SAMSUNG SW AI ACADEMY FOR YOUTH

對外秘

프로젝트 명세서

웹 애플리케이션 프로젝트

(Python)

“추천 알고리즘을 통한 영화 추천 커뮤니티 서비스”

1 회차

對外秘

목차

對外秘

1. Python을 활용한 데이터 수집 1

PJT명	Python을 활용한 데이터 수집 1	
단계	01 PJT	
진행일자	2026.01.23	
예상 구현 시간	필수기능	6H
	심화기능	2H

1.1 목표

이번 'Python을 활용한 데이터 수집 1' 프로젝트의 목표는 다음과 같습니다.

- Python 기초 문법에 대하여 이해하고 활용할 수 있다.
- 파일 입출력에 대하여 이해한다.
- JSON 파일로 주어진 데이터를 프로그램에 활용할 수 있다.
- 데이터 구조를 분석하고 재구성하여 프로그램에 활용할 수 있다.
- Python을 활용하여 데이터를 가공하고 JSON 형태로 구성할 수 있다.

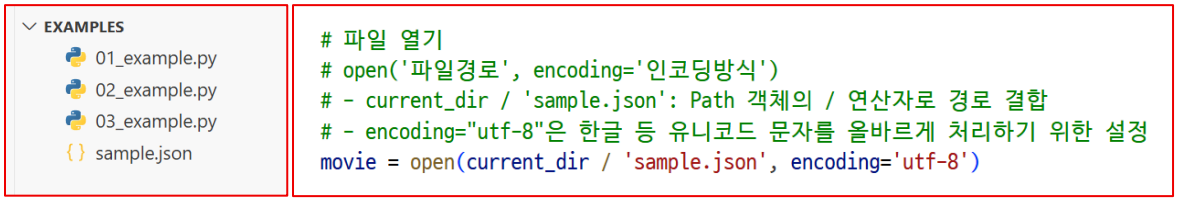
1.2 준비사항

1) 프로젝트 구조

- 프로젝트는 총 두 개의 폴더로 구성되어 있으며, 각 폴더별로 사용하는 데이터가 다릅니다.
  - tmdb
  - spotify
- 각 폴더에는 다음의 내용이 제공됩니다.
  - problem_*.py : 요구사항 구현을 위한 스켈레톤 코드
  - examples 폴더 : 아래 '2) 사용 데이터'에 설명된 examples 폴더
  - data 폴더 : 아래 '2) 사용 데이터'에 설명된 data 폴더

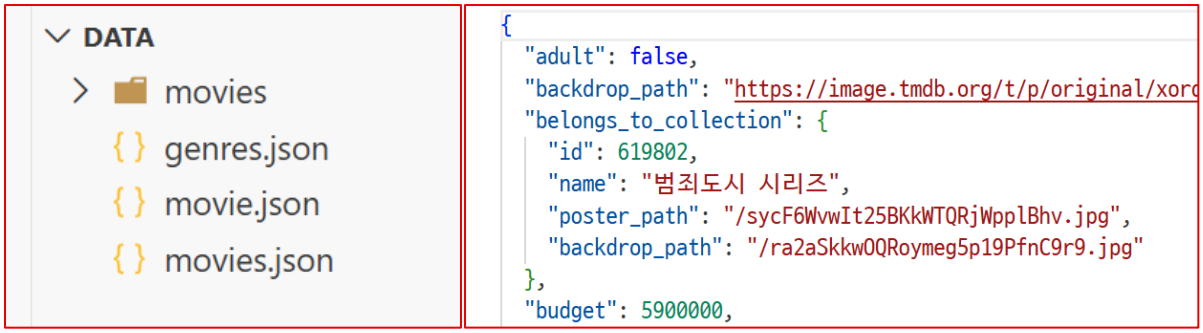
2) 사용 데이터

- examples 폴더 : 이번 프로젝트 해결을 위해 알아야 하는, 혹은 직접적인 도움이 될 수 있는 코드



<그림 1> examples 폴더 및 작업 경로 설정 예시

- data 폴더 : 실제 문제 풀이에 사용되는 데이터



<그림 2> data 폴더 및 json 파일 내용 예시

3) 개발언어 및 툴

- Python 3.11+ / Visual Studio Code

4) 필수 라이브러리 / 오픈소스

- Python 내장 json 모듈

1.3 작업 순서

1) 팀원과 같이 요구사항(기본/추가/심화)을 확인하고, GitLab 에 프로젝트를 생성한다.

