

Session 0.

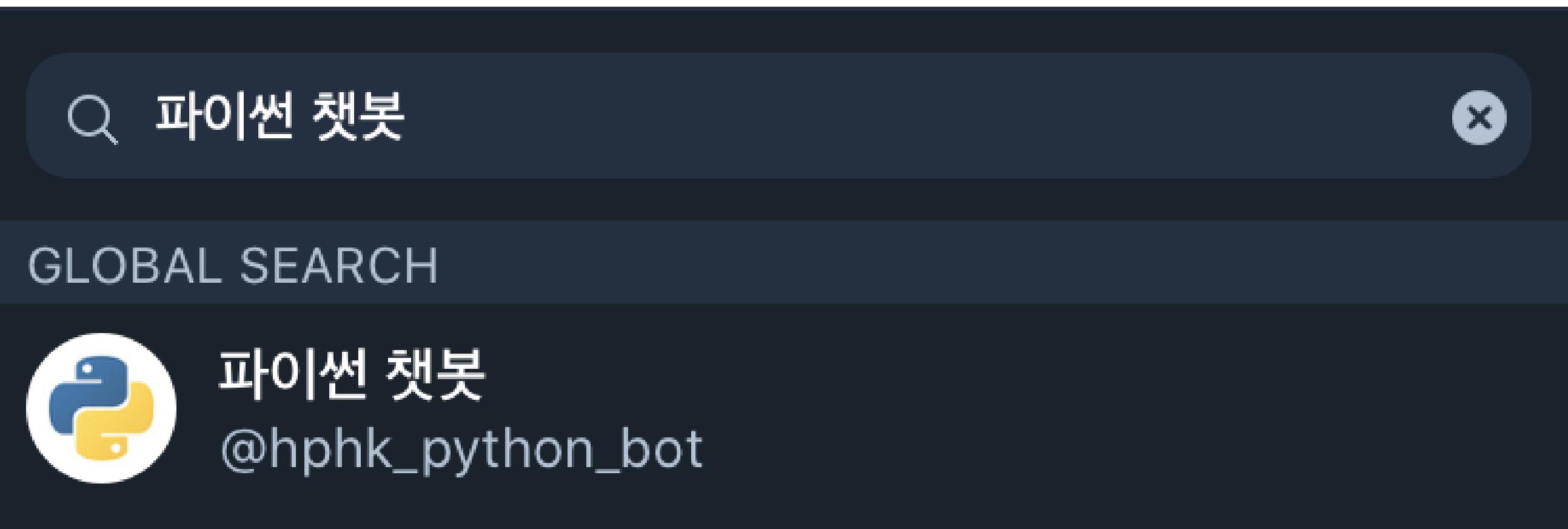
Python Chatbot 맛보기



Telegram 설치

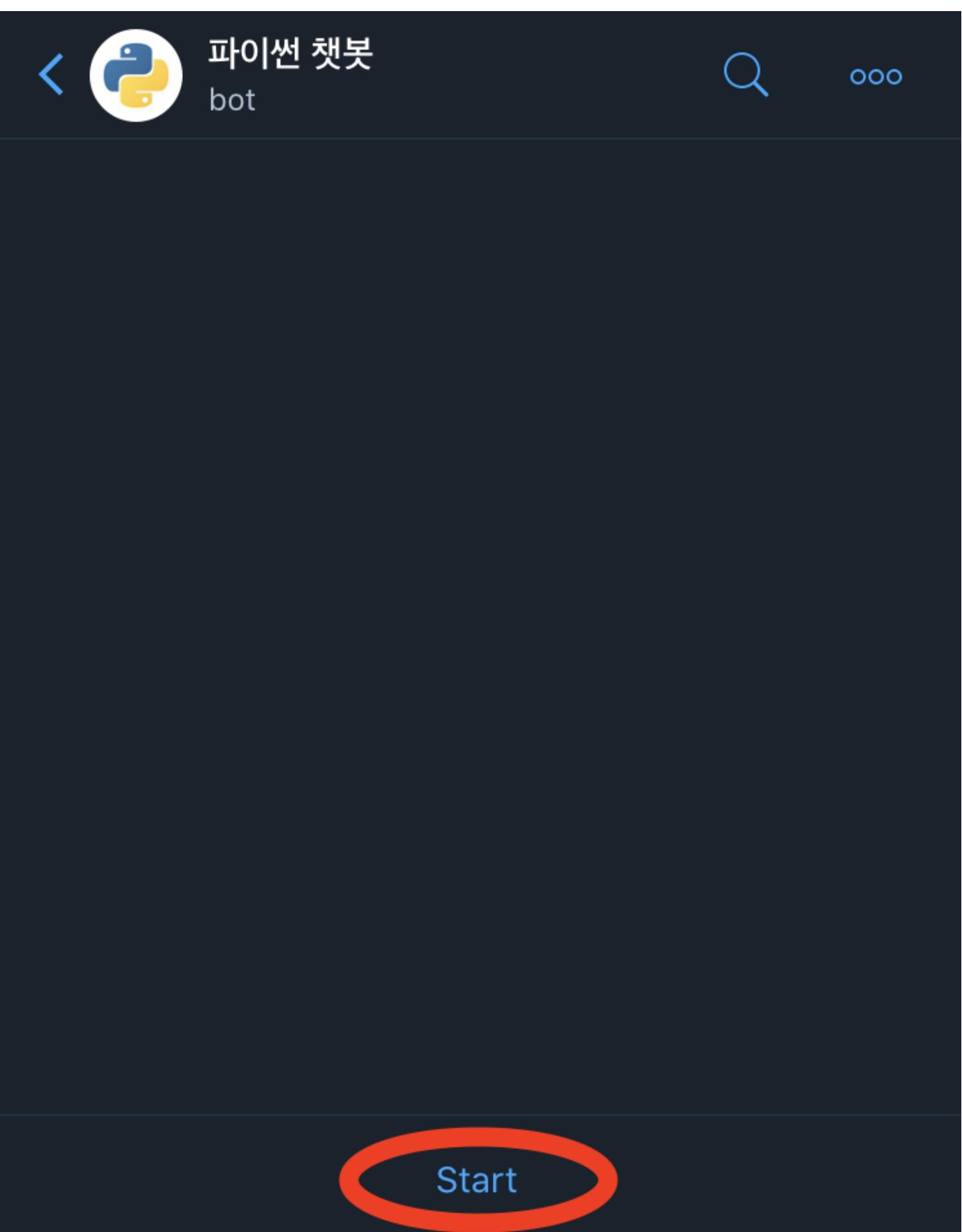
Life Hacking

1. ‘파이썬 챗봇’ 검색



Life Hacking

2. '파이썬 챗봇' Start!



Life Hacking

3. ‘파이썬 챗봇’에게 인사, ‘안녕’



