Python Programming and Practice

Development of Jeju Island Travel Course Recommendation Program

Final Report

Date : 2023.12.23.
Name : 이영채
ID : 213954

1. Introduction

1) Background

Since February 2023, the number of tourists visiting domestic tourist destinations has been steadily recovering as the COVID-19 situation has gradually improved. Among domestic tourist destinations, Jeju Island is particularly visited by tourists from before COVID-19 to the present.

However, due to the large area of Jeju Island than tourists think, it is difficult to determine the route efficiently, it takes time to find and decide on tourist attractions, and problems related to the distance and time from accommodation facilities and airports to tourist attractions, tourists are constantly complaining of difficulties in deciding the travel course from the beginning. In order to solve this problem, we intend to design a program that recommends Jeju Island's travel course to provide efficient travel routes and satisfactory travel experiences for Jeju Island tourists.

2) Project goal

The project aims to classify Jeju Island into four major areas and design a program that recommends Jeju Island travel courses according to the area when tourists choose the area they want to travel.

3) Differences from existing programs

The existing travel course recommendation program works in such a way that when a user enters a predetermined departure point and destination, the optimal visit order between the destinations is determined. Such a program can be said to be more suitable for users with clear destinations to visit. However, the program to be designed here is different from existing programs because it recommends courses to users who have not decided on a destination to visit by selecting several famous tourist attractions and recently emerging places.

In addition, Naver's "Jeju worth visiting" recommendation service provides users with only the places of famous tourist destinations in Jeju Island by categorizing them. This method is specialized in browsing various places, but there is a hassle that the user must make his or her own course in consideration of the location or distance of the place. However, in order to solve this hassle, this program can provide users with the optimal travel course derived after selecting tourist destinations by category.

2. Functional Requirement

1) Classification and selection of desired travel region

- The functionality involves classifying Jeju Island into your regions and then allowing users to select the desired region for their travel.

(1) Classification of Jeju Island into four regions

- Jeju Island's various areas are broadly classified into four regions: Western region, Jeju City, Seogwipo City, and Eastern region.

(2) User selection of region

- The user receives one of the four areas. The reference points of the tour course are arbitrarily set within the four zones.

(3) Display area details

- If the user desires information about a specific region, the system presents details about the region the user has chosen.

2) Classifying places within Jeju Island by category

- A feature that categorizes places into categories such as tourist attractions, cafes, restaurants, cultural facilities, etc

(1) Collection and classification of Jeju Island tourism data

- Utilizing the Kakao Maps API, the data of places within Jeju Island is crawled, preprocessed, and categorized into groups such as cafes, restaurants, tourist attractions, and more.

3) Providing a travel course in the selected area

- The function of deriving a travel course according to the area selected by the user and providing it to the user.

(1) Selection of travel courses

- Based on the designated point within the selected region, a one-day tour course is selected using the tourist destination data classified above.

(2) Provide a travel course to the user

- The selected travel course is provided to the user. If the user wants a different course, it provides a different travel course in the same area.

3. Implementation

(1) Classification of Jeju Island into four regions

- input : information for 4 regions

- output : List containing information for 4 regions

- Jeju Island into 4 regions and store the administrative districts included in each region in a list.

- Applied knowledge : Variable, List

- Code screenshot (jeju_region.py)

```python
# 구역별 분류
# 1구역 - 제주시
jeju1 = ['일도일동', '일도이동', '이도일동', '이도이동', '삼도일동',
         '삼도이동', '건입동', '용담일동', '용담이동', '도두동', '이호동',
         '외도동', '노형동', '연동', '오라동', '아라동', '화북동', '삼양동',
         '봉개동']

# 2구역 - 제주서부
jeju2 = ['애월읍', '한림읍', '한경면', '대정읍', '안덕면']

# 3구역 - 서귀포시
jeju3 = ['중앙동', '정방동', '천지동', '예래동', '중문동', '대천동',
         '대륜동', '서홍동', '동홍동', '영천동', '송산동', '효돈동']

# 4구역 - 제주동부
jeju4 = ['조천읍', '구좌읍', '남원읍', '표선면', '성산읍', '구좌읍', '우도면']
```

(2) User selection of region

- input : Number of the desired travel region

- output : Message confirming the selection of the region

- When the user selected the number of the desired travel region, a message confirming the completion of the region selection is displayed.

