # 이메일 보내기
        server.sendmail(sender_email, receiver_email, message.as_string())
```

· The given code defines a function that sends an email.

```
def send_email(sender_email, sender_password, receiver_email, subject, body):
```

The function receives five factors: the sender's e-mail address, the sender's e-mail password, the receiver's e-mail address, the subject of the e-mail, and the text of the e-mail.

```
    message = MIMEMultipart()
```

Create a base structure that can contain multiple parts of an email message by creating a MIMEMPpart object.

```
    message['From'] = sender_email
    message['To'] = receiver_email
    message['Subject'] = subject
```

Sets header information for the generated message object: sender, recipient, and email title.

```
    message.attach(MIMEText(body, 'plain'))
```

Create a MIMEText object to set the body content of the email. The body is set in plain text format.

```
    smtp_server = "smtp.gmail.com"
    smtp_port = 587
```

Saves the address and port number of the Gmail SMTP server in a variable.

```
    with smtplib.SMTP(smtp_server, smtp_port) as server:
```

Create a smtplib.SMTP object to connect to an SMTP server.Use with statements to safely manage SMTP sessions.

```
        server.starttls()
```

Initiate Transport Layer Security (TLS) encryption. This setting is required to encrypt communications with the Gmail SMTP server.

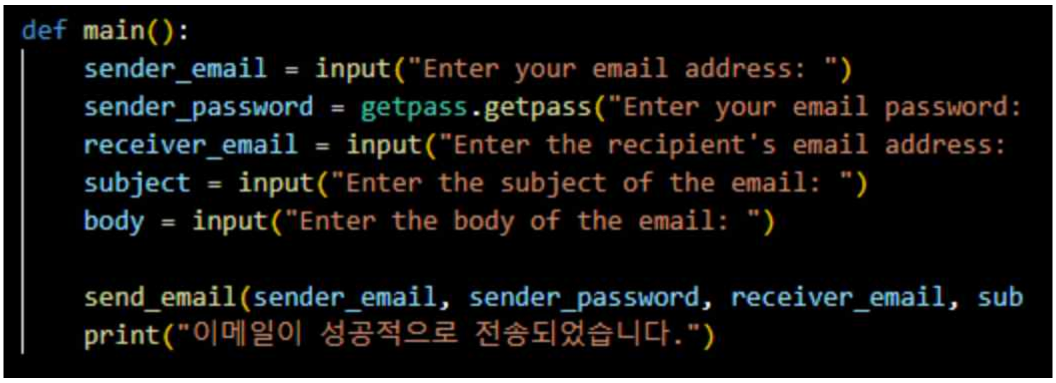
```
        server.login(sender_email, sender_password)
```

Log in to the SMTP server. Authentication is performed using the sender's email address and password.

```
        server.sendmail(sender_email, receiver_email, message.as_string())
```

Send an email using the server.sendmail method which forwards the sender, receiver and message content as a factor. message.as_string() converts the MIME Multipart object into a string to make it transferable.

part_3 function_2 main



```
def main():
    sender_email = input("Enter your email address: ")
    sender_password = getpass.getpass("Enter your email password:
    receiver_email = input("Enter the recipient's email address:
    subject = input("Enter the subject of the email: ")
    body = input("Enter the body of the email: ")

    send_email(sender_email, sender_password, receiver_email, sub
    print("이메일이 성공적으로 전송되었습니다.")
```

· This function is the main function of this sending_email.py program. In this function,

get the user informations and write the contents of the mail. If we sent the mail successfully, the message "Email was sent successfully" will appear. The only email you can receive in this process is gmail. This is because the server connected in this program is a gmail server. And the password requires setting the app password, and I will leave the related information in the note. Also, even if you enter the password, it will not be visible by getpass.

- Explanation

This function serves to organize and record information obtained using the program's search function within the program and deliver it easily and quickly through e-mail. It performs the function of improving the program's information utilization ability and improving users' freedom. Mail is compatible with this program. In the process, you must use an app password.

· app password

This content is for using the ‘Data mail transfer function’. App password is different password which you know. To get this password, we need some process.

< process >

First, Enter “google chrome” and click your profile.

Second, Enter “Managing google account”.

Third, Enter the category which name is “security”.

Fourth, Enter ‘2-step-authentication’ and scroll to the bottom.

Fifth, Click ‘App password’.

Sixth, Input the app’s name and make your password.

If you click the button ‘Make’, you can see the password which given from google.

Copy it and Use it to using my program.

After getting password, Come back to my program.

Enter your gmail on ID and Enter the app password in the blank. You can’t see the password while you wirte the password in the blank. Because The program is protecting your password’s exposure.

- The contents which we have learned.

```
fromemail.mime.textimportMIMEText
fromemail.mime.multipartimportMIMEMultipart
```

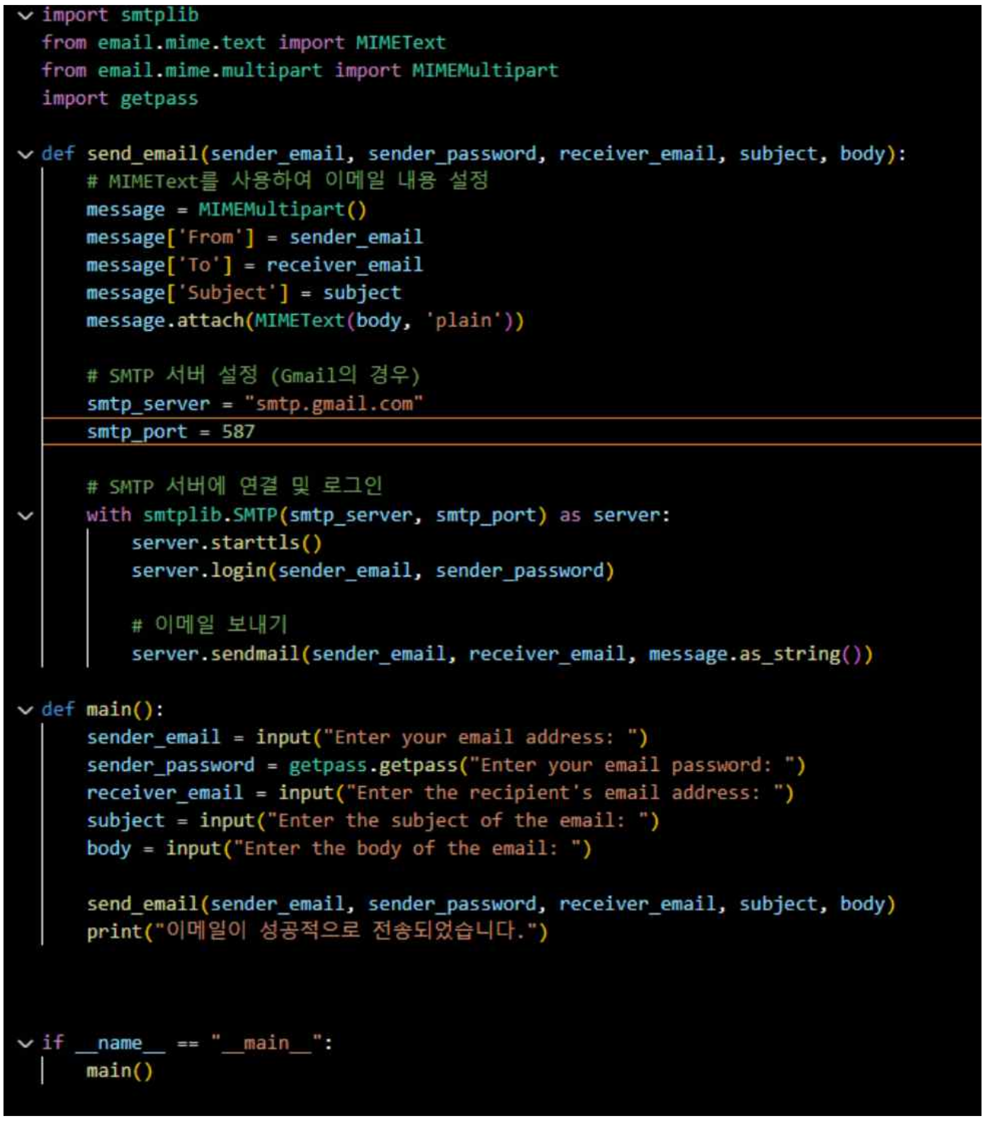
· module

There are many ways when we load modules. One of them is the from-import method. This can be used as a case where you want to use only a specific function or class in the module, import multiple functions or classes in the module, or finally import everything in the module.

· class

Classes were applied in this part. MIMEMultipart is the class to make the multipart type of MIME message. Classes are designs or frameworks for creating objects, including attributes and methods. Object means an instance of the class. Allocated to actual memory based on the class is called an object. For example, “apples” is a class and “multiple apples” is a set of objects. Attributes is a variable that represents a characteristic of a class or object. For example, in the “People” class, names, ages and so on can be attributes. Method is a function defined in a class or object that performs a specific action. A method can manipulate properties of that class or call another method.

- Code screenshot