Life Hacking

4. 링크를 클릭하여 회원가입

파이썬 챗봇 회원가입

이름 *

비밀번호 *

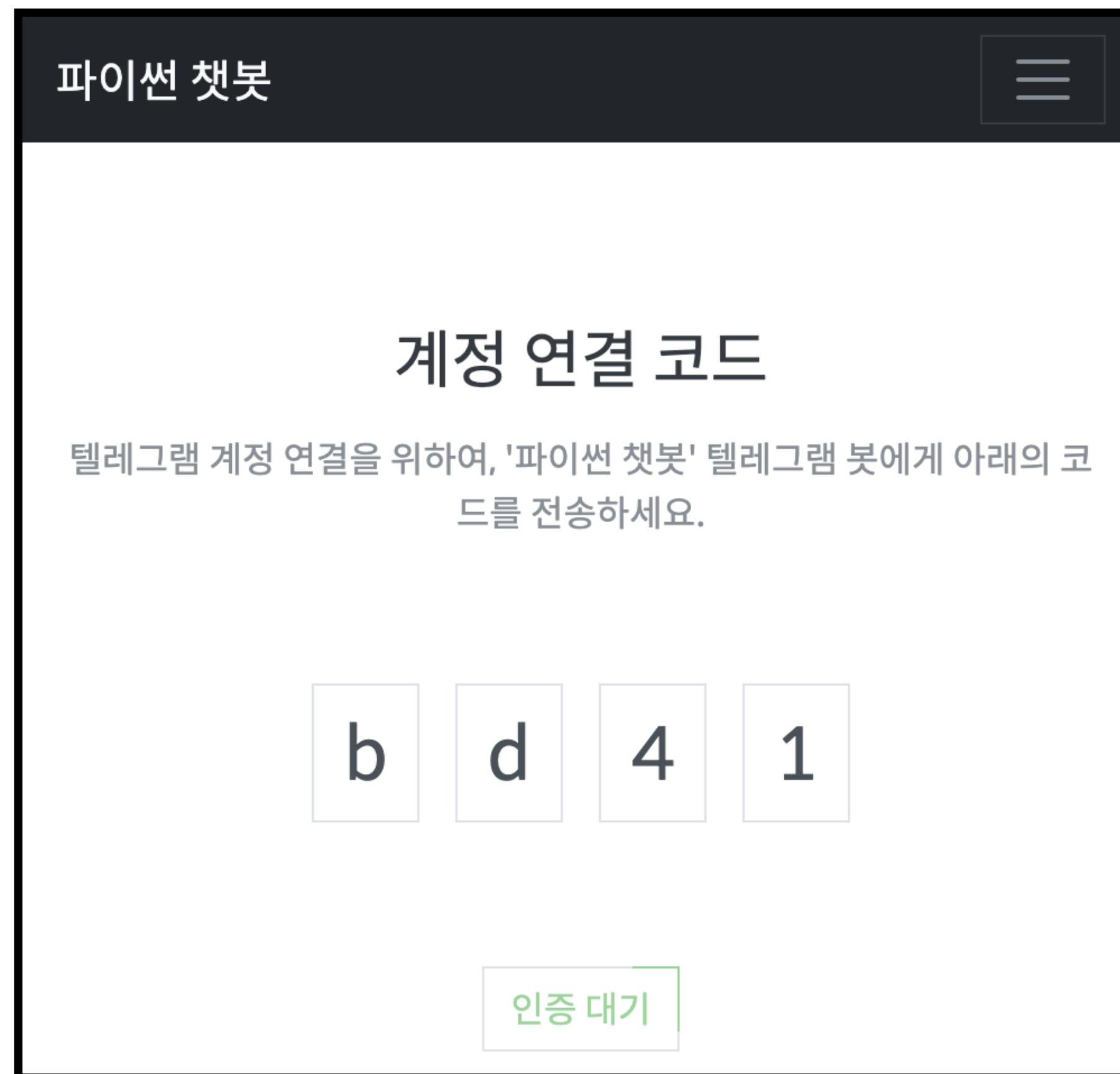
비밀번호는 4자 이상이어야 합니다.

비밀번호 확인 *

[회원가입](#) [로그인](#)

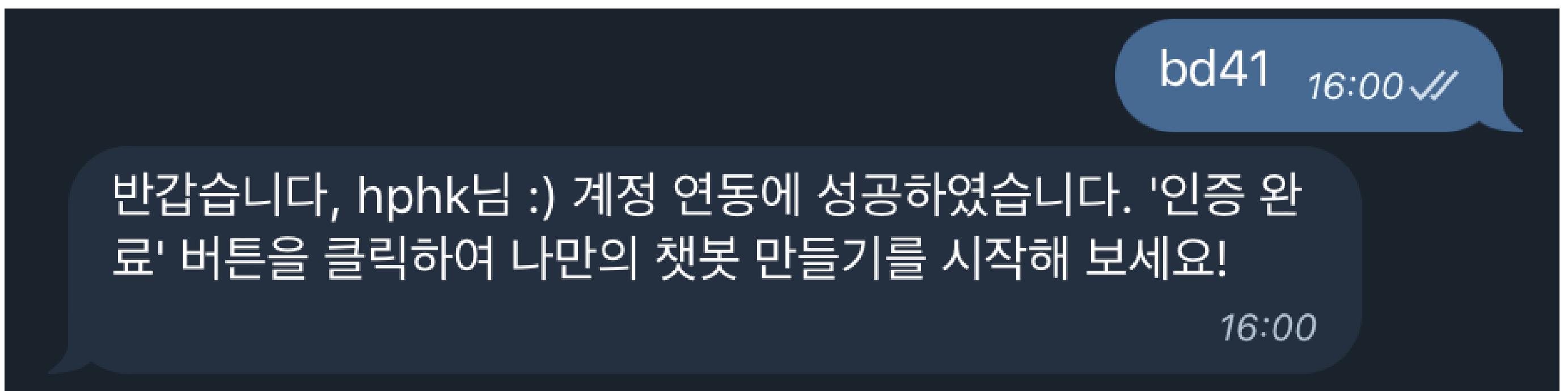
Life Hacking

5. 계정 연결 코드를 확인 후, 텔레그램 화면으로 이동



Life Hacking

6. 계정 연결 코드 입력



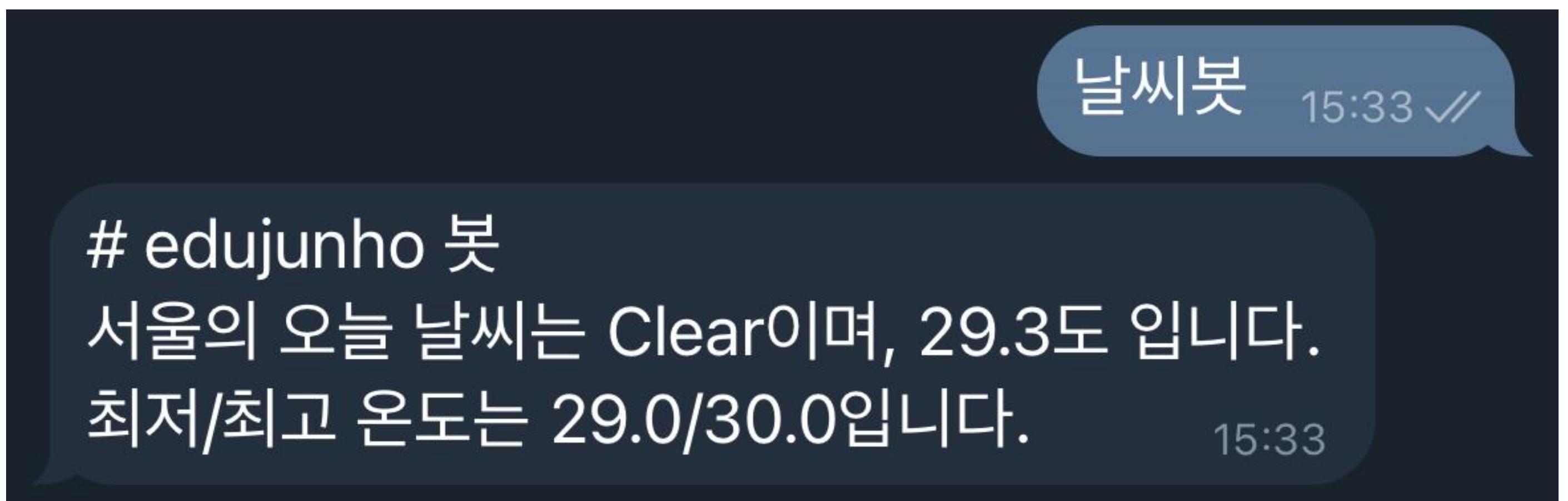
Life Hacking

7. '안녕봇' 입력



Life Hacking

8. '날씨봇' 입력



Life Hacking

9. '점심메뉴봇'
10. '로또봇'
11. '미세먼지봇'
12. '환율봇'
13. '코스피봇'

```
import requests
from bs4 import BeautifulSoup

KEY = '...'
url = f'http://apis.data.go.kr/B552584/ArpltnInforInqireSvc/getCtprvnRltmMesureDnsty?serviceKey={KEY}&numOfRows=10&pageNo=3&sidoName=서울&ver=1.0'

response = requests.get(url).text
data = BeautifulSoup(response, 'xml')
item = data('item')[7]
time = item.dataTime.text
station = item.stationName.text
dust = int(item.pm10Value.text)

print(f'{time} 기준 {station}의 미세먼지 농도는 {dust} 입니다.')

if 150 < dust:
    print('매우나쁨')
elif 80 < dust <= 150:
    print('나쁨')
elif 30 < dust and dust <= 80:
    print('보통')
else:
    print('좋음')
```

‘미세먼지봇’ 전체 코드

하지만,
이렇게 쉽게
가능할까?

가능하다!
프로그래밍이
쉬워졌기 때문.

왜 쉬워졌는가?

1. 언어가 쉬워졌다.
2. 환경이 좋아졌다.
3. 좋은 강사가 있다.

1. 프로그래밍 언어가 쉬워졌다.

과거의 C언어

```
#include <stdio.h>

int main(void)

{
    print('Hello');
    return 0;
}
```

1.
프로그래밍
언어가
쉬워졌다.