- Applied knowledge : Variable, Loop, Condition, Exception handling, Module, Function

- Code screenshot (main.py)

```python
try:
    import jeju_region as rg
    import course_recommendation as cr

except ModuleNotFoundError:
    print("실행에 필요한 모듈이 설치되어 있지 않습니다. pandas, random, geopy 모듈을 설치한 후 실행해주세요.")
    exit(0) # 모듈 오류 메시지 출력 후 프로그램 즉시 종료

# 사용자가 권역을 선택하고 출발지를 입력함
try:

    while True:
        print("========== 제주도 여행 코스 추천 프로그램 ==========")
        print("1. 제주시  2. 제주서부  3. 서귀포시  4. 제주동부")
        print("5. 권역의 세부 지역 정보 설명")
        region_num = int(input("여행을 원하는 권역의 번호를 선택하세요. -> "))

        # 선택한 번호에 맞게 코스 추천 메시지 출력
        if region_num == 1:
            print("\n제주시의 여행 코스를 추천해드릴게요.\n")
            cr.CourseRecommend(region_num) # 코스 추천 함수 호출
            break
        elif region_num == 2:
            print("\n제주서부의 여행 코스를 추천해드릴게요.\n")
            cr.CourseRecommend(region_num) # 코스 추천 함수 호출
            break
        elif region_num == 3:
            print("\n서귀포시의 여행 코스를 추천해드릴게요.\n")
            cr.CourseRecommend(region_num) # 코스 추천 함수 호출
            break
        elif region_num == 4:
            print("\n제주동부의 여행 코스를 추천해드릴게요.\n")
            cr.CourseRecommend(region_num) # 코스 추천 함수 호출
            break
```

(3) Display area details

- input : Number selection of the region for desired information

- output : Administrative district information of the selected region

- When the user selects the number of the region for which they want information, the program displays the administrative district information belonging to the selected region.

- Applied knowledge : Variable, Condition, Module, Exception handling

- Code screenshot (main.py)

```
        # 세부 지역 정보 표시하기(각 권역 별 동 정보)
        elif region_num == 5:
            region_info = int(input("어느 권역의 세부 지역 정보가 필요하신가요?(숫자 입력) "))
            print()
            if region_info == 1:
                print("1. 제주시 권역의 세부 지역 정보는 다음과 같습니다.")
                print(rg.jeju1, "\n")
                continue
            elif region_info == 2:
                print("2. 제주서부 권역의 세부 지역 정보는 다음과 같습니다.")
                print(rg.jeju2, "\n")
                continue
            elif region_info == 3:
                print("3. 서귀포시 권역의 세부 지역 정보는 다음과 같습니다.")
                print(rg.jeju3, "\n")
                continue
            elif region_info == 4:
                print("4. 제주동부 권역의 세부 지역 정보는 다음과 같습니다.")
                print(rg.jeju4, "\n")
                continue
            else:
                print("올바른 권역 번호를 입력해주세요.\n")
        # 잘못된 권역 번호 입력 시
        else:
            print("올바른 권역 번호를 입력해주세요.")
            continue

# 오류 발생 시 예외처리
# 파일을 찾을 수 없는 경우 예외처리
except FileNotFoundError:
    print("실행에 필요한 장소 csv 파일이 존재하지 않습니다. 파일을 올바른 경로에 위치시킨 후 실행해주세요.")

# 그외 모든 오류 예외처리
except:
    print("오류가 발생했습니다. 모듈 설치 여부나 파일 존재 여부 등을 확인해주세요.")