```python
import smtplib
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
import getpass

def send_email(sender_email, sender_password, receiver_email, subject, body):
    # MIMEText를 사용하여 이메일 내용 설정
    message = MIMEMultipart()
    message['From'] = sender_email
    message['To'] = receiver_email
    message['Subject'] = subject
    message.attach(MIMEText(body, 'plain'))

    # SMTP 서버 설정 (Gmail의 경우)
    smtp_server = "smtp.gmail.com"
    smtp_port = 587

    # SMTP 서버에 연결 및 로그인
    with smtplib.SMTP(smtp_server, smtp_port) as server:
        server.starttls()
        server.login(sender_email, sender_password)

        # 이메일 보내기
        server.sendmail(sender_email, receiver_email, message.as_string())

def main():
    sender_email = input("Enter your email address: ")
    sender_password = getpass.getpass("Enter your email password: ")
    receiver_email = input("Enter the recipient's email address: ")
    subject = input("Enter the subject of the email: ")
    body = input("Enter the body of the email: ")

    send_email(sender_email, sender_password, receiver_email, subject, body)
    print("이메일이 성공적으로 전송되었습니다.")


if __name__ == "__main__":
    main()
```

(3) Recommendation of the searching keywords

- input & output

part_1 import part

```
importre
```

· ‘re’ is a built-in module included in Python's standard library that provides functions and classes for regular expression processing.

· Function of ‘re’

1. Search for strings and match patterns

re.search(): Finds the first location within the string that matches the regular expression pattern.
re.match(): Checks the regular expression pattern only at the beginning of the string.
re.findall(): Finds any part of the string that matches the regular expression pattern and returns it to the list.

2. Replace and modify patterns

re.sub(): Replace the part within the string that matches the regular expression pattern with a new string.

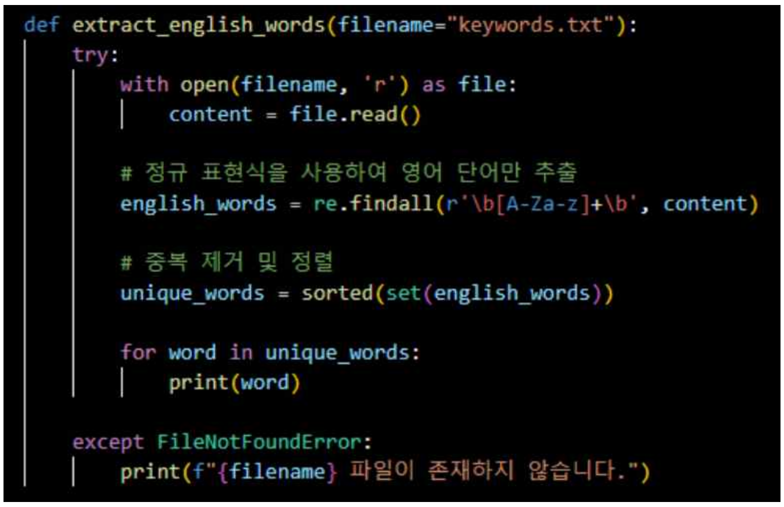
3. Split pattern

re.split(): Slices a string based on a regular expression pattern and returns it to the list.

4. Compiled Patterns

re.compile(): returns a re object by compiling regular expression patterns, which can be quickly searched through a variety of methods.

part_2 function_1 extract_english_words



```python
def extract_english_words(filename="keywords.txt"):
    try:
        with open(filename, 'r') as file:
            content = file.read()

        # 정규 표현식을 사용하여 영어 단어만 추출
        english_words = re.findall(r'\b[A-Za-z]+\b', content)

        # 중복 제거 및 정렬
        unique_words = sorted(set(english_words))

        for word in unique_words:
            print(word)

    except FileNotFoundError:
        print(f"{filename} 파일이 존재하지 않습니다.")
```

This function performs a keyword recommendation function that receives a search word and a file in which the search frequency is stored and provides search records from the order of the highest frequency to the lowest frequency.The text file in which the search history is stored consists of keywords and the number of searches, and includes numbers together. In order to prevent the number, which is a frequency, from being counted, the code was written to extract only search words composed of English words.

```
def extract_english_words(filename="keywords.txt"):
```

Define a function called extract_english_words, which essentially extracts English words from the "keywords.txt" file.You can specify a different file name with the filename parameter.

```
    try:
        with open(filename, 'r') as file:
            content = file.read()
        english_words = re.findall(r'₩b[A-Za-z]+₩b', content)
        unique_words = sorted(set(english_words))
        for word in unique_words:
            print(word)
    except FileNotFoundError:
        print(f"{filename} 파일이 존재하지 않습니다.")
```

Start the try block to run the code to handle exceptions. Use the with statement to safely open the file, referring the file object to the name file. Read the entire contents of the file and save it in the variable content. Use the re.findall function to find all the English words that match the regular expression pattern ₩b[A-Za-z]+₩b in the content and return them to the list. Use the set to remove duplicate words and sort the results into a sorted function. This creates a list of English words that have been deduplicated. Start a for loop to iterate each word in the list of English words that have been deduplicated. Output each word to the screen. If a FileNotFoundError exception occurs when attempting to open a file, the message "File does not exist" with the name of that file.

- Explanation

This function serves to recommend which keywords were searched a lot in the order of the highest frequency of search terms by using the search history storage text file formed while using the program's search function. In other words, it serves as a guide for users to use the program as a search word recommendation program. This influences deciding which contents to search among road-related regulations.

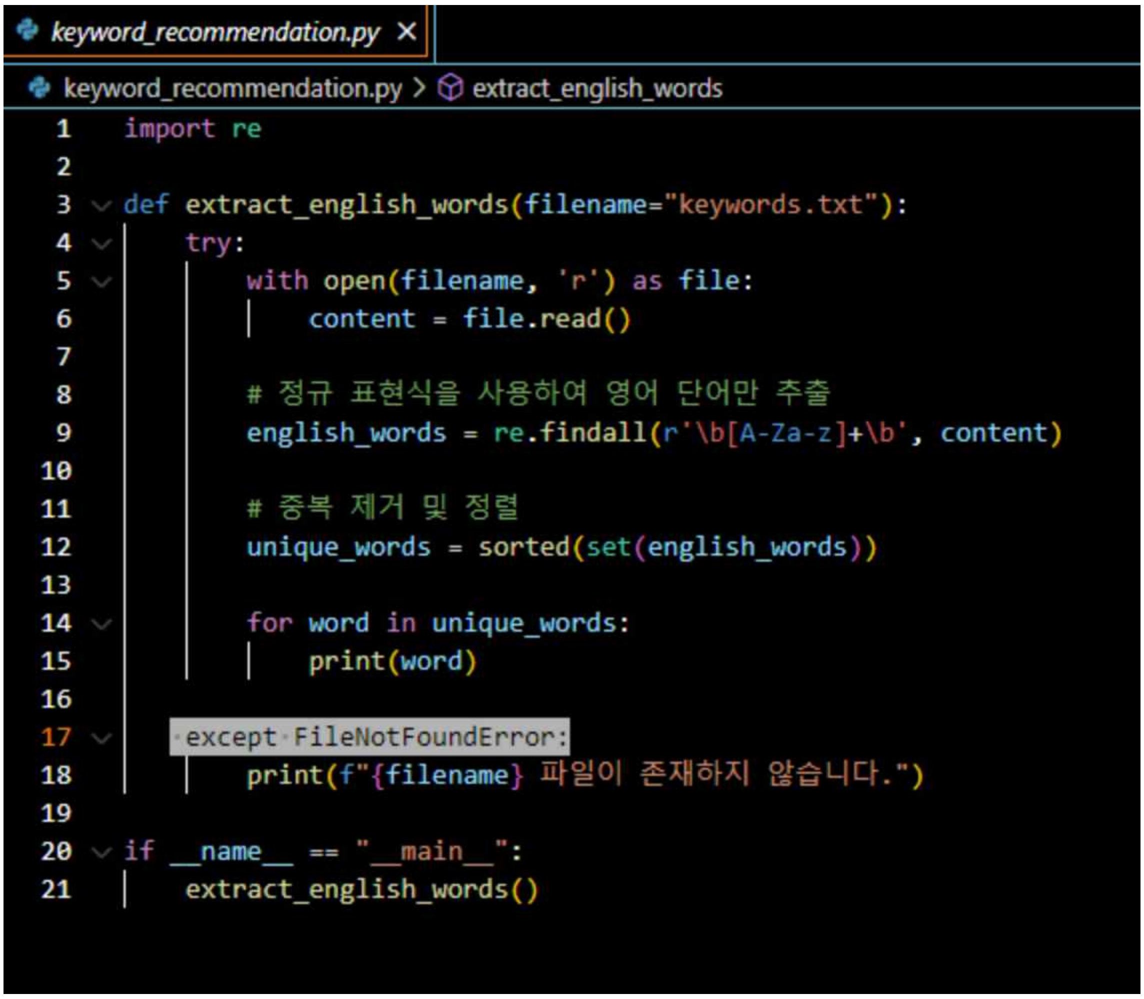
- The contents which we have learned.

