

PJT명	Python을 활용한 데이터 수집 2	
단계	[Python PJT]	
진행일자	2025.07.25	
예상 구현 시간	필수기능	5H
	추가기능	2H
	심화기능	1H

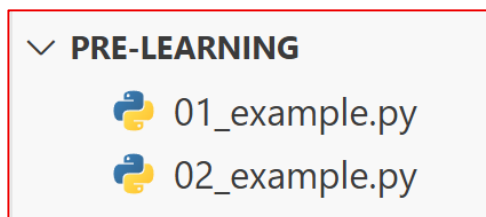
1. 목표

- API와 서버와의 요청 응답에 대하여 이해한다.
- requests 라이브러리를 통해 데이터를 요청할 수 있다.
- API를 통해 받은 JSON 데이터를 다룰 수 있다.
- 데이터 타입 List와 Dictionary를 조작할 수 있다.
- JSON 데이터를 재구성할 수 있다.

2. 준비사항

1) 프로젝트 구조

- 프로젝트는 총 세개의 폴더로 구성되어 있음
 1. pre-learning
 2. aladin
 3. tmdb
- pre-learning 폴더는 사전학습 자료로, 두개의 Python 코드가 제공됨



1. 01_example.py: 강아지 사진 API를 활용한 requests 사용 예시

```
# API 요청을 보낼 주소(Endpoint)를 변수에 저장
# 이 API는 랜덤한 강아지 사진 URL을 반환합니다.
URL = 'https://dog.ceo/api/breeds/image/random'

# requests.get()으로 요청을 보내고, json()으로 응답을
response = requests.get(URL).json()
```






2. 02_example.py: 알라딘 API를 활용한 requests 사용 예시

```
# API 요청에 필요한 파라미터들을 변수로 선언
ttbkey = '부여받은 TTBKey' # 알라딘 API 인증 키
query_type = 'ItemNewSpecial' # 신간 도서 요청 타입
max_results = 20 # 한 번에 받아올 결과 수
start = 1 # 검색 시작 위치
search_target = 'Book' # 검색 대상(도서)
output = 'js' # 출력 형식(json)
version = 20131101 # API 버전

# f-string을 사용하여 쿼리 파라미터가 포함된 전체 URL 생성
URL = f'http://www.aladin.co.kr/ttb/api/ItemList.aspx?ttbkey=
```







- aladin 폴더는 Aladin API를 활용한 프로젝트 요구사항 구현을 위한 스켈레톤 코드 5가지가 제공됨

▼ ALADIN

-  problem_a.py
-  problem_b.py
-  problem_c.py
-  problem_d.py
-  problem_e.py

- tmdb 폴더는 TMDB API를 활용한 프로젝트 요구사항 구현을 위한 스켈레톤 코드 5가지와 API 사용 예시 코드가 제공됨

▼ TMDB

- >  examples
 -  problem_a.py
 -  problem_b.py
 -  problem_c.py
 -  problem_d.py
 -  problem_e.py