지금의 Python

```
print('Hello')
```

2. 프로그래밍 환경이 좋아졌다.

- 과거 : 모든 것을 스스로 제작
- 현재 : 이미 만들어진 것을 활용
(오픈소스)

Open Source

: 모두에게 열려있는 (소스)코드

오픈 소스

Open Source

제작자의 권리 를 지키면서,
누구나 코드 열람이 가능한,
오픈 소스 라이센스가 등장.

공동 참여 프로젝트들이 생겨나는 등의
생태계가 구성 됨.



기술생산 == 기술활용

- Samsung
- Microsoft
- Oracle, SAP



기술 이해만 있다면 누구든 활용 가능

- Google Tensorflow – Airbnb, Xiaomi
- AWS – Pinterest, Netflix
- IBM Watson – Legends, Minter Ellison

실제 오픈소스 활용 예시 : 카카오톡

설정 – 고객센터/도움말 – 오픈소스 라이선스

**수레바퀴를
두 번 만들지 마라**

모든 것을 바닥부터 만들지 않아도 된다.

거인의 어깨 위에서 프로그래밍 시작하기

이미 만들어진 것을 잘 활용하면 된다.

Session 1. Python 기초

Session 1. Python 기초

#Lifeisshort # Youneedpython

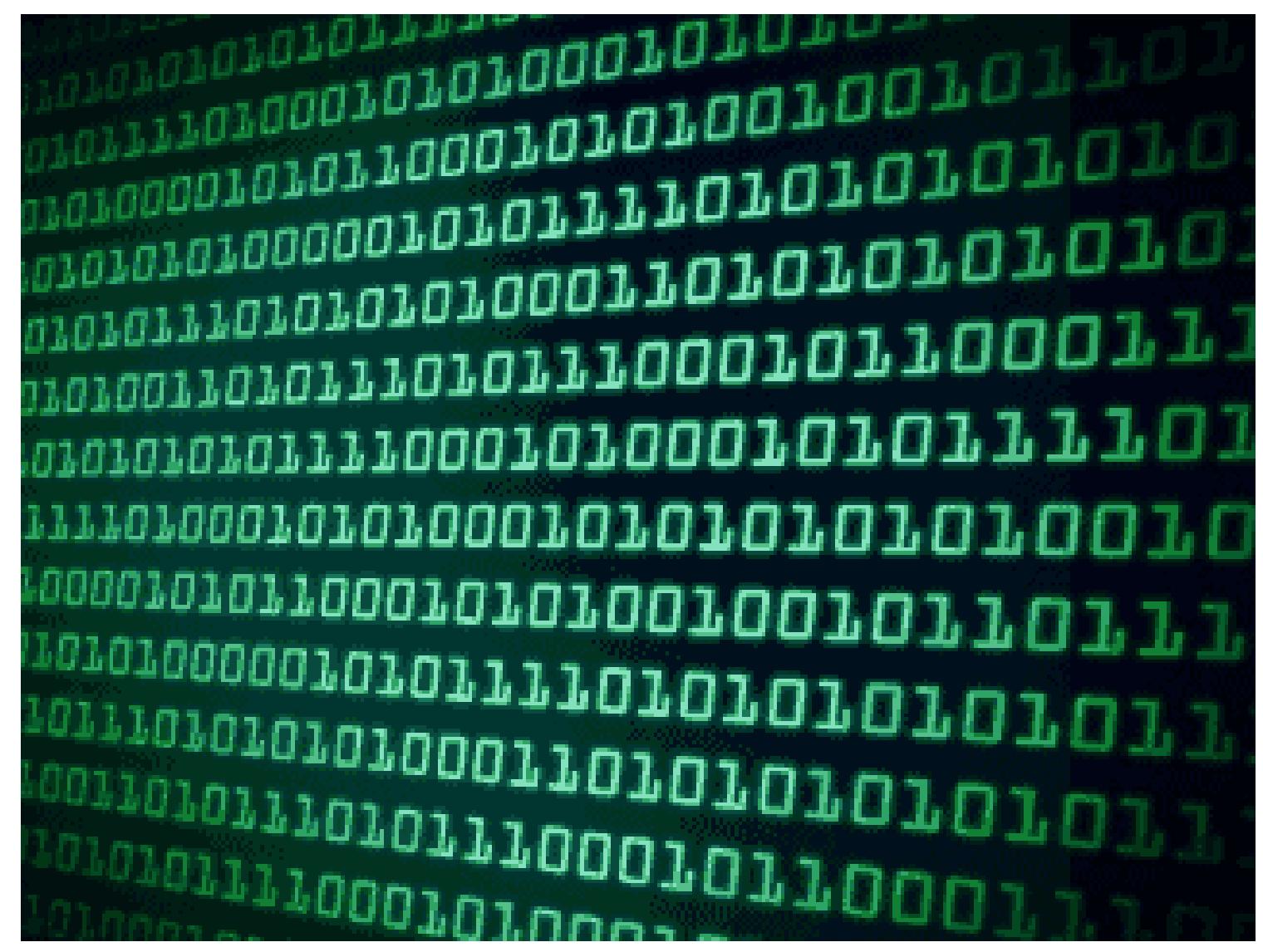
컴퓨터 프로그래밍 언어

기술을 사용하기 위한, 결과물을 만들기 위한 기초 학습.

컴퓨터 프로그래밍 언어

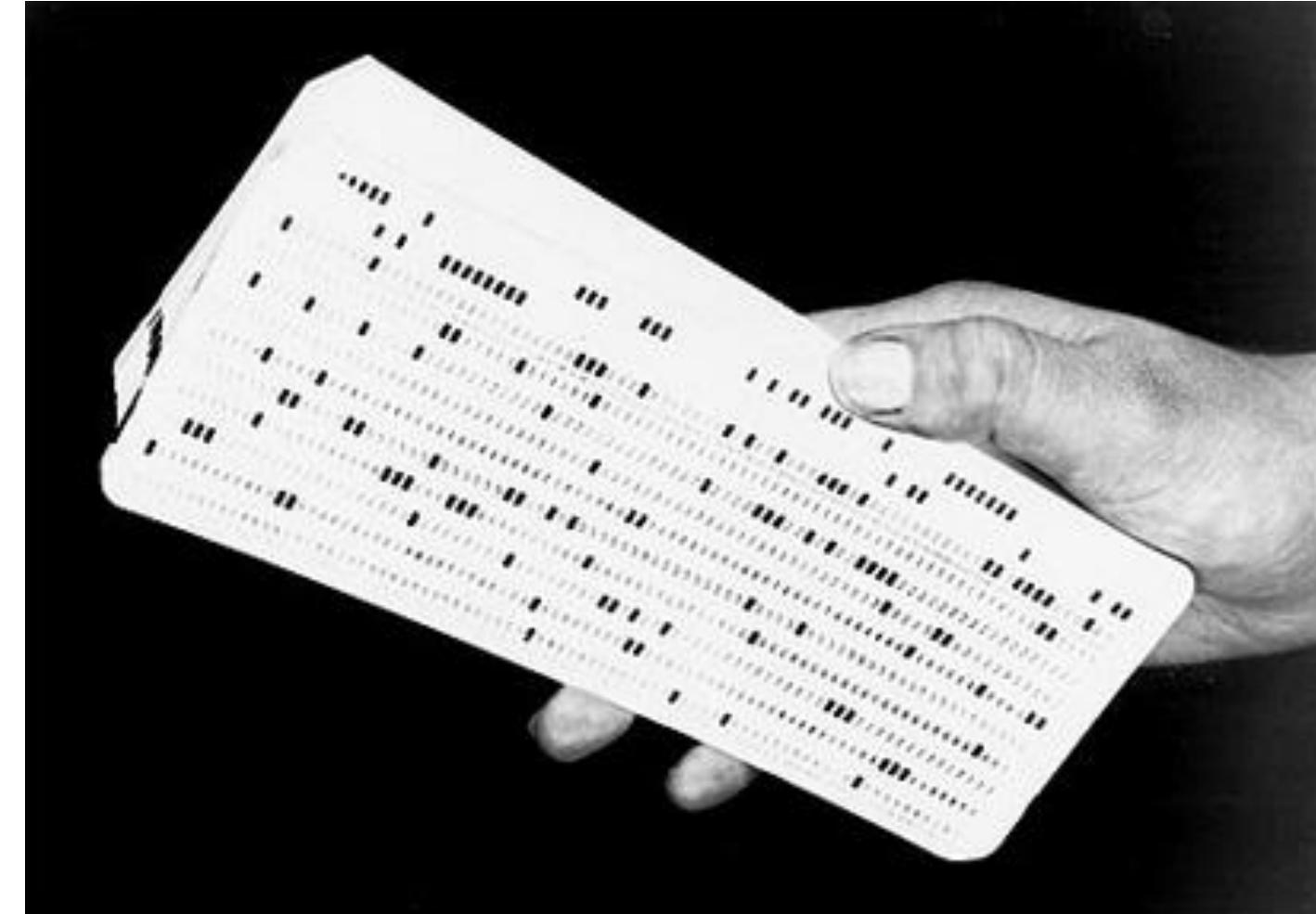
-- 컴퓨터에게 무언가를 시킬 때 쓰는 말

컴퓨터 프로그래밍 언어
: 원래는 아주 어려웠음..



