Week 2 Homework #1-1 Report

- 작성자 : B889047 윤준호
- 작성일 : 2022 / 09 / 19, Mon

아래 사이트에서 해당 내용을 공부하여 간단 정리, 그리고 코드를 (가능하면) python tutor/ 본인의 개발 환경에서 수행한 snapshot 추가. (필요하면 Chrome 영어 번역 기능 활용)

- (DocString) https://cs50.harvard.edu/python/2022/notes/9/#docstrings (https://cs50.harvard.edu/python/2022/notes/9/#docstrings)
- (Random, Sys) https://cs50.harvard.edu/python/2022/notes/4/ (https://cs50.harvard.edu/python/2022/notes/4/)

About Doc-String

- `Doc-String` 은 함수, 클래스, 메소드의 목적이나, 설명을 기술하기 위한 가장 표준적인 방법이다. 예시 코드 작성해보면 아래와 같이 작성할 수 있다.
- `Doc-String` 을 작성하기 위해서는 Python문법중 `Multiline-String` 을 사용해 주어야 한다(`"""` 혹은 `'''`)
- `Doc-String` 에는 parameter에 대한 정보, parameter타입, 어떤 예외가 발생할 수 있는지, 반환값에 대한 정보와 반환 값의 타입등의 정보들이 적혀있다.
- `Doc-String` 에 관한 Convention은 PEP-257 (https://peps.python.org/pep-0257/)에 명시되어있다.
- `Doc-String` 을 출력을 하기 위해서는 아래 두가지 방법이 존재한다
  - help() 내장함수 사용 : help안에 함수 혹은 클래스 객체 자체를 넘겨주어야 한다
  - .__doc__ 매직메소드 (https://wikidocs.net/83755) 사용

In [1]:

```
from typing import Callable,Any
```

In [2]:

```
# Function

def meow(n:int) -> None:
    """
    Meow n times

    :param n: Number of times to meow
    :type n: int
    :raise TypeError: If n is not an int
    :return: A string of n meows, one per line
    :rtype: str
    """
    return "meow\n" * n

number:int = int(input("Numbers : "))
meows:str = meow(number)
print(meows,end="")
```

```
Numbers : 5
meow
meow
meow
meow
meow
```

In [3]:

```
help(meow)
```

```
Help on function meow in module __main__:

meow(n: int) -> None
    Meow n times

    :param n: Number of times to meow
    :type n: int
    :raise TypeError: If n is not an int
    :return: A string of n meows, one per line
    :rtype: str
```

In [4]:

```
meow.__doc__
```

Out[4]:

```
'\n    Meow n times\n\n    :param n: Number of times to meow\n    :typ
e n: int\n    :raise TypeError: If n is not an int\n    :return: A str
ing of n meows, one per line\n    :rtype: str\n    '
```

In [5]:

```
class calculate(object):
    """
    calculate : calculate integer

    multiply

    :param x: variable x
    :param y: varaible y
    :type x: int
    :type y: int
    :raise TypeError: If parameter is not type of int
    :return: result of multiply
    :rtype: int
    """

    def __init__(self):
        pass

    def multiply(self,x:int,y:int) -> int:
        """
        multiply two numbers

        :param x: variable x
        :param y: varaible y
        :type x: int
        :type y: int
        :raise TypeException: If parameter is not type of int
        :return: result of multiply
        :rtype: int
        """
        return x * y
```

In [6]:

```
help(calculate)
```

```
Help on class calculate in module __main__:

class calculate(builtins.object)
 |  calculate : calculate integer
 |
 |  multiply
 |
 |  :param x: variable x
 |  :param y: varaible y
 |  :type x: int
 |  :type y: int
 |  :raise TypeError: If parameter is not type of int
 |  :return: result of multiply
 |  :rtype: int
 |
 |  Methods defined here:
 |
 |  __init__(self)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  multiply(self, x: int, y: int) -> int
 |      multiply two numbers
 |
 |      :param x: variable x
 |      :param y: varaible y
 |      :type x: int
 |      :type y: int
 |      :raise TypeException: If parameter is not type of int
 |      :return: result of multiply
 |      :rtype: int
 |
 |  ----------------------------------------------------------------------
----
 |  Data descriptors defined here:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
```