2) 개발언어 및 툴

- Python 3.11+ / Visual Studio Code

3) 필수 라이브러리 / 오픈소스

- Python 내장 모듈 json
- Python 패키지 requests
- 알라딘 API
- TMDB API

3. 사전 학습

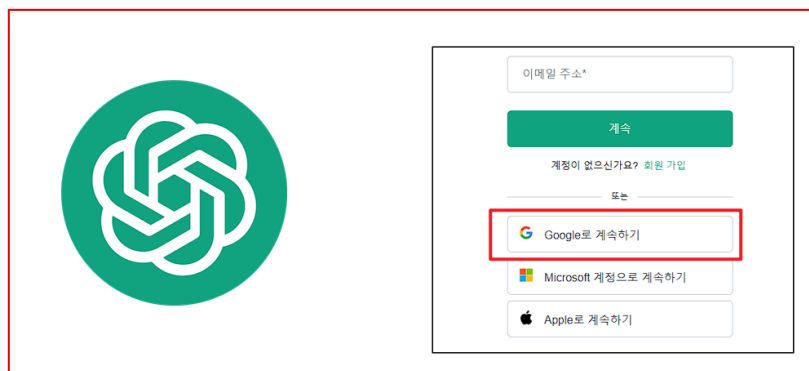
API – Application Programming Interface

두 소프트웨어가 서로 통신할 수 있게 하는 메커니즘

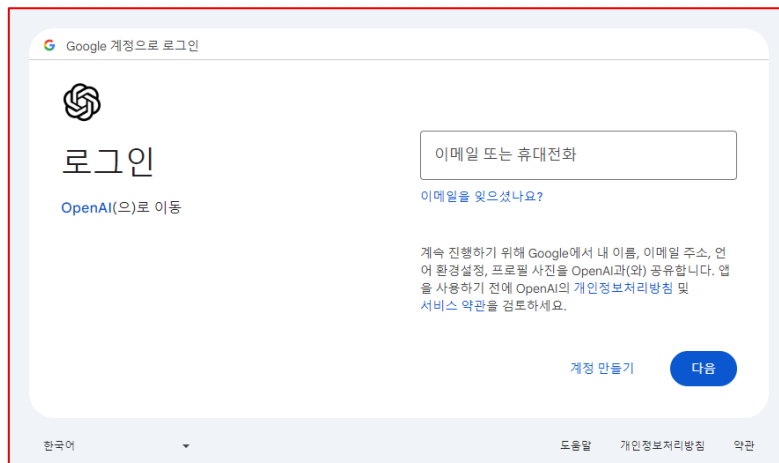
- 예시1: 소셜 로그인

특정 사이트에 직접 회원가입을 하지 않고 다른 소셜 미디어 계정으로 회원가입 및 로그인이 가능

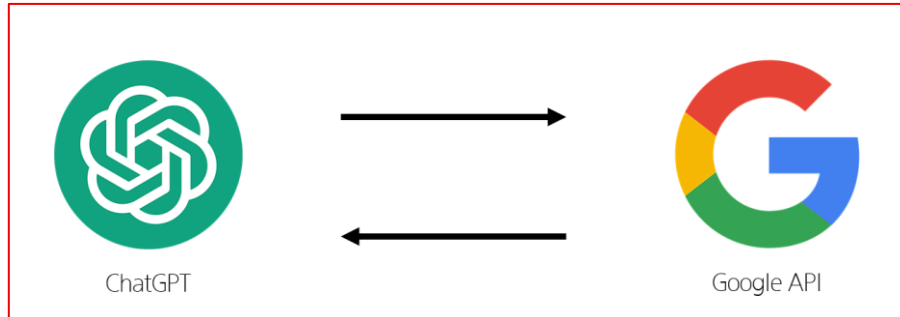
1. ChatGPT에서 Google로 계속하기 클릭



2. ChatGPT가 제공한 Google 로그인 화면에서 Google 계정으로 로그인 진행

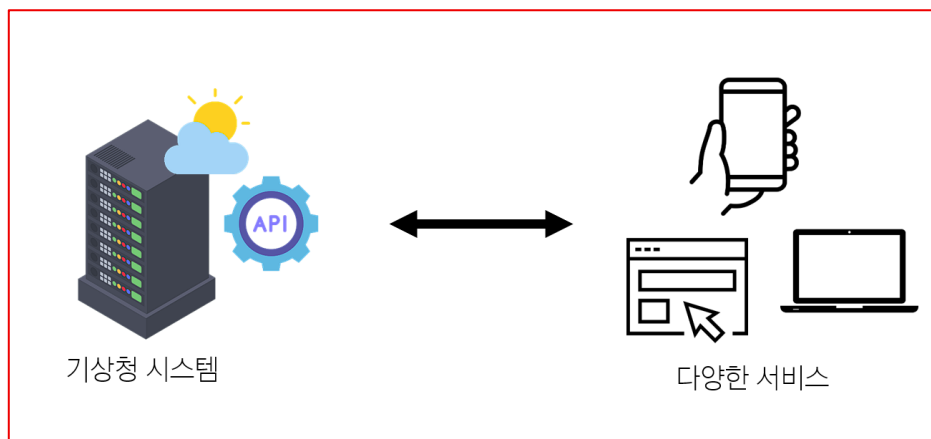


3. Google 로그인 계정으로 로그인을 성공했을 경우 Google API는 ChatGPT에게 로그인에 성공한 인증된 사용자 정보를 넘겨줌



4. 사용자 정보를 넘겨받은 ChatGPT는 해당 정보를 활용해 회원가입 및 로그인을 진행

● 예시2: 날씨 데이터 받기

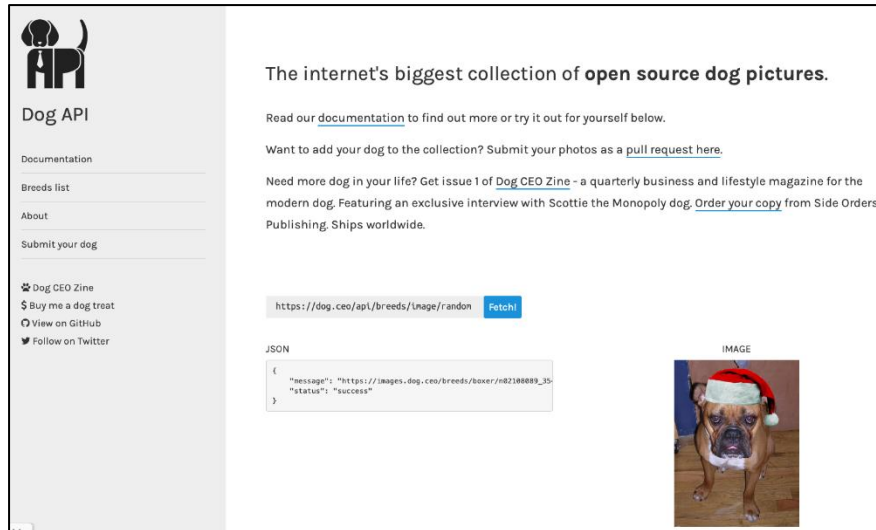


기상청의 시스템에 들어있는 기상 데이터를, 스마트폰 날씨 앱, 웹사이트의 날씨 정보 등이 기상청 시스템으로부터 데이터를 요청해서 받아감

- 기상청 시스템에는 정보들을 요청하는 지정된 형식이 있음
- 지역, 날짜, 조회할 내용들(온도, 바람 등)을 제공하는 매뉴얼
- “이렇게 요청을 보내면, 이렇게 정보를 제공해줄 것이다”라는 매뉴얼

API 활용 연습

1) 01_example.py



<https://dog.ceo/dog-api/>

강아지 사진을 요청하면 응답해주는 Dog API

- <https://dog.ceo/dog-api/documentation/random>
무작위 강아지 사진 URL을 응답해주는 API에 관한 문서
- 한장의 강아지 사진이 필요할 경우

DISPLAY SINGLE RANDOM IMAGE FROM ALL DOGS COLLECTION

<https://dog.ceo/api/breeds/image/random> [Fetch!](#)

- 여러장의 강아지 사진이 필요할 경우

DISPLAY MULTIPLE RANDOM IMAGES FROM ALL DOGS COLLECTION

<https://dog.ceo/api/breeds/image/random/3> [Fetch!](#)

Max number returned is 50. More info [here](#).

최대 50장 사용 가능

- 응답은 JSON 형태로 제공되며, message 키에 이미지를 확인 가능한 URL이 제공됨

JSON

```
{
  "message": "https://images.dog.ceo/breeds/terrier-russell/",
  "status": "success"
}
```

여러장의 경우 message가 list로 제공됨

JSON

```
{
  "message": [
    "https://images.dog.ceo/breeds/terrier-scottish/n02097707",