```
try:
exceptFileNotFoundError:
```

· Exception handling _ try _ except

If an error occurs while running a program, the program may be stopped. These errors are called exceptions, and if they are not handled properly, an error message may appear to the user or the program may terminate abnormally. Dealing with exceptions is a technique in programming that predicts and provides countermeasures accordingly.

- Code screenshot


keyword_recommendation.py

keyword_recommendation.py > extract_english_words

```python
import re

def extract_english_words(filename="keywords.txt"):
    try:
        with open(filename, 'r') as file:
            content = file.read()

        # 정규 표현식을 사용하여 영어 단어만 추출
        english_words = re.findall(r'\b[A-Za-z]+\b', content)

        # 중복 제거 및 정렬
        unique_words = sorted(set(english_words))

        for word in unique_words:
            print(word)

    except FileNotFoundError:
        print(f"{filename} 파일이 존재하지 않습니다.")

if __name__ == "__main__":
    extract_english_words()
```

(4) Design Criteria Search Engine and Search History Retention Function

- input & output

part_1 import part

```
import re
```

```
from collections import Counter
```

re: Module for using functions related to regular expressions.

Counter: A dictionary subclass that counts the number of elements in the container.

part_2 function 1 load_file(filename)

```
def load_file(filename):
    filename = filename.strip('"')
    with open(filename, 'r', encoding='utf-8') as file:
        return file.read()
```

Open the file with the given filename and read and return the contents.

Remove the start and end quotes from the file path.

part_3 function 2 preprocess_text(text)

```
def preprocess_text(text):
    sentences = re.split(r'[.!?]', text)
    sentences = [s.strip() for s in sentences if s.strip()]
    return sentences
```

Split the given text into sentence units.

Remove the front and back spaces of each segmented sentence.

part_4 function 3 compute_similarity(keyword,sentence)

```
def compute_similarity(keyword, sentence):
    keyword_count = Counter(keyword.split())
    sentence_count = Counter(sentence.split())
    ...
```

Calculate the similarity between the given keywords and sentences.

Count the frequency of each word to calculate the proportion of common words.

part_5 function 4 save_keyword(keyword)

```
def save_keyword(keyword):
    with open('keywords.txt', 'a+', encoding='utf-8') as file:
        ...
```

Save the given keyword and its frequency in the keywords.txt file.
For keywords that already exist, update the frequency.

part_6 function 5 main

```
def main():
    ...
    text = load_file(filename)
    ...
    for rank, (similarity, sentence) in enumerate(top_10_sentences, start=1):
        print(f"{rank}. {similarity:.2f}: {sentence}")
```

Enter the file path and keyword from the user.
Read the contents of the entered file, divide it into sentences, and calculate the similarity with the keywords in each sentence.
Outputs the top 10 sentences based on the calculated similarity.

part_7 Main Execution Code

```
if __name__ == "__main__":
    main()
```

Be sure to call the main function only when the script runs directly.

- Explanation

This function serves to analyze text files containing expertise related to roads, such as classification according to the role of roads and legal classification of roads, and derive results. When the user enters a search word, the most relevant sentences corresponding to the keyword are output up to 10 sentences. Through this, professional knowledge is easily and quickly provided to users only with keywords.

- The contents which we have learned.

```
withopen(filename, 'r', encoding='utf-8') asfile:
```

```
withopen('keywords.txt', 'a+', encoding='utf-8') asfile:
```

· File input and output

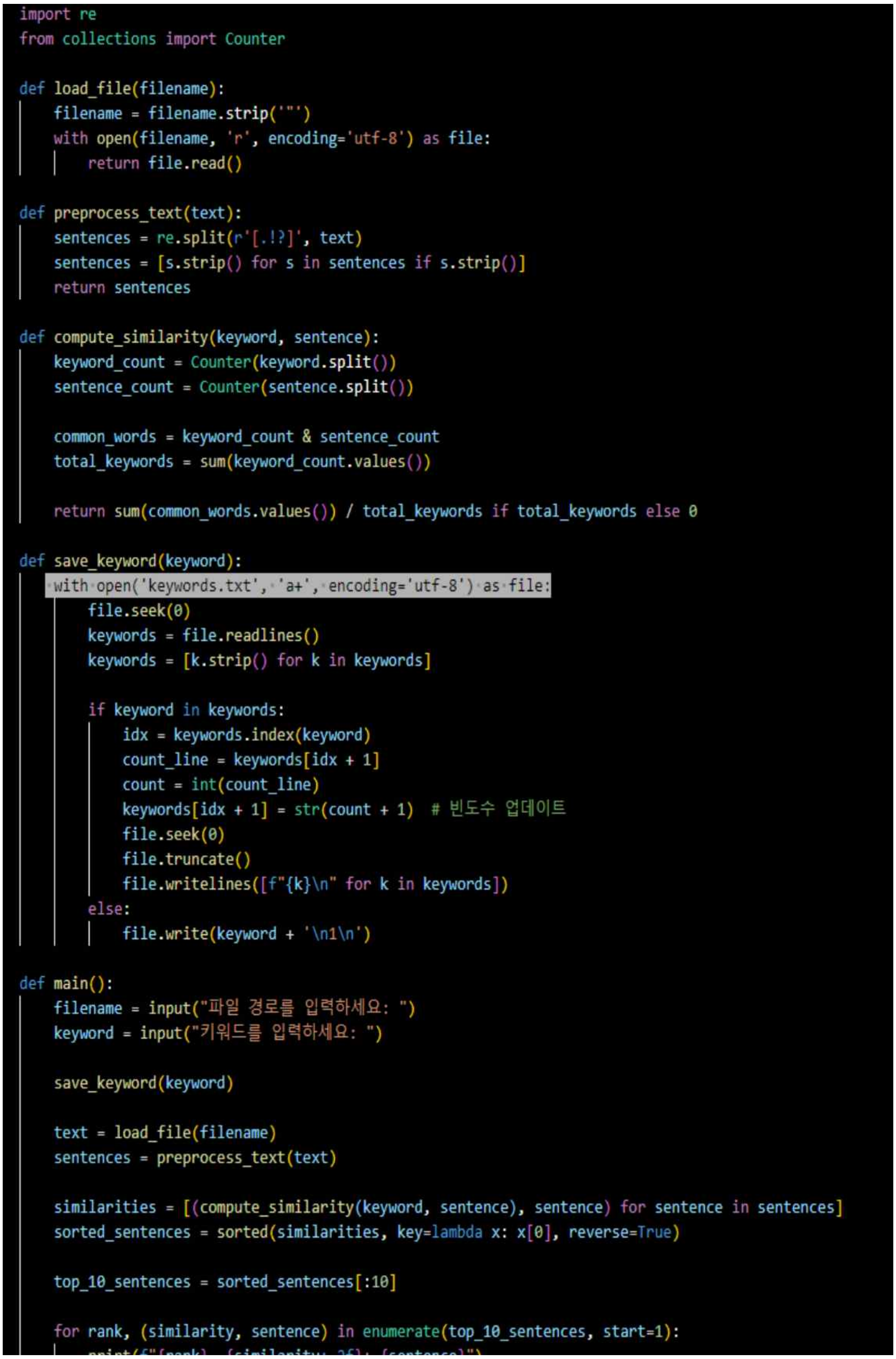
In Python, file input/output refers to the action of storing data in a file or reading data from a file. Python provides a variety of built-in functions and methods that allow you to simply and intuitively perform these file input/output.

· File mode

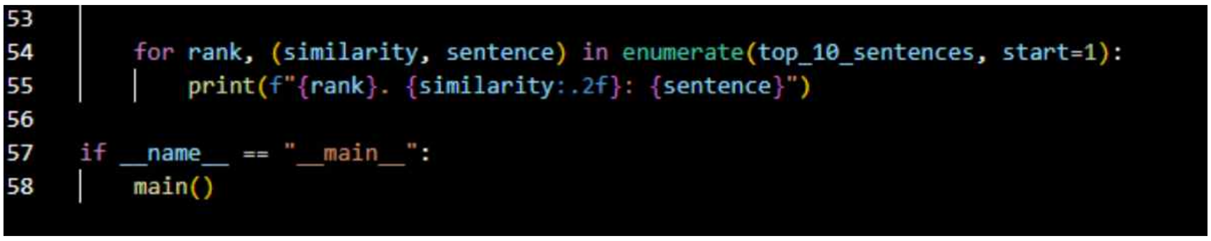
There are several modes that you use to perform file input/output operations.

r : Read mode (default). Open the file as read-only.
w : Write mode. Open the file as write-only, and it will be overwritten if the file already exists. If the file does not exist, create a new one.
a : Additional mode. Open the file for write-only, and add content to the end of the file.
b : Binary mode. Open the file in binary mode. For example, it is used to handle binary data such as images and music files.
+ : Read/write mode. Opens the file in read and write mode.

- Code screenshot