In [7]:

```
help(calculate.multiply)
```

```
Help on function multiply in module __main__:

multiply(self, x: int, y: int) -> int
    multiply two numbers

    :param x: variable x
    :param y: varaible y
    :type x: int
    :type y: int
    :raise TypeException: If parameter is not type of int
    :return: result of multiply
    :rtype: int
```

About Random

- `Random` 파이썬에 기본적으로 탑재된 내장 모듈이다. 즉, 따로 설치 없이 `import` 를 통해 바로 사용할 수 있다는 것이다.
- `Random` 모듈 안에는 내장함수로 `random.choice(seq)` 가 존재한다. 여기서 매개변수로 가지는 `seq` 는 `sequence` 를 의미한다. 한번 `random.choice()` 를 사용해 보자

In [8]:

```
import random

coin = random.choice(["heads","tails"])
print(coin)
```

```
heads
```

- `Sequence` 를 매개변수로 받기 때문에 리스트 뿐만 아니라 튜플을 인자로 넣어줘도 된다.

In [9]:

```
coin2 = random.choice(("heads","tails"))
print(coin)
```

```
heads
```

- 추가적으로 `random` 을 `import` 하는 방식을 변경하여 코드를 개선해 봅니다. 아래와 같이 import만 해서 모듈을 사용시, `(모듈).(함수)` 형태로 사용하지만 `from` 키워드를 사용하면 원하는 내장함수만 선언하여 사용할 수 있다.

```
# 기존 방식
import random

# 개선 방식
from random import choice

# 만약 해당 모듈의 모든 함수를 쓰고 싶다면
from random import *
```

In [10]:

```
from random import choice

coin = choice(["heads","tails"])
coin
```

Out[10]:

```
'heads'
```

- `random.randint(a,b)` 를 사용해 봅니다. `randint(a,b)` 는 a와 b 사이에 있는 숫자를 랜덤으로(a,b포함) 선택해서 반환해 줍니다

In [11]:

```
from random import randint

number = random.randint(1,10)
print(number)
```

```
9
```

- `random.randint(a,b)` 를 응용해서 10개의 랜덤 넘버를 포함한 리스트를 반환해 봅니다

In [12]:

```
randomnumbers = [ randint(1,10) for _ in range(10)]
print(*randomnumbers,end=" ")
```

```
8 5 8 3 10 3 10 2 7 2
```

- `random.shuffle(x)` 를 사용해 봅니다. `random.shuffle` 은 리스트 안에 있는 값들을 무작위로 섞어서 반환해 줍니다. 위에서 보았던 다른 함수들과 달리 새로운 값을 반환하는것이 아닌 기존, 값(리스트)을 변경하는방식입니다.

In [13]:

```
import random


def make_cardset_shuffle(cards:list) -> None:
    random.shuffle(cards)

cards_list = ["jack","queen","king"]
make_cardset_shuffle(cards_list)

print(*cards_list,sep="\n")
```

```
queen
king
jack
```

- 기존 값을 변경하는 방식이기 때문에, 불변 속성을 가진 시퀀스 타입인 튜플로는 작동하지 않습니다

In [14]:

```
cards_tuple = ("jack","queen","king")
make_cardset_shuffle(cards_tuple)

print(*cards_tuple,sep="\n")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call
 last)
Input In [14], in <module>
      1 cards_tuple = ("jack","queen","king")
----> 2 make_cardset_shuffle(cards_tuple)
      4 print(*cards_tuple,sep="\n")

Input In [13], in make_cardset_shuffle(cards)
      4 def make_cardset_shuffle(cards:list) -> None:
----> 5     random.shuffle(cards)

File /opt/homebrew/Cellar/python@3.9/3.9.13_2/Frameworks/Python.framew
ork/Versions/3.9/lib/python3.9/random.py:362, in Random.shuffle(self,
 x, random)
    359     for i in reversed(range(1, len(x))):
    360         # pick an element in x[:i+1] with which to exchange x
[i]
    361         j = randbelow(i + 1)
--> 362         x[i], x[j] = x[j], x[i]
    363 else:
    364     _warn('The *random* parameter to shuffle() has been deprec
ated\n'
    365           'since Python 3.9 and will be removed in a subsequen
t '
    366           'version.',
    367           DeprecationWarning, 2)

TypeError: 'tuple' object does not support item assignment
```

