2019-07-08(월) | StartCamp01

1. 4차 산업혁명과 소프트웨어

4차 산업혁명

- Amazon의 KIVA 로봇, 드론 배송
- Google 어시스턴트 미용실 예약, 구글 렌즈
- YOLO : You Only Look Once
- 포브스 2019 기술 트렌드

  : 5G / Chatbots / Connected Clouds / Blockchain / Data Analytic / GDPR / AR(feat.VR) / Much more IoT

2. (파이썬) 프로그래밍의 개념과 문법

1. 프로그래밍 개념

Hack Your Life

Life Hacking, Programmable Life

- 당근마켓 식물 물주기 2명 랜덤 pick
- YouTube 동영상 끝까지 봤는지

오픈소스

> **제작자의 권리를 지키면서, 누구나 코드 열람이 가능한!**
>
> - 수레바퀴를 두 번 만들지 마라
> - 거인의 어깨 위에서 프로그래밍 시작하기

2. 프로그래밍 문법

※ 주의 ※

> 1. **대/소문자 구분**
> 2. **띄어쓰기 및 들여쓰기(indent) >> spacebar*4 == 1 tab**
> 3. **스펠링**

< 프로그래밍 언어 3형식 >

- **저장**

- 저장할 값
  1. **숫자**
  2. **문자**
  3. **boolean**값 (True/False)

- 저장하는 방법
  1. **변수** 박스 1개

```
dust = 60   #dust에 60을 저장한다.
dust == 60  #dust = 60이다.
```

  2. **리스트** 박스 여러개 **(순서가 있음)**

```
dust = [1, 2, 3]
```

  3. **딕셔너리** 견출지 붙인 박스들의 묶음 **(key와 value로 구성)**

```
dust = {'영등포구': 58, '강남구': 40}
print(["영등포구"])  #key로 value를 불러냄
```

- **조건**
  - **if / elif / else**

```
if dust > 150:
    print

elif 80 < dust <= 150:    #80 < dust and dust <= 150 이렇게도 가능
    print

else:
    print
```

- **반복**
  - **while문** : *종료 조건이 필요*

```
while True:
    print('계속해주세요.')
```

  - **for문** : 정해진 범위 내의 반복이라 *종료 조건 필요하지X*

```
for i in list:
    print
```

3. (파이썬) 프로그래밍 심화

1. API : Application Programming Interface

서로 다른 서비스간의 대화 방식 (요청 & 응답)

요청 _____주소(URL)_____>> 응답

(정보를 원하는 사람) <<___문서(HTML, XML, Json 등)___ (정보를 주는 사람)

오픈 소스 기술 + API 통신 = 쉬워진 프로그래밍

2. CLI : Command Line Interface

★ CLI에서는 항상 ***자신이 어디에 있는지*** 주의하자!!! (pwd : print working directory)

3. 크롤링

사람이 보라고 만들어 놓은 문서에서 억지로 긁어 오는 것

4. Package

4. 마크다운 (from. hyunny0463)

4.1. Text

4.1.1. Italic

기울임체를 사용하기 위해 사용한다.

특수문자 `*` 그리고 `_` 를 사용하거나, `ctrl + i` 단축키를 사용해도 된다.

```
*기울임체를 사용하려면 이렇게 하세요*
_기울임체를 사용하려면 이렇게 하세요_
```

4.1.2. Bold

글자를 굵은 글꼴로 바꿀 때 사용한다.

특수문자 `**` 를 사용하거나, `ctrl + b` 단축키를 사용해도 된다.

```
**볼드체를 사용하려면 이렇게 하세요**
```

4.1.3. Strikethrough

~~글자에 취소선을 넣을 때 사용한다.~~

특수문자 `~` 를 사용하거나, `alt + shift + 5` 단축키를 사용해도 된다.

```
~~취소선을 넣으려면 이렇게 하세요~~
```

4.2. Headers

문서 제목을 표현할 때 사용한다.

각 행에서 맨 앞에 `#` 을 넣어 사용할 수 있다.

```
# This is a H1          # Ctrl + 1
## This is a H2         # Ctrl + 2
### This is a H3        # Ctrl + 3
#### This is a H4       # Ctrl + 4
##### This is a H5      # Ctrl + 5
###### This is a H6     # Ctrl + 6
```

4.3. Quotation

인용문을 사용할 때 사용한다.

각 행에서 맨 앞에 `>` 을 넣어 사용할 수 있다.

> *The gratification comes in the doing, not in the results." - James dean*
>
> *만족은 결과가 아니라 과정에서 찾아온다. - 제임스 딘*

4.4. Code

4.4.1. Short source code

간단한 소스코드를 한 줄 입력할 때 사용한다.

소스코드에 `` ` 를 둘러싸서 사용한다. 단축키는 `ctrl + shift + ``

```
print('Hello, SSAFY!')
```

4.4.2. Complete source code

긴 소스코드를 입력할 때 사용한다.

소스코드를 ```` ``` 입력하고 enter `를 친 후 넣으면 된다. 단축키는 ctrl + shift + K ````

```
#include <iostream>
using namespace std;

int main(int argc, char** argv)
{
    cout << "Hello, SSAFY!" << '\n';
    return 0;
}
```

4.5. Link

markdown 문서에 링크를 걸고 싶을 때 사용한다.

4.5.1. 일반 링크

일반 링크는 하나의 링크를 연결할 때 사용한다.

`[링크 이름](실제 주소)` 의 규칙으로 작성하면 된다.

hyunny0463님의 github

4.5.2. 참조 링크

참조 링크는 하나의 링크를 여러번 참조하고 싶을 때 사용한다.

`[링크 이름][숫자]` 로 링크를 작성하고, 추가적으로 `[숫자]:{실제 주소}` 로 원하는 링크를 입력한다.

hyunny0463님의 github

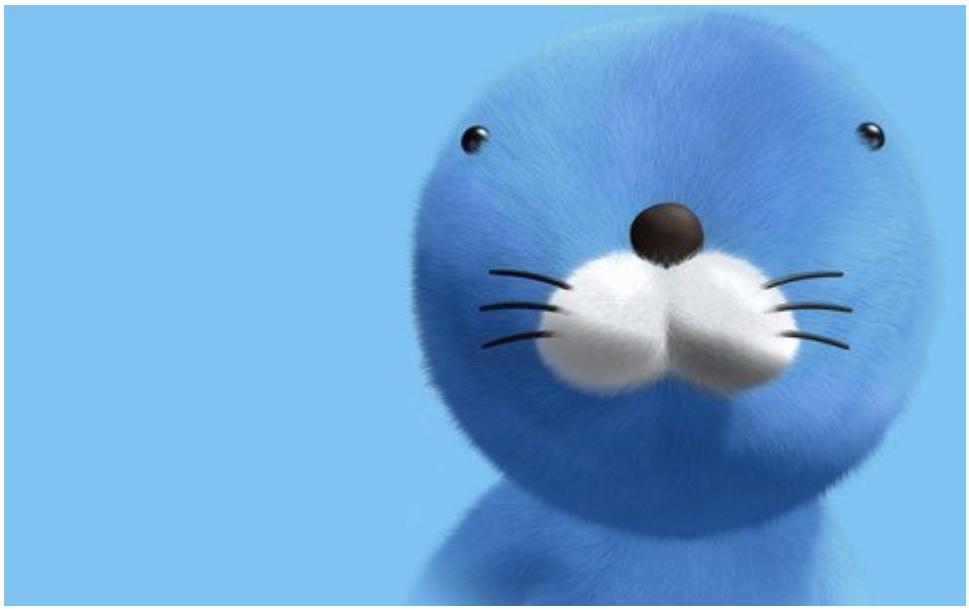
몰라유

4.6. Image

markdown 문서에 이미지를 넣을 때 사용한다.