```
import re
from collections import Counter

def load_file(filename):
    filename = filename.strip('"')
    with open(filename, 'r', encoding='utf-8') as file:
        return file.read()

def preprocess_text(text):
    sentences = re.split(r'[.!?]', text)
    sentences = [s.strip() for s in sentences if s.strip()]
    return sentences

def compute_similarity(keyword, sentence):
    keyword_count = Counter(keyword.split())
    sentence_count = Counter(sentence.split())

    common_words = keyword_count & sentence_count
    total_keywords = sum(keyword_count.values())

    return sum(common_words.values()) / total_keywords if total_keywords else 0

def save_keyword(keyword):
    with open('keywords.txt', 'a+', encoding='utf-8') as file:
        file.seek(0)
        keywords = file.readlines()
        keywords = [k.strip() for k in keywords]

        if keyword in keywords:
            idx = keywords.index(keyword)
            count_line = keywords[idx + 1]
            count = int(count_line)
            keywords[idx + 1] = str(count + 1)  # 빈도수 업데이트
            file.seek(0)
            file.truncate()
            file.writelines([f"{k}\n" for k in keywords])
        else:
            file.write(keyword + '\n1\n')

def main():
    filename = input("파일 경로를 입력하세요: ")
    keyword = input("키워드를 입력하세요: ")

    save_keyword(keyword)

    text = load_file(filename)
    sentences = preprocess_text(text)

    similarities = [(compute_similarity(keyword, sentence), sentence) for sentence in sentences]
    sorted_sentences = sorted(similarities, key=lambda x: x[0], reverse=True)

    top_10_sentences = sorted_sentences[:10]

    for rank, (similarity, sentence) in enumerate(top_10_sentences, start=1):
        print(f"{rank}. {similarity:.2f}: {sentence}")
```



```

    for rank, (similarity, sentence) in enumerate(top_10_sentences, start=1):
        print(f"{rank}. {similarity:.2f}: {sentence}")

if __name__ == "__main__":
    main()
```

(5) Image Search Engine

- input & output

part_1 Module and package import

```
import os
```

```
from PIL import Image, ImageTk
```

```
from tkinter import Tk, Label, Entry, Button, StringVar, filedialog, Canvas, Frame, Scrollbar
```

os: Module for file and directory related operations.
PIL: Module for image processing. (As part of the Pillow library, it provides image processing and conversion capabilities.)
tkinter: a standard Python library for creating a graphical user interface (GUI).

part_2 Global variables and initialization

```
root = None
```

```
image_list = []
```

```
matching_images = []
```

```
images_folder = ""
```

```
canvas = None
```

root: The initiative of the GUI.
image_list: A list that stores PhotoImage objects in an image.
matching_images: A list of image files that match the keyword.

images_folder: The path to the folder where the image files are stored.
canvas: GUI elements for displaying images.

part_3 Function 1 show_images_with_keyword(keyword)

```
def show_images_with_keyword(keyword):
    ...
```

Find the matching image files according to the keywords and display them in the GUI.

part_4 Function 2 on_closing()

```
def on_closing():
    ...
```

A function that runs at the end of a program that releases the image objects used from memory and shuts down the GUI.

part_5 Function 3 get_keyword_and_show_images(root)

```
def get_keyword_and_show_images(root):
    ...
```

It receives a keyword from the user and displays images for that keyword on the GUI.

part_6 Function 4 get_root()

```
def get_root():
    ...
```

Returns the root object of the current GUI.

part_7 Function 5 main()

```
def main():
    ...
```

The main execution part of the program. Initialize the GUI and start the main loop.

part_8 Main Execution Code

```
if __name__ == "__main__":
    main()
```

Call the main function when the script runs directly.

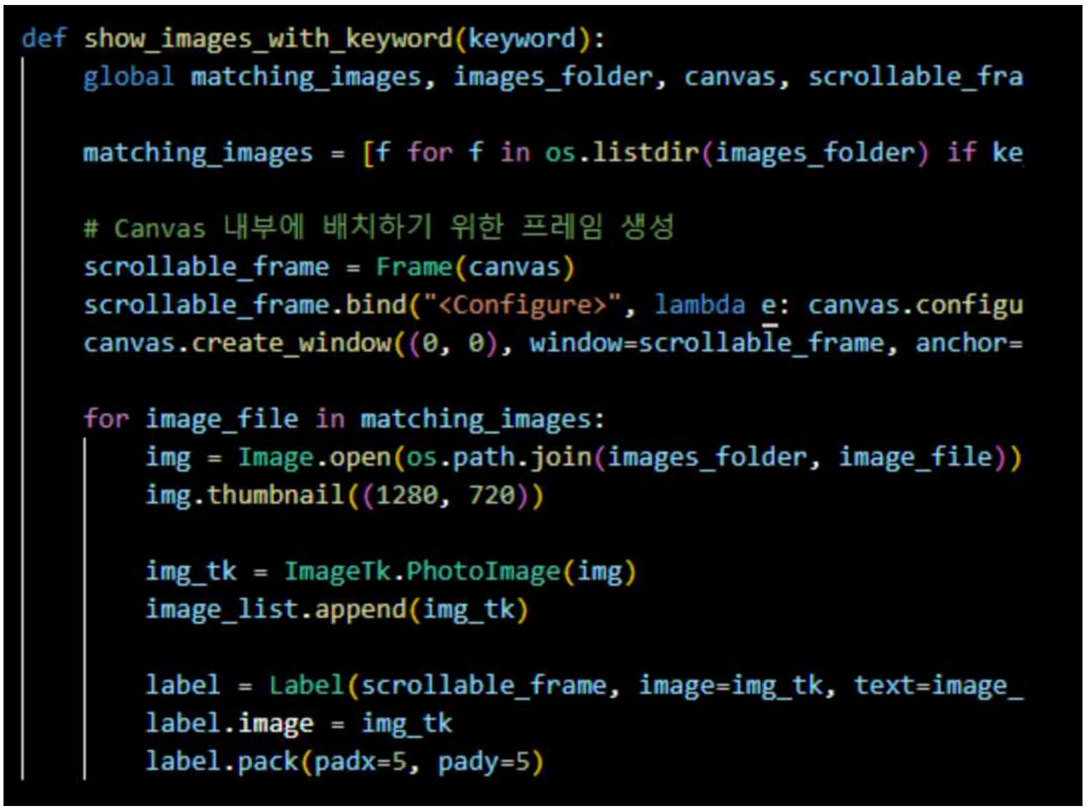
- Explanation

Professional content, such as the classification of roads by road laws and regulations related to the road or by classification according to the function and role of the road, is difficult to understand only through writing. Therefore, there are many schematic data, charts, and tables, and images must be used to efficiently utilize them and to be understood by users. For this part, if you search for keywords, you will see related photos. In particular, since the contents of the tunnel method require a lot of visual data, data were prepared with an emphasis on that part.

Recommended Keywords : NATM

NATM is one of the construction methods of tunnels, and many processes and machines are used. We have prepared a lot of data on this, so please check it.

- The contents which we have learned.



```
def show_images_with_keyword(keyword):
    global matching_images, images_folder, canvas, scrollable_fra

    matching_images = [f for f in os.listdir(images_folder) if ke

    # Canvas 내부에 배치하기 위한 프레임 생성
    scrollable_frame = Frame(canvas)
    scrollable_frame.bind("<Configure>", lambda e: canvas.configu
    canvas.create_window((0, 0), window=scrollable_frame, anchor=

    for image_file in matching_images:
        img = Image.open(os.path.join(images_folder, image_file))
        img.thumbnail((1280, 720))

        img_tk = ImageTk.PhotoImage(img)
        image_list.append(img_tk)

        label = Label(scrollable_frame, image=img_tk, text=image_
        label.image = img_tk
        label.pack(padx=5, pady=5)
```

· function

A function is a block of code that performs a specific task. Using a function can increase the reusability of the code and improve the readability of the code. function is defined "def_function's name(parameter1,parameter2,...)". After you define a function, you can call it using the function name and the required parameters (factors).

· for sentences

In Python, the for statement is one of the control statements used for iterations. The for statement performs an action by sequentially accessing each element of a rotatable object.