About sys - Command Line Arguments

- 우리가 일반적으로 command line, 즉 cli환경에서 코드를 실행할때 아래 예시에서 첫번째와 같이 명령어를 입력한다. 두번째 형태 같은 경우에 파일 이외 다른 값들을 넘겨주고는 한다. 저러한 것들을 우리는 `parameter` 라고 부른다.

  ```
  python3 example.py

  python3 example.py "example string" 23
  ```

- `sys` 모듈을 이용하면, 이 실행시 매개변수를 사용해 줄 수 있다. `sys.argv` 클래스 변수를 사용해서 매개변수를 들고 올 수 있다. 한가지 짚고 넘어갈 점은 `sys.argv` 는 `list` 타입의 변수이며, 0번째 index는 실행파일의 이름이 담긴다. 즉 사실상 사용자가 입력한 매개변수는 Index 1번부터라는 것이다. 예를 들어 argumet_example폴더에 example.py라는 파이썬 폴더에 아래와 같이 코드를 입력했다고 가정하자.

```
import sys

print(sys.argv)
print(f"Execution file name : {sys.argv[0]}")
print(f"My name is {sys.argv[1]}")
```

In [15]:

```
!python3 argument_example/example.py Hoplin
```

```
['argument_example/example.py', 'Hoplin']
Execution file name : argument_example/example.py
My name is Hoplin
```

- 만약 사용자가 매개변수를 입력하지 않는 경우가 발생할 수 도 있다. 그런 경우에 코드 런타임 관점에서 두가지 해결방안이 있다.
  - `IndexError` 가 날 것이므로, 해당 부분에 대한 예외처리 후 종료
  - `IndexErrr` 검사를 한 후 기본값으로 대체
- example2, example3 각각에 두가지 경우를 모두 작성해 보자.

```
# Case  : example2.py

import sys

try:
    print(f"My name is {sys.argv[1]}")
except IndexError as e:
    print("Argument not found")
```

In [16]:

```
!python3 argument_example/example2.py
```

```
Argument not found
```

```
# Case 2

import sys

name = "Default Name"
try:
    name = sys.argv[1]
except IndexError:
    pass

print(f"My name is {name}")
```

In [17]:

```
!python3 argument_example/example3.py
```

```
My name is Default Name
```

- `argument` 는 기본적으로 하나를 가지고 시작한다. 그렇기 때문에 매개변수를 하나 받는다고 하면 `argument` 에 두개가 들어오게 되는것이다. 이 성질을 통해서 사용자가 `argument` 를 입력하지 않았을때에 대한 대응을 해줄 수 있다.

- `sys.argv` 는 리스트 형태이므로, 길이를 바탕으로 입력의 여부를 확인해 줄 수 있다. 아래 예시를 살펴보자 아래 예시는 총 두개의 `argument` 가 필요한 예시이다. 코드를 작성하기 전에 살펴봐야할 것은, 그럼 `sys.argv` 안에는 총 3개의 원소가 들어있어야 한다. 하나는 실행파일 이름, 나머지 두개는 입력 `argument` 가 되는것이다. 그렇기에, `sys.argv` 의 길이가 3개 미만, 혹은 3개 초과인 경우에 대해 잘못된 값이라고 예외를 발생시킬 수 있다
- 여기에 추가적으로 첫번째 argument는 string타입, 두번째 argument는 int타입이라는 제약이 있다고 가정하자.

```
import sys

try:
    if len(sys.argv) < 3 or len(sys.argv) > 3:
        raise IndexError("Argument count not matched")
    _,name,age = sys.argv
    try:
        age = int(age)
    except ValueError as e:
        print("Argument type not matched")
        sys.exit()
    print(f"My name is {sys.argv[1]} and my age is {sys.argv[2]}")
except IndexError as e:
    print(e.args[0])
```

In [18]:

```
!python3 argument_example/example4.py
```

```
Argument count not matched
```

In [19]:

```
!python3 argument_example/example4.py 20 example
```

```
Argument type not matched
```

In [20]:

```
!python3 argument_example/example4.py Hoplin 24
```

```
My name is Hoplin and my age is 24
```

- 위에서는 `sys.argv` 의 성질을 살펴보았다. 우리가 우선적으로 알 수 있는것은, `sys.argv` 는 클래스 변수이며, `list` 타입이라는 것이다.

```
import sys

if len(sys.argv) < 2:
    sys.exit("Too few arguments")

for arg in sys.argv:
    print("hello, my name is", arg)
```