컴퓨터가 이해하는 언어

????



옛날 옛적..

컴퓨터한테 뭐라도 시키려면 이 고생을…



A large rectangular area filled with binary code (0s and 1s) arranged in a grid pattern.

통역기를 사용하기 시작!

컴퓨터한테 얘기하기가 쉬워짐

```
1 .386
2 .model flat, c
3 .stack 4096
4 .data
5     message db "Hello, World!", 0Dh, 0Ah, 00h
6
7 .code
8     extern _heap_init: proc
9     extern _mtinit: proc
10    extern _ioinit: proc
11
12    extern exit: proc
13    extern printf: proc
14
15    main proc
16        ; _heap_init(1)
17        push 1
18        call _heap_init
19        add esp, 4
20
21        ; _mtinit(1)
22        push 1
23        call _mtinit
24        add esp, 4
26    ; _ioinit()
27    call _ioinit
28
29    ; main()
30
31    push offset message
32    call printf
33    add esp, 4
34
35    ; exit(0)
36    push 0
37    call exit
38    add esp, 4
39
40    ; return 0
41    xor eax, eax
42    ret
43    main endp
44
45    end main
```

어셈블리어

외계어

Link: <http://codepad.org/QOmLUb6y> [[raw code](#) | [output](#) | [fork](#)]

C, pasted 1 second ago:

```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("Hello World");
6     return 0;
7 }
```

Output:

```
1 Hello World
```

C언어

이해 될 듯, 말 듯

```
print('Hello World')
```

Python

거의 영어

"Life is short,
You need Python."

-- Bruce Eckel

파이썬(Python)

1. 쉽다.
2. 많은 사람들이 사용한다.
3. 많은 것을 할 수 있다.

Python

설치하기

<https://abit.ly/ssafy7-document>

시작하기 전에…

1. 대/소문자
2. 띄어쓰기
3. 스펠링

Python의 문법

1. 저장
2. 조건
3. 반복

1. 저장

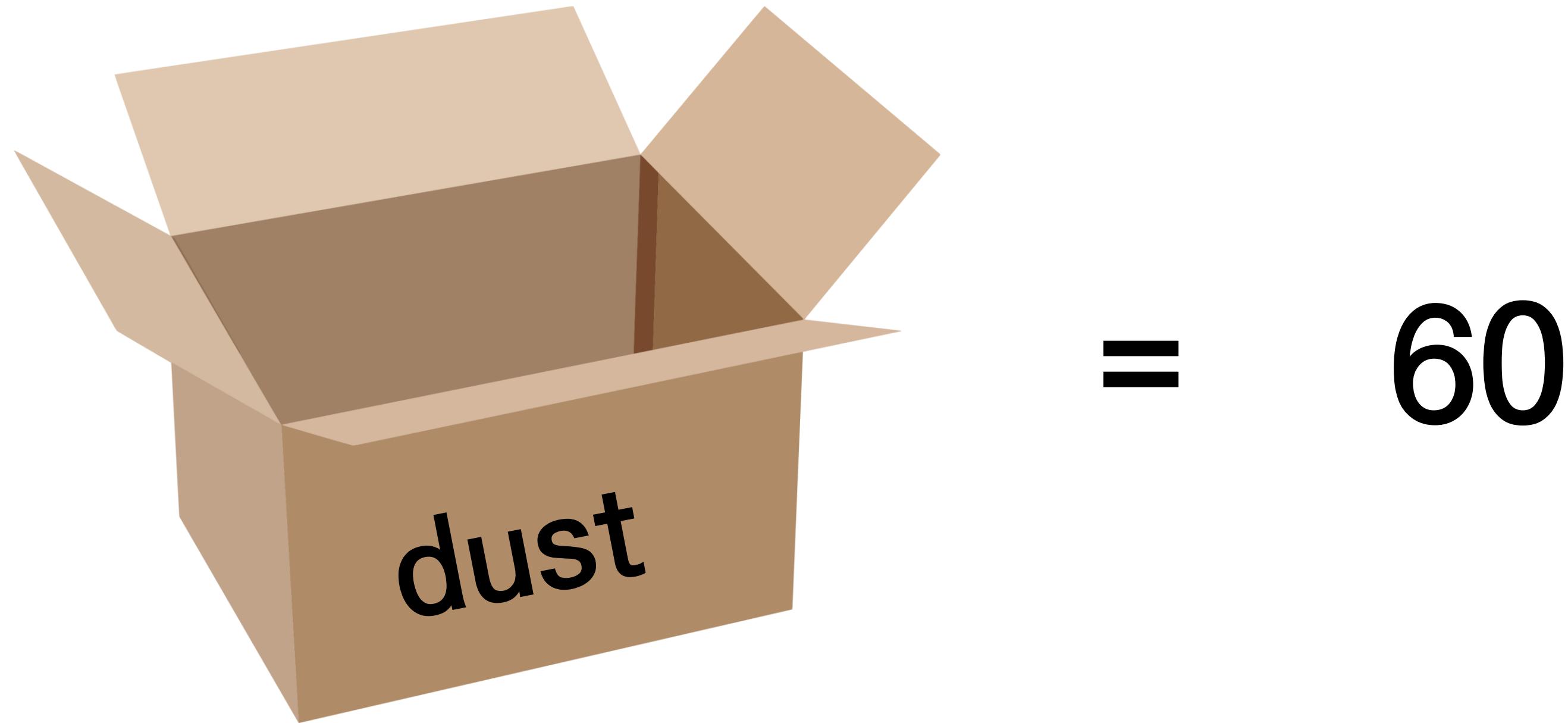
- 저장의 개념은? Save?
- 무엇을, 어떻게 저장할 수 있는가?