```
for variables in iterable_object:
    # code blocks
```

It can use with range() function. The range() function is used to generate consecutive integers and is often used with for statements.

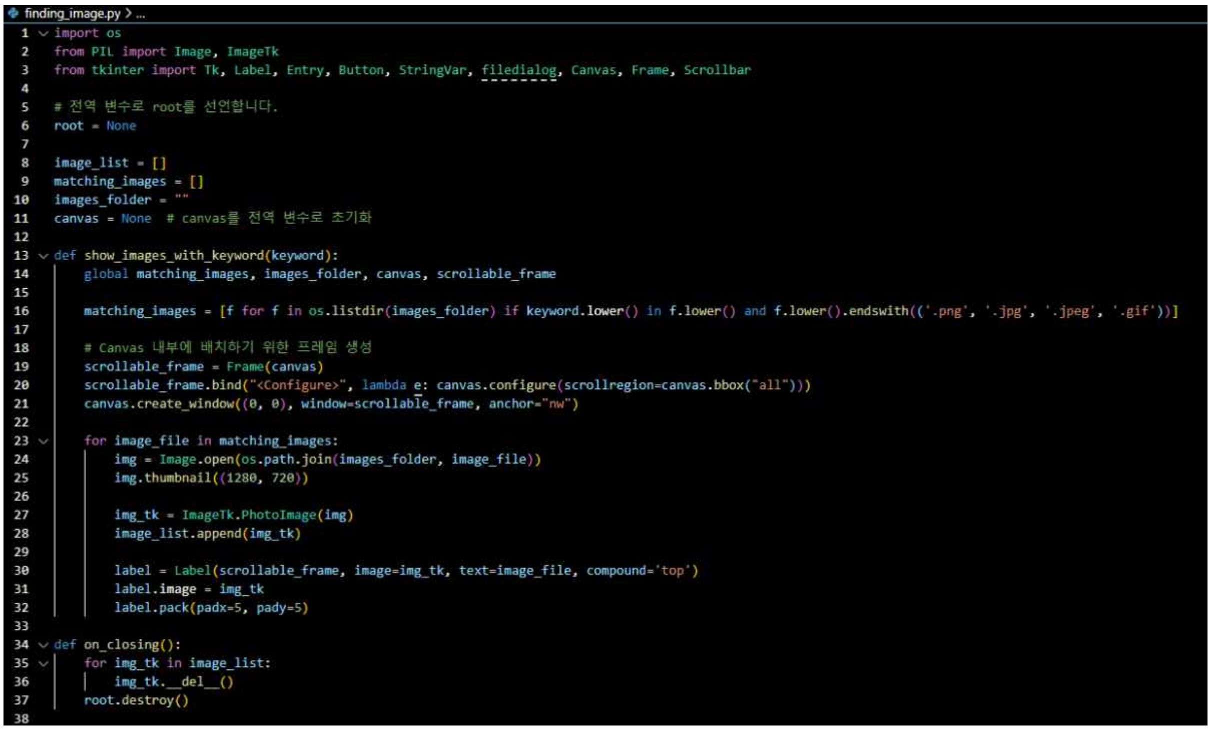
· For-else syntax:

When the for loop is complete, the else block is executed. However, when the for loop ends through a break statement in the middle, the else block is not executed.

· Nested for statements:

A for statement can be superimposed inside another for statement.

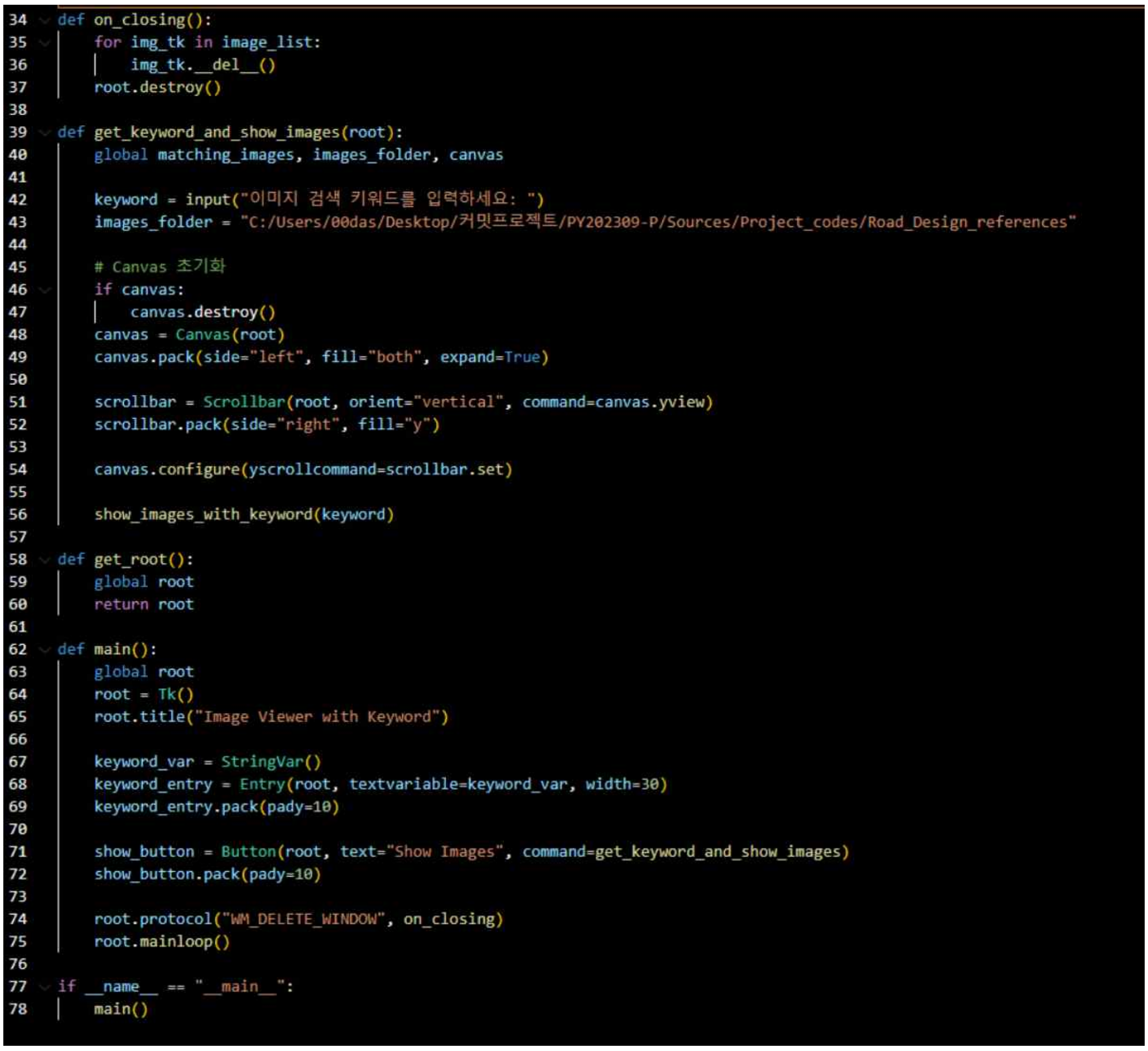
- Code screenshot_1



finding_image.py > ...