```

(4) Collection and classification of Jeju Island tourism data

- input : 1)List containing place keywords, 2)Place csv file derived after place collection

- output : 1)CSV file containing the name, latitude, longitude, address, and website URL of places obtained through place keyword searches, 2)CSV file that categorizes place data according to categories(tourist attractions, cafes, souvenir shops, restaurants) by process 1) into 4 regions

- 1)Use the Kakao Maps API to retrieve results of place keyword searches within Jeju Island.

2) Data in which an address is not written in the place data creates an address and excludes places that do not fit the category. Finally, the preprocessed data is divided into 4 areas and stored as a csv file.

- Applied knowledge : Variable, Condition, Loop, Module, Function, List, Dictionary, File, Pandas

- Code screenshot (region_data.ipynb)

```python
# 라이브러리 불러오기
import requests
import pandas as pd
import numpy as np
import folium # 지도 시각화 라이브러리
from folium.plugins import MiniMap
```

```python
# 장소 데이터를 가져오는 함수
def elec_location(region, page_num):
    url = 'https://dapi.kakao.com/v2/local/search/keyword.json'
    # region에는 원하는 검색어가 들어감
    params = {'query':region, 'page':page_num}
    # 카카오맵 API 키 입력
    headers = {"Authorization":"KakaoAK 976f96e6f22f30bb25f87793cc172f62"}

    places = requests.get(url, params=params, headers=headers).json()['documents']
    total = requests.get(url, params=params, headers=headers).json()['meta']['total_count']
    # 카카오맵 API는 최대 45개의 장소만을 가져올 수 있음
    if total > 45:
        print(total, '개 중 45개 데이터만을 가져왔습니다.')
    else:
        print("모든 데이터를 가져왔습니다.")
    return places

```

```python
# elec_location()에서 얻은 결과값을 하나씩 분리하여 저장
def elec_info(places):
    x = []
    y = []
    stores = []
    road_address = []
    place_url = []
    ID = []
    for place in places:
        x.append(float(place['x'])) # 위도
        y.append(float(place['y'])) # 경도
        stores.append(place['place_name']) # 장소명
        road_address.append(place['road_address_name']) # 장소의 도로명주소
        place_url.append(place['place_url']) # 장소의 사이트 주소
        ID.append(place['id']) # 장소의 ID

    ar = np.array([ID, stores, x, y, road_address, place_url]).T
    # 데이터프레임 형태로 위의 정보들을 저장
    df = pd.DataFrame(ar, columns=['ID', 'stores', 'x', 'y', 'road_address', 'place_url'])
    return df

```

```python
# 여러 개의 키워드를 검색할 때 사용
def keywords(location_name):
    df = None
    for loca in location_name:
        # 1페이지 당 15개의 결과를 가짐. 총 3페이지의 정보만 불러올 수 있음
        for page in range(1,4):
            local_name = elec_location(loca, page)
            local_elec_info = elec_info(local_name)
```

```python
# 지도를 표출하는 함수
def make_map(dfs):
    # 기준지 - 제주국제공항
    m = folium.Map(location=[33.5101562, 126.4861157], zoom_start=12)
    # 미니맵 추가
    minimap = MiniMap()
    m.add_child(minimap)
    # 마커 추가
    for i in range(len(dfs)):
        folium.Marker([dfs['y'][i], dfs['x'][i]],
                      tooltip=dfs['stores'][i],
                      popup=dfs['place_url'][i],).add_to(m)
    return m
```

```python
# 검색을 원하는 키워드를 리스트 형태로 입력
# 관광지
location = ['제주 오름', '제주 테마파크', '제주 식물원', '제주 박물관',
            '제주 해수욕장', '제주 산봉우리', '제주 해안도로']
df = keywords(location)
df = df.drop_duplicates(['ID']) # 중복 결과 제거
df = df.reset_index() # 인덱스 리셋

make_map(df) # 지도로 표시
```

```python
# df의 데이터 확인 및 경도 값을 기준으로 정렬
df_sorted = df.sort_values(by=['road_address'])
display(df_sorted.head())
```

```python
# 데이터를 csv 파일로 저장
df_sorted.to_csv("tour_region.csv", index=False, encoding="cp949")
```

