

01 PJT

Python을 활용한 데이터 수집 1

INDEX

Python을 활용한 데이터 수집 1

- Aladin API
- Spotify API

목표

프로젝트 목표

- Python 기본 문법 습득
- 파일 입출력에 대한 이해
- 데이터 구조에 대한 분석과 이해
- 데이터를 가공하고 JSON 형태로 구성하기

Aladin API

준비사항

개발도구 및 라이브러리

- 개발도구
 - Visual Studio Code
 - Python 3.9+
- 필수 라이브러리
 - <https://docs.python.org/3.9/library/json.html>

요구사항

공통 요구사항

- 커뮤니티 서비스 개발을 위한 데이터 구성 단계로, 필요한 도서 데이터를 직접 추출하고 구성하는 과정입니다.
- 반드시 제공된 examples/ 폴더의 예시 파일을 먼저 참고합니다.
 - 예시 파일에는 이번 프로젝트 해결을 위해 알아야 하는 혹은 직접적인 도움이 될 수 있는 코드가 작성되어 있습니다.

A. 제공되는 도서 데이터의 주요내용 수집 (problem_a)

- 샘플 도서 데이터(book.json)가 주어집니다.

이중 서비스 구성에 필요한 정보만 추출해 반환하는 함수를 단계적으로 완성합니다.

완성된 함수는 다음 문제의 기본 기능으로 사용됩니다.

1. 데이터

- 제공되는 book.json을 활용합니다.
- book.json은 도서 '노인과 바다' 정보를 담고 있습니다.

A. 제공되는 도서 데이터의 주요내용 수집 (problem_a)

2. 풀이

- 필요한 정보
 - id, name, author, priceSales, description, cover, categoryId
 - book.json에서 필요한 정보에 해당하는 값을 추출합니다.
 - 필요한 정보를 새로운 dictionary로 반환하는 함수 book_info를 완성합니다.

A. 제공되는 도서 데이터의 주요내용 수집 (problem_a)

3. 결과

- problem_a.py 실행 예시

```
{'author': '어니스트 헤밍웨이 (지은이), 김욱동 (옮긴이)',  
 'categoryId': [151128, 50919],  
 'cover': 'https://image.aladin.co.kr/product/1452/24/coversum/8937462788_2.jpg',  
 'description': '노벨 문학상, 풀리처상 수상 작가, 20세기 미국 문학을 개척한 작가 어니스트 헤밍웨이의 대표작. 미국 현대  
 "문학의 개척자라 불리는 헤밍웨이는 제1차 세계대전 후 삶의 좌표를 잊어버린 '길 잃은 세대'를 대표하는 "  
 "작가이다. '민음사 세계문학 전집' 278권으로 출간된 <노인과 바다>는 헤밍웨이의 마지막 소설로, "  
 '작가 고유의 소설 수법과 실존 철학이 짧은 분량 안에 집약되어 있다.',  
 'id': 14522431,  
 'priceSales': 7200,  
 'title': '노인과 바다'}
```

주의) [pprint](#) 함수로 인해 dictionary의 key 순서가 정렬되어서 출력됩니다.

B. 제공되는 도서 데이터의 주요내용 수정 (problem_b)

- 이전 단계에서 만들었던 데이터 중 categoryId를 카테고리 번호가 아닌 카테고리 이름 리스트 categoryName으로 바꿔 반환하는 함수를 완성합니다. 완성된 함수는 다음 문제의 기본 기능으로 사용됩니다.

1. 데이터

- 제공되는 book.json, categories.json 을 활용합니다.
- categories.json은 모든 장르의 id, name 정보를 담고 있습니다.

B. 제공되는 도서 데이터의 주요내용 수정 (problem_b)

2. 풀이

- 필요한 정보
 - id, name, author, priceSales, description, cover, categoryId
 - book.json에서 필요한 정보에 해당하는 값을 추출합니다.
 - 이때, categoryName대신 categoryId를 추출합니다.
 - categories.json을 이용하여 categoryId를 각 장르 번호에 맞는 name 값으로 대체한 categoryName 키를 생성합니다.
 - 위 요구사항을 반영한 새로운 dictionary를 반환하는 함수 book_info를 완성합니다.