`![그림 설명](이미지 링크)` 를 입력해서 사용할 수 있다.

이미지가 깨진다면 `repository > issue > new issue > 이미지 드래그&드롭 > [주소]` 를 활용한다.



4.7. etc

`ctrl + shift + 1 ~ 3` : 파일의 목차와 파일트리를 볼 수 있다.

`ctrl + /` : markdown 문서로 작성된 문서의 소스코드를 볼 수 있다.

`---` `___` `***` : 아래와 같은 horizontal rule을 생성할 때 사용할 수 있다.

##

2019-07-09(화) | StartCamp02

Python IDE(Integrated Development Environment 통합개발환경)

Python IDLE : Python으로 내 컴퓨터를 조작하는 툴

웹 브라우저 조작하기

import webbrowser

webbrowser.open_new_tab() : web browser를 새 tab에서 열어줘

code . --> 지금 디렉토리에서 코드 파일 열어달라

```
#웹 브라우저 열기
import webbrowser

webbrowser.open('https://google.com')
webbrowser.open_new('https://naver.com')
webbrowser.open_new_tab('https://daum.net')
```

```
#원하는 검색어로 검색하기
import webbrowser

searching = ['구미맛집', '구미가볼곳', '구미마트', '구미다이소']

for mysearch in searching:
    print(mysearch)
    webbrowser.open('https://search.naver.com/search.naver?query=' + mysearch)
    # +mysearch★가 중요 포인트! 이 안에 들어있는건 str 타입이다.
print(type(mysearch)) 해보면 str 나옴
```

두 번째 조작, 정보 스크랩하기

자주 확인하는 정보를 (자동)스크랩하기

import requests

1) **requests.get('주소')**

'주소'에 요청 보내서(request), 정보를 받아줘(get)

2) **requests.get('주소').text**

'주소'에 요청 보내서, 정보 받아서, 글(text)만 뽑아줘

3) **requests.get('주소').status_code** >>> 200이 가장 잘 보낸 것

'주소'에 요청 보내서, 정보 받아서, 상태(status_code)만 뽑아줘

import bs4

bs4.BeatifulSoup(문서)

받은 문서를 좀 예쁘게(보기 좋게, 검색하기 좋게) 만들어줘

.select(selector)

문서 안에 있는 특정 내용을 뽑아줘(select) >>> 형식 list로 나옴! 내용'들을' 뽑아주는거라서

.select_one(selector)

문서 안에 있는 특정 내용 하나만 뽑아줘 >>> 형식 str으로 나옴!

- 내장함수 : 책상 위
- 외장함수 : 책상 서랍
- 라이브러리 : 거실에서 가져오는 것 `pip install 라이브러리 --user`

ctrl + Enter : 코드 가운데 커서가 있어도 아랫줄로 내려올 수 있음

```
#코스피 지수 가져오기
import requests
from bs4 import BeautifulSoup  #bs4라는 모듈에서 BeautifulSoup을 가져와서 쓴다.

url = 'https://finance.naver.com/sise/'

html = requests.get(url).text
soup = BeautifulSoup(html, 'html.parser') #파씽한다 : 유용한 정보를 가져온다.
kospi = soup.select_one('#KOSPI_now').text  #id라서 #을 찍음
print(f'오늘의 코스피 지수는 {kospi}입니다.')



#결과값
student@DESKTOP MINGW64 ~/TIL/00_StartCamp/01_Day
$ python 01_naver_kospi.py
오늘의 코스피 지수는 2,065.12입니다.
```

```
#원/달러 환율 가져오기
import requests
from bs4 import BeautifulSoup

url = 'https://finance.naver.com/marketindex/'

html = requests.get(url).text
soup = BeautifulSoup(html, 'html.parser')
exchange = soup.select_one('#exchangeList > li.on > a.head.usd > div >
span.value')
print(f'현재의 원/달러 환율은 {exchange.text}입니다.')
```

```
#결과값
student@DESKTOP MINGW64 ~/TIL/00_StartCamp/01_Day
$ python 02_naver_exchange.py
현재의 원/달러 환율은 1,180.10입니다.
```

```
#실시간 검색순위 가져오기
import requests
from bs4 import BeautifulSoup

url = 'https://www.naver.com/'

html = requests.get(url).text
soup = BeautifulSoup(html, 'html.parser')
names = soup.select('#PM_ID_ct > div.header > div.section_navbar >
div.area_hotkeyword.PM_CL_realtimeKeyword_base >
div.ah_roll.PM_CL_realtimeKeyword_rolling_base > div > ul > li .ah_k')
#실시간 검색어는 여러개. 뱉어내는게 list
#print(type(names))   #>>> list

'''
for name in names:
    print(name.text)
'''  #읽지 않는 스트링으로 만들기 (주석은 X)

for idx, name in enumerate(names):  #enumerate는 인덱스 번호까지 같이 빼내주는 함수
    print(f'{idx+1}위: {name.text}')



#결과값
student@DESKTOP MINGW64 ~/TIL/00_StartCamp/01_Day
$ python 03_naver_real_time_search.py
1위: 토스머니5만원이벤트
2위: 천우희
3위: 자사고
4위: 위메프투어 아시아나항공
5위: 티르티르
6위: 브로콜리 새싹
7위: 23사단
8위: 안재홍
9위: 분홍코끼리
10위: 블라디미르 게레로 주니어
11위: 토스 삼성카드 5만원
12위: 블라우스
13위: 멜로가 체질
14위: 거제 살인사건
15위: 서울항아리물회
16위: 한국사능력검정시험
17위: 작 피더슨
18위: 모스키노 쿠션
19위: 베오울프
20위: 분양가 상한제
```

API vs Crawling

프로그래밍 하라고 준 데이터 쓰기 vs 사람 보라고 준 데이터 긁기

API는 충분히 친절하다

우리가 hacking 할 수 있다면 얼마든지 가져다 쓸 수 있다.(무료라고는 안했다)

But, 코드가 더 짧았으면 좋겠다 ---> Python Package 사용하기

git: (분산) 버전 관리 시스템

git이 있다면 차이(diff)가 무엇이고 수정 이유를 log로 남길 수 있다.