- Code screenshot (region_distribution.ipynb)

```python
import pandas as pd
import make_map # make_map.py
```

Python

```python
# 관광지 데이터의 값들을 미리 나눠둔 구역별로 분할한 후 csv로 저장
# 지도 출력을 위해 구역에 맞게 새롭게 분할한 데이터프레임을 반환함
def distribution(input_file, output_file, region_list):
    # 파일 불러오기
    df = pd.read_csv(input_file, encoding="cp949")
    jeju_n_list = region_list # 제주 구역별 행정동 리스트
    jeju_df = pd.DataFrame(columns=df.columns) # 데이터프레임 생성

    # road_address(도로명주소 컬럼)에서 제주 구역별 행정동이 포함된 데이터 찾기
    for i in jeju_n_list:
        contain = df[df['road_address'].str.contains(i)]
        jeju_df = pd.concat([jeju_df, contain], ignore_index=True)

    # 구역에 맞게 분할한 데이터를 csv파일로 저장
    jeju_df.to_csv(output_file, encoding="cp949", index=False)

    return jeju_df
```

Python

```python
# 1, 3구역에서  2, 4구역과 겹치는 부분을 제외한 후 분할해서 csv로 저장
# 지도 출력을 위해 구역에 맞게 새롭게 분할한 데이터프레임을 반환함
def distribution_remove(input_file, output_file, contain_region, not_contain_list):
    # 파일 불러오기
    df = pd.read_csv(input_file, encoding="cp949")
    contain_list = contain_region # 포함시킬 부분 (제주시 / 서귀포시)
    not_contain_list = not_contain_list # 2,4구역과 겹치는 부분(제외시킬 부분)

    jeju_df = pd.DataFrame(columns=df.columns)

    # contain_region에 해당하는 데이터 추출
    for region in contain_list:
        contain = df[df['road_address'].str.contains(region)]
        jeju_df = pd.concat([jeju_df, contain])

    # not_contain_list에 해당하는 데이터 제외
    for not_contain_region in not_contain_list:
        jeju_df = jeju_df[~jeju_df['road_address'].str.contains(not_contain_region)]

    # 구역에 맞게 분할한 데이터를 csv파일로 저장
    jeju_df.reset_index(drop=True, inplace=True)
    jeju_df.to_csv(output_file, encoding="cp949", index=False)

    return jeju_df
```

Python

```python
# 1구역 - 제주시
# 2,4구역 일부가 제주시에 속하므로 겹칠 수 있기 때문에 2,4구역은 제외해줌

jeju1_contain = ["제주시"]
jeju1_not_contain = ['애월읍', '한림읍', '한경면', '대정읍', '안덕면',
                     '조천읍', '구좌읍', '남원읍', '표선면', '성산읍', '구좌읍', '우도면']

jeju1_tour_df = distribution_remove('tour_region.csv',
                                    'jeju1_tour_region.csv',
                                    jeju1_contain,
                                    jeju1_not_contain)

jeju1_cf_souv_df = distribution_remove('tour_cafe_souvenir.csv',
                                       'jeju1_cafe_souvenir.csv',
                                       jeju1_contain,
                                       jeju1_not_contain)

jeju1_restr_df = distribution_remove('tour_restaurant.csv',
                                     'jeju1_restaurant.csv',
                                     jeju1_contain,
                                     jeju1_not_contain)
```
Python

(The code above is the process of classifying region 1, and region 3 uses the same function as this. For the code of region 3, please refer to the region_distribution.ipynb file.)

```python
# 2구역 - 제주서부
jeju2 = ['애월읍', '한림읍', '한경면', '대정읍', '안덕면']

# 관광지
jeju2_tour_df = distribution("tour_region.csv",
                             "jeju2_tour_region.csv",
                             jeju2)

# 카페/기념품
jeju2_cf_souv_df = distribution("tour_cafe_souvenir.csv",
                                "jeju2_cafe_souvenir.csv",
                                jeju2)

# 음식점
jeju2_restr_df = distribution("tour_restaurant.csv",
                              "jeju2_restaurant.csv",
                              jeju2)
```
Python

(The code above is the process of classifying region 2, and region 4 uses the same function as this. For the code of region 4, please refer to the region_distribution.ipynb file.)