B. 제공되는 도서 데이터의 주요내용 수정 (problem_b)

3. 결과

- problem_b.py 실행 예시

```
{'author': '어니스트 헤밍웨이 (지은이), 김육동 (옮긴이)',  
 'categoryName': ['문학', '영미소설'],  
 'cover': 'https://image.aladin.co.kr/product/1452/24/coversum/8937462788_2.jpg',  
 'description': '노벨 문학상, 폴리처상 수상 작가, 20세기 미국 문학을 개척한 작가 어니스트 헤밍웨이의 대표작. 미국 현대 '문학의 개척자'라 불리는 헤밍웨이는 제1차 세계대전 후 삶의 좌표를 잊어버린 '길잃은 세대'를 대표하는 "작가이다. '민음사 세계문학전집' 278권으로 출간된 <노인과 바다>는 헤밍웨이의 마지막 소설로, "작가 고유의 소설 수법과 실존 철학이 짧은 분량 안에 집약되어 있다.',  
 'id': 14522431,  
 'priceSales': 7200,  
 'title': '노인과 바다'}
```

주의) [pprint](#) 함수로 인해 dictionary의 key 순서가 정렬되어서 출력됩니다.

C. 다중 데이터 분석 및 수정 (problem_c)

- books.json에는 평점이 높은 20개의 도서 데이터가 주어집니다.
이중 서비스 구성에 필요한 정보만 반환하는 함수를 완성합니다.
완성된 함수는 향후 커뮤니티 서비스에서 제공되는 추천 도서 목록을
제공하기 위한 기능으로 사용됩니다.

1. 데이터

- 제공되는 books.json, categories.json 을 활용합니다.
- books.json은 전체 도서 정보를 담고 있습니다.

C. 다중 데이터 분석 및 수정 (problem_c)

2. 풀이

- 필요한 정보
 - id, name, author, priceSales, description, cover, categoryName
- 이전 단계의 함수 구조를 재사용합니다.
- 위 요구사항을 반영한 새로운 list를 반환하는 함수 books_info를 완성합니다.

C. 다중 데이터 분석 및 수정 (problem_c)

3. 결과

- problem_c.py 실행 예시

```
[{'author': '제인 오스틴 (지은이), 윤지관, 전승희 (옮긴이)',  
 'categoryName': ['문학', '영미소설'],  
 'cover': 'https://image.aladin.co.kr/product/43/68/coversum/8937460882_3.jpg',  
 'description': '제인 오스틴의 대표작 <오만과 편견>이 보다 정확하고 말끔한 번역으로 재출간됐다. 역자인 윤지관과 전승희는  
 '10여 년에 걸친 기간 동안 철저한 원문대조를 통해, 본래의 의미와 문체를 생생하게 되살려내기 위해 '  
 '노력했다고.',  
 'id': 436838,  
 'priceSales': 11700,  
 'title': '오만과 편견'},  
 {'author': '어니스트 헤밍웨이 (지은이), 김욱동 (옮긴이)',  
 'categoryName': ['문학', '영미소설'],  
 'cover': 'https://image.aladin.co.kr/product/1452/24/coversum/8937462788_2.jpg',  
 'description': '노벨문학상, 폴리처상 수상 작가, 20세기 미국 문학을 개척한 작가 어니스트 헤밍웨이의 대표작. 미국 현대  
 '문학의 개척자라 불리는 헤밍웨이는 제1차 세계대전 후 삶의 좌표를 잊어버린 '길잃은 세대'를 대표하는 "  
 "작가이다. '민음사 세계문학전집' 278권으로 출간된 <노인과 바다>는 헤밍웨이의 마지막 소설로, "  
 '작가 고유의 소설 수법과 실존 철학이 짧은 분량 안에 집약되어 있다.',  
 'id': 14522431,  
 'priceSales': 7200,  
 'title': '노인과 바다'},  
 {'author': '서머싯 몽 (지은이), 송무 (옮긴이)',  
 'categoryName': ['문학', '영미소설'],  
 'cover': 'https://image.aladin.co.kr/product/23/77/coversum/s692639624_1.jpg',  
 'description': '<달과 6펜스>는 15종에 이르는 번역본이 이미 소개되어 있을 만큼 국내에서 크게 환영받는 작품이다. 이  
 '작품은 서머싯 몽을 전세계에 널리 알린 결정적인 작품으로 제1차 세계대전이 끝난 이듬해인 1919년에 '  
 '출판되어 대단한 인기를 끌었다.'},  
 'id': 237763,  
 'priceSales': 9000,  
 '# 이하 생략'
```

D. 알고리즘을 사용한 데이터 출력 (problem_d)

- 도서 회원 리뷰 평점 정보(customerReviewRank)를 이용하여 모든 도서 중 리뷰 평점이 가장 높은 도서를 출력하는 알고리즘을 작성합니다. 해당 과정은 향후 커뮤니티 서비스에서 회원 리뷰 평점 순으로 도서를 정렬하여 출력하는 정보로 사용됩니다.