> 과거 모습 그대로 복원 가능(반대도 마찬가지)
>
> 히스토리 돌아보고 차이를 볼 수 있음
>
> 파일 크기 현저히 줄어듦

git의 작업 흐름

- **add** 커밋할 목록에 추가
- **commit** 커밋(create a snapshot) 만들기
- **push** 현재까지의 역사(commits)가 기록되어 있는 곳에 새로 생성한 커밋들 반영하기

    - git을 사용하여 끝말잇기

      git clone : 나한테 없는 파일 다른 사람한테서 받아오기

      pull : 시작한 사람 다시 끌고 오기

GitHub Pages 템플릿 가지고 와서 나만의 블로그 만들기

```
#1. pyformat

# name = '홍길동'
# english_name = 'hong'

# print('안녕하세요, {}입니다. My name is {}'.format(name, english_name))
# print('안녕하세요, {1}입니다. My name is {0}'.format(name, english_name))
# print('안녕하세요, {1}입니다. My name is {1}'.format(name, english_name))

#2. f-strings

# name = '홍길동'
# print(f'안녕하세요, {name}입니다.')
# print('안녕하세요,', name, '입니다.')

# import random

# menu = ['김밥천구', '별별벅스', '흥부부대찌개']
```

```
# lunch = random.choice(menu)
# print('오늘의 점신은 {}입니다.'.format(lunch))
# print(f'오늘의 점심은 {lunch}입니다.')



import random

numbers = list(range(1, 46))
lotto = random.sample(numbers, 6)
print(f'오늘의 행운 번호는 {sorted(lotto)}입니다.')
```

import os

1) os.chdir(r'폴더주소')

```
import os

os.chdir(r'C:\Users\student\TIL\00_StartCamp\02_Day\dummy')

for filename in os.listdir('.'):
    #os.rename(filename, f'SAMSUNG_{filename}')      #모든 파일 이름 앞에 SAMSUNG_ 붙이기
    os.rename(filename, filename.replace('SAMSUNG_', 'SSAFY_'))  #SAMSUNG_ 대신 SSAFY_로 바꾸기
```

```
#1. 파일 쓰기(옛날 방식)

f = open('ssafy.txt', 'w')
for i in range(10):
    f.write(f'This is line {i}.\n')
f.close()  #파일을 열었으면 닫아줘야지! 이거 반드시 반드시 해주기

#2. 파일 쓰기(최신 방식)

with open('with_ssafy.txt', 'w') as f:       #with 구문 이용한 파일 쓰기
    for i in range(10):
        f.write(f'This is line {i}.\n')      #f.close() 쓰지 x



#결과값 >>> ssafy.txt랑 with_ssafy.txt 파일이 생기고 각각에 아래 내용 입력됨.
This is line 0.
This is line 1.
This is line 2.
This is line 3.
This is line 4.
This is line 5.
This is line 6.
This is line 7.
This is line 8.
This is line 9.
```

2019-07-10(수) | StartCamp03

개행

- \n
- print()

reverse() : 메서드

reversed() : 함수

> return을 하는가 아닌가 함수는 return 을 뱉음 (원본은 그대로 두고 뱉는게 달라지는 것)
>
> 메서드도 함수인데 함수가 더 큰 개념
>
> 메서드 : 클래스 안에 정의된 함수들

```
#1. 각각의 라인을 리스트의 요소로 불러오기
with open('writelines_ssafy.txt', 'r') as f:
    lines = f.readlines()

#2. 뒤집기
lines.reverse()

#3. line을 작성하기
with open('reverse_ssafy.txt', 'w') as f:
    for line in lines:
        f.write(line)



#결과값 (reverse_ssafy.txt 파일에 적힌 내용)
3
2
1
0
```

csv : comma seperated value

> 파일의 한 형태

딕셔너리는 '튜플' 자료형로 뱉어냄

('1', '2', '3') 이런게 튜플

```
lunch = {
    '양자강':'02-557-4211', '김밥카페':'02-553-3181', '순남시레기':'02-508-0887'
}

#1. 방법 1

with open('lunch.csv', 'w', encoding = 'utf-8') as f:  #encoding-'utf-8'은 깨짐
방지
    for item in lunch.items():                         #item은 튜플형
        f.write(f'{item[0]}, {item[1]}\n')



#결과값(lunch.csv라는 파일에)
양자강, 02-557-4211
김밥카페, 02-553-3181
순남시레기, 02-508-0887
```

split() : 쪼개서 리스트 만들어줌

ex) split(',') : , 를 기준으로 리스트

```
#csv 파일 read의 2가지 방법

#1. split
with open('lunch.csv', 'r', encoding = 'utf-8') as f:
    lines = f.readlines()
    for line in lines:
        print(line.strip())
        print(line.strip().split(','))

#2. csv reader
import csv

with open('lunch.csv', 'r', encoding = 'utf-8') as f:
    lines = csv.reader(f)

    for line in lines:
        print(line)



#결과값
student@DESKTOP MINGW64 ~/TIL/00_StartCamp/02_Day (master)
$ python 07_csv_read.py
['양자강', ' 02-557-4211']
['김밥카페', ' 02-553-3181']
['순남시레기', ' 02-508-0887']
```

```
import requests
from bs4 import BeautifulSoup
```

```
from datetime import datetime

html = requests.get('https://www.naver.com').text
soup = BeautifulSoup(html, 'html.parser')
tags = soup.select('#PM_ID_ct > div.header > div.section_navbar >
div.area_hotkeyword.PM_CL_realtimeKeyword_base >
div.ah_roll.PM_CL_realtimeKeyword_rolling_base > div > ul > li .ah_k')

for tag in tags:
    print(tag.text)

now = datetime.now()
with open('naver.txt', 'w', encoding = 'utf-8') as f:
    f.write(f'{now} 기준 네이버 실시간 검색어\n')
    for idx, tag in enumerate(tags):    #enumerate()는 인덱스까지 꺼내주는. idx 자료
형 '튜플'
        f.write(f'{idx+1}위: {tag.text}\n')



#결과값 (naver.txt 파일에)
2019-07-10 11:29:19.606143 기준 네이버 실시간 검색어
1위: 강지환
2위: 안다르 워터레깅스
3위: 패션플러스
4위: 현대건설
5위: 이랜드몰 반값데이
6위: 임은경
7위: 김혜수
8위: 류현진 올스타전
9위: 배지현
10위: 돌벽지
11위: 부산 지하철 파업
12위: 메이썸
13위: 진관사
14위: 조선생존기
15위: 진화
16위: 화사
17위: 두올산업
18위: 카라스코
19위: 타짜3
20위: 트러블 패치
```