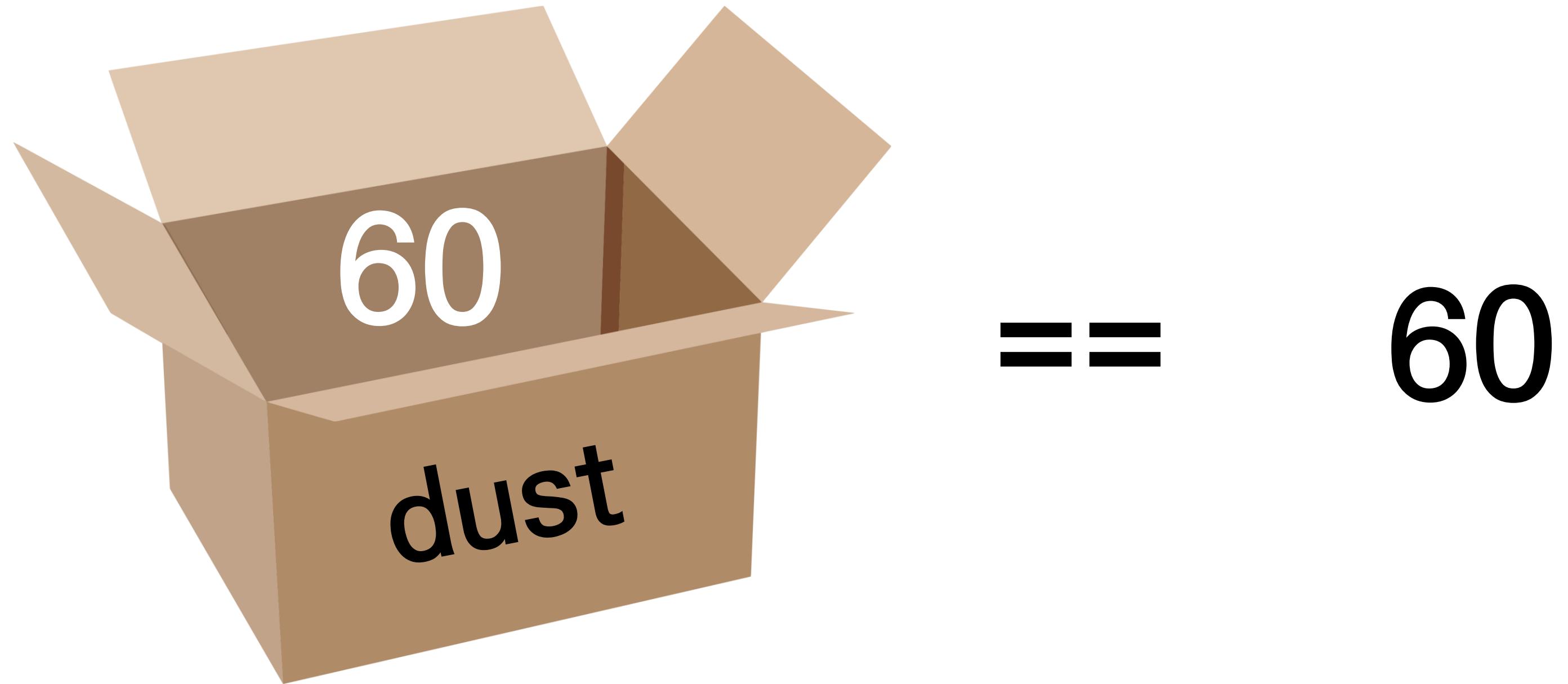
박스를 떠올리세요



dust = 60



dust 는 60이다 (X)
dust 에 60을 저장(할당)한다 (O)



`dust == 60`
`dust` 에 저장된 값은 60과 같다.

1. 저장

- 저장의 개념은? Save?
- 무엇을, 어떻게 저장할 수 있는가?

무엇을 저장하는가

컴퓨터는 무엇을 인식할 수 있나?
박스에 담을 수 있는 것들

무엇을
저장하는가

1) 숫자
2) 글자
3) 참/거짓

1. 현실세계에 존재하는 모든 숫자.
(글자가 들어가면 안됨.)
 2. 기본적인 연산이 가능. (계산기)
- O : 12, -365, 3.14159265358
X : 23ab, ba123

무엇을
저장하는가

- 1) 숫자
- 2) 글자
- 3) 참/거짓

1. 현실세계에 존재하는 모든 글자.
2. 따옴표로 둘러싼 글자 or 숫자.

'미세먼지'
'58도 글자입니다.'
'글자는 반드시 따옴표를 붙여야합니다!'

58
숫자

VS

'58'
글자

무엇을 저장하는가

- 1) 숫자
- 2) 글자
- 3) 참/거짓

True, False
단 두 가지!

조건 / 반복에 사용됨.

$$\begin{array}{ll} 300 > 200 & \Rightarrow \text{True} \\ 150 == 161 & \Rightarrow \text{False} \end{array}$$

실습 1

- 변수 활용하기 (hello.py)
 - (공손하게) 여러번 인사하는 챗봇

어떻게 저장하는가.

다양한 종류의 박스

어떻게 저장하는가

1) 변수 (variable)

- 저장된 값을 변경 할 수 있는(variable) 박스.
- 숫자, 글자, 참거짓을 담을 수 있다.

```
dust = 58  
dust = 60  
print(dust)
```

어떻게 저장하는가

1) 변수 (variable)

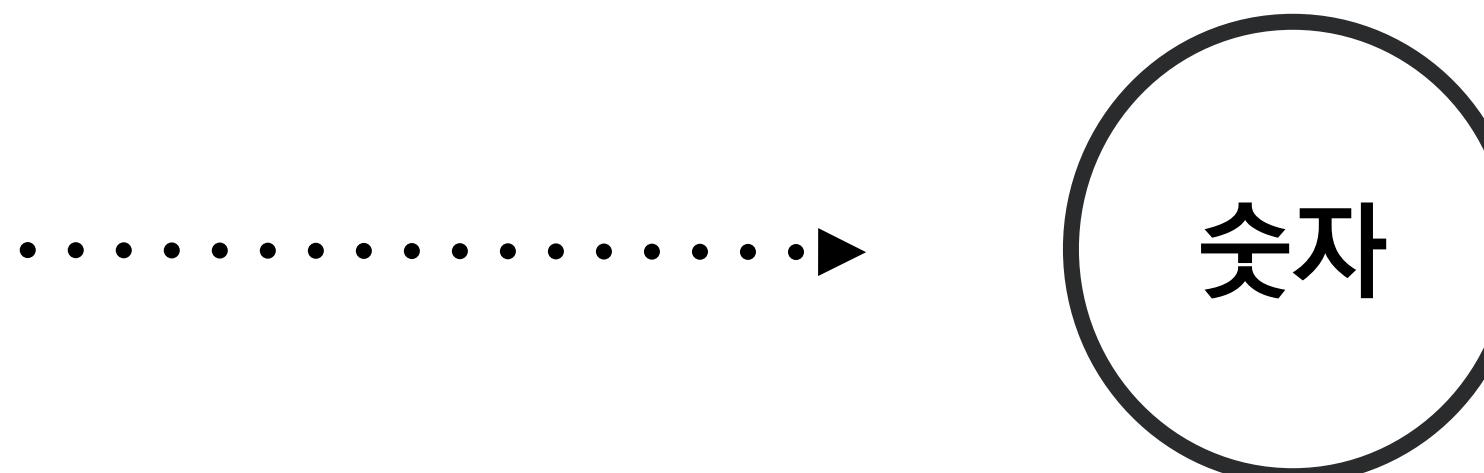
- 저장된 값을 변경 할 수 있는(variable) 박스.
- 숫자, 글자, 참거짓을 담을 수 있다.

```
dust = 58  
dust = 60  
print(dust)
```