1. 데이터

- 제공되는 books.json과 books 폴더 내부의 파일들을 활용합니다.
- books 폴더 내부의 파일들은 각 도서의 세부 정보를 가지고 있습니다.

D. 알고리즘을 사용한 데이터 출력 (problem_d)

2. 풀이

- 반복문을 통해 books 폴더 내부의 파일들을 오픈해야 합니다.
 - 03_example 또는 파일 하단의 코드를 참고합니다.
- books 폴더 내부의 json에서 리뷰 평점이 가장 높은 도서의 제목을 출력하는 함수 best_book을 완성합니다.

D. 알고리즘을 사용한 데이터 출력 (problem_d)

3. 결과

- problem_d.py 실행 예시

멋진 신세계

E. 알고리즘을 사용한 데이터 출력 (problem_e)

- 도서 세부 정보 중 출판연도(pubDate)를 이용하여 모든 도서 중 2023년에 출판한 도서들의 제목 리스트를 출력하는 알고리즘을 작성합니다. 해당 과정은 향후 커뮤니티 서비스에서 추천 기능의 일부로 사용됩니다.

1. 데이터

- 제공되는 books.json과 books 폴더 내부의 파일들을 활용합니다.
- books 폴더 내부의 파일들은 각 도서의 세부 정보를 가지고 있습니다.

E. 알고리즘을 사용한 데이터 출력 (problem_e)

2. 풀이

- 반복문을 통해 books 폴더 내부의 파일들을 오픈해야 합니다.
 - 03_example 또는 파일 하단의 코드를 참고합니다.
- 출판연도가 2023년인 도서들의 제목을 리스트로 출력하는 함수 new_books를 완성합니다.
 - 02_example 코드를 참고합니다.

E. 알고리즘을 사용한 데이터 출력 (problem_e)

3. 결과

- problem_e.py 실행 예시

['회복탄력성의 힘 - 쉽게 포기하지 않고 결국 해내는 아이의 비밀 ', '사피엔스 - 유인원에서 사이보그까지 , 인간 역사의 대담하고 위대한 질문 ']

F. 선택 과제

- 제공된 도서 데이터를 사용하여 내가 원하는 데이터를 추출하고 나만의 데이터 구조를 만들어봅니다.
- 예시
 - [problem_f_1.py] 2023년 출판 도서 중 회원 리뷰 평점이 가장 높은 도서 제목
 - [problem_f_2.py] 카테고리가 컴퓨터 공학인 도서들을 판매가격(priceSales)이 높은 순서대로 제목 정렬한 리스트

F. 선택 과제

3. 결과

- problem_f_1.py 실행 예시

사피엔스 - 유인원에서 사이보그까지, 인간 역사의 대담하고 위대한 질문

- problem_f_2.py 실행 예시

['한 권으로 읽는 컴퓨터 구조와 프로그래밍 - 더 나은 소프트웨어 개발을 위한 하드웨어, 자료구조, 필수 알고리즘 등 프로그래머의 비밀 노트 ', '클린 코드 Clean Code - 애자일 소프트웨어 장인 정신 ']

Spotify API

준비사항

개발도구 및 라이브러리

- 개발도구
 - Visual Studio Code
 - Python 3.9+
- 필수 라이브러리
 - <https://docs.python.org/3.9/library/json.html>

요구사항

공통 요구사항

- 커뮤니티 서비스 개발을 위한 데이터 구성 단계로, 필요한 아티스트 데이터를 직접 추출하고 구성하는 과정입니다.
- 반드시 제공된 examples/ 폴더의 예시 파일을 먼저 참고합니다.
 - 예시 파일에는 이번 프로젝트 해결을 위해 알아야 하는 혹은 직접적인 도움이 될 수 있는 코드가 작성되어 있습니다.

A. 제공되는 아티스트 데이터의 주요내용 수집 (problem_a)

- 샘플 아티스트 데이터(artist.json)가 주어집니다.

이중 서비스 구성에 필요한 정보만 추출해 반환하는 함수를 단계적으로 완성합니다.

완성된 함수는 다음 문제의 기본 기능으로 사용됩니다.

1. 데이터

- 제공되는 artist.json을 활용합니다.
- artist.json은 아티스트 'Jimin' 정보를 담고 있습니다.