멜론 차트

웹페이지와 다른 점 : header

```
import requests


response = requests.get('https://www.melon.com/chart/index.htm').status_code

print(response)


#결과값
student@DESKTOP MINGW64 ~/TIL/00_StartCamp/02_Day (master)
$ python 09_melon_rank.py
406  >>>>>>406 에러 '4'는 (사용자가 잘못)
```

```
import requests
from bs4 import BeautifulSoup

headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/75.0.3770.100 Safari/537.36'
}

response = requests.get('https://www.melon.com/chart/index.htm', headers =
headers)
response.encoding = 'utf-8'
res = response.text

soup = BeautifulSoup(res, 'html.parser')

# print(response)
songs = soup.select('.lst50')

for song in songs:
    rank = song.select_one('#lst50 > td:nth-child(2) > div > span.rank').text
    name = song.select_one('#lst50 > td:nth-child(6) > div > div >
div.ellipsis.rank01 > span > a').text
    artist = song.select_one('#lst50 > td:nth-child(6) > div > div >
div.ellipsis.rank02 > a').text
    print(f'{rank}위 : {name} / {artist}')


#csv 파일로 옮기기

with open('melon.csv', 'w', encoding = 'utf-8') as f:
    for song in songs:
        rank = song.select_one('#lst50 > td:nth-child(2) > div >
span.rank').text
        name = song.select_one('#lst50 > td:nth-child(6) > div > div >
div.ellipsis.rank01 > span > a').text
        artist = song.select_one('#lst50 > td:nth-child(6) > div > div >
div.ellipsis.rank02 > a').text
        f.write(f'{rank}위: {name} / {artist}\n')
```

```
#결과값(melon.csv 파일에)
1위: 인사 / 멜로망스
2위: 헤어져줘서 고마워 / 벤
3위: 술이 문제야 / 장혜진
4위: We don`t talk together (Feat. 기리보이) (Prod. SUGA) / 헤이즈 (Heize)
5위: 니 소식 / 송하예
6위: Snapping / 청하
7위: 솔직하게 말해서 나 / 김나영
8위: 사랑에 연습이 있었다면 (Prod. 2soo) / 임재현
9위: Speechless (Full) / Naomi Scott
10위: 2002 / Anne-Marie
11위: bad guy / Billie Eilish (빌리 아일리시)
12위: 너에게 못했던 내 마지막 말은 / 다비치
13위: 포장마차 / 황인욱
14위: 작은 것들을 위한 시 (Boy With Luv) feat. Halsey / 방탄소년단
15위: 오늘도 빛나는 너에게 (To You My Light) (Feat.이라온) / 마크툽 (MAKTUB)
16위: 사랑이 식었다고 말해도 돼 / 먼데이 키즈 (Monday Kiz)
17위: 괜찮아도 괜찮아 (That`s okay) / 디오 (D.O.)
18위: 서울 밤 (feat. 빈지노) / 어반자카파
19위: 비가 내리는 날에는 / 윤하 (YOUNHA)
20위: FANCY / TWICE (트와이스)
21위: 주저하는 연인들을 위해 / 잔나비
22위: 누구 없소 (NO ONE) (Feat. B.I of iKON) / 이하이
23위: A Whole New World / Mena Massoud
24위: 사월이 지나면 우리 헤어져요 (Beautiful goodbye) / 첸 (CHEN)
25위: Goodbye / 박효신
26위: 짐살라빔 (Zimzalabim) / Red Velvet (레드벨벳)
27위: 노래방에서 / 장범준
28위: 사계 (Four Seasons) / 태연 (TAEYEON)
29위: 열대야 (Fever) / 여자친구 (GFRIEND)
30위: 그때가 좋았어 / 케이시 (Kassy)
31위: U GOT IT / 갓츄 (GOT U)
32위: 모든 날, 모든 순간 (Every day, Every Moment) / 폴킴
33위: BIRTHDAY / 전소미
34위: 아퍼 (Feat. Kid Milli, Lil tachi, 김승민, NO:EL, C JAMM) / 기리보이
35위: 나만, 봄 / 볼빨간사춘기
36위: Uh-Oh / (여자)아이들
37위: 너를 만나 / 폴킴
38위: 사계 (하루살이) / 엠씨더맥스
39위: 이뻐이뻐 (Pretty girl) / 크레파스
40위: 대충 입고 나와 / 우디 (Woody)
41위: 움직여 (MOVE) (Prod. by ZICO) / SIXC (6 crazy)
42위: Paris In The Rain / Lauv
43위: BAND / 창모 (CHANGMO)
44위: 소우주 (Mikrokosmos) / 방탄소년단
45위: 비워 (Beer) (Prod. Way Ched) / 창모 (CHANGMO)
46위: 봄날 / 방탄소년단
47위: AH YEAH (아예) / WINNER
48위: 벌써 12시 / 청하
49위: 옥탑방 (Rooftop) / 엔플라잉 (N.Flying)
50위: 달라달라 / ITZY (있지)
```

문제풀기

1.
```
"""
문제1. 문자열을 입력받아 문자열의 첫 글자와 마지막 글자를 출력하는 프로그램을 만드세요.
"""

str = input('문자를 입력하세요: ')

print(f'첫 글자 : {str[0]}\n마지막 글자 : {str[-1]}')
```

2. **input**으로 받으면 '문자' >>> **숫자**로 받으려면 **int**(input('숫자를 입력하세요: '))

```
"""
문제2. 자연수 n이 주어질 때, 1부터 n까지 한 줄에 하나씩 출력하는 프로그램을 만드세요.
"""

number = int(input('자연수를 입력하세요: '))


for i in range(1, number+1):
    print (i)

#또는,
for i in range(number)
    print(i+1)
```

3.
```
"""
문제3. 숫자를 입력 받아 짝수/홀수를 구분하는 코드를 작성하세요.
"""

number = int(input('숫자를 입력하세요: '))

if (number%2 == 0):
    print(number, '는 짝수 입니다.')
else:
    print(number, '는 홀수 입니다.')

#또는, nuber%2가 1이면 참(True)이니까 if 조건에 걸림. 0이면 거짓(False)니까 else로!
if number%2:
    print('홀수')
else:
    print('짝수')
```

4.
```
"""
문제4. 표준 입력으로 국어, 영어, 수학, 과학 점수가 입력됩니다.
국어는 90점 이상,
영어는 80점 초과,

수학은 85점 초과,
과학은 80점 이상일 때 합격이라고 정했습니다.
(한 과목이라도 조건에 만족하지 않으면 불합격)
다음 코드를 완성하여 합격이면 True, 불합격이면 False가 출력되도록 작성하세요.
"""
```

```python
kor = int(input('국어: '))
eng = int(input('영어: '))
mat = int(input('수학: '))
sci = int(input('과학: '))

if (kor >= 90) and (eng > 80) and (mat > 85) and (sci >= 80):
    print('True')
else:
    print('False')
```

5.
```python
"""
문제5. 표준 입력으로 물품 가격 여러 개가 문자열 한 줄로 입력되고,
각 가격은 ;(세미콜론)으로 구분되어 있습니다.
입력된 가격을 높은 가격순으로 한 줄에 하나씩 출력하는 프로그램을 만드세요.
입력예시: 30000;2000;40000
출력예시
40000
30000
2000
"""

prices = input('물품 가격을 입력하세요(;로 구분하세요.): ').split(';')
boxs = []
#prices.sort(reverse = True)
#sorted(prices, reverse=True)

for price in prices:
    boxs.append(int(price))

boxs.sort(reverse=True)

for box in boxs:
    print(box)


#결과값
student@DESKTOP MINGW64 ~/TIL/00_StartCamp/03_Day (master)
$ python 00_problem_set_list.py
물품 가격을 입력하세요(;로 구분하세요.): 300;20;10000
10000
300
20
```