In [21]:

```
!python3 argument_example/example5.py 'alice' 'sam' 'hoplin' 'smith' 'edward'
```

```
hello, my name is argument_example/example5.py
hello, my name is alice
hello, my name is sam
hello, my name is hoplin
hello, my name is smith
hello, my name is edward
```

- 여기서 하나 더 생각할 수 있는것은 `sys.argv` 는 `list` 타입이므로 `slice` 가 가능하다는 것이다. 그렇다면 `slice` 를 어떻게 활용할 수 있을까? 우선 우리가 argument를 받을때 0번째 index는 필요 없다. 이유는 위 예시에서도 볼 수 있듯이 파일 명을 의미하기 때문이다. 그렇기 때문에, 시작할 때 sys.argv를 1번째 index부터 슬라이싱 한거로 치환해줄 수 도 있다.
- 예시코드로 정해지지 않은 여러개의 argument를 받는 코드라고 가정을 해보자. 우리가 유의해야할 점은, 여러개의 argument를 받는 코드이므로, `sys.argv` 는 최소 두개 이상의 원소를 가지고 있어야 한다. 그럼 아래와 같이 작성해볼 수 있다.

```
import sys

if len(sys.argv) < 2:
    sys.exit("Too few arguments")

for arg in sys.argv[1:]:
    print("hello, my name is", arg)
```

In [22]:

```
!python3 argument_example/example6.py 'alice' 'sam' 'hoplin' 'smith' 'edward'
```

```
hello, my name is alice
hello, my name is sam
hello, my name is hoplin
hello, my name is smith
hello, my name is edward
```

- 조금 더 응용해 보면 아래와 같이 작성해 줄 수 도 있다.

```
import sys

try:
    if len(sys.argv) < 2:
        raise Exception("Argument count should be more than 1")
except Exception as e:
    print(e.args[0])

# 0번째 index를 제외한 나머지로 대체한다
sys.argv = sys.argv[1:]
for i,j in enumerate(sys.argv,start=1):
    print(f"Argument {i} : {j}")
```

In [23]:

```
!python3 argument_example/example7.py 'alice' 'sam' 'hoplin' 'smith' 'edward'
```

```
Argument 1 : alice
Argument 2 : sam
Argument 3 : hoplin
Argument 4 : smith
Argument 5 : edward
```

Packages

- Python이 유명하고, 많이 사용되는 이유중 하나는 강력한 third-party 라이브러리들이 많이 존재하기 때문이고, 이는 `packages` 라는 형태로 제공이 되고, 이는 폴더 형태로 제공이 된다.
- 이러한 패키지들은 PyPI (https://pypi.org/)라는 디렉토리에서 설치가 가능하다. 직접 들어가서 찾을 필요없이 python이 설치될때 같이 다운로드되는 `pip` 라는걸 사용해서 설치해줄 수 있다.
- 한번 `cowsay` 라는 패키지를 설치해보자.

In [24]:

```
!pip3 install cowsay
```

```
Requirement already satisfied: cowsay in /opt/homebrew/lib/python3.9/s
ite-packages (5.0)

[notice] A new release of pip available: 22.2.1 -> 22.2.2
[notice] To update, run: python3.9 -m pip install --upgrade pip
```

- 이제 cowsay를 사용하여 소가 인사하는 간단한 코드를 작성해 보자. 코드는 아래와 같다

```
import cowsay
import sys

if len(sys.argv) == 2:
    cowsay.cow("hello, " + sys.argv[1])
```

In [25]:

```
!python3 argument_example/say.py hoplin
```

```
 _______________
| hello, hoplin |
  =============
              \
               \
                 ^__^
                 (oo)\_______
                 (__)\       )\/\
                     ||----w |
                     ||     ||
```

APIs

- API는 "Application Program Interace"의 약자이다. API는 다른 사람의 코드에 접속하여, 해당 코드로부터 결과값을 돌려받을 수 있다.
- `requests` 패키지를 사용하면, 코드가 웹 브라우저처럼 동작할 수 있도록 도와준다
- 우선 `requests` 를 설치해준다. PyPI에서 requests를 검색해보자 (https://pypi.org/project/requests/ (https://pypi.org/project/requests/))
- `pip` 를 사용해서 `requests` 를 설치해준다