    "https://images.dog.ceo/breeds/mudhol-indian/Indian-Mudhol",
    "https://images.dog.ceo/breeds/terrier-silky/n02097658"
  ],
  "status": "success"
}
```

해당 API를 활용하는 코드가 01_example.py에 제공됨

```
import requests

URL = 'https://dog.ceo/api/breeds/image/random'

response = requests.get(URL).json()
print(response)

results = response.get('message')
print(results)
```

2) 02_example.py

알라딘 Open API 매뉴얼

1. 요청 (Request)

1) 상품 검색 API

2) 상품 리스팅 API

3) 상품 조회 API

4) 중고상품 보유 매장 검색 API

2. 응답 (Response)

1) 상품 검색 응답

2) 상품 리스팅 응답

3) 상품 조회 응답

4) 중고상품 보유 매장 검색 응답

• 최종업데이트 : 2022-07-13

1. 요청 (Request)

1) 상품 검색 API

- 요청 방법

- 요청 URL : <http://www.aladin.co.kr/ttb/api/ItemSearch.aspx>
- 요청 URL 설명 : <http://www.aladin.co.kr/ttb/api/ItemSearch.aspx?ttbkey={TTBKey}&Query={aladdin}&QueryType=Title&MaxResults=10&start=1&SearchTarget=Book&output=xml&Version=20131101>
- 결과 설명
 - XML 형식 : http://www.aladin.co.kr/ttb/api/test/ItemSearch_20131101.xml
 - javascript 형식 : http://www.aladin.co.kr/ttb/api/test/ItemSearch_20131101.js
- 한 페이지에 최대 50개, 총 결과는 200개까지만 검색 가능

- 상품 검색 API 요청(Request) 파라미터

구분	요청변수	변수종류	설명
필수	TTBKey	문자열	부여받은 TTBKey값
	Query	문자열	검색어
옵션 (옵션 조정 변수가 없을 경우에는 기본값으 로 검색)	QueryType	Keyword (기본값) : 제목+저자 Title : 제목검색 Author : 저자검색 Publisher : 출판사검색	검색어 종류
	SearchTarget	Book(기본값) : 도서 Foreign : 외국도서 Music : 음반 DVD : DVD Used : 중고상(도서/음반/DVD 등) eBook : 전자책 All : 위의 모든 타겟(모)	검색 대상 Mail
	Start	1이상, 양의 정수(기본값:1)	검색결과 시작페이지
	MaxResults	1이상 100이하, 양의 정수(기본값:10)	검색결과 한 페이지당 최대 출력 개수

https://docs.google.com/document/d/1mX-WxuoGs8Hy-QalhHcvuV17n50uGI2Sg_GHofgiePE/edit

알라딘 API와 문서

알라딘을 비롯한 많은 서비스는 클라이언트에게 고유의 키를 발급해 들어온 요청이 누구의 요청인지 확인(인증)한다.