- HTML (HyperText Markup Language)

> Hyper : 문장을 넘나든다
>
> ! + tab : 자동으로 틀 만들어줌
>
> 태그들로 이루어짐(Head Tag + Body Tag)

- CSS (Cascading Styling Sheets)

> HTML을 꾸며주는 친구

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="style.css">
    <title>지니지니진 페이지</title>
</head>
<body>
    <h1>SSAFY 2기</h1>
    <h2>HTML & CSS</h2>
    <p>이건 문단을 구분하는 p 태그</p>
    <ol>
        <li>이건 순서가 있는 태그</li>
        <li>이건 순서가 있는 태그</li>
    </ol>
    <ul>
        <li>이건 순서가 있는 태그</li>
        <li>이건 순서가 있는 태그</li>
    </ul>
</body>
</html>
```

```
h1 {
    color:skyblue;
}

h2{
    color:plum;
}

p{
    font-size:40px;
}

li{
    text-align:center;
}
```

git init

git add .

git commit -m " "

git remote add origin 주소

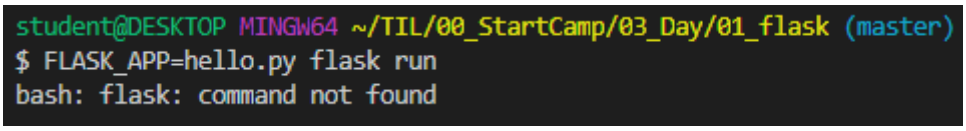
git push origin master //이 때 동기화가 되는 것

Web Framework 왜 사용?

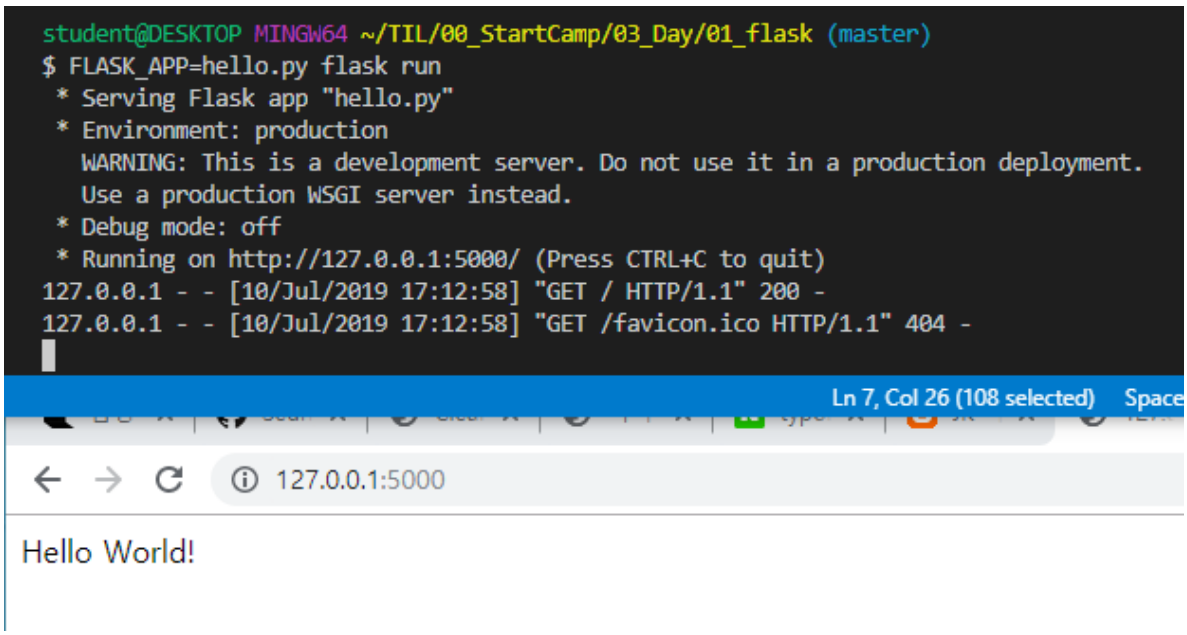
> 프랜차이즈냐 / All by myself~ 의 차이
>
> 뼈대 코드 제공해줄게, 너는 사용자들에게 좋은 서비스 제공에만 집중해
>
> 파이썬의 두 가지 Framework
>
> 1. **Flask(마이크로)**
> 2. **Django(덩어리)**

Flask

```
#hello.py 폴더
from flask import flask

app = Flask(__name__)

@app.route('/')
def hello():
    return 'Hello World!'
```




```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return 'Hello World!' >>>우리만의 FLASK 서버가 생긴 것
```



Ctrl + C : 서버 끄기

git bash

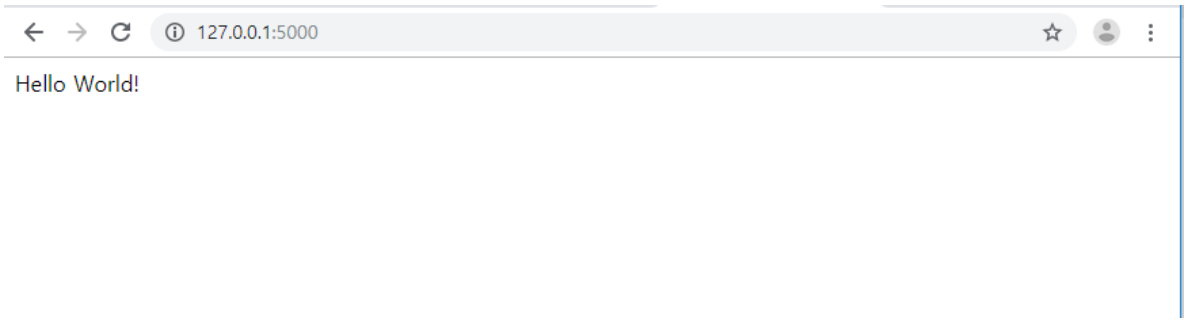
my_portfolio

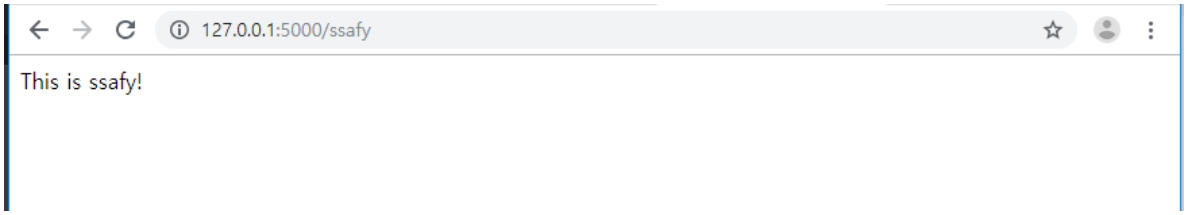
code . >>> 코드 다 열림

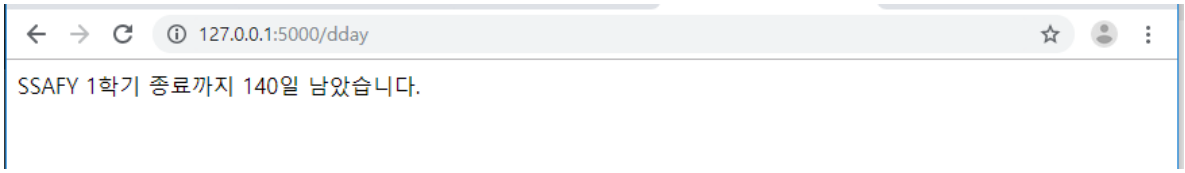
Flask 정적 라우팅, 동적 라우팅(Variable Routing)

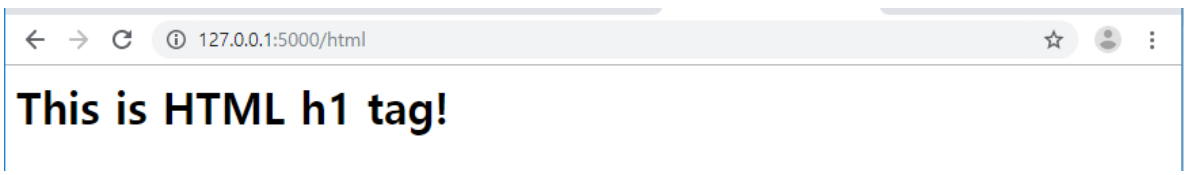
```
from flask import Flask
import datetime