(5) Providing a travel course in the selected area

- input : Area number selected by the user and location data by category for that area csv file

- output : A travel course based on six randomly selected places by category, calculating the distance between places based on location data by region

- Using the tourist destination data classified above, select and print a one-day tour course in your area. It prints out tourist attractions, cafes, souvenir shops, and restaurants to a total of 6 places, 2 each. The places where the accommodations are concentrated in 4

areas were randomly selected as the reference points, and this reference point was selected as the starting point. The course is printed by selecting a place within a 7km radius from the previous place.

- Applied knowledge : Variable, Condition, List, Dictionary, Module, Loop, File, Function, Pandas

- Code screenshot (course_recommendation.py)

```python
import pandas as pd
import random
from geopy.distance import geodesic

# csv파일 읽는 함수
def read_file(filename):
    df = pd.read_csv(filename, encoding='cp949')
    return df

# 위도, 경도를 이용하여 두 지점 간의 거리를 계산하는 함수
def calculate_distance(coord1, coord2):
    return geodesic(coord1, coord2).km

# 여행 코스를 추천해주는 함수
def CourseRecommend(region):
    # 구역 번호에 맞는 구역 장소 파일 이름을 저장함
    if region == 1:
        files = ["jeju1_tour_region.csv",
                 "jeju1_cafe_souvenir.csv",
                 "jeju1_restaurant.csv"]
    elif region == 2:
        files = ["jeju2_tour_region.csv",
                 "jeju2_cafe_souvenir.csv",
                 "jeju2_restaurant.csv"]
    elif region == 3:
        files = ["jeju3_tour_region.csv",
                 "jeju3_cafe_souvenir.csv",
                 "jeju3_restaurant.csv"]
    elif region == 4:
        files = ["jeju4_tour_region.csv",
                 "jeju4_cafe_souvenir.csv",
                 "jeju4_restaurant.csv"]

    courses = {"1": {"data": []},
               "2": {"data": []},
               "3": {"data": []}}
```

```python
    # 파일에서 장소를 불러와 선택
    for idx, file in enumerate(files):
        df = read_file(file) # 파일 읽기

        # csv 파일 안의 위도, 경도 값을 가져옴
        if 'x' in df.columns and 'y' in df.columns:
            coord_list = df.apply(lambda row: (row['y'], row['x']), axis=1).tolist()

            # 파일순으로 하나씩 선택
            selected_index = random.randrange(len(coord_list))
            selected_coord = coord_list[selected_index]

            # 관광지 선택
            place = df.loc[selected_index, 'stores']
            url = df.loc[selected_index, 'place_url'] if 'place_url' in df.columns else "url 없음"
            courses["1"]["data"].append((place, url)) # 장소 추가
```

```
        # 카페 및 기념품 선택
        # 선택된 관광지에서 7km 반경 이내의 장소를 선택
        cafe_souvenir_candidates = [i for i, coord in enumerate(coord_list) if
                                    calculate_distance(selected_coord, coord) ≤ 7 and df.loc[i, 'stores'] not in
                                    [p[0] for p in courses["1"]["data"]]]
        selected_cafe_souvenir = random.choice(cafe_souvenir_candidates) if cafe_souvenir_candidates else None
        if selected_cafe_souvenir is not None:
            place = df.loc[selected_cafe_souvenir, 'stores']