```python
import os
from PIL import Image, ImageTk
from tkinter import Tk, Label, Entry, Button, StringVar, filedialog, Canvas, Frame, Scrollbar

# 전역 변수로 root를 선언합니다.
root = None

image_list = []
matching_images = []
images_folder = ""
canvas = None  # canvas를 전역 변수로 초기화

def show_images_with_keyword(keyword):
    global matching_images, images_folder, canvas, scrollable_frame

    matching_images = [f for f in os.listdir(images_folder) if keyword.lower() in f.lower() and f.lower().endswith(('.png', '.jpg', '.jpeg', '.gif'))]

    # Canvas 내부에 배치하기 위한 프레임 생성
    scrollable_frame = Frame(canvas)
    scrollable_frame.bind("<Configure>", lambda e: canvas.configure(scrollregion=canvas.bbox("all")))
    canvas.create_window((0, 0), window=scrollable_frame, anchor="nw")

    for image_file in matching_images:
        img = Image.open(os.path.join(images_folder, image_file))
        img.thumbnail((1280, 720))

        img_tk = ImageTk.PhotoImage(img)
        image_list.append(img_tk)

        label = Label(scrollable_frame, image=img_tk, text=image_file, compound='top')
        label.image = img_tk
        label.pack(padx=5, pady=5)

def on_closing():
    for img_tk in image_list:
        img_tk.__del__()
    root.destroy()

```

- Code screenshot_2



```python
def on_closing():
    for img_tk in image_list:
        img_tk.__del__()
    root.destroy()

def get_keyword_and_show_images(root):
    global matching_images, images_folder, canvas

    keyword = input("이미지 검색 키워드를 입력하세요: ")
    images_folder = "C:/Users/00das/Desktop/커밋프로젝트/PY202309-P/Sources/Project_codes/Road_Design_references"

    # Canvas 초기화
    if canvas:
        canvas.destroy()
    canvas = Canvas(root)
    canvas.pack(side="left", fill="both", expand=True)

    scrollbar = Scrollbar(root, orient="vertical", command=canvas.yview)
    scrollbar.pack(side="right", fill="y")

    canvas.configure(yscrollcommand=scrollbar.set)

    show_images_with_keyword(keyword)

def get_root():
    global root
    return root

def main():
    global root
    root = Tk()
    root.title("Image Viewer with Keyword")

    keyword_var = StringVar()
    keyword_entry = Entry(root, textvariable=keyword_var, width=30)
    keyword_entry.pack(pady=10)

    show_button = Button(root, text="Show Images", command=get_keyword_and_show_images)
    show_button.pack(pady=10)

    root.protocol("WM_DELETE_WINDOW", on_closing)
    root.mainloop()

if __name__ == "__main__":
    main()
```

4. Test Result

(1) Design Criteria Search Engine

- Explanation

1. To use the search engine function, enter '1' after executing the main program.

2. Since this function is a search function within a file, enter the path of the text file you want to search for.

Copy and paste the path and it will be done automatically.

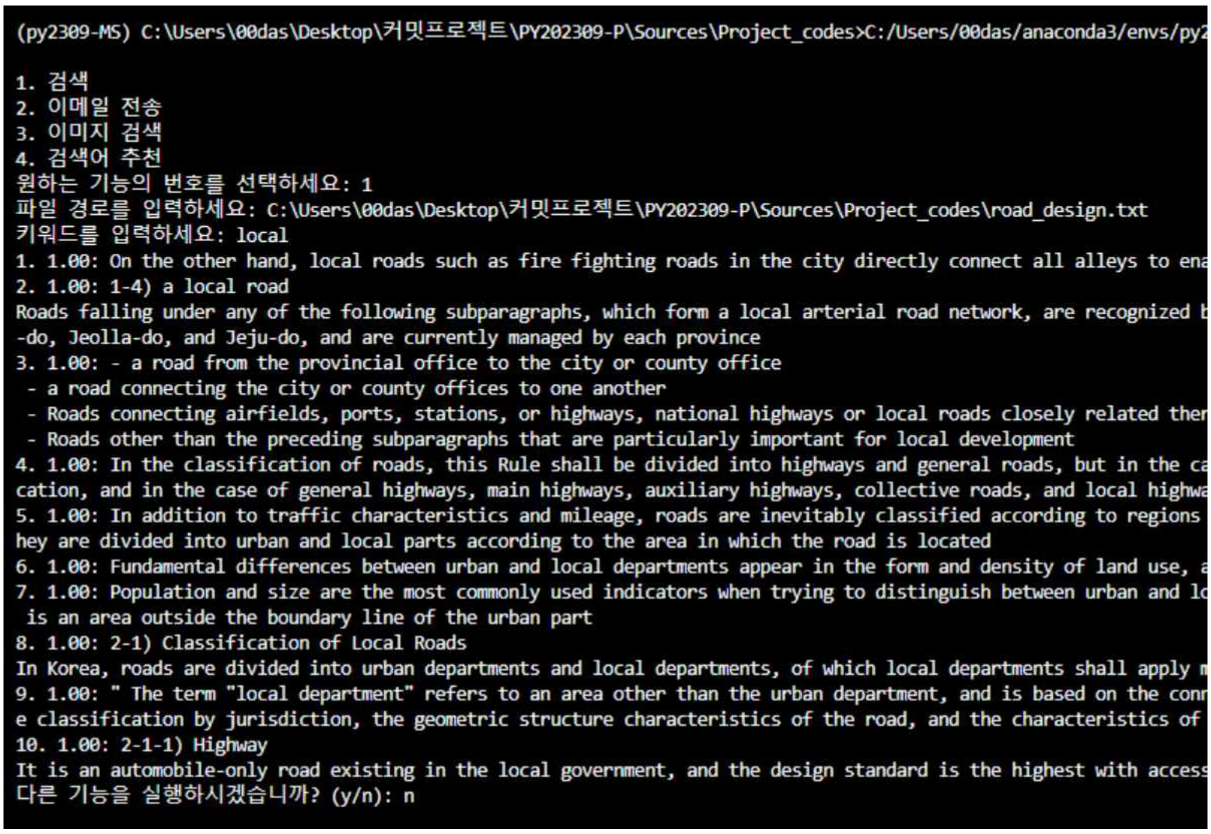
3. After entering the path, enter the keyword you want to find. Enter English words and enter 'local' to test.

4. Then, 10 sentences containing 'local' are output in the order of high relevance.

5. After that, if you want to run more other functions, enter 'y', and if you want to stop, enter 'n'

6. If y is entered, a restart menu is output, and if 'n' is entered, the program ends.

- Screenshot of the test result



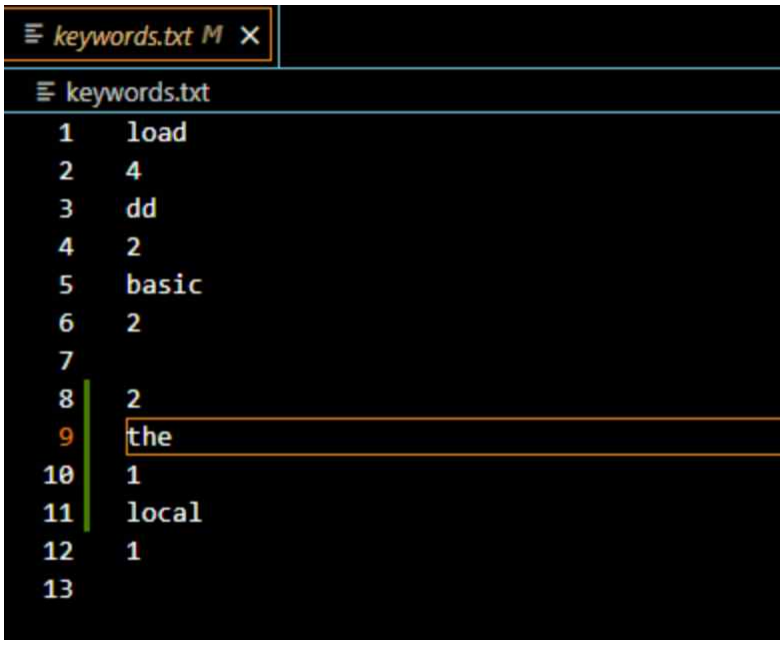
```
(py2309-MS) C:\Users\00das\Desktop\커밋프로젝트\PY202309-P\Sources\Project_codes>C:/Users/00das/anaconda3/envs/py2

1. 검색
2. 이메일 전송
3. 이미지 검색
4. 검색어 추천
원하는 기능의 번호를 선택하세요: 1
파일 경로를 입력하세요: C:\Users\00das\Desktop\커밋프로젝트\PY202309-P\Sources\Project_codes\road_design.txt
키워드를 입력하세요: local
1. 1.00: On the other hand, local roads such as fire fighting roads in the city directly connect all alleys to ena
2. 1.00: 1-4) a local road
Roads falling under any of the following subparagraphs, which form a local arterial road network, are recognized b
-do, Jeolla-do, and Jeju-do, and are currently managed by each province
3. 1.00: - a road from the provincial office to the city or county office
 - a road connecting the city or county offices to one another
 - Roads connecting airfields, ports, stations, or highways, national highways or local roads closely related ther
 - Roads other than the preceding subparagraphs that are particularly important for local development
4. 1.00: In the classification of roads, this Rule shall be divided into highways and general roads, but in the ca
cation, and in the case of general highways, main highways, auxiliary highways, collective roads, and local highwa
5. 1.00: In addition to traffic characteristics and mileage, roads are inevitably classified according to regions
hey are divided into urban and local parts according to the area in which the road is located
6. 1.00: Fundamental differences between urban and local departments appear in the form and density of land use, a
7. 1.00: Population and size are the most commonly used indicators when trying to distinguish between urban and lo
 is an area outside the boundary line of the urban part
8. 1.00: 2-1) Classification of Local Roads
In Korea, roads are divided into urban departments and local departments, of which local departments shall apply m
9. 1.00: " The term "local department" refers to an area other than the urban department, and is based on the conn
e classification by jurisdiction, the geometric structure characteristics of the road, and the characteristics of
10. 1.00: 2-1-1) Highway
It is an automobile-only road existing in the local government, and the design standard is the highest with access
다른 기능을 실행하시겠습니까? (y/n): n
```

(2) Search History Retention Function

- Explanation

This function stores search history so that you can check search terms searched through the search engine later. Search terms are sorted and stored from the top in the order of high frequency of search in "keywords.txt". Records are provided to the user in the form of text files so that I can know what contents I have searched for later.

- Screenshot of the test result