- 공용 문서에서 "[Aladin API Key 발급 가이드](#)" 진행
- 요청 방법

http://www.aladin.co.kr/ttb/api/ItemList.aspx?ttbkey={key}&QueryType={query_type}

- <http://www.aladin.co.kr/ttb/api/ItemList.aspx>
endpoint: 어떤 자료를 요청할지
- [ttbkey={key}&QueryType={query_type}](#)
query: 자료 요청에 필요한 정보
- endpoint와 query는 '?'로 연결 및 구분
- 각각의 query는 & 로 연결

해당 API를 활용하는 코드가 02_example.py에 제공됨


```
URL = 'http://www.aladin.co.kr/ttb/api/ItemList.aspx'
```

```
params = {  
    'ttbkey': '부여받은 TTBKey',  
    'QueryType': 'ItemNewAll',  
    'MaxResults': 20,  
    'start': 1,  
    'SearchTarget': 'Book',  
    'output': 'js',  
    'Version': '20131101',  
}
```

```
response = requests.get(URL, params=params).json()
```

응답 받은 결과 중 item에 도서 정보를 담은 리스트가 응답됨

```
{  
    "version": "20131101",  
    "logo": "http://image.aladin.co.kr/img/header/2011/aladin_logo_new.gif",  
    "title": "알라딘 주목할 만한 신간 리스트 - 국내도서",  
    "link": "https://www.aladin.co.kr/shop/common/wnew.aspx?NewType=SpecialNew",  
    "pubDate": "Fri, 25 Jul 2025 01:51:00 GMT",  
    "totalResults": 2206,  
    "startIndex": 1,  
    "itemsPerPage": 20,  
    "query": "QueryType=ITEMNEWSPECIAL;SearchTarget=Book",  
    "searchCategoryId": 0,  
    "searchCategoryName": "국내도서",  
    "item": [  
        {  
            "title": "[세트] 파도관찰자를 위한 가이드 + 구름관찰자를 위한 가이드",  
            "link": "https://www.aladin.co.kr/shop/wproduct.aspx?ItemId=368681633",  
            "author": "개빈 프레터피니 (지은이), 김성훈, 홍한걸 (옮긴이)",  
            "pubDate": "2025-07-16",  
            "description": "",  
            "isbn": "K982030507",  
            "isbn13": "",  
            "itemId": 368681633,  
            "priceSales": 40500
```

4. 작업 순서

- 1) 팀원과 같이 요구사항(기본/추가/심화)을 확인하고, GitLab에 프로젝트를 생성한다.
 - 프로젝트 이름은 02-pjt로 지정한다.
 - 각 반 담당 강사님을 Maintainer로 설정한다.
- 2) 제공된 예시 코드를 확인하고, 요구사항에 필요한 코드를 파악한다.
- 3) 각 문제 코드를 확인하고, 예시 코드의 내용을 활용하여 필수 요구사항을 구현한다.
- 4) 작성한 코드들을 정리하고, README를 작성한다.

5. 요구사항

추천 알고리즘을 통한 영화 추천 커뮤니티 서비스를 구축하려고 한다. 다양한 스트리밍 플랫폼에서 제공되는 영화 정보를 수집 및 관리하고, 이를 기반으로 개인화된 영화 추천, 장르별 영화 탐색, 유사 영화 추천 등 다채로운 추천 기능을 설계 및 구현한다. 또한 영화에 대한 사용자 리뷰 및 감상평 공유 커뮤니티 기능을 제공하여, 사용자들이 활발하게 소통하고 정보를 교환할 수 있는 기능을 제공한다. 사용자는 자신이 본 영화를 평가하고, 다른 사용자의 리뷰를 참고하여 다음 영화를 선택하는데 도움을 받을 수 있다. 나아가, 관심 영화 목록을 맞춤형으로 구성하는 등 다양한 편의 기능을 제공한다. 팀원과 상의하여 아래 요구사항을 만족할 수 있도록 요구 사항 명세서를 작성 및 구현해보자.

영화 정보를 전부 보유하기는 어려우므로, 외부에서 영화 데이터를 가져와야 한다. API를 활용하여 데이터를 받아오고, 필요한 형태로 정리하는 연습을 해보자.