- 프로젝트 이름은 01-pjt 로 지정한다.
- 각 반 담당 강사님을 Maintainer 로 설정한다.

2) 제공된 examples 의 코드를 확인하고, 요구사항에 필요한 코드를 파악한다.
3) 각 문제 코드를 확인하고, data 에 주어진 파일을 활용하여 필수 요구사항을 구현한다.
4) 작성한 코드들을 정리하고, README 를 작성한다.
5) README 작성이 완료되면 심화 학습을 진행한다.
6) 제출 기한에 맞춰 모든 산출물이 GitLab 에 업로드 될 수 있도록 한다.

1.4 요구사항

본 프로젝트는 추천 알고리즘을 통한 영화 추천 커뮤니티 서비스 구축을 목표로 합니다. 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 구현합니다.

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 요구사항 명세서를 작성 및 구현해 봅시다.

본격적인 서비스 구현에 앞서 제공되는 데이터 파일을 기반으로, 필요한 정보를 파싱한 후 프로그램 내부에서 정리할 수 있는 프로그램을 구현해봅시다.

■ 요구사항 예시 (참고용)

- 아래의 내용을 참고하여 추가적인 아이디어에 대해 요구사항을 추가 또는 수정하여 기능을 구현합니다. 단, 필수 기능은 반드시 구현해야 하며 수정할 수 없습니다.

기능적 요구사항

번호	분류	요구사항명	요구사항 상세	우선순위
F101	영화 데이터	데이터 수집	주어진 파일에서 필요한 정보를 추출하는 기능	필수
F102	영화 데이터	데이터 변환	F01에서 추출한 정보를 재사용 가능하게 변환하는 기능	필수
F103	영화 데이터	데이터 수정	추출한 데이터를, 다른 데이터를 바탕으로 수정하는 기능	필수
F104	영화 데이터	다중 데이터 분석	여러 개체에 대한 정보가 주어진 파일에서 각각의 필요한 정보를 추출하는 기능	필수
F105	영화 데이터	다중 데이터 변환	F04에서 추출한 다중 데이터를 재사용 가능하게 변환하는 기능	필수
F106	영화 데이터	다중 파일 데이터 수집	여러 파일에 걸쳐서 데이터를 추출하는 기능	필수
F107	영화 데이터	특정 데이터 출력	F06에서 추출한 데이터 중 특정 조건을 만족하는 데이터를 출력하는 기능	필수
F108	영화 데이터	특정 데이터 수집	F06에서 추출한 데이터 중 특정 조건을 만족하는 데이터를 재사용 가능하게 변환하는 기능	필수
F109	영화 데이터	특정 데이터 수집	생성형 AI를 활용하여 특정 조건을 만족하는 데이터를 추출하는 기능	심화
F110	영화 데이터	특정 데이터 수집 및 정리	생성형 AI를 활용하여 특정 조건을 만족하는 데이터를 추출하고 정리하는 기능	심화
F111	음악 데이터	데이터 수집	주어진 파일에서 필요한 정보를 추출하는 기능	필수
F112	음악 데이터	데이터 변환	F11에서 추출한 정보를 재사용 가능하게 변환하는 기능	필수
F113	음악 데이터	데이터 수정	추출한 데이터를, 다른 데이터를 바탕으로 수정하는 기능	필수
F114	음악 데이터	다중 데이터 분석	여러 개체에 대한 정보가 주어진 파일에서 각각의 필요한 정보를 추출하는 기능	필수
F115	음악 데이터	다중 데이터 변환	F14에서 추출한 다중 데이터를 재사용 가능하게 변환하는 기능	필수
F116	음악 데이터	다중 파일 데이터 수집	여러 파일에 걸쳐서 데이터를 추출하는 기능	필수