            url = df.loc[selected_cafe_souvenir, 'place_url'] if 'place_url' in df.columns else "url 없음"
            courses["2"]["data"].append((place, url)) # 장소 추가

            # 음식점 선택
            # 선택된 카페 및 기념품에서 7km 반경 이내의 장소를 선택
            restaurant_candidates = [i for i, coord in enumerate(coord_list) if
                                     calculate_distance(coord_list[selected_cafe_souvenir], coord) ≤ 7 and df.loc[
                                        i, 'stores'] not in [p[0] for p in courses["2"]["data"]]]
            selected_restaurant = random.choice(restaurant_candidates) if restaurant_candidates else None
            if selected_restaurant is not None:
                place = df.loc[selected_restaurant, 'stores']
                url = df.loc[selected_restaurant, 'place_url'] if 'place_url' in df.columns else "url 없음"
                courses["3"]["data"].append((place, url)) # 장소 추가
```

```
    # 코스 추천 메시지 출력
    print("================ 여행 코스 추천 ================")
    print(f"선택하신 {region}구역의 하루 여행 코스를 랜덤으로 추천해 드립니다.")
    print("( 1.관광지 > 2.카페 및 기념품 > 3.음식점 > 4.관광지 > 5.카페 및 기념품 > 6.음식점 순서로 추천합니다.)\n")
    print("                    시작")
    print("                    ↓")

    # 선택한 구역별로 출발 기준지를 출력 (출발기준지 선택기준 - 숙박시설이 상대적으로 많은 장소로 선택)
    if region == 1:
        print("- 출발 기준지: 제주국제공항 (https://place.map.kakao.com/10808261)")
    elif region == 2:
        print("- 출발 기준지: 고내포구 (https://place.map.kakao.com/8175875)")
    elif region == 3:
        print("- 출발 기준지: 서귀포버스터미널 (https://place.map.kakao.com/7938158)")
    elif region == 4:
        print("- 출발 기준지: 고성교차로 (https://place.map.kakao.com/15140307)")
    print("                    ↓")

    # 관광지 > 카페 및 기념품 > 음식점 > 관광지 > 카페 및 기념품 > 음식점 순서로 출력
    for category in ["2", "3"]:
        for i in range(3):
            place, url = courses[category]["data"][i]
            print(f"- {place} ({url})") # 장소와 url을 함께 출력
            print("                    ↓")

    print("            하루 여행 코스 끝")
    ContinueRecommend(region) # 코스를 계속 추천받을지 사용자에게 입력받음
```

```
# 코스를 계속 추천받을지 입력받는 함수
def ContinueRecommend(region_num):
    while True:
        diff_course = input("\n다른 코스를 추천받으시겠어요? (Y / N(종료)) ")
        print()
        # y 입력 시 새로운 코스 랜덤 추천
        if diff_course.lower() == "y":
            CourseRecommend(region_num)
        # n 입력 시 프로그램을 종료함
        elif diff_course.lower() == "n":
            print("여행 코스 추천 프로그램을 종료합니다.")
            break
        # 다른 문자 입력 시 문자를 재입력받음
        else:
            print("올바른 문자를 입력해주세요.")
            continue
        break

# 코드 테스트
if __name__ == "__main__":
    CourseRecommend(1)  # 1구역
```

4. Test Result

(1) User selection of region

- When the user selected the number of the desired travel region, a message confirming the completion of the region selection is displayed.

- Test result screenshot

```
========== 제주도 여행 코스 추천 프로그램 ==========
1. 제주시  2. 제주서부  3. 서귀포시  4. 제주동부
5. 권역의 세부 지역 정보 설명
여행을 원하는 권역의 번호를 선택하세요. -> 1

제주시의 여행 코스를 추천해드릴게요.
```

(2) Display area details

- When the user selects the number of the region for which they want information, the program displays the administrative district information belonging to the selected region.