- 요구사항 예시(참고용)
 - 아래의 내용을 참고하여 추가적인 아이디어에 대해 요구사항을 추가 또는 수정하여 기능을 구현한다. 단, **필수 기능은 구현해야 하며, 수정할 수 없다.**

번호	분류	요구사항명	요구사항 상세	우선순위
기능적 요구사항				
F01	도서 데이터	API 호출	API를 활용해 저자를 기준으로 도서를 검색하는 기능	필수
F02	도서 데이터	데이터 변환	반환된 정보 중 일부 항목 만 추출하는 기능	필수
F03	도서	데이터 수집	반환된 정보 중 특정 조건을 만족하는	필수

	데이터		데이터를 추출하는 기능	
F04	도서 데이터	데이터 수집	반환된 정보를 특정 기준으로 정리한 후 일부를 추출하는 기능	
F05	도서 데이터	API 호출	API를 활용해 제목을 기준으로 도서를 검색하는 기능	필수
F06	도서 데이터	데이터 수집	반환된 정보 중 특정 조건을 만족하는 데이터의 일부 항목을 추출하는 기능	필수
F07	도서 데이터	API 재호출	반환된 정보와 API를 활용해 연관된 새로운 정보를 검색하는 기능 I	필수
F08	도서 데이터	API 호출	API에 추가 설정을 포함해 제목을 기준으로 도서를 검색하는 기능	필수
F09	도서 데이터	API 호출 심화	생성형 AI를 활용하여 다른 도서 데이터 API를 통해 데이터를 검색하는 기능	도전
F10	영화 데이터	API 호출	API를 활용해 장르를 기준으로 영화를 검색하는 기능	필수
F11	영화 데이터	데이터 변환	반환된 정보 중 일부 항목 만 추출하는 기능	필수
F12	영화 데이터	데이터 수집	반환된 정보 중 특정 조건을 만족하는 데이터를 추출하는 기능	필수
F13	영화 데이터	API 호출	API를 활용해 배우를 이름으로 검색하는 기능	필수
F14	영화 데이터	API 호출	API를 활용해 출연 배우를 기준으로 영화를 검색하는 기능	필수
F15	영화 데이터	데이터 정리	추출한 데이터를 특정 기준으로 정리하는 기능	필수
F16	영화 데이터	API 호출	API를 활용해 영화의 제작진을 조회하는 기능	필수
F17	영화 데이터	데이터 수집	조회된 데이터에서 특정 데이터를 수집하는 기능	필수
F18	영화 데이터	데이터 집계	조회된 데이터의 특징을 집계하는 기능	필수
F19	영화 데이터	API 호출 심화	생성형 AI를 활용하여 다른 영화 데이터 API를 통해 데이터를 검색하는 기능	도전
...

1) 기본(필수) 기능 (aladin)

API 요청 시 버전은 가장 최신 버전인 20131101을 사용한다.

A. 저자의 도서 조회

알라딘 API를 활용해 저자 '파울로 코엘료'의 도서를 검색한다. 최대 20개의 도서를 검색한 후, 제목(title)을 추출하는 함수 `get_author_books`를 작성한다.

- 요구사항 번호: F01, F02
- 주어진 `aladin/problem_a.py`를 수정하여 구현
- 알라딘 상품 검색 API 응답 데이터는 시기에 따라 예시 출력과 다를 수 있음.
- 구현해야 할 화면 (콘솔 출력)

```
['연금술사',  
 '11분',  
 '브리다',  
 '흐르는 강물처럼',  
 '포르토벨로의 마녀',  
 '마법의 순간 (리커버)',  
 '오 자히르',  
 '불륜',  
 '순례자 - 개정판',  
 '일러스트 연금술사',  
 '아처',  
 '내가 빛나는 순간',  
 '스파이',  
 '피에트라 강가에서 나는 울었네',  
 '아크라 문서',  
 '알레프',  
 '승자는 혼자다 1',  
 '악마와 미스 프랭',  
 '다섯번째 산',  
 '승자는 혼자다 2']
```

B. 저자의 도서 조회 & 조건 I

알라딘 API를 활용해 저자 '파올로 코엘료'의 도서를 검색한다. 최대 20개의 도서를 검색한 후, 리뷰 평점(customerReviewRank)이 9점 이상인 도서 목록을 추출하는 함수 get_best_review_books를 작성한다.