app = Flask(__name__)

@app.route('/')
def hello():
    return 'Hello World!'

@app.route('/ssafy')
def ssafy():
    return 'This is ssafy!'

@app.route('/dday')
def dday():
    today = datetime.datetime.now()
    endgame = datetime.datetime(2019, 11, 29)
    td = endgame - today
    return f'SSAFY 1학기 종료까지 {td.days}일 남았습니다.'

@app.route('/html')
def html():
    return '<h1>This is HTML h1 tag!</h1>'

@app.route('/html_line')
def html_line():
```

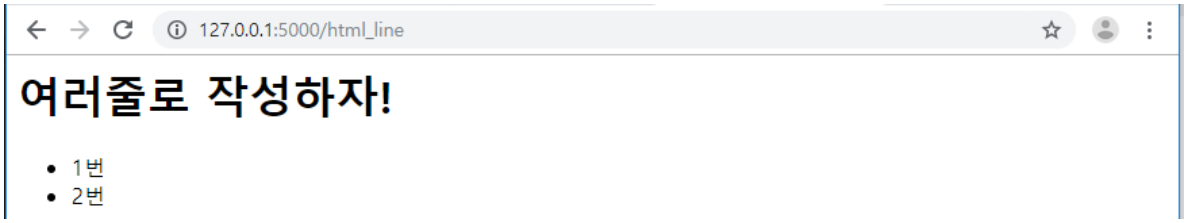
```
    return """
    <h1>여러줄로 작성하자!</h1>
    <ul>
        <li>1번</li>
        <li>2번</li>
    </ul>
    """

#Variable Routing 동적 라우팅
@app.route('/greeting/<string:name>')
def greeting(name):
    return f'반갑습니다! {name}님'

@app.route('/cube/<int:number>')
def mul(number):
    return f'{number}의 3제곱은 {number**3}입니다.'

# 점심 메뉴 리스트(5개)에서 2개를 뽑아 출력하기
# >>> flask는 문자형만 return. str()으로 감싸서 출력해줘야!

@app.route('/lunch/<int:person>')
def lunch(person):
    menu = ['쇠고기 덮밥', '빨간 우동', '카레 갈치 구이', '테이크아웃']
    td_lunch = random.sample(menu, person)
    return f'오늘의 점심 메뉴는 {td_lunch}입니다.'
```

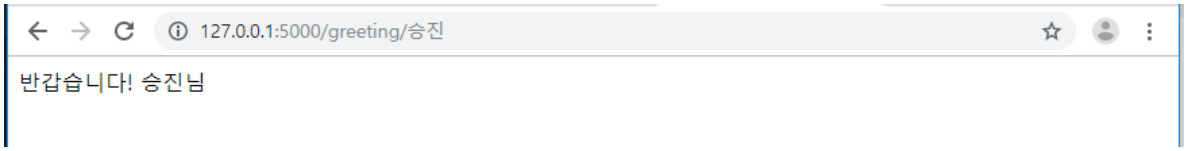


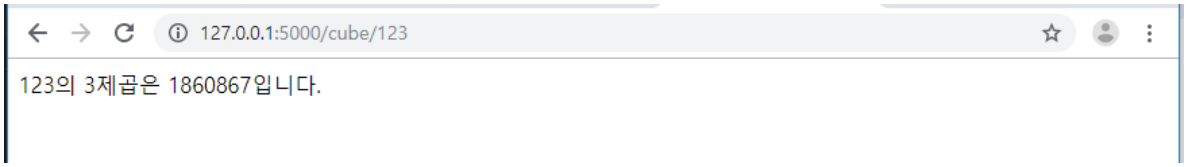


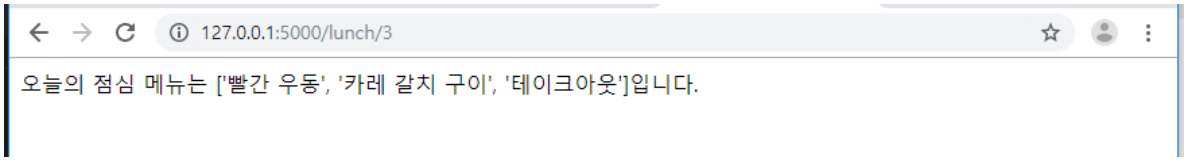













```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def hello():
    return render_template('index.html')  #render_template() 함수

@app.route('/greeting/<string:name>')
def greeting(name):
    return render_template('greeting.html', html_name=name) #template에서
greeting.html 파일 찾아서 열어줌

@app.route('/cube/<int:number>')
def cube(number):
    result = number ** 3
    return render_template('cube.html', result=result, number=number)

@app.route('/movie')
def movie():
    movies = ['토이스토리4', '스파이더맨', '비스트', '기생충', '마담 싸이코', '알라딘',
'라이온킹']
    return render_template('movie.html', movies=movies)
```

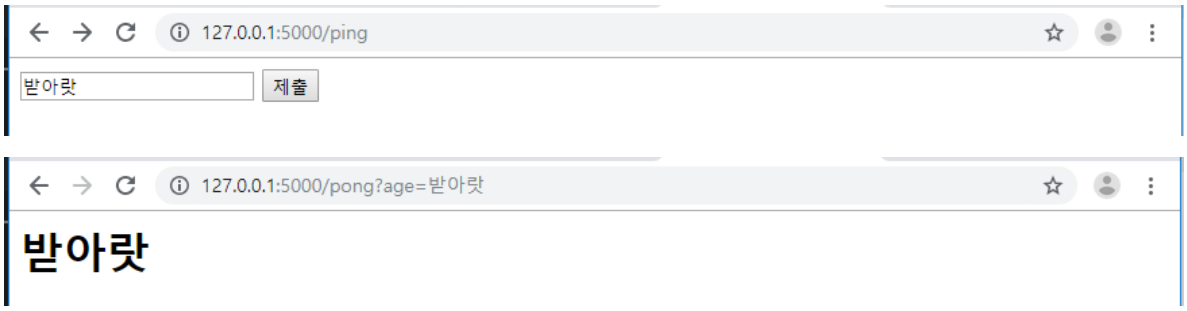
/ping /pong