- Test result screenshot

```
====== 제주도 여행 코스 추천 프로그램 ======
1. 제주시  2. 제주서부  3. 서귀포시  4. 제주동부
5. 권역의 세부 지역 정보 설명
여행을 원하는 권역의 번호를 선택하세요. -> 5
어느 권역의 세부 지역 정보가 필요하신가요?(숫자 입력) 1

1. 제주시 권역의 세부 지역 정보는 다음과 같습니다.
['일도일동', '일도이동', '이도일동', '이도이동', '삼도일동', '삼도이동', '건
입동', '용담일동', '용담이동', '도두동', '이호동', '외도동', '노형동', '연동'
, '오라동', '아라동', '화북동', '삼양동', '봉개동']
```

(3) Collection and classification of Jeju Island tourism data

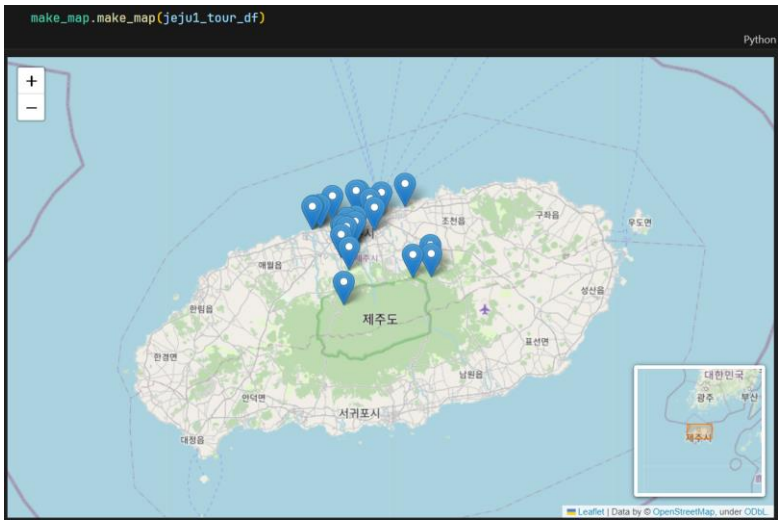
- Use the Kakao Maps API to retrieve results of place keyword searches within Jeju Island. Returns a CSV file containing the name, latitude, longitude, address, and website URL of the place obtained through a place keyword search and a map image containing the location of the place.

- Test result screenshot

	A	B	C	D	E	F	G	H	I	J
1	index	ID	stores	x	y	road_address	place_url			
2	0	10626439	용눈이오름	126.8327	33.45988		http://place.map.kakao.com/10626439			
3	241	17475343	월랑봉	126.8216	33.47582		http://place.map.kakao.com/17475343			
4	240	18742637	삼의봉	126.5619	33.44025		http://place.map.kakao.com/18742637			
5	121	18039453	방림원 백화동산	126.2692	33.33703		http://place.map.kakao.com/18039453			
6	239	24022942	삼각봉	126.5266	33.3738		http://place.map.kakao.com/24022942			

- In addition, the data obtained in the above process are classified into four regions, each saved as a csv file, and the map image containing the location is returned.

- Test result screenshot (jeju1_tour_region.csv)