F117	음악 데이터	특정 데이터 출력	F16에서 추출한 데이터 중 특정 조건을 만족하는 데이터를 출력하는 기능	필수
F118	음악 데이터	특정 데이터 수집	F16에서 추출한 데이터 중 특정 조건을 만족하는 데이터를 재사용 가능하게 변환하는 기능	필수
F119	음악 데이터	특정 데이터 수집	생성형 AI를 활용하여 특정 조건을 만족하는 데이터를 추출하는 기능	심화
F120	음악 데이터	특정 데이터 수집	생성형 AI를 활용하여 특정 조건을 만족하는 데이터를 추출하는 기능	심화
…	…	…	…	…

1. [tmdb] 기본(필수) 기능

1) 제공되는 영화 데이터의 주요 내용 수집

① 요구사항 번호 : F101, F102

② 주어진 tmdb/problem_a.py 을 수정하여 구현

- 필요한 정보 : id, title, revenue, overview, genre_ids, poster_path
- movie.json 파일에서 필요한 정보에 해당하는 값을 추출하여, 새로운 Dictionary 로 반환하는 함수 movie_info 를 작성
- 완성된 함수는 다음 문제의 기본 기능으로 사용됨
- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary 의 key 순서가 정렬되어서 출력됨

```
{'genre_ids': [28, 80, 53],
 'id': 479718,
 'overview': '2004년 서울. 중국 하얼빈에서 활동하다 피신한 신흥 범죄조직의 악랄한 보스 장첸. 가리봉동 일대로 넘어온 장첸과 '
             '그의 일당들은 단숨에 기존 조직들을 장악하고 가장 강력한 세력인 춘식이파 보스 황사장까지 위협하며 도시 일대의 '
             '최강자로 급부상한다. 한편 대한민국을 뒤흔든 장첸 일당을 잡기 위해 오직 주먹 한방으로 도시의 평화를 유지해 온 '
             '괴물형사 마석도와  인간미 넘치는 든든한 리더 전일만 반장이 이끄는 강력반은 눈에는 눈 방식의  소탕 작전을 '
             '기획하는데...',
 'poster_path': 'https://image.tmdb.org/t/p/original/ayk6y2D5v5VACFqrPfF05MARZ9n.jpg',
 'revenue': 52946454,
 'title': '범죄도시'}
```

<그림 3> 요구사항 F101, F102 결과물 예시

2) 제공되는 영화 데이터의 주요 내용 수정

① 요구사항 번호 : F103

② 앞선 요구사항의 코드를 활용하여 주어진 tmdb/problem_b.py 을 수정하여 구현

- 필요한 정보 : id, title, revenue, overview, genres, poster_path
- movie.json 파일에서 필요한 정보에 해당하는 값을 추출한 다음, categoryId는 genres.json 파일에서 각각 ID에 매칭되는 객체의 name으로 변환하여 genres 로 수집한 뒤 새로운 Dictionary 로 변환하는 함수 movie_info를 작성
- 완성된 함수는 다음 문제의 기본 기능으로 사용됨
- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary의 key 순서가 정렬되어서 출력됨

```
{'genres': ['액션', '범죄', '스릴러'],
 'id': 479718,
 'overview': '2004년 서울. 중국 하얼빈에서 활동하다 피신한 신흥 범죄조직의 악랄한 보스 장첸. 가리봉동 일대로 넘어온 장첸과 '
             '그의 일당들은 단숨에 기존 조직들을 장악하고 가장 강력한 세력인 춘식이파 보스 황사장까지 위협하며 도시 일대의 '
             '최강자로 급부상한다. 한편 대한민국을 뒤흔든 장첸 일당을 잡기 위해 오직 주먹 한방으로 도시의 평화를 유지해 온 '
             '괴물형사 마석도와  인간미 넘치는 든든한 리더 전일만 반장이 이끄는 강력반은 눈에는 눈 방식의  소탕 작전을 '
             '기획하는데...',
 'poster_path': 'https://image.tmdb.org/t/p/original/ayk6y2D5v5VACFqrPfF05MARZ9n.jpg',
 'revenue': 52946454,
 'title': '범죄도시'}
```