```
#app.py 안의 코드
@app.route('/ping')
def ping():
    return render_template('ping.html')

@app.route('/pong')
def pong():
    age = request.args.get('age')
    return render_template('pong.html', age=age)
```

template 폴더 안에 있는 html 파일(ping.html, pong.html)

```
#ping.html
<form action="/pong">
    <input type="text" name="age">  #text 입력하는 창
    <input type="submit">  #submit버튼 생김
</form>
```

```
#pong.html
<h1>{{ age }}</h1>
```



검색사이트 연결하기

```
# NAVER 검색 연결하기
@app.route('/naver')
def naver():
    return render_template('naver.html')

# Google 검색 연결하기
@app.route('/google')
def google():
    return render_template('google.html')
```

```
#naver.html
<h1>네이버 검색</h1>
<form action="https://search.naver.com/search.naver">
    <input type="text" name="query">      #넣어 보내는 박스 이름이 query
    <input type="submit" value="검색">
</form>
```

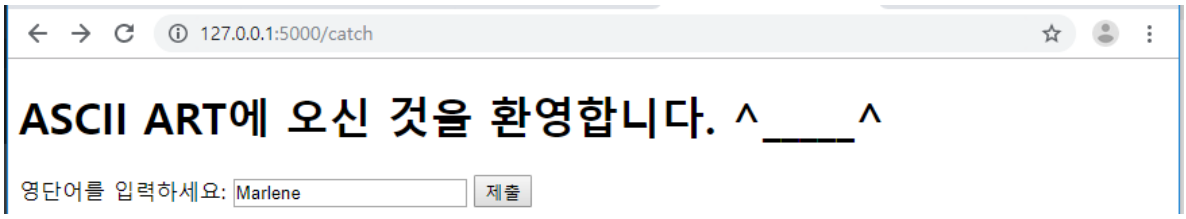
```
#google.html
<h1>Google Search</h1>
<form action="https://www.google.com/search">
    <input type="text" name="query">
    <input type="submit" value="검색">
</form>
```

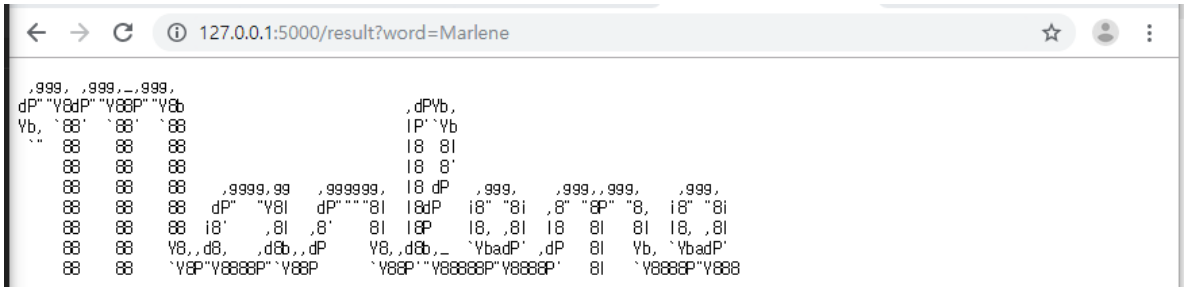
실습 : ACII ARTII

```
#app.py
from flask import Flask, render_template, request
import requests, random

app = Flask(__name__)

@app.route('/catch')
def catch():
    return render_template('catch.html')

@app.route('/result')
def result():
    #1. form태그로 날린 데이터를 받는다.
    word = request.args.get('word')

    #2. ARTII api로 (font_list)요청을 보내 받은 응답 결과를 text로 fonts(변수)에 저장
    fonts = requests.get('http://artii.herokuapp.com/fonts_list').text

    #3. fonts(str)를 fonts(list)로 바꾼다. >>형변환
    fonts = fonts.split('\n')

    #4. fonts(list)안에 들어있는 요소 중 하나를 선택해 font에 담는다.
    font = random.choice(fonts)

    #5. 위에서 우리가 만든 word와 font를 가지고 다시 요청을 보낸다.
    # 요청 결과(응답)를 result에 저장한다.
    result = requests.get(f'http://artii.herokuapp.com/make?text={word}&font=
{font}').text

    #6. 최종적으로 result에 담긴 문자열을 result.html에서 보여준다.
    return render_template('result.html', result=result)
```

```
#catch.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    <h1>ASCII ART에 오신 것을 환영합니다. ^_____^</h1>
    <form action="/result">
        영단어를 입력하세요: <input type="text" name="word">
        <input type="submit">
    </form>
</body>
</html>
```

```
#result.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    <pre>{{ result }}</pre>
</body>
</html>
```





2019-07-11(목) | StartCamp04

Dictionary

```
#1. 딕셔너리 만들기
lunch = {
    '중국집':'02'
}

lunch = dict(중국집='02')
#print(lunch)


#2. 딕셔너리 내용 추가하기
lunch['분식집'] = '031'
# print(lunch)


idol = {
    'BTS' : {
        '지민': 24,
        'RM': 25
    }
}

#3. 딕셔너리 value(값) 가져오기 >> RM 나이 출력하기
#>>1) [] 대괄호로 뽑아오기 /값이 없으면 Error가 남
print(idol['BTS']['RM'])
#>>2) 값이 없어도 Error 안 남
print(idol.get('BTS').get('RM'))


#4. 딕셔너리 반복문 활용하기
#4-1. 기본활용
for key in lunch:
    print(key)
    print(lunch[key])     #키를 넣으면 나오는건 value(값)

#4-2. .items() - key, value 모두 가져오기
for key, value in lunch.items():
    print(key, value)

#4-3. .values() - value만 가져오기
for value in lunch.values():
    print(value)

#4-4. .keys() - key만 가져오기
for key in lunch.keys():
    print(key)
```

"""

"""

더블 쿼트가 컨벤션. 공식 문서에서 사용. ''' 싱글 쿼트도 가능하긴 함'''

Dictionary 문제 풀기

1.
```
#1. 평균을 구하세요.
score = {
    '수학':80,
    '국어':90,
    '음악':100
}

# for key, value in score.items():
#     print('과목 평균은 '+ mean(value) + '입니다.')

total_score = 0
for subject_score in score.values():
    total_score += subject_score

average_score = total_score / len(score)
print(average_score)
```

2.
```
#2. 반 평균을 구하세요. >> 전체 평균
score = {
    'a': {
        '수학': 80,
        '국어': 90,
        '음악': 100
    },

    'b': {
        '수학': 80,
        '국어': 90,
        '음악': 100
    }
}

total_score = 0
length = 0

for person_score in score.values():
    for subject_score in person_score.values():
        total_score += subject_score
        length += 1

average_score = total_score / length
print(average_score)
```

3.
```
#3. 도시별 최근 3일 온도입니다.

city = {
    '서울': [-6, -10, 5],
    '대전': [-3, -5, 2],
```

```
    '광주': [0. -2, 10],
    '부산': [2, -2, 9]
}

#3-1. 도시별 최근 3일의 온도 평균은?
for name, temp in city.items():
    average_temp = sum(temp) / len(temp)
    print(f'{name} : {average_temp}')

#3-2. 도시 중 최근 3일 중에 가장 추웠던, 가장 더웠던 곳은?