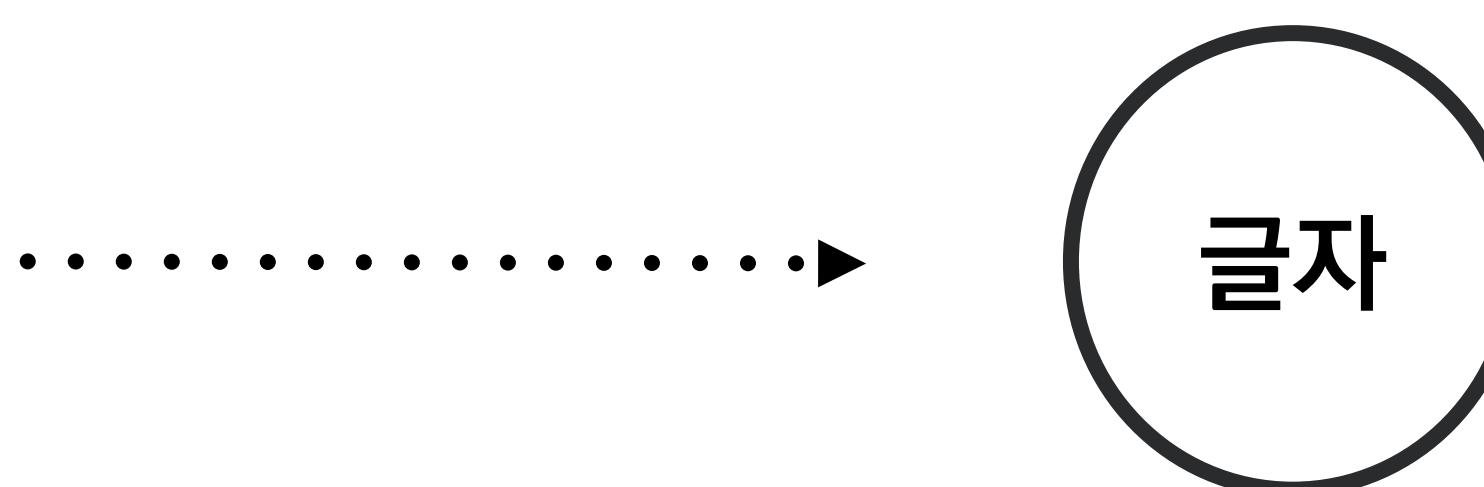
60



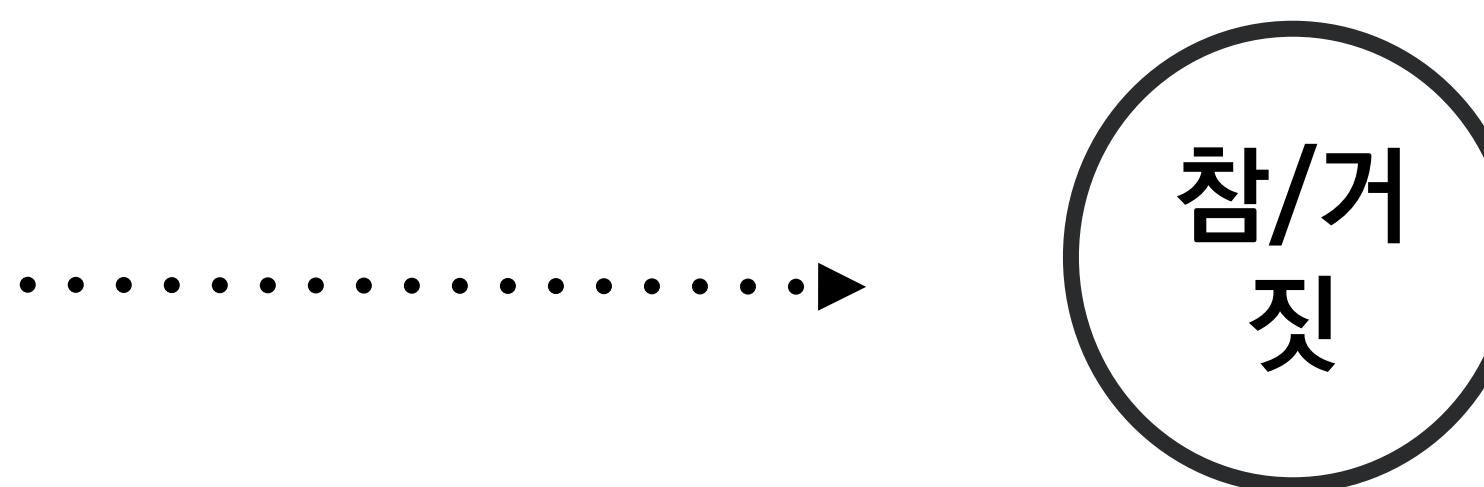
dust = 60



input = '미세먼지'



today = True



print(hello)
hello 변수에 담긴 내용을 출력해주세요.

VS

print('hello')
'hello'라는 글자를 출력해주세요.

3일치의 정보를 저장한다면?



100일치의 정보를 저장한다면?



...



코드로 작성해보면,

dust_today = 58

dust_1d_ago = 40

dust_2d_ago = 70

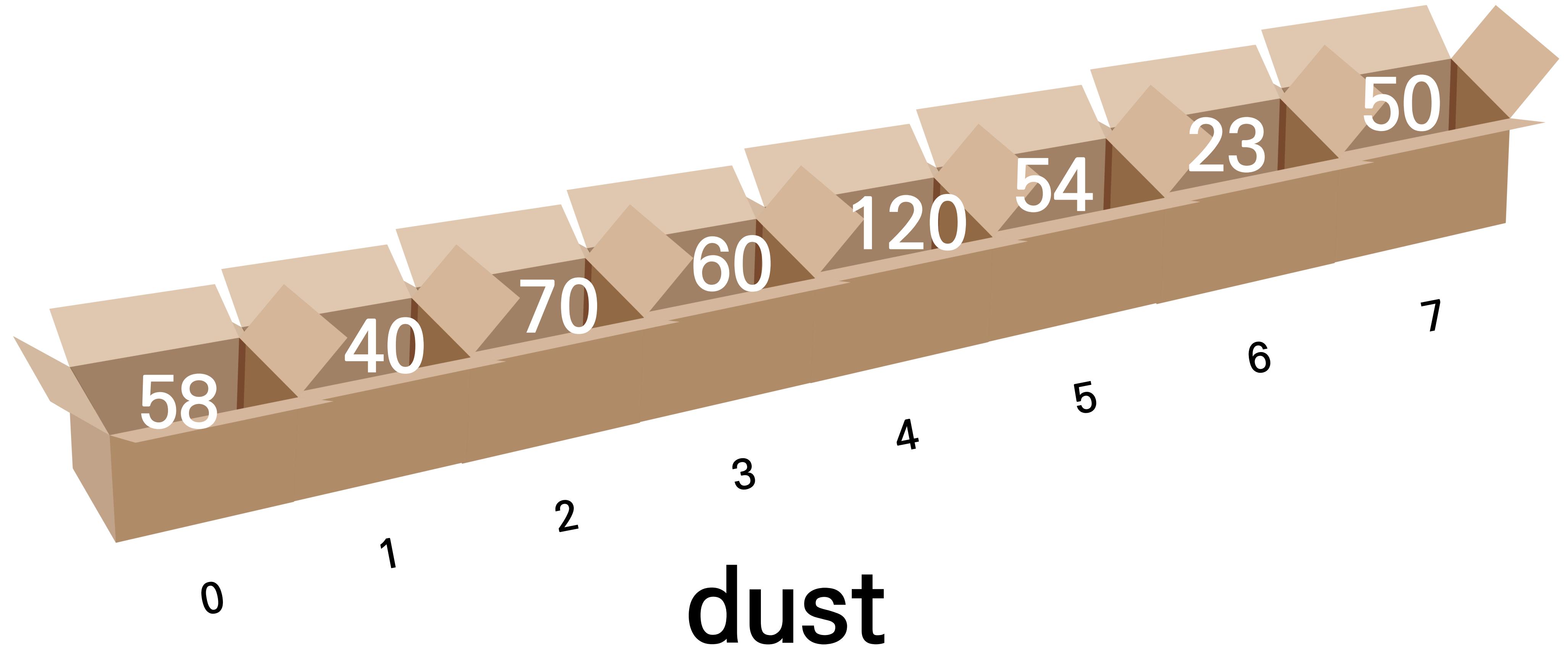
...

dust_100d_ago = 66

더 편한 방법은 없을까?



박스들을 붙이자.



어떻게 저장하는가

2) 리스트 (List)

박스의 리스트(리스트)
박스가 순서대로 여러 개가 붙어있다.

```
dust = [58, 40, 70]  
print(dust[1])
```

어떻게 저장하는가

2) 리스트 (List)

박스의 리스트(리스트)
박스가 순서대로 여러 개가 붙어있다.

```
dust = [58, 40, 70]  
print(dust[1])
```

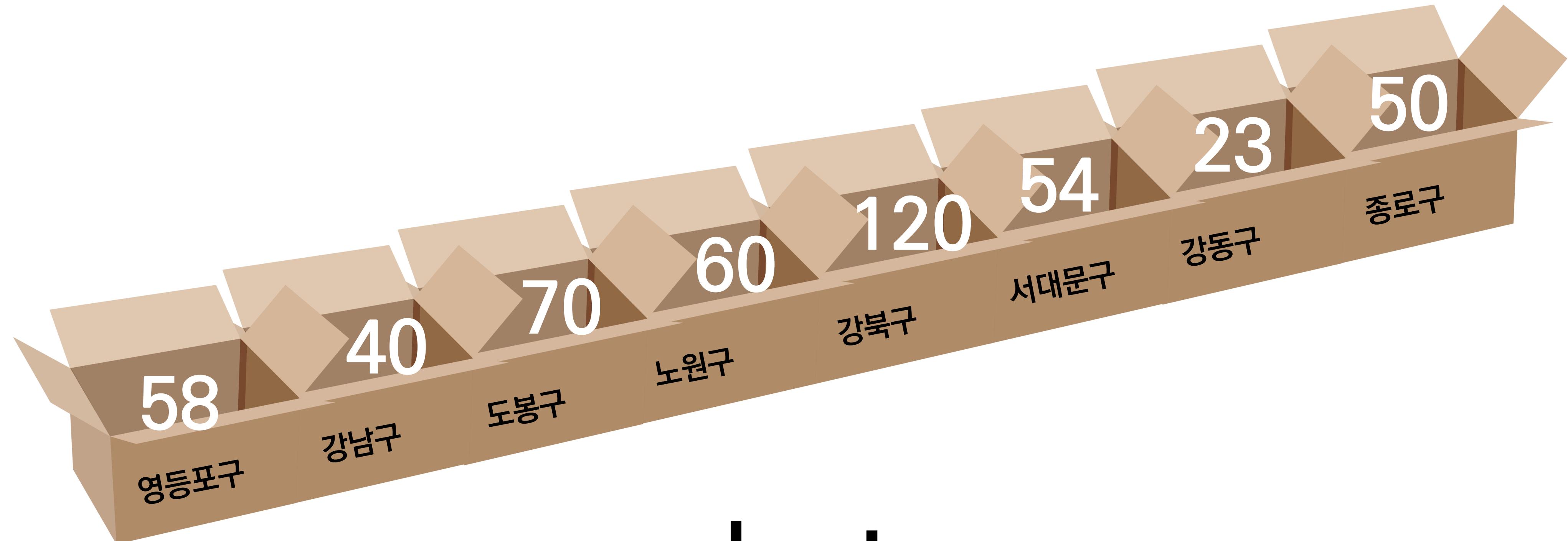
40

```
location = ['영등포구', '강남구', ...]  
value = [58, 40, ...]
```

더 편한 방법은 없을까?