<그림 4> 요구사항 F103 결과물 예시

3) 다중 데이터 분석 및 수정

① 요구사항 번호 : F104, F105

② 앞선 요구사항의 코드를 활용하여 주어진 tmdb/problem_c.py 을 수정하여 구현

- 필요한 정보 : id, title, revenue, overview, genres, poster_path

- movies.json 파일에는 여러 영화에 대한 데이터가 제공됨. 각 영화에 대한 데이터 중 필요한 정보를 추출하여 Dictionary 로 변환한 다음, 해당 Dictionary 들을 담고 있는 새로운 List 를 반환하는 함수 movies_info 를 작성

- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary 의 key 순서가 정렬되어서 출력됨

```
[{'genres': ['액션', '범죄', '스릴러', '모험'],
  'id': 619803,
  'overview': '가리봉동 소탕작전으로부터 4년이 지난 2008년, 금천경찰서 강력반은 베트남으로 도주한 용의자를 인도받아 오라는 '
              '미션을 받는다. 그렇게 현지를 물색하던 마석도와 전일만은 용의자에게 미심쩍음을 느껴 추궁한 끝에, 악랄한 강해상의 '
              '존재를 알게 된다. 두 형사는 수사권이 없는 상황에도 한국과 베트남을 오가며 역대급 범죄를 저지르는 그를 잡겠다는 '
              '일념 하나로 호찌민 이곳저곳을 거침없이 누빈다.',
  'poster_path': 'https://image.tmdb.org/t/p/original/6dl0f9322VuPUPRSf7TX4P6u1rB.jpg',
  'revenue': 101166707,
  'title': '범죄도시 2'},
 {'genres': ['드라마', '음악'],
  'id': 529859,
  'overview': '발렛 파킹, 편의점 아르바이트로 빡센 청춘을 보내지만, 쇼미더머니 6년 개근의 열정을 불태우는 무명 래퍼 학수 '
              'a.k.a 심빡. 또 다시 예선 탈락을 맞이한 인생 최악의 순간, 한 통의 전화를 받고 잊고 싶었던 고향 변산으로 '
              '향한다. 짝사랑 선미의 꼼수에 제대로 낚여 고향에 강제로 소환된 학수. 징글징글하게 들러 붙는 옛 친구들로 인해 '
              '지우고 싶었던 흑역사는 하나, 둘 떠오르고 하루 빨리 고향을 뜨고 싶었던 학수는 예측 불허의 사건들을 겪으면서 '
              '인생 최대 위기를 맞이하게 되는데...',
  'poster_path': 'https://image.tmdb.org/t/p/original/odLMoZQy38CiA97tOPJvPx33NB6.jpg',
  'revenue': 0,
  'title': '변산'},
 {'genres': ['다큐멘터리', '역사'],
  'id': 1070032,
  'overview': '20여 년 전 북한을 탈출한 이현서의 충격적 증언. 북한에 남겨 두고 온 아들을 어떻게 든 데리고 오려는 탈북한 '
              '엄마 이소연. 어린 자매부터 할머니까지 목숨을 걸고 국경을 넘어 탈출하려는 5명의 일가족. 낙원이라 믿었던 땅을 '
              '떠나 자유를 향한 이들의 탈출을 헌신적으로 돕는 김성은 목사. 거짓의 유토피아 북한에서 자행되고 있는 인권의 '
              '실태를 보여주며 충격과 분노를, 낙원이라고 믿고 자란 땅을 탈출하려는 이들의 위험한 여정과 탈출을 위한 김성은 '
              '목사의 용감한 헌신을 생생하게 담아내 안타까움과 감동을 선사하는 다큐멘터리.',
  'poster_path': 'https://image.tmdb.org/t/p/original/2EKSTCRzv7KvsLsOhCVXcUtwSJR.jpg',
  'revenue': 6800,
  'title': '비욘드 유토피아'},
```

<그림 5> 요구사항 F104, F105 결과물 예시

4) 가장 리뷰가 높은 영화 데이터 수집

① 요구사항 번호 : F106, F107

② 주어진 tmdb/problem_d.py 을 수정하여 구현

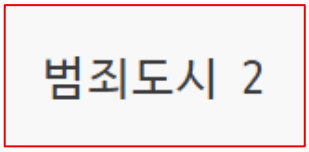
- movies 폴더에는 <영화 id>.json 형식으로 각 영화에 대한 정보가