In [26]:

```
!pip3 install requests
```

```
Requirement already satisfied: requests in /opt/homebrew/lib/python3.
9/site-packages (2.26.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/homebrew/lib/pytho
n3.9/site-packages (from requests) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/homeb
rew/lib/python3.9/site-packages (from requests) (2.0.9)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/homebrew/
lib/python3.9/site-packages (from requests) (1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in /opt/homebrew/li
b/python3.9/site-packages (from requests) (2021.10.8)

[notice] A new release of pip available: 22.2.1 -> 22.2.2
[notice] To update, run: python3.9 -m pip install --upgrade pip
```

예시코드 작성

- Apple Itunes는 `https://itunes.apple.com/search?entity={}&limit={}&term={}` 라는 endpoint를 가진 링크를 통해서 정보를 가져올 수 있다. `Query String` 을 살펴보면 `entity` , `limit` , `term` 세 가지 매개변수를 받는것을 볼 수 있다. 여기서는 `entity` 는, song `limit` 은 1 `term` 으로는 weezer로 해서 크롬 URL창에서 들어가보자. 그러면 아래와 같은 텍스트 파일이 다운로드 받아지는것을 볼 수 있다

```
{
 "resultCount":1,
 "results": [
{"wrapperType":"track", "kind":"song", "artistId":115234, "collectionId":144
0878798, "trackId":1440879551, "artistName":"Weezer", "collectionName":"Weez
er", "trackName":"Say It Ain't So", "collectionCensoredName":"Weezer", "trac
kCensoredName":"Say It Ain't So", "artistViewUrl":"https://music.apple.com/u
s/artist/weezer/115234?uo=4", "collectionViewUrl":"https://music.apple.com/u
s/album/say-it-aint-so/1440878798?i=1440879551&uo=4", "trackViewUrl":"http
s://music.apple.com/us/album/say-it-aint-so/1440878798?i=1440879551&uo=4",
"previewUrl":"https://audio-ssl.itunes.apple.com/itunes-assets/AudioPreview1
22/v4/5c/07/84/5c078405-d5db-0762-d346-0f6ae3ccb530/mzaf_537061158510225480
3.plus.aac.p.m4a", "artworkUrl30":"https://is2-ssl.mzstatic.com/image/thumb/
Music125/v4/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.j
pg/30x30bb.jpg", "artworkUrl60":"https://is2-ssl.mzstatic.com/image/thumb/Mu
sic125/v4/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jp
g/60x60bb.jpg", "artworkUrl100":"https://is2-ssl.mzstatic.com/image/thumb/Mu
sic125/v4/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jp
g/100x100bb.jpg", "collectionPrice":10.99, "trackPrice":1.29, "releaseDat
e":"1994-05-10T12:00:00Z", "collectionExplicitness":"notExplicit", "trackExp
licitness":"notExplicit", "discCount":1, "discNumber":1, "trackCount":10, "t
rackNumber":7, "trackTimeMillis":258853, "country":"USA", "currency":"USD",
 "primaryGenreName":"Rock", "isStreamable":true}]
}
```


- 뭔가 생긴것이 딕셔너리와 비슷하다. 하지만 딕셔너리 타입은 아니다. 웹에서 이러한 형태로 데이터를 주고 받는 경우가 많다. 이러한 형태를 JSON(JavasScript Object Notation) 포맷이라고 부른다. 기존에는 XML(eXtensible Markup Language)를 썼지만, 무겁고, 파싱하기 어렵다는 문제로 JSON으로 대부분 대체되었다
- 단순히 데이터를 read 하기 위해서는 `HTTP Method` 중 `GET` 메소드를 사용해서 결과를 받아올 수 있다.그리고 `.json()` 을 사용해서 request객체에서 json()만 추출할 수 있다
- 위에서 활용한 `sys.argv` 를 활용해서 예시코드를 작성해보자

In [27]:

```
import requests

# Response객체가 출력되는것을 볼 수 있다
res = requests.get('https://itunes.apple.com/search?entity=song&limit=1&term=weezer'
res
```

Out[27]:

```
<Response [200]>
```

- 예제코드를 아래와 같이 작성했다고 가정하자

```
import requests
import sys

if len(sys.argv) != 2:
    sys.exit()

response = requests.get("https://itunes.apple.com/search?entity=song&limit=1
&term=" + sys.argv[1])

print(response.json())

print(type(response.json()))
```