```
load
4
dd
2
basic
2

2
the
1
local
1

```

(3) Recommendation of the searching keywords

- Explanation

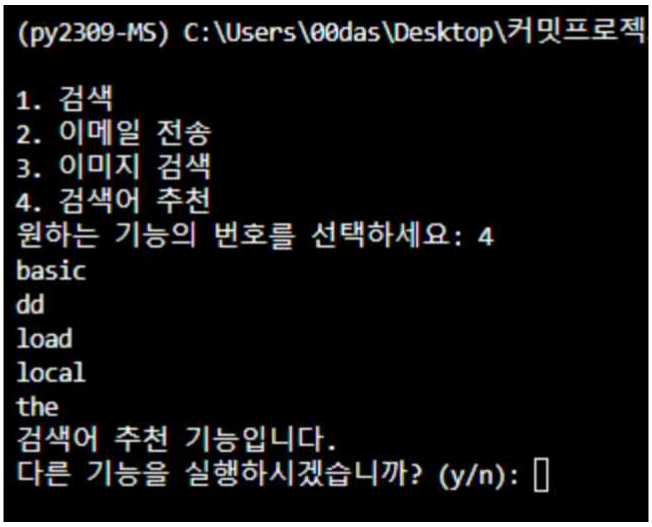
This function can be a guideline for users who use this program for the first time, and it is a function that can guess the user's convenience and disposition. It stores frequently searched search terms and provides recommended keywords by outputting them from the order of high frequency.

1. Run main.py to enter the Start menu.

2. Enter "4" in the number of the desired function.
3. Check the recommended keywords.

4. Enter y or n to decide whether to use or end the program further.

- Screenshot of the test result



```
(py2309-MS) C:\Users\00das\Desktop\커밋프로젝

1. 검색
2. 이메일 전송
3. 이미지 검색
4. 검색어 추천
원하는 기능의 번호를 선택하세요: 4
basic
dd
load
local
the
검색어 추천 기능입니다.
다른 기능을 실행하시겠습니까? (y/n): 
```

(4) Image Search Engine

- Explanation

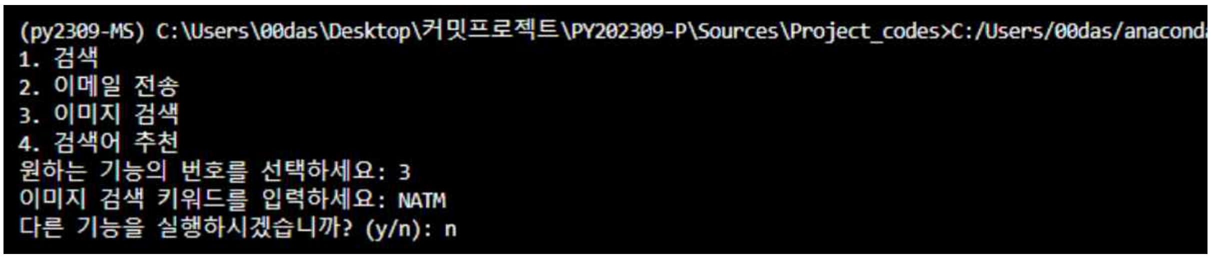
This function provides content that is difficult to understand in writing, such as tunnel excavation method or road construction method, through an easy-to-understand image file. In addition, specialized knowledge such as classification according to the function and role of the road can be searched and utilized through visual data that are easy to see at a glance. When a keyword is entered, related visual data is provided to the user. When there are several related visual data, convenience has been added so that multiple data can be easily compared through scrolling.

1. Run main.py to enter the Start menu.
2. Enter '3' in the number of the desired function.
3. In the image search keyword, type the keyword you want to search for, such as "NATM".

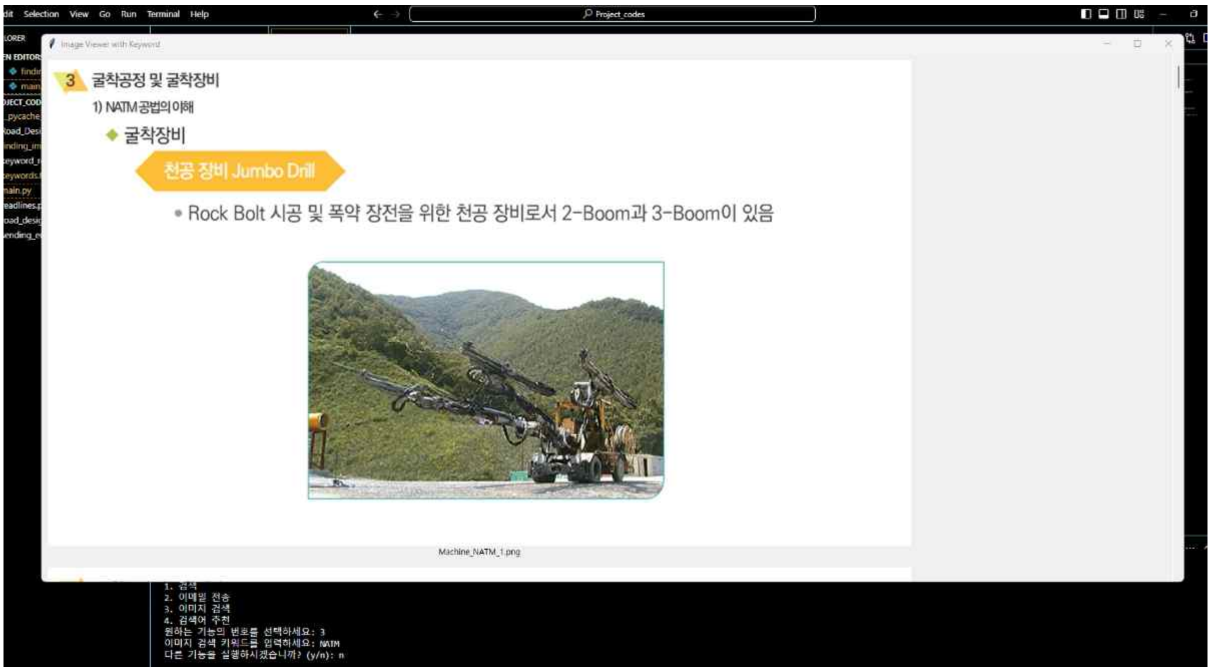
4. The "feather" image upload window will appear on the taskbar. That's the "Tkinter window."
5. When you enter the created window, you can check the related images.
6. When you exit the window, the program ends, and you select through y, n whether to continue using the program.

Recommended Keywords: NATM

- Screenshot of the test result



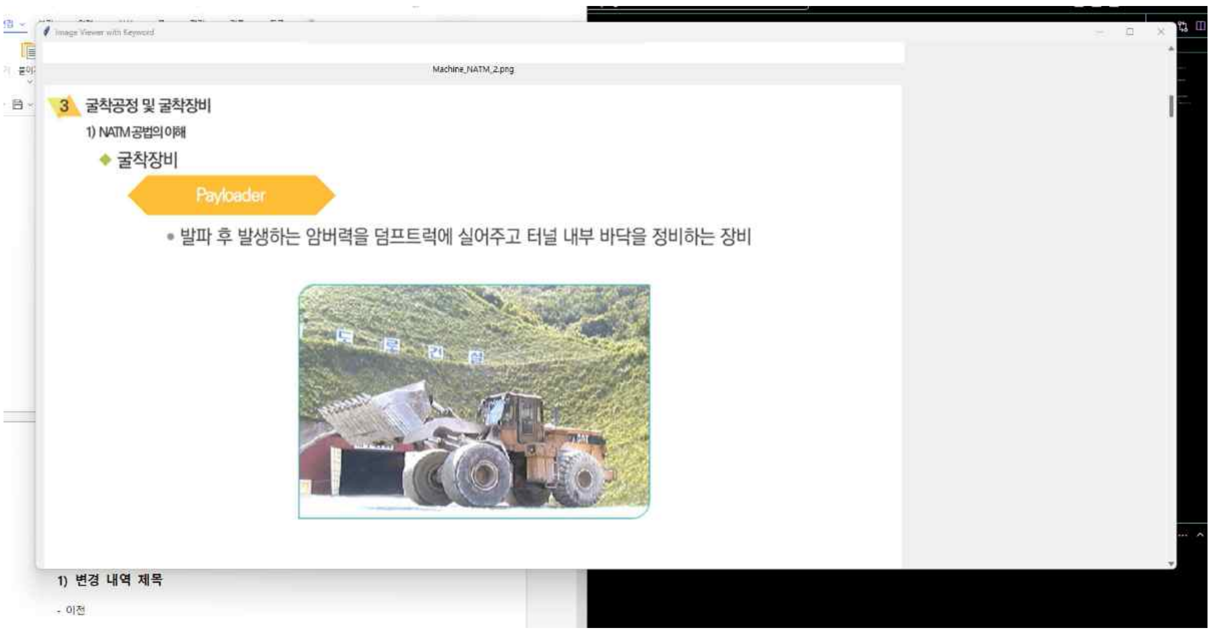
```
(py2309-MS) C:\Users\00das\Desktop\커밋프로젝트\PY202309-P\Sources\Project_codes>C:/Users/00das/anacond
1. 검색
2. 이메일 전송
3. 이미지 검색
4. 검색어 추천
원하는 기능의 번호를 선택하세요: 3
이미지 검색 키워드를 입력하세요: NATM
다른 기능을 실행하시겠습니까? (y/n): n
```



• Rock Bolt 시공 및 폭약 장전을 위한 천공 장비로서 2-Boom과 3-Boom이 있음

Machine_NATM_1.png