제공되며, 영화의 인기도는 popularity 에 기록되어 있음

- movies.json 파일에 제공된 각 영화 정보의 id 를 활용하여 해당 영화의 인기도 정보를 확인하고, 이 중 인기도가 가장 높은 영화의 제목을 출력하는 함수 best_movie 을 작성

③ 주어진 tmdb/examples/03_exmaple.py 코드를 참고하여 작성

- 반복문을 통해 tmdb/data/movies 폴더 내부의 파일들을 열기



<그림 6> 요구사항 F106, F107 결과물 예시

5) 특정 연도에 개봉한 영화 데이터 수집

④ 요구사항 번호 : F108

⑤ 주어진 tmdb/problem_e.py 을 수정하여 구현

- movies 폴더에는 <영화 id>.json 형식으로 각 영화에 대한 정보가 제공되며, 영화의 개봉 날짜는 release_date 에 포함되어 있음.

- movies.json 파일에 제공된 각 영화 정보의 id 를 활용하여 해당 영화의 개봉 날짜를 확인하고, 개봉 연도가 2022 년 또는 그 이후인 영화들의 원제목(original_title)을 리스트로 출력하는 함수 recent_movies 를 작성

⑥ 주어진 tmdb/examples/03_exmaple.py 코드를 참고하여 작성

- 반복문을 통해 tmdb/data/movies 폴더 내부의 파일들을 열기

```
['범죄도시 2', 'Beyond Utopia', '경관의 피', 'The Simpsons
Meet the Bocellis in "Feliz Navidad"']
```

<그림 7> 요구사항 F108 결과물 예시

2. [tmdb] 심화 기능

기본 기능 구현 후, 생성형 AI 를 사용해 다음 두 심화 과제 요구사항을 해결합니다.

1) 특정 연도의 회원 평균 평점이 가장 높은 영화 제목 수집

① 요구사항 번호 : F109

② 주어진 tmdb/problem_f_1.py 를 수정하여 구현

- 2022 년 개봉 영화 중 회원 평균 평점이 가장 높은 영화의 원제목(original_title)을 수집하는 함수 best_recent_movies 를 작성

```
Beyond Utopia
```

<그림 8> 요구사항 번호 F109 결과물 예시

2) 특정 장르에서 수익 높은 순의 영화 제목 수집

① 요구사항 번호 : F110

② 주어진 tmdb/problem_f_2.py 를 수정하여 구현

- 장르가 액션인 영화의 제목을 수익(revenue)이 높은 순서대로 수집하는

함수 sorted_action_movies_by_revenue 를 작성

```
['범죄도시 2', '의형제', 'Buffalo Boys', 'Contratiempo']
```

<그림 9> 요구사항 번호 F110 결과물 예시

3. [spotify] 기본(필수) 기능

1) 제공되는 아티스트 데이터의 주요 내용 수집

① 요구사항 번호 : F111, F112

② 주어진 spotify/problem_a.py 를 수정하여 구현

- 필요한 정보 : id, name, genres_ids, images, type
- artist.json 파일에서 필요한 정보에 해당하는 값을 추출하여, 새로운 Dictionary 로 반환하는 함수 artist_info 를 작성
- 완성한 함수는 다음 문제의 기본 기능으로 사용됨
- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary 의 key 순서가 정렬되어서 출력

```
{'genres_ids': [651, 816],
 'id': 178,
 'images': [{'height': 640,
             'url': 'https://i.scdn.co/image/ab6761610000e5eb59f8cfc8e71dcaf8c6ec4bde',
             'width': 640},
            {'height': 320,
             'url': 'https://i.scdn.co/image/ab67616100005174 59f8cfc8e71dcaf8c6ec4bde',
             'width': 320},
            {'height': 160,
             'url': 'https://i.scdn.co/image/ab6761610000f17859f8cfc8e71dcaf8c6ec4bde',
             'width': 160}],
 'name': 'Jimin',
 'type': 'artist'}
```