- 요구사항 번호: F03
- 주어진 aladin/problem_b.py를 수정하여 구현
- 구현해야 할 화면 (콘솔 출력, 결과 일부 생략)

```
[{'adult': False,
  'author': '파올로 코엘료 (지은이), 최정수 (옮긴이)',
  'categoryId': 50920,
  'categoryName': '국내도서>소설/시/희곡>스페인/중남미소설',
  'cover': 'https://image.aladin.co.kr/product/30/73/cover/8982814477_3.jpg',
  'customerReviewRank': 9,
  'description': '"세상을 두루두루 여행하기 위해 양치기가 된 청년 산티아고의 \'자  
아의 신화\' 찾기 여행담. 자칫 딱딱하게 보일 "  
"수 있는 제목과는 달리 간결하고 경쾌한 언어들로 쓰여 있어서,  
이 흘러가듯 수월하게 읽히는 작품이다."',
  'fixedPrice': True,
  'isbn': '8982814477',
  'isbn13': '9788982814471',
  'itemId': 307361,
  'link': 'https://www.aladin.co.kr/shop/wproduct.aspx?ItemId=307361&partne  
=openAPI&start=api',
  'mallType': 'BOOK',
  'mileage': 670,
  'priceSales': 12150,
  'priceStandard': 13500,
  'pubDate': '2001-12-01',
  'publisher': '문학동네',
  'salesPoint': 83224,
  'stockStatus': '',
  'subInfo': {},
  'title': '연금술사'},
```

- 예시 화면의 경우 pprint 모듈을 활용하여 출력되어 Dictionary의 key 순서가 정렬되어서 출력

C. 저자의 도서 조회 & 조건 II

알라딘 API를 활용해 저자 '파올로 코엘료'의 도서를 검색한다. 최대 20개의 도서를 검색한 후, 판매 지수(salesPoint)가 높은 5개의 도서의 제목과 판매지수를 추출하는 함수 `get_bestseller_books`를 작성한다.

- 요구사항 번호: F04
- 주어진 `aladin/problem_c.py`를 수정하여 구현
 - sort 메서드
<https://docs.python.org/3.11/library/stdtypes.html#list.sort>
 - sorted 함수
<https://docs.python.org/3.11/library/functions.html#sorted>
 - 제목과 판매지수를 포함할 것
- 구현해야 할 화면 (콘솔 출력)

```
[{'제목': '연금술사', '판매지수': 83224},  
 {'제목': '11분', '판매지수': 10322},  
 {'제목': '브리다', '판매지수': 10092},  
 {'제목': '흐르는 강물처럼', '판매지수': 10047},  
 {'제목': '포르토벨로의 마녀', '판매지수': 8364}]
```

D. 특정 도서 저자의 다른 도서 조회

알라딘 API를 활용해 제공된 도서 제목을 검색하여, 응답 받은 첫 번째 도서의 저자가 집필한 다른 5개의 도서 제목을 추출하는 함수 `get_author_other_books`를 작성한다.

- 요구사항 번호: F05
- 주어진 `aladin/problem_d.py`를 수정하여 구현
 - 응답 받은 첫번째 도서를 제외한 다른 도서 5개를 출력해야 함
 - 검색된 도서가 없는 경우 `None`을 반환
- 구현해야 할 화면 (콘솔 출력)

```
{'"베니스의 상인"'의 저자 "윌리엄 셰익스피어"의 다른 도서 목록': ['햄릿',  
                                                                    '셰익스피어 4대 비극 세트 : 햄릿.오셀로.맥베스.리어 왕 - '  
                                                                    '전4권',  
                                                                    '맥베스',  
                                                                    '로미오와 줄리엣',  
                                                                    '한여름 밤의 꿈']}]  
{'"'죄와 벌"'의 저자 "표도르 도스토옙스키"의 다른 도서 목록': ['카라마조프 가의 형제들 세트 - 전3권',  
                                                                    '죄와 벌 2',  
                                                                    '카라마조프가의 형제들 1',  
                                                                    '죄와 벌 - 상',  
                                                                    '카라마조프가의 형제들 2']}]
```

None

[제목 없음]

E. 가장 저렴한 중고 도서 가격 조회

알라딘 API를 활용해 제공된 도서 제목을 검색하여, 응답 받은 첫 번째 도서의 중고 도서 정보를 검색한다. 이후 중고 도서 배송처 중 가장 저렴하게 판매중인 도서의 배송처와 가격을 추출한다.

- 요구사항 번호: F08
- 주어진 `aladin/problem_e.py`를 수정하여 구현
 - 중고 도서 재고가 없는 경우 새 도서의 가격을 출력
 - 검색된 도서가 없는 경우 `None`을 반환
- 구현해야 할 화면 (콘솔 출력)

```
'도서 "죄와 벌"의 가장 저렴한 중고는 광활한 우주점이 보유 중이며, 7200원에 판매 중입니다.'  
'도서 "로미오와 줄리엣"의 가장 저렴한 중고는 알라딘이 보유 중이며, 4600원에 판매 중입니다.'  
None
```

2) 도전 과제 (aladin)

기본 기능 구현 후 생성형 AI를 사용해 도전과제 요구사항을 해결한다.