A. 제공되는 아티스트 데이터의 주요내용 수집 (problem_a)

2. 풀이

- 필요한 정보
 - id, name, genres_ids, images, type
 - artist.json에서 필요한 정보에 해당하는 값을 추출합니다.
 - 필요한 정보를 새로운 dictionary로 반환하는 함수 artist_info를 완성합니다.

A. 제공되는 아티스트 데이터의 주요내용 수집 (problem_a)

3. 결과

- problem_a.py 실행 예시

```
{'genres_ids': [651, 816],  
 'id': 178,  
 'images': [{  
     'height': 640,  
     'url': 'https://i.scdn.co/image/ab6761610000e5eb59f8cf8e71dcaf8c6ec4bde',  
     'width': 640},  
     {'height': 320,  
     'url': 'https://i.scdn.co/image/ab6761610000517459f8cf8e71dcaf8c6ec4bde',  
     'width': 320},  
     {'height': 160,  
     'url': 'https://i.scdn.co/image/ab6761610000f17859f8cf8e71dcaf8c6ec4bde',  
     'width': 160}],  
 'name': 'Jimin',  
 'type': 'artist'}
```

다음과 같이

주의) [pprint](#) 함수로 인해 dictionary의 key 순서가 정렬되어서 출력됩니다.

B. 제공되는 아티스트 데이터의 주요내용 수정 (problem_b)

- 이전 단계에서 만들었던 데이터 중 genres_ids를 장르 번호가 아닌 장르 이름 리스트 genres_names로 바꿔 반환하는 함수를 완성합니다.
완성된 함수는 다음 문제의 기본 기능으로 사용됩니다.

1. 데이터

- 제공되는 artist.json, genres.json 을 활용합니다.
- genres.json은 모든 장르의 id, name 정보를 담고 있습니다.

B. 제공되는 아티스트 데이터의 주요내용 수정 (problem_b)

2. 풀이

- 필요한 정보
 - id, name, images, type, genres_names
- artist.json에서 필요한 정보에 해당하는 값을 추출합니다.
 - 이때, genres_names대신 genres_ids를 추출합니다.
- genres.json을 이용하여 genres_ids를 각 장르 번호에 맞는 name 값으로 대체한 genres_names 키를 생성합니다.
- 위 요구사항을 반영한 새로운 dictionary를 반환하는 함수 artist_info를 완성합니다.

B. 제공되는 아티스트 데이터의 주요내용 수정 (problem_b)

3. 결과

- problem_b.py 실행 예시

```
{'genres_names': ['punk-rock', 'anime'],
 'id': 178,
 'images': [{['height': 640,
            'url': 'https://i.scdn.co/image/ab676161000e5eb59f8cfc8e71dcaf8c6ec4bde',
            'width': 640},
            {'height': 320,
            'url': 'https://i.scdn.co/image/ab676161000517459f8cfc8e71dcaf8c6ec4bde',
            'width': 320},
            {'height': 160,
            'url': 'https://i.scdn.co/image/ab676161000f17859f8cfc8e71dcaf8c6ec4bde',
            'width': 160}],
 'name': 'Jimin',
 'type': 'artist'}
```

주의) [pprint](#) 함수로 인해 dictionary의 key 순서가 정렬되어서 출력됩니다.

C. 다중 데이터 분석 및 수정 (problem_c)

- artists.json에는 k-pop 장르 20명의 아티스트 데이터가 주어집니다.
이 중 서비스 구성에 필요한 정보만 추출해 반환하는 함수를 완성합니다.
완성된 함수는 향후 커뮤니티 서비스에서 제공되는 아티스트 목록을 제공하기 위한 기능
으로 사용됩니다.

1. 데이터

- 제공되는 artists.json, genres.json 을 활용합니다.
- artists.json은 전체 아티스트 정보를 담고 있습니다.

C. 다중 데이터 분석 및 수정 (problem_c)

2. 풀이

- 필요한 정보
 - id, name, images, type, genres_names
- genres_names 대신 genres_ids를 추출합니다.
 - 이전 단계의 함수 구조를 재사용합니다.
- 위 요구사항을 반영한 새로운 list를 반환하는 함수 artist_info를 완성합니다.