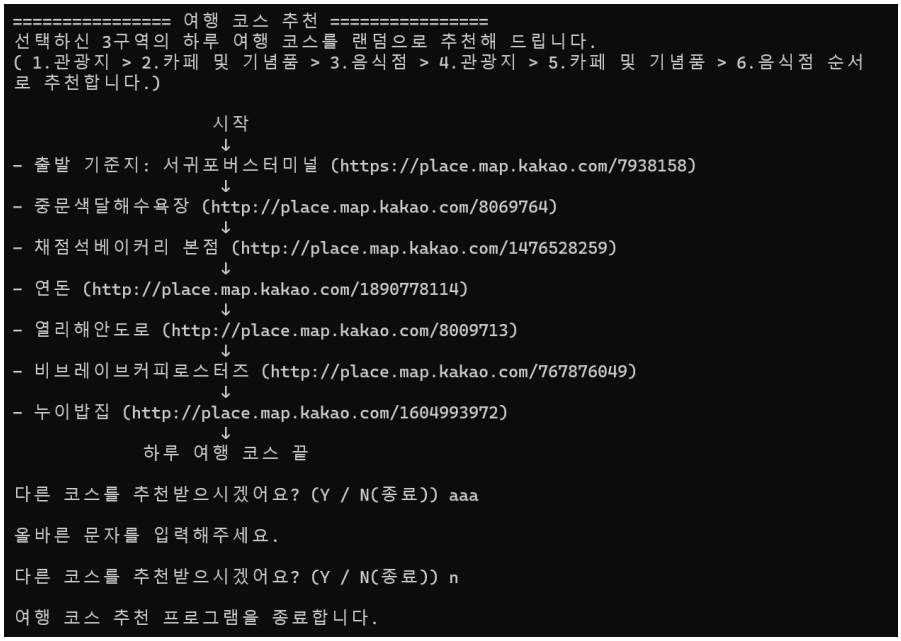
	A	B	C	D	E	F	G	H	I	J
1	index	ID	stores	x	y	road_address	place_url			
2	55	1.83E+09	브릭캠퍼스 제주	126.4853	33.45855	제주특별자치도 제주시 1100로 304	http://place.map.kakao.com/1826295011			
3	107	17583340	한라생태숲 수생식물원	126.5978	33.43236	제주특별자치도 제주시 516로 259	http://place.map.kakao.com/17583340			
4	86	1.57E+09	히어로플레이파크 제주점	126.4946	33.4807	제주특별자치도 제주시 과원로 128	http://place.map.kakao.com/1566739723			
5	212	13323388	알작지해변	126.4398	33.49593	제주특별자치도 제주시 내도동 465	http://place.map.kakao.com/13323388			

(The above result is a map of tourist destination data in area 1 among tourist destinations, cafes, souvenir shops, and restaurants in area 1 to 4, and the same method was used for data in other areas. For more information, please refer to the region_distribution.ipynb file.)

(4) Providing a travel course in the selected area

- Using the tourist destination data classified above, the same-day tour course in the region is selected and printed. Currently, only one place is derived for each category of tourist destination, cafe, souvenir shop, and restaurant, but in the future, we plan to set the standard location in the region and select more places to derive a course that considers efficiency.

- Test result screenshot



5. Changes in Comparison to the Plan

1) Collection and classification of Jeju Island tourism data

- Before : Pre-processing is performed by collecting Jeju Island tourism data provided by the Korea Tourism Organization. In addition, to reflect the places that are emerging these days, the results are derived after crawling using certain keywords related to Jeju Island on Instagram.

- After : Using the Kakao Map API, data on tourist attractions in Jeju Island are collected, preprocessed, and CSV files are returned.

- I have determined that using the Kakao Maps API is more suitable for finding places, and since it allows searching for popular places, I have decided to change the plan.

2) User selection of region

- Before : Jeju Island tourism data provided by the Korea Tourism Organization is collected and preprocessed, and classified into categories such as cafes,

- After : Set the travel course reference point to any place within 4 regions

- Since the starting point input by the user may not be registered on the Kakao Map or may not appear in the Kakao Map search result, the starting point is changed to be arbitrarily set.

3) Display area details

- Before : Do not provide details for each region

- After : Provide detailed information such as information on administrative consent in the selected area

- Provide administrative consent information for each area to help users select areas.

6. Lessons Learned & Feedback

It was nice to learn more about Python's overall content during the class. Not only was I able to learn the theory, but I was able to apply it directly in practice, which helped me improve my skills a lot. Also, I think it was better to take the midterm exam by writing code directly on a computer, not in a hand-coding method like other classes' tests. There was a little burden to proceed with the final project, but I was able to get a lot of things and feel a sense of accomplishment in the process of planning and completing the project myself. And as I can use Git and Github skillfully, I think it will be useful for future academic activities. I think it was a lecture that helped me a lot in many areas.