A. 도서 데이터를 제공하는 다른 API를 사용하여 내가 원하는 데이터를 추출해 재구성하기

- 요구사항 번호: F09
- API 목록
 - 국가 자료 종합 목록
<https://www.nl.go.kr/NL/contents/N31101030400.do>
 - 네이버 도서검색 API
<https://developers.naver.com/docs/serviceapi/search/book/book.md>
 - 카카오 도서검색 API
<https://developers.kakao.com/docs/latest/ko/daum-search/dev-guide>

3) 기본(필수) 기능 (TMDB)

API 요청 시 언어(language)는 ko-KR로 진행한다.

- 공용 문서를 참고하여 [API 키](#)를 발급받는다.
- examples/01_example.py 코드를 확인하여 TMDB API 사용법을 익힌다.

```
URL = 'https://api.themoviedb.org/3'
HEADERS = {
    'accept': 'application/json',
    'Authorization': f'Bearer <TMDB Access Token>',
}

def get_multi_search():
    # search movie API 문서: https://developer.themoviedb.org/reference/search-movie
    params = {
        'query': '범죄도시', # 필수 파라미터
        'include_adult': True,
        'language': 'ko-KR',
        'page': 1,
    }

    # 요청 보내 받아온 결과는 requests 타입의 데이터이고, 파이썬에서 바로 쓸 수 없으며
    response = requests.get(f'{URL}/search/movie', headers=HEADERS, params=params)
    # 파이썬에서 쓸 수 있도록 하기 위해 json() 메서드를 사용해 json 타입의 데이터를 파
    response = response.json()
    # response 구조는 위의 공식 문서에서 확인할 수 있다.
    result = response.get('results')

    return result
```

- TMDB API 문서

<https://developer.themoviedb.org/reference/intro/getting-started>

- 주요 활용 API

- 장르 목록

<https://developer.themoviedb.org/reference/genre-movie-list>

- 영화 탐색

<https://developer.themoviedb.org/reference/discover-movie>

- 이름으로 인물 검색

<https://developer.themoviedb.org/reference/search-person>

- 영화 제작진 조회

<https://developer.themoviedb.org/reference/movie-credits>

A. 영화 정보 조회

TMDB API를 활용해 Action 장르의 영화에 대한 정보를 조회한다.

결과를 바탕으로 최대 20편의 영화 제목(title)을 추출하는 함수

get_movies를 작성한다.

- 요구사항 번호: F10, F11
- 주어진 tmdb/problem_a.py를 수정하여 구현
 - origin_country가 대한민국(KR)인 영화만 검색
- 구현해야 할 화면 (콘솔 출력)

```
['범죄도시 4',  
'부산행',  
'하이파이브',  
'용감한 시민',  
'마녀(魔女) Part2. The Other One',  
'올드보이',  
'마녀',  
'악인전',  
'베테랑 2',  
'하이재킹',  
'설국열차',  
'탈출: 프로젝트 사일런스',  
'거룩한 밤: 데몬 헌터스',  
'범죄도시 3',  
'범죄도시 2',  
'외계+인 1부',  
'탈주',  
'무도실무관',  
'늑대사냥',  
'더 킬러: 죽어도 되는 아이']
```

B. 영화 정보 조회 & 조건 I

TMDB API를 활용해 Action 장르의 영화에 대한 정보를 조회한다.
결과를 바탕으로 최대 20편의 영화를 검색하고, 평점이 7.0 이상인
영화의 제목과 평점(vote_average)을 추출하는 함수
get_good_score_movies를 작성한다.

- 요구사항 번호: F12
- 주어진 tmdb/problem_b.py를 수정하여 구현
 - origin_country가 대한민국(KR)인 영화만 검색
- 구현해야 할 화면 (콘솔 출력)

```
[{'title': '범죄도시 4', 'vote_average': 7.031},  
 {'title': '부산행', 'vote_average': 7.753},  
 {'title': '하이파이브', 'vote_average': 7.0},  
 {'title': '용감한 시민', 'vote_average': 7.0},  
 {'title': '마녀(魔女) Part2. The Other One', 'vote_average': 7.329},  
 {'title': '올드보이', 'vote_average': 8.252},  
 {'title': '마녀', 'vote_average': 7.889},  
 {'title': '악인전', 'vote_average': 7.8},  
 {'title': '탈출: 프로젝트 사일런스', 'vote_average': 7.016},  
 {'title': '범죄도시 3', 'vote_average': 7.2},  
 {'title': '범죄도시 2', 'vote_average': 7.3},  
 {'title': '탈주', 'vote_average': 7.4},  
 {'title': '무도실무관', 'vote_average': 7.8},  
 {'title': '더 킬러: 죽어도 되는 아이', 'vote_average': 7.687}]
```

C. 영화 정보 조회 & 조건 II

TMDB API를 활용해 이름을 기준으로 인물 정보를 검색하고, 결과 중 배우로 알려진 인물 중 첫번째 인물 정보를 추출한다. 이후 해당 배우의 영화 중 가장 최근에 개봉한 최대 영화 5편을 검색하고, 평점(vote_average)이 높은 순서대로 추출하는 함수 `get_actor_movies`를 작성한다.

