이번 레슨에서는 유용한 스탠다드 모듈 몇 개를 소개해 드릴 건데요. 모듈이 어떤 기능을 제공해 주는지, 그리고 모듈을 어떻게 사용하는지 간단히 보여드리겠습니다.

모듈들에 대한 자세한 정보는 파이썬 공식 문서에서 찾을 수 있습니다: https://docs.python.org/ko/3/library/

math

math는 기본적인 수학 모듈입니다. 여러 수학적인 함수를 제공해 줍니다.

import math

# 코사인 함수 (모든 삼각함수는 라디안을 사용합니다)

print(math.cos(0))

# 로그 함수

print(math.log10(100))

1.0

2.0

random

random 모듈은 랜덤 한 숫자를 생성하기 위한 다양한 함수들을 제공해 줍니다.

import random

# 랜덤한 정수 1 <= N <= 20

print(random.randint(1, 20))

# 랜덤한 소수 0 <= x <= 1

print(random.uniform(0, 1))

3

0.599056286966887

datetime

datetime 모듈은 날짜와 시간을 다루기 위한 다양한 '클래스'를 갖추고 있습니다. 클래스의 개념을 잘 모르셔도 이 모듈을 사용하는 데에는 문제없습니다.

import datetime

# 현재 시간과 날짜

today = datetime.datetime.now()

print(today)

# 출력값을 "요일, 월 일 연도"로 포매팅

print(today.strftime("%A, %B %dth %Y"))

# 특정 시간과 날짜

pi\_day = datetime.datetime(2020, 3, 14, 13, 6, 15)

print(pi\_day)

# 두 datetime의 차이

print(today - pi\_day)

2020-08-26 14:09:36.361025

Wednesday, August 26th 2020

2020-03-14 13:06:15

165 days, 1:03:21.361025

os

OS는 Operating System, 즉 운영체제의 약자입니다. os 모듈을 통해서 파이썬으로 운영체제를 조작하거나 운영체제에 대한 정보를 가져올 수 있습니다.

import os

# 현재 어떤 계정으로 로그인 돼있는지 확인

print(os.getlogin())

# 현재 파일의 디렉토리 확인

print(os.getcwd())

# 현재 프로세스 ID 확인

print(os.getpid())

codeit

/Users/codeit/PycharmProjects/standard\_modules

2346

os.path

os.path 모듈은 파일 경로를 다룰 때 쓰입니다.

import os.path

# 프로젝트 디렉토리 경로 '/Users/codeit/PycharmProjects/standard\_modules'

# 현재 파일 경로 '/Users/codeit/PycharmProjects/standard\_modules/main.py'

# 주어진 경로를 절대 경로로

print(os.path.abspath('..'))

# 주어진 경로를 현재 디렉토리를 기준으로 한 상대 경로로

print(os.path.relpath('/Users/codeit/PycharmProjects'))

# 주어진 경로들을 병합

print(os.path.join('/Users/codeit/PycharmProjects', 'standard\_modules'))

/Users/codeit/PycharmProjects

..

/Users/codeit/PycharmProjects/standard\_modules

re

프로그래밍에서 Regular Expression (RegEx, re, 한국어로는 정규 표현식)은 특정한 규칙/패턴을 가진 문자열을 표현하는 데 사용됩니다.

import re

# 알파벳으로 구성된 단어들만 매칭

pattern = re.compile('^[A-Za-z]+$')

print(pattern.match('I'))

print(pattern.match('love'))

print(pattern.match('python3'))

print()

# 숫자가 포함된 단어들만 매칭

pattern = re.compile('.\*\d+')

print(pattern.match('I'))

print(pattern.match('love'))

print(pattern.match('python3'))

<re.Match object; span=(0, 1), match='I'>

<re.Match object; span=(0, 4), match='love'>

None

None

None

<re.Match object; span=(0, 7), match='python3'>

pickle

pickle 을 사용하면 파이썬 오브젝트(객체)를 바이트(byte) 형식으로 바꿔서 파일에 저장할 수 있고 저장된 오브젝트를 읽어올 수도 있습니다.

import pickle

# 딕셔너리 오브젝트

obj = {'my': 'dictionary'}

# obj를 filename.pickle 파일에 저장

with open('filename.pickle', 'wb') as f:

pickle.dump(obj, f)

# filename.pickle에 있는 오브젝트를 읽어옴

with open('filename.pickle', 'rb') as f:

obj = pickle.load(f)

print(obj)

{'my': 'dictionary'}

json

json 모듈은 pickle과 비슷하지만 오브젝트를 JSON 형식으로 바꿔줍니다. JSON 형식에 맞는 데이터 (기본 데이터 타입들, 리스트, 딕셔너리)만 바꿀 수 있습니다.

import json

# 딕셔너리 오브젝트

obj = {'my': 'dictionary'}

# obj를 filename.json 파일에 저장

with open('filename.json', 'w') as f:

json.dump(obj, f)

# filename.json에 있는 오브젝트를 읽어옴

with open('filename.json', 'r') as f:

obj = json.load(f)

print(obj)

{'my': 'dictionary'}

copy

copy 모듈은 파이썬 오브젝트를 복사할 때 쓰입니다.

import copy

# '=' 연산자는 실제로 리스트를 복사하지 않음

# 리스트를 복사하려면 슬라이싱을 사용하거나 copy.copy() 함수를 사용해야 함

a = [1, 2, 3]

b = a

c = a[:]

d = copy.copy(a)

a[0] = 4

print(a, b, c, d)

# 하지만 오브젝트 안에 오브젝트가 있는 경우 copy.copy() 함수는 가장 바깥에 있는 오브젝트만 복사함

# 오브젝트를 재귀적으로 복사하려면 copy.deepcopy() 함수를 사용해야 함

a = [[1,2,3], [4,5,6], [7,8,9]]

b = copy.copy(a)

c = copy.deepcopy(a)

a[0][0] = 4

print(a, b, c)

[4, 2, 3] [4, 2, 3] [1, 2, 3] [1, 2, 3]

[[4, 2, 3], [4, 5, 6], [7, 8, 9]] [[4, 2, 3], [4, 5, 6], [7, 8, 9]] [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

sqlite3

sqlite3 모듈을 통해 파이썬에서 SQLite 데이터베이스를 사용할 수 있습니다.

import sqlite3

# 데이터베이스 연결

conn = sqlite3.connect('example.db')

# SQL 문 실행

c = conn.cursor()

c.execute('''SELECT ... FROM ... WHERE ... ''')

# 가져온 데이터를 파이썬에서 사용

rows = c.fetchall()

for row in rows:

print(row)

# 연결 종료

conn.close()