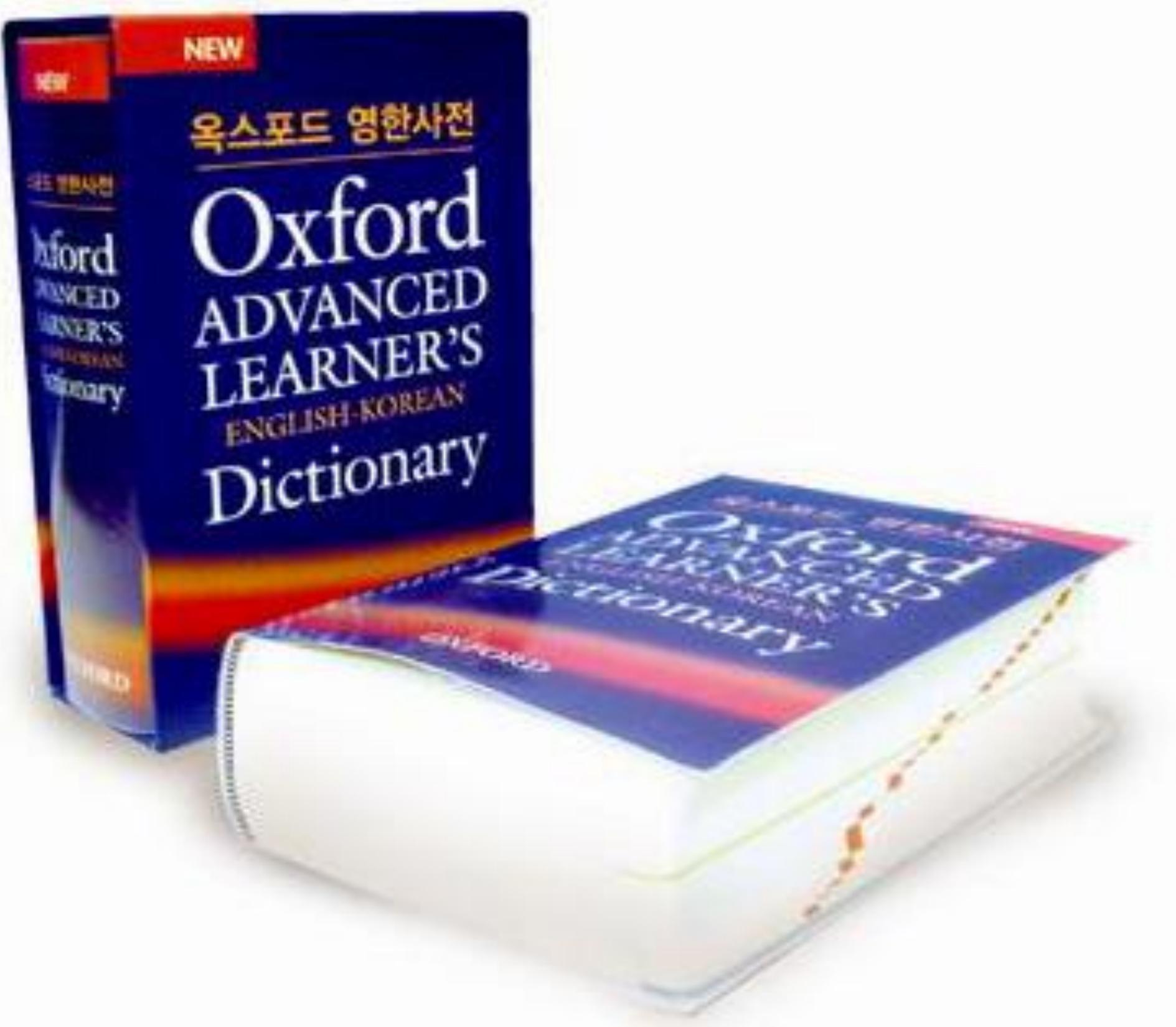


견출지를 붙여보자



dust

SW StartCamp



어떻게 저장하는가

3) 딕셔너리 (dictionary)

궁극의 박스 dictionary
“견출지 불인 박스들의 묶음”

```
dust = {'영등포구': 58, '강남구': 40}  
print(dust['영등포구'])
```

어떻게 저장하는가

3) 딕셔너리 (dictionary)

궁극의 박스 dictionary
“견출지 불인 박스들의 묶음”

```
dust = {'영등포구': 58, '강남구': 40}  
print(dust['영등포구'])
```

58

	저장하는 법	불러오는 법	출력값
변수 (박스 1개)	dust = 40	dust	40
리스트 (박스 여러개)	dust = [40, 50, 80]	dust[0]	40
딕셔너리 (견출지 붙인 박스)	dust = {'영등포구': 40, '강남구': 50}	dust['영등포구']	40

	저장하는 법	불러오는 법	출력값
변수 (박스 1개)	dust = 40	dust	40
리스트 (박스 여러개)	dust = [40, 50, 80]	dust[0]	40
딕셔너리 (견출지 붙인 박스)	dust = {'영등포구': 40, '강남구': 50}	dust['영등포구']	40