<그림 10> 요구사항 번호 F111, F112 결과물 예시

2) 제공되는 아티스트 데이터의 주요 내용 수정

① 요구사항 번호 : F113

② 앞선 요구사항의 코드를 활용하여 주어진 spotify/problem_b.py 를 수정하여 구현

- 필요한 정보 : id, name, images, type, genre_names
- artist.json 파일에서 필요한 정보에 해당하는 값을 추출한 다음, genres_ids 는 genres.json 파일에서 각 ID 에 매칭되는 genre_name 으로 변환하여 genres_names 로 수집한 뒤 새로운 Dictionary 로 반환하는 함수 artist_info 를 작성
- 완성한 함수는 다음 문제의 기본 기능으로 사용됨
- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary 의 key 순서가 정렬되어서 출력

```
{'genres_names': ['punk-rock', 'anime'],
 'id': 178,
 'images': [{'height': 640,
             'url': 'https://i.scdn.co/image/ab6761610000e5eb59f8cfc8e71dcaf8c6ec4bde',
             'width': 640},
            {'height': 320,
             'url': 'https://i.scdn.co/image/ab676161000051745 9f8cfc8e71dcaf8c6ec4bde',
             'width': 320},
            {'height': 160,
             'url': 'https://i.scdn.co/image/ab6761610000f17859f8cfc8e71dcaf8c6ec4bde',
             'width': 160}],
 'name': 'Jimin',
 'type': 'artist'}
```

<그림 11> 요구사항 번호 F113 결과물 예시

3) 다중 데이터 분석 및 수정

① 요구사항 번호 : F114, F115

② 앞선 요구사항의 코드를 활용하여 주어진 spotify/problem_c.py 를 수정하여 구현

- 필요한 정보 : id, name, images, type, genre_names
- artists.json 파일에는 여러 아티스트에 대한 데이터 제공됨. 각 아티스트에 대한 데이터 중 필요한 정보를 추출하여 Dictionary 로 변환한 다음, 해당 Dictionary 들을 담고 있는 새로운 List 를 반환하는 함수 artist_info 를 작성
- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary 의 key 순서가 정렬되어서 출력

```
[{'genres_names': ['acoustic', 'electro', 'j-rock'],
  'id': 451,
  'images': [{'height': 640,
              'url': 'https://i.scdn.co/image/ab6761610000e5ebd642648235ebf3460d2d1f6a',
              'width': 640},
             {'height': 320,
              'url': 'https://i.scdn.co/image/ab67616100005174d642648235ebf3460d2d1f6a',
              'width': 320},
             {'height': 160,
              'url': 'https://i.scdn.co/image/ab6761610000f178d642648235ebf3460d2d1f6a',
              'width': 160}],
  'name': 'BTS',
  'type': 'artist'},
 {'genres_names': ['edm'],
  'id': 116,
  'images': [{'height': 640,
              'url': 'https://i.scdn.co/image/ab6761610000e5eb6199c3c2f414880e2b9077a9',
              'width': 640},
             {'height': 320,
              'url': 'https://i.scdn.co/image/ab67616100051746199c3c2f414880e2b9077a9',
              'width': 320},
             {'height': 160,
              'url': 'https://i.scdn.co/image/ab6761610000f1786199c3c2f414880e2b9077a9',
              'width': 160}],
  'name': 'NewJeans',
  'type': 'artist'}]
```


<그림 12> 요구사항 번호 F114, F115 결과물 예시

4) 가장 인기도가 높은 아티스트 데이터 수집

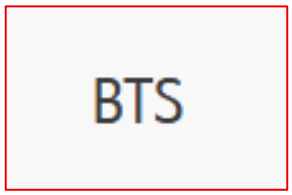
① 요구사항 번호 : F116, F117

② 앞선 요구사항의 코드를 활용하여 주어진 spotify/problem_d.py 를 수정하여 구현

- artist 폴더에는 <아티스트 id>.json 형식을로 각 아티스트에 대한 정보가 제공되어 있으며, 아티스트의 인기도는 popularity 에 기록되어 있음
- artists.json 파일에 제공된 각 아티스트 정보의 id 를 활용하여, 해당 아티스트의 인기도를 확인하고, 이 중 인기도가 가장 높은 아티스트의 이름을 출력하는 함수 max_popularity 를 작성