In [28]:

```
!python3 argument_example/itunes.py weezer
```

```
{'resultCount': 1, 'results': [{'wrapperType': 'track', 'kind': 'son
g', 'artistId': 115234, 'collectionId': 1440878798, 'trackId': 1440879
551, 'artistName': 'Weezer', 'collectionName': 'Weezer', 'trackName':
"Say It Ain't So", 'collectionCensoredName': 'Weezer', 'trackCensoredN
ame': "Say It Ain't So", 'artistViewUrl': 'https://music.apple.com/us/
artist/weezer/115234?uo=4', 'collectionViewUrl': 'https://music.apple.
com/us/album/say-it-aint-so/1440878798?i=1440879551&uo=4', 'trackViewU
rl': 'https://music.apple.com/us/album/say-it-aint-so/1440878798?i=144
0879551&uo=4', 'previewUrl': 'https://audio-ssl.itunes.apple.com/itune
s-assets/AudioPreview122/v4/5c/07/84/5c078405-d5db-0762-d346-0f6ae3ccb
530/mzaf_5370611585102254803.plus.aac.p.m4a', 'artworkUrl30': 'http
s://is2-ssl.mzstatic.com/image/thumb/Music125/v4/fc/74/67/fc74674a-1eb
0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/30x30bb.jpg', 'artworkUr
l60': 'https://is2-ssl.mzstatic.com/image/thumb/Music125/v4/fc/74/67/f
c74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/60x60bb.jpg',
'artworkUrl100': 'https://is2-ssl.mzstatic.com/image/thumb/Music125/v
4/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/1
00x100bb.jpg', 'collectionPrice': 10.99, 'trackPrice': 1.29, 'releaseD
ate': '1994-05-10T12:00:00Z', 'collectionExplicitness': 'notExplicit',
'trackExplicitness': 'notExplicit', 'discCount': 1, 'discNumber': 1,
'trackCount': 10, 'trackNumber': 7, 'trackTimeMillis': 258853, 'countr
y': 'USA', 'currency': 'USD', 'primaryGenreName': 'Rock', 'isStreamabl
e': True}]}
<class 'dict'>
```

- 위 결과에서도 볼 수 있듯이, `response` 객체에 대해 `.json()` 으로 불러온 값은 `dictionary` 자료구조로 불러와지는것을 볼 수 있다.그렇기 때문에 일반적인 딕셔너리에서 값을 가져오듯이 가져올 수 있다. 여기서는 trackName을 한번 불러와보자. 대신 `limit` 을 50으로 변경하여 총 50개의 검색결과에 대해서 검색후, 각 검색 내용에서의 trackName을 출력해보자

```
import json
import requests
import sys

if len(sys.argv) != 2:
    sys.exit()

response = requests.get("https://itunes.apple.com/search?entity=song&limit=5
0&term=" + sys.argv[1])

o = response.json()
for result in o["results"]:
    print(result["trackName"])
```

In [29]:

```
!python3 argument_example/itunes4.py weezer
```

```
Say It Ain't So
Buddy Holly
Undone - The Sweater Song
My Name Is Jonas
Holiday
Surf Wax America
The World Has Turned and Left Me Here
Only in Dreams
In the Garage
No One Else
Say It Ain't So
Buddy Holly
Undone - The Sweater Song
Susanne
Say It Ain't So
My Name Is Jonas
Island In the Sun
Jamie
No One Else
Jamie
In the Garage
Mykel and Carli
The World Has Turned and Left Me Here
Undone -- The Sweater Song
Paperface
Lullabye For Wayne
Surf Wax America
My Evaline
My Name Is Jonas
Surf Wax America
I Swear It's True
Only In Dreams
No One Else
Weezer
Pork and Beans
Hash Pipe
Africa
Troublemaker
Lost in the Woods
Only In Dreams
Holiday
The Greatest Man That Ever Lived (Variations On a Shaker Hymn)
Pork and Beans
Photograph
Beverly Hills
Take on Me
Thank God for Girls
Heart Songs
Everybody Wants to Rule the World
Everybody Get Dangerous
```