#3-3. 서울은 영상 2도였던 적이 있나요? >> ex. 네 있어요! or 없어요!
result = 2 in city['서울']
if result == 1:
    print('네 있어요!')
else:
    print('없어요!')
```

로또 Pick

```
from flask import Flask, render_template, request
from bs4 import BeautifulSoup
import requests, random

app = Flask(__name__)

@app.route('/lotto_check')
def lotto_check():
    return render_template('lotto_check.html')

@app.route('/lotto_result')
def lotto_result():
    num = request.args.get('num')
    res = requests.get('https://www.dhlottery.co.kr/common.do?
method=getLottoNumber&drwNo=866')
    lotto = res.json()

    #1. 번호 6개 가져오기
    winner = []
    for i in range(1, 7):
        winner.append(lotto[f'drwtNo{i}'])

    #2. 내 번호 리스트 만들기
    numbers = []
    for num in request.args.get('numbers').split():
        numbers.append(int(num))

    if len(numbers) == 6:
        # 번호 교집합 개수 찾기
        matched = 0
        for num in numbers:
            if num in winner:
                matched += 1
        if matched == 6:
```

```
            result = '축하합니다~ 1등입니다!!! 퇴사하세요^^'
        elif matched == 5:
            if lotto['bnusNo'] in numbers:
                result = '2등입니다. 보너스 숫자 잘고르셨네요.'
            else:
                result = '3등입니다. 아깝네요. 그래도 이게 어딘가요.'
        elif matched == 4:
            result = '4등입니다. 택시비 나왔네요.'
        elif matched == 3:
            result = '5등입니다. 이번엔 숫자 잘 골라보세요.'
        else:
            result = '꽝 입니다. 돈 날리셨네요!'
    else:
        result = '번호의 수가 6개가 아닙니다.'

    return render_template('lotto_result.html', winner=winner, numbers=numbers,
result=result)
```

```
#lotto_check.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    <h1>로또로로또로또로로토르로로또</h1>
    <form action="/lotto_result">
        로또회차: <input type="number" name="num"><br>
        로또번호: <input type="text" name="numbers"><br>
        <input type="submit" value="내 집 마련">
    </form>
</body>
</html>
```

```
#lotto_result.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    <p>당첨번호!!!!!!: {{ winner }}</p>
    <p>내 번호!!!!!!!: {{ numbers }}</p>
    <p>결과는!!!: {{ result }}</p>
</body>
</html>
```

개행 :

2019-07-12(금) | StartCamp05

Telegram

```
#파이썬 코드로 텔레그램 메시지 보내기
from decouple import config
import requests

api_url = 'https://api.telegram.org'
token = config('TELEGRAM_BOT_TOKEN')
chat_id = config('CHAT_ID')
text = '오늘 점심 메뉴는 삼계탕이래 > 3<'

send_message = requests.get(f'{api_url}/bot{token}/sendMessage?chat_id=
{chat_id}&text={text}')

print(send_messagae.text)
```

```
#app.py
from flask import Flask, render_template, request
from decouple import config
import requests

app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'

@app.route('/write')
def write():
    return render_template('write.html')

@app.route('/send')
def send():
    api_url = 'https://api.telegram.org'
    token = config('TELEGRAM_BOT_TOKEN')
    chat_id = config('CHAT_ID')
    text = request.args.get('message')

    requests.get(f'{api_url}/bot{token}/sendMessage?chat_id={chat_id}&text=
{text}')

    return render_template('send.html')
```

```
#write.html
<h1>Telegram</h1>

<form action="/send">
    <input type="text" name="message">
    <input type="submit" value="메시지 전송">
</form>


#send.html
<h1>메시지가 성공적으로 전송되었습니다.</h1>
```

토큰이랑 chat id는 주민등록번호랑 마찬가지.

키는 유니크하기 때문에 키에 딕셔너리가 들어갈 수는 없음

```
from flask import Flask, render_template, request
from decouple import config
import requests

app = Flask(__name__)

api_url = 'https://api.telegram.org'
token = config('TELEGRAM_BOT_TOKEN')
chat_id = config('CHAT_ID')

naver_client_id = config('NAVER_CLIENT_ID')
naver_client_secret = config('NAVER_CLIENT_SECRET')

@app.route('/')
def hello_world():
    return 'Hello, World!'

@app.route('/write')
def write():
    return render_template('write.html')

@app.route('/send')
def send():
    text = request.args.get('message')

    requests.get(f'{api_url}/bot{token}/sendMessage?chat_id={chat_id}&text=
{text}')

    return render_template('send.html')

@app.route(f'/{token}', methods=['POST'])   #/{token} 이건 아무나 못 들어오게 랜덤
숫자 주소 뒤에 주는 효과
def telegram():
    #1. 구조 print 해보기 & 변수 저장
    # print(request.get_json())
    # print(type(request.get_json()))
    from_telegram = request.get_json()
```

```
    print(from_telegram)

    #2. 메시지를 그대로 돌려 보내기
    if from_telegram.get('message') is not None: #텔레그램에서 수정된 메세지 다시 보내면 none으로 표시됨. 그 경우 제외시키는 것.(예외처리)
        chat_id = from_telegram.get('message').get('from').get('id')
        text = from_telegram.get('message').get('text')
        # print(text)

        # print(res)

        #3. Keyword(번역)
        if text[0:4] == '/번역 ':
            headers = {'X-Naver-Client-Id': naver_client_id, 'X-Naver-Client-Secret': naver_client_secret}
            data = {'source': 'ko', 'target': 'en', 'text': text[4:]}

            papago_res = requests.post('https://openapi.naver.com/v1/papago/n2mt', headers=headers, data=data)
            print(papago_res.json())
            text = papago_res.json().get('message').get('result').get('translatedText')
            print(text)
        res = requests.get(f'{api_url}/bot{token}/sendMessage?chat_id={chat_id}&text={text}')

    return '', 200
```

```
TELEGRAM_BOT_TOKEN='837930925:AAGKAE4lNZNrPz3Mtd7tHyAt-4nM-Z8SQH4'
CHAT_ID='864904475'

NAVER_CLIENT_ID='uaP3zPLgyMkvx7cY9ddK'
NAVER_CLIENT_SECRET='3z521lCBua'
```

> **pythonanywhere를 통해 배포**