C. 다중 데이터 분석 및 수정 (problem_c)

3. 결과

- problem_c.py 실행 예시

```
[{'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/ab676161000051746199c3c2f414880e2b9077a9',
               'width': 320},
              {'height': 160,
               'url': 'https://i.scdn.co/image/ab6761610000f1786199c3c2f414880e2b9077a9',
               'width': 160}],
   'name': 'NewJeans',
   'type': 'artist'},
   # 이하 생략
]
```

D. 알고리즘을 사용한 데이터 출력 (problem_d)

- 아티스트 세부 정보 중 인기도(popularity)를 이용하여 모든 아티스트 중 가장 높은 인기도를 가진 아티스트를 출력하는 알고리즘을 작성합니다. 해당 과정은 향후 커뮤니티 서비스에서 인기도순으로 아티스트를 정렬하여 출력하는 정보로 사용됩니다.

1. 데이터

- 제공되는 artists.json과 artists 폴더 내부의 파일들을 활용합니다.
- artists 폴더 내부의 파일들은 각 아티스트의 세부 정보를 가지고 있습니다.

D. 알고리즘을 사용한 데이터 출력 (problem_d)

2. 풀이

- 반복문을 통해 artists 폴더 내부의 파일들을 오픈해야 합니다.
 - 03_example 또는 파일 하단의 코드를 참고합니다.
- 제공된 아티스트 데이터에서 인기도(popularity)가 같은 아티스트는 없습니다.
- 인기도가 가장 높은 아티스트의 이름을 출력하는 함수 max_popularity를 완성합니다.

D. 알고리즘을 사용한 데이터 출력 (problem_d)

3. 결과

- problem_d.py 실행 예시

NewJeans

E. 알고리즘을 사용한 데이터 출력 (problem_e)

- 아티스트 세부 정보 중 팔로워 수(followers)를 이용하여 모든 아티스트 중 팔로워 수의 총합(total)이 10,000,000이상인 아티스트의 정보를 리스트로 출력하는 알고리즘을 작성합니다. 해당 과정은 향후 커뮤니티 서비스에서 추천 기능의 일부로 사용됩니다.

1. 데이터

- 제공되는 artists.json과 artists 폴더 내부의 파일들을 활용합니다.
- artists 폴더 내부의 파일들은 각 아티스트의 세부 정보를 가지고 있습니다.

E. 알고리즘을 사용한 데이터 출력 (problem_e)

2. 풀이

- 반복문을 통해 artists 폴더 내부의 파일들을 오픈해야 합니다.
 - 03_example 또는 파일 하단의 코드를 참고합니다.
- 팔로워 수의 총 합이 10,000,000이상인 아티스트들의 이름(name)과 uri-id를 출력하는 함수 dec_artists를 완성합니다.
 - uri-id는 아티스트 세부정보에서 uri값 중 두번째 세미콜론 뒤의 값을 말합니다.
예시: spotify:artist:**6HvZYsbFfjnjFrWF950C9d**
 - 02_example 코드를 참고합니다.

E. 알고리즘을 사용한 데이터 출력 (problem_e)

3. 결과

- problem_e.py 실행 예시

```
[{'name': 'BTS', 'uri-id': '3Nrfpe0tUJi4K4DXWgMUX'},  
 {'name': 'BLACKPINK', 'uri-id': '41MozSoPIsD1dJM0CLPjZF'},  
 {'name': 'Stray Kids', 'uri-id': '2dIgFjalVxs4ThymZ67YCE'},  
 {'name': 'j-hope', 'uri-id': '0b1sIQumIAsNbqAoIClSpy'}]
```

F. 선택 과제

- 제공된 아티스트 데이터를 사용하여 내가 원하는 데이터를 추출하고 나만의 데이터 구조로 만들어봅니다.
- 예시
 - [problem_f_1.py] 팔로워가 5,000,000이상, 10,000,000미만인 아티스트들을 아티스트 이름과 팔로워를 목록으로 출력하기
 - [problem_f_2.py] 장르에 acoustic이 포함된 아티스트 이름 목록 출력하기

F. 선택 과제

결과

- problem_f_1.py 실행 예시

```
[{'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'}]
```

- problem_f_2.py 실행 예시

```
['BTS']
```

제출

제출 시 주의사항

- 제출기한은 금일 18시까지입니다. 제출기한을 지켜 주시기 바랍니다.
- 반드시 README.md 파일에 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낀 점 등을 상세히 기록하여 제출합니다.
 - 단순히 완성된 코드만을 나열하지 않습니다.
- 위에 명시된 요구사항은 최소 조건이며, 추가 개발을 자유롭게 진행할 수 있습니다.
- <https://lab.ssafy.com/>에 프로젝트를 생성하고 제출합니다.
 - 프로젝트 이름은 '프로젝트 번호 + pjt'로 지정합니다 (ex. **01-pjt**)
- 반드시 각 반 담당 교수님을 Maintainer로 설정해야 합니다.