- `.json()` 으로 json값을 가져오면, 이는 파이썬 딕셔너리 타입으로 값을 불러오게 된다. python의 `json` 내장 모듈의 `.dumps()` 메소드는 `indent` 속성을 활용하여, 가독성이 좋게끔 json포맷으로 변경하여 출력하게 해준다. 아래와 같이 예제코드를 작성하고 실행해보자. 하지만 주의할 점은 `json.dumps()` 는 `str` 타입이라는 것이다.

```
import json
import requests
import sys

if len(sys.argv) != 2:
    sys.exit()

response = requests.get("https://itunes.apple.com/search?entity=song&limit=1
&term=" + sys.argv[1])
j = json.dumps(response.json(), indent=4)
print(j)
print(type(j))
```

In [30]:

```
!python3 argument_example/itunes2.py weezer
```


```
{
    "resultCount": 1,
    "results": [
        {
            "wrapperType": "track",
            "kind": "song",
            "artistId": 115234,
            "collectionId": 1440878798,
            "trackId": 1440879551,
            "artistName": "Weezer",
            "collectionName": "Weezer",
            "trackName": "Say It Ain't So",
            "collectionCensoredName": "Weezer",
            "trackCensoredName": "Say It Ain't So",
            "artistViewUrl": "https://music.apple.com/us/artist/weeze
r/115234?uo=4",
            "collectionViewUrl": "https://music.apple.com/us/album/say
-it-aint-so/1440878798?i=1440879551&uo=4",
            "trackViewUrl": "https://music.apple.com/us/album/say-it-a
int-so/1440878798?i=1440879551&uo=4",
            "previewUrl": "https://audio-ssl.itunes.apple.com/itunes-a
ssets/AudioPreview122/v4/5c/07/84/5c078405-d5db-0762-d346-0f6ae3ccb53
0/mzaf_5370611585102254803.plus.aac.p.m4a",
            "artworkUrl30": "https://is2-ssl.mzstatic.com/image/thumb/
Music125/v4/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM5297
1.rgb.jpg/30x30bb.jpg",
            "artworkUrl60": "https://is2-ssl.mzstatic.com/image/thumb/
Music125/v4/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM5297
1.rgb.jpg/60x60bb.jpg",
            "artworkUrl100": "https://is2-ssl.mzstatic.com/image/thum
b/Music125/v4/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM529
71.rgb.jpg/100x100bb.jpg",
            "collectionPrice": 10.99,
            "trackPrice": 1.29,
            "releaseDate": "1994-05-10T12:00:00Z",
            "collectionExplicitness": "notExplicit",
            "trackExplicitness": "notExplicit",
            "discCount": 1,
            "discNumber": 1,
            "trackCount": 10,
            "trackNumber": 7,
            "trackTimeMillis": 258853,
            "country": "USA",
            "currency": "USD",
            "primaryGenreName": "Rock",
            "isStreamable": true
        }
    ]
}
<class 'str'>
```


- 조금 더 응용하여 `json.dumps()` 를 활용하여 `json` 파일로 저장할 수 있다.

```
import json
import requests
import sys

if len(sys.argv) != 2:
    sys.exit()

response = requests.get("https://itunes.apple.com/search?entity=song&limit=1
&term=" + sys.argv[1])

with open('./itune.json','w') as j:
    j.write(json.dumps(response.json(), indent=4))
    j.close()
```

In [31]:

```
!python3 argument_example/itunes3.py weezer
```

In [32]:

```
!ls
```

```
Homework #1-1.ipynb
Homework #1-1_윤준호_B889047.pdf
argument_example
itune.json
```

- json파일이 잘 저장된것을 볼 수 있다.

Making your own libraries

- 나만의 라이브러리를 만들어서 사용해 보자. 간단하게 `saying.py` 라는 파일에 아래 두 함수를 작성해보자.

```
def hello(name):
    print(f"hello, {name}")


def goodbye(name):
    print(f"goodbye, {name}")
```

- 이제 자신이 작성한 라이브러리를 import한 후, 함수 매개변수를 넘길때 argument로 받은값으로 넘길 수 있게끔 작성해보자.

```
import sys

from saying import *

if len(sys.argv) == 2:
    hello(sys.argv[1])
    goodbye(sys.argv[1])
```

In [33]:

```
!python3 argument_example/library_test.py hoplin
```

```
hello, hoplin
goodbye, hoplin
```