```
1. 검색
2. 이메일 전송
3. 이미지 검색
4. 검색어 추천
원하는 기능의 번호를 선택하세요: 3
이미지 검색 키워드를 입력하세요: NATM
다른 기능을 실행하시겠습니까? (y/n): n
```



3 굴착공정 및 굴착장비

1) NATM공법의이해

◆ 굴착장비

Payloader

• 발파 후 발생하는 암버력을 덤프트럭에 실어주고 터널 내부 바닥을 정비하는 장비

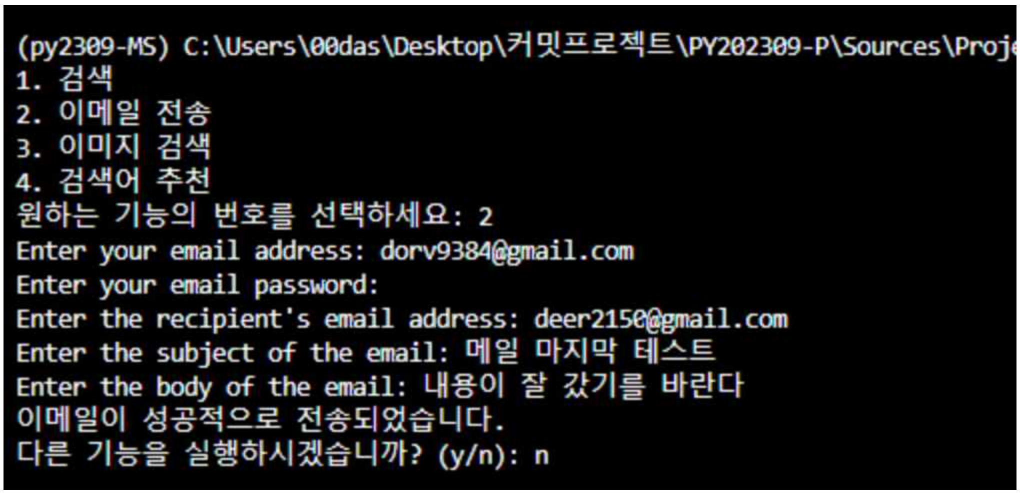
You can see more pictures by holding and lowering the scroll on the right side of the screen.

(5) Data mail transfer function

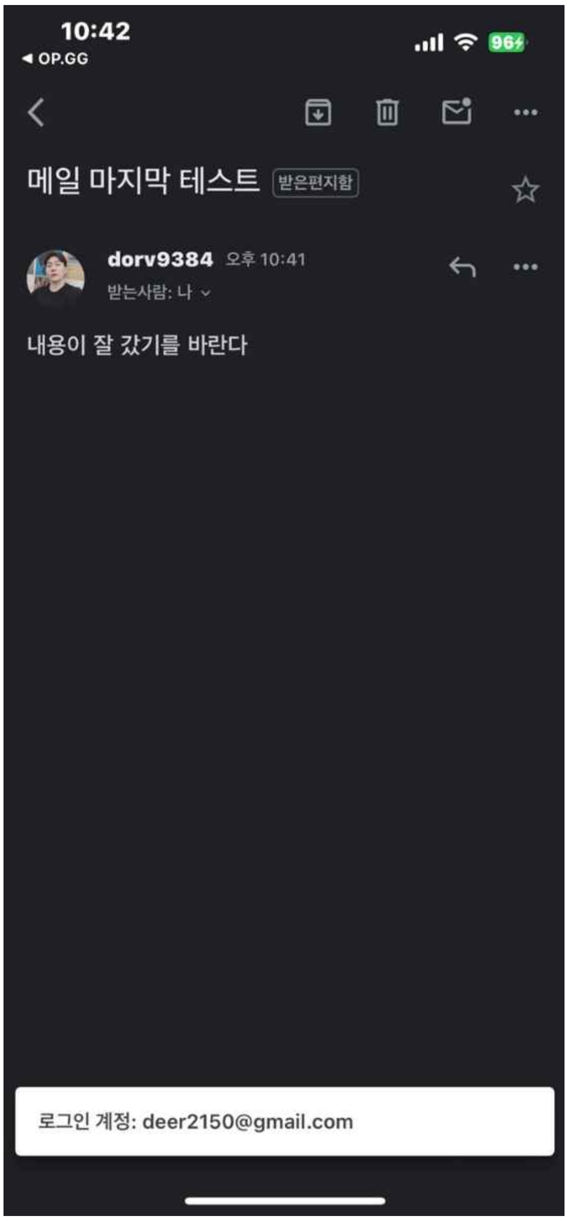
- Explanation

This function is to send knowledge and information obtained through programs such as search engines by e-mail. Only gmail is available in this process and an app password is required. For more information, please refer to the function description above.

- Screenshot of the test result



```
(py2309-MS) C:\Users\00das\Desktop\커밋프로젝트\PY202309-P\Sources\Proj
1. 검색
2. 이메일 전송
3. 이미지 검색
4. 검색어 추천
원하는 기능의 번호를 선택하세요: 2
Enter your email address: dorv9384@gmail.com
Enter your email password:
Enter the recipient's email address: deer2150@gmail.com
Enter the subject of the email: 메일 마지막 테스트
Enter the body of the email: 내용이 잘 갔기를 바란다
이메일이 성공적으로 전송되었습니다.
다른 기능을 실행하시겠습니까? (y/n): n
```



10:42
◀ OP.GG

메일 마지막 테스트 받은편지함

dorv9384 오후 10:41
받는사람: 나

내용이 잘 갔기를 바란다

로그인 계정: deer2150@gmail.com

5. Changes in Comparison to the Plan

1) Add function

- before

Recommendation of the searching keywords was not included in our program.

- after

Recommendation of the searching keywords has been added to our program.

- reason

I added the ability to save search history to this program, but I added it because I thought that the ability to create second information using stored data, not just storing it, would be innovative and very helpful to users. And I thought I could solve it with my own technology.

6. Lessons Learned & Feedback

Through this project, I was able to have a good experience not only learning Python's grammar, but also using it in real life. And while carrying out project-type tasks, I was able to gain a deep understanding of the process of developing the program. Through the process of solving problems around us through programming, I was able to develop a new way of thinking. And through the process of creating new results by combining programming that I had never encountered with my major before, I felt really rewarding and confident that I had my own competitiveness. The overall contents and materials of the class were satisfactory, and it was not easy for me as a non-major to follow the class, but I was able to come this far thanks to the professor's kind explanation. The one thing that was disappointing was that the method of calculating the midterm score result by summing the assistant's score and the professor's score was unfortunate. Most of the questions were asked by the professor in class, and the method of setting up tests and calculating scores was also in line with the professor's tendency, but it was a little disappointing that the assistant's evaluation came in. The difference in rank before and after the sum was also very different, so I was embarrassed and thought that the equity was not right.

And what was very disappointing about this project was that all the functional units of my program were difficult to write, so I couldn't upload the commit frequently due to errors and the process of correcting, correcting, and correcting. It was a shame that I lacked confidence in the function I implemented.