- 요구사항 번호: F13, F14, F15
- 주어진 `tmdb/problem_c.py`를 수정하여 구현
 - sort 메서드
<https://docs.python.org/3.11/library/stdtypes.html#list.sort>
 - sorted 함수
<https://docs.python.org/3.11/library/functions.html#sorted>
 - 검색되는 배우가 없을 경우 `None`을 반환
- 구현해야 할 화면 (콘솔 출력)

```
[{'release_date': '2022-08-03', 'title': '비상선언', 'vote_average': 7.533},  
 {'release_date': '2022-04-29', 'title': '이창동: 아이러니의 예술', 'vote_average': 7.5},  
 {'release_date': '2024-12-04', 'title': '1승', 'vote_average': 7.333},  
 {'release_date': '2022-06-08', 'title': '브로커', 'vote_average': 7.183},  
 {'release_date': '2023-09-27', 'title': '거미집', 'vote_average': 6.6}]  
None
```

D. 배우의 작품별 감독 정보 조회

TMDB API를 활용해 이름을 기준으로 인물 정보를 검색하고, 결과 중 배우로 알려진 인물 중 첫번째 인물 정보를 추출한다. 이후 해당 배우의 대표작(known_for)의 감독 정보를 조회하여, 제목과 함께 정리하여 반환하는 함수 `get_actor_movie_directors`를 작성한다.

- 요구사항 번호: F16, F17
- 주어진 `tmdb/problem_d.py`를 수정하여 구현
 - 검색되는 배우가 없을 경우 `None`을 반환
- 구현해야 할 화면 (콘솔 출력)

```
[{'director': '봉준호', 'title': '기생충'},  
 {'director': '봉준호', 'title': '설국열차'},  
 {'director': '봉준호', 'title': '살인의 추억'}]  
None
```

E. 배우의 작품별 장르 정리

TMDB API를 활용해 이름을 기준으로 인물 정보를 검색하고, 결과 중 배우로 알려진 인물 중 첫번째 인물 정보를 추출한다. 이후 해당 배우의 영화 중 가장 최근에 개봉한 최대 영화 5편을 검색하고, 각 영화 별 장르를 확인하여 각 장르에 해당하는 영화가 몇편인지 정리하여 반환하는 함수 `get_actor_genres`를 작성한다.

- 요구사항 번호: F18
- 주어진 `tmdb/problem_e.py`를 수정하여 작업
 - 검색되는 배우가 없을 경우 `None`을 반환
- 구현해야 할 화면 (콘솔 출력)

```
{'Action': 1,  
 'Comedy': 3,  
 'Crime': 1,  
 'Documentary': 1,  
 'Drama': 4,  
 'Thriller': 1}  
None
```


4) 도전 과제 (tmdb)

기본 기능 구현 후 생성형 AI를 사용해 도전과제 요구사항을 해결한다.

A. 영화 데이터를 제공하는 다른 API를 사용하여 내가 원하는 데이터를 추출해 재구성하기

- 요구사항 번호: F19
- API 목록
 - IMDb Developer
<https://developer.imdb.com/>
 - KMDb Open API
<https://www.kmdb.or.kr/info/api/apiList>
 - 영화진흥위원회 Open API
<https://www.kobis.or.kr/kobisopenapi/homepg/apiservice/searchServiceInfo.do>
 - 네이버 영화 검색 API
<https://developers.naver.com/docs/serviceapi/search/movie/movie.md>

상기된 과제를 전부 완료한 경우 명시된 요구사항 외 추가 개발을 자유롭게 진행한다.

6. 참고자료

- requests – HTTP for Humans
<https://requests.readthedocs.io/en/latest/>
- 알라딘 API
https://docs.google.com/document/d/1mX-WxuoGs8Hy-QalhHcvuV17n50uGI2Sg_GHofgiePE/edit
- TMDB API
<https://developer.themoviedb.org/reference/intro/getting-started>

7. 결과

제출 기한은 진행일 18시까지이므로 제출 기한을 지킬 수 있도록 한다. 제출은 GitLab을 통해서 이뤄진다.

- 산출물과 제출
 - 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낀 점을 상세히 기록한 README.md
 - 완성된 각 문제 별 소스코드 및 실행 화면 캡처본
 - 프로젝트 이름은 02_pjt로 지정, 각자의 계정에 생성할 것
 - 각 반 담당 강사님을 Maintainer로 설정

- 끝 -