③ 주어진 spotify/exampes/03_example.py 의 코드를 참고하여 작성

- 반복문을 통해 spotify/data/artists 폴더 내부의 파일들을 열기



<그림 13> 요구사항 번호 F116, F117 결과물 예시

5) 특정 팔로워 수 이상의 아티스트 데이터 수집

① 요구사항 번호 : F118

② 앞선 요구사항의 코드를 활용하여 주어진 spotify/problem_e.py 를 수정하여 구현

- artists 폴더에는 <아티스트 id>.json 형식을로 각 아티스트에 대한 정보가 제공되어 있으며, 아티스트의 팔로워 수는 followers 에 기록되어 있음

- artists.json 파일에 제공된 각 아티스트 정보의 id 를 활용하여, 해당 아티스트의 팔로워 수를 확인하고, 팔로워 수의 총합이 10,000,000 이상인 아티스트 이름과 URI 를 리스트로 출력하는 함수 dec_artists 를 작성

- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary 의 key 순서가 정렬되어서 출력

③ 주어진 spotify/exampes/03_example.py 의 코드를 참고하여 작성

- 반복문을 통해 spotify/data/artists 폴더 내부의 파일들을 열기

```
[{'name': 'BTS', 'uri-id': '3Nrfpe0tUJi4K4DXYWgMUX'},
 {'name': 'BLACKPINK', 'uri-id': '41MozSoPIsD1dJM0CLPjZF'},
 {'name': 'Stray Kids', 'uri-id': '2dIgFjalVxs4ThymZ67YCE'},
 {'name': 'j-hope', 'uri-id': '0b1sIQumIAsNbqAoIClSpy'}]
```

<그림 14> 요구사항 번호 F118 결과물 예시

4. [spotify] 심화 기능

기본 기능 구현 후, 생성형 AI 를 사용해 다음 두 심화 과제 요구사항을 해결합니다. 하기된 과제를 전부 완료한 경우, 명시된 요구사항 외 추가 개발을 자유롭게 진행할 수 있습니다.

1) 특정 팔로워 수 구간의 아티스트 데이터 수집

① 요구사항 번호 : F119

② 주어진 spotify/problem_f_1.py 를 수정하여 구현

- 팔로워 수가 5,000,000 이상 ~ 10,000,000 미만인 아티스트들의 이름과 팔로워 수를 수집하는 함수 get_popular 를 작성
- 예시 화면의 경우, pprint 모듈을 활용하여 출력되어 Dictionary 의 key 순서가 정렬되어서 출력

```
[{'followers': 5901644, 'name': 'Jimin'},
 {'followers': 9571638, 'name': 'SEVENTEEN'},
 {'followers': 8041766, 'name': 'IU'},
 {'followers': 8000368, 'name': 'Jung Kook'},
 {'followers': 6278033, 'name': '(G)I-DLE'},
 {'followers': 6785638, 'name': 'NCT DREAM'}]
```

<그림 15> 요구사항 번호 F119 결과물 예시

2) 장르에 따른 아티스트 데이터 수집

③ 요구사항 번호 : F120

④ 주어진 spotify/problem_f_2.py 를 수정하여 구현

- 장르에 acoustic 이 포함된 아티스트의 이름을 수집하는 함수 acoustic_artists 함수를 작성

```
['BTS']
```

<그림 16> 요구사항 번호 F120 결과물 예시

1.5 참고자료

- json - JSON encoder and decoder
  https://docs.python.org/3.11/library/json.html
- pprint - Data pretty printer
  https://docs.python.org/3.11/library/pprint.html

1.6 프로젝트 결과

제출 기한은 진행일 18 시까지이므로 제출 기한을 지킬 수 있도록 합니다. 제출은 GitLab 을 통해서 진행합니다.

- **산출물과 제출**

  1) 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분,
     새로 배운 것들 및 느낀 점을 상세히 기록한 README.md

  2) 완성된 각 문제 별 소스코드 및 실행 화면 캡쳐본

  3) 프로젝트 이름은 01-pjt 로 지정, 각자의 계정에 생성할 것

  4) 각 반 담당 강사님을 Maintainer 로 설정

- 以上