실습 2

- 리스트 활용하기 (`lunch.py`)
 - 점심 메뉴 후보 정하기

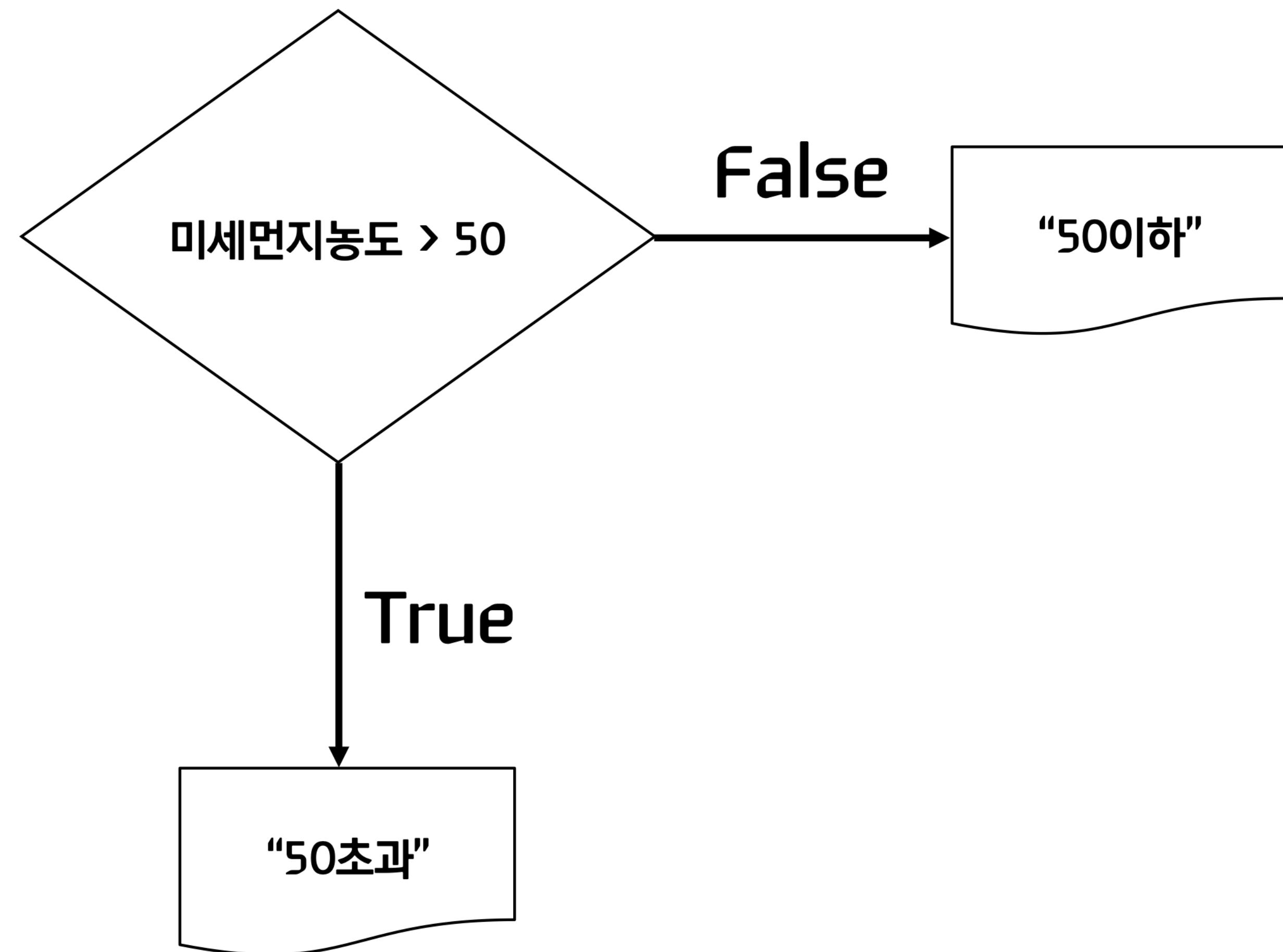
실습 3

- 딕셔너리 활용하기 (`phone_book.py`)
 - 식당 + 전화번호 알려주기

2. 조건

```
if True:  
    print('조건문입니다.')
```

조건 if/else



조건
if/else

```
if dust > 50:  
    print('50초과')  
else:  
    print('50이하')
```

조건
if/else

```
if dust > 50:  
    print('50초과')  
else:  
    print('50이하')
```

조건

if/else

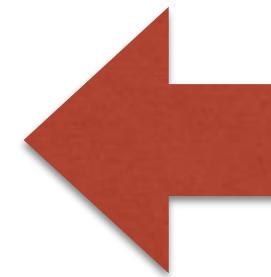
dust = 60

if 60 > 50: True

print('50초과')

else:

print('50이하')



50초과

조건

if/else

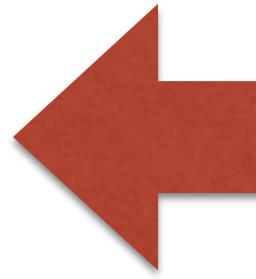
dust = 40

```
if 40 > 50: False
```

```
print('50초과')
```

```
else:
```

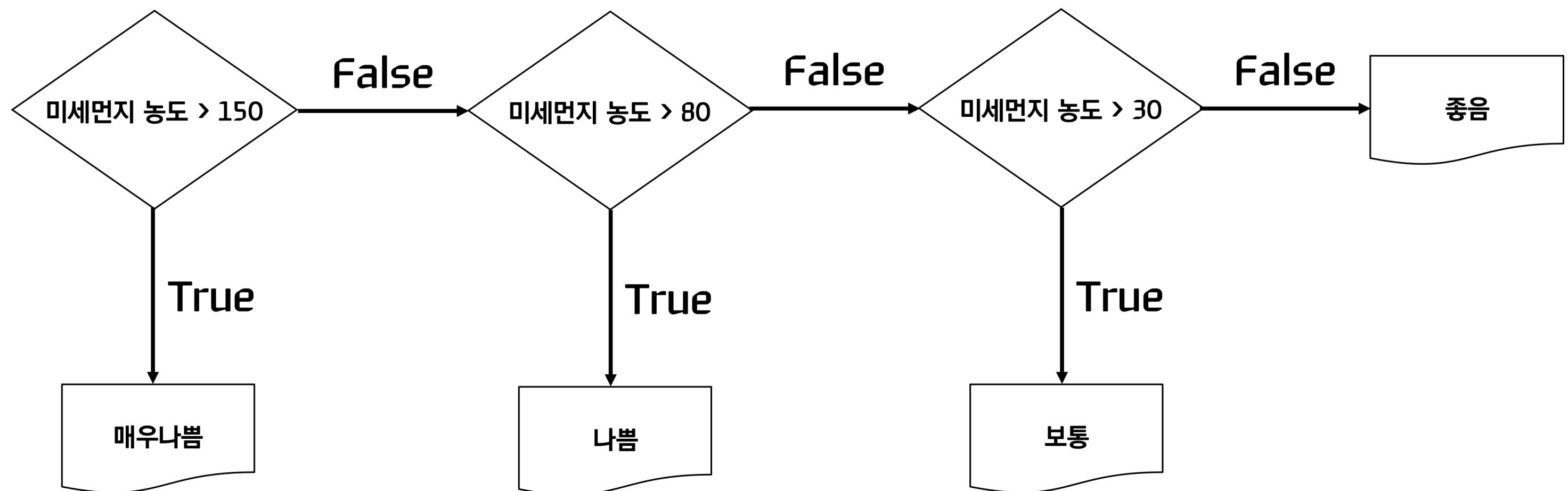
```
print('500이하')
```



500이하



조건 if/elif/else



조건

if/elif/else

```
if dust > 150:  
    print('매우나쁨')  
elif 조건:  
    print('나쁨')  
elif 조건:  
    print('보통')  
else:  
    print('좋음')
```

미세먼지농도	
등급	
151~	매우나쁨
81 ~ 150	나쁨
31 ~ 80	보통
0 ~ 30	좋음

```
if 150 < dust:  
    print('매우나쁨')
```

1. dust라는 변수에 농도가 API로부터 받은 정보가 저장되어있다.
2. 기상청에 미세먼지 농도 등급은 아래와 같다.

Q. 조건문과 출력하는 법을 통하여 코드를 작성하라.

Ex) 55, 보통

미세먼지농도	등급
151 ~	매우나쁨
81 ~ 150	나쁨
31 ~ 80	보통
0 ~ 30	좋음

<Hint>

만약 (미세먼지 농도가 150보다 크면):

출력하자('매우나쁨')

아닌데 만약에 (미세먼지 농도가 150이하 80초과면):

출력하자('나쁨')

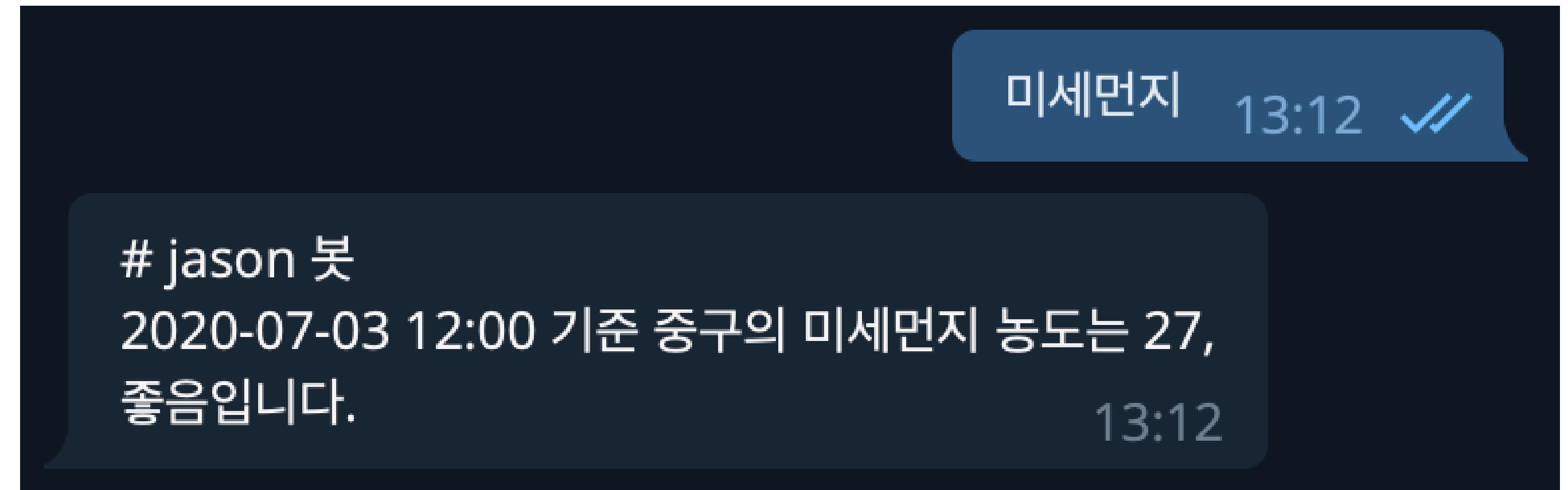
아닌데 만약에 (미세먼지 농도가 80이하 30초과면):

출력하자('보통')

아니면:

출력하자('좋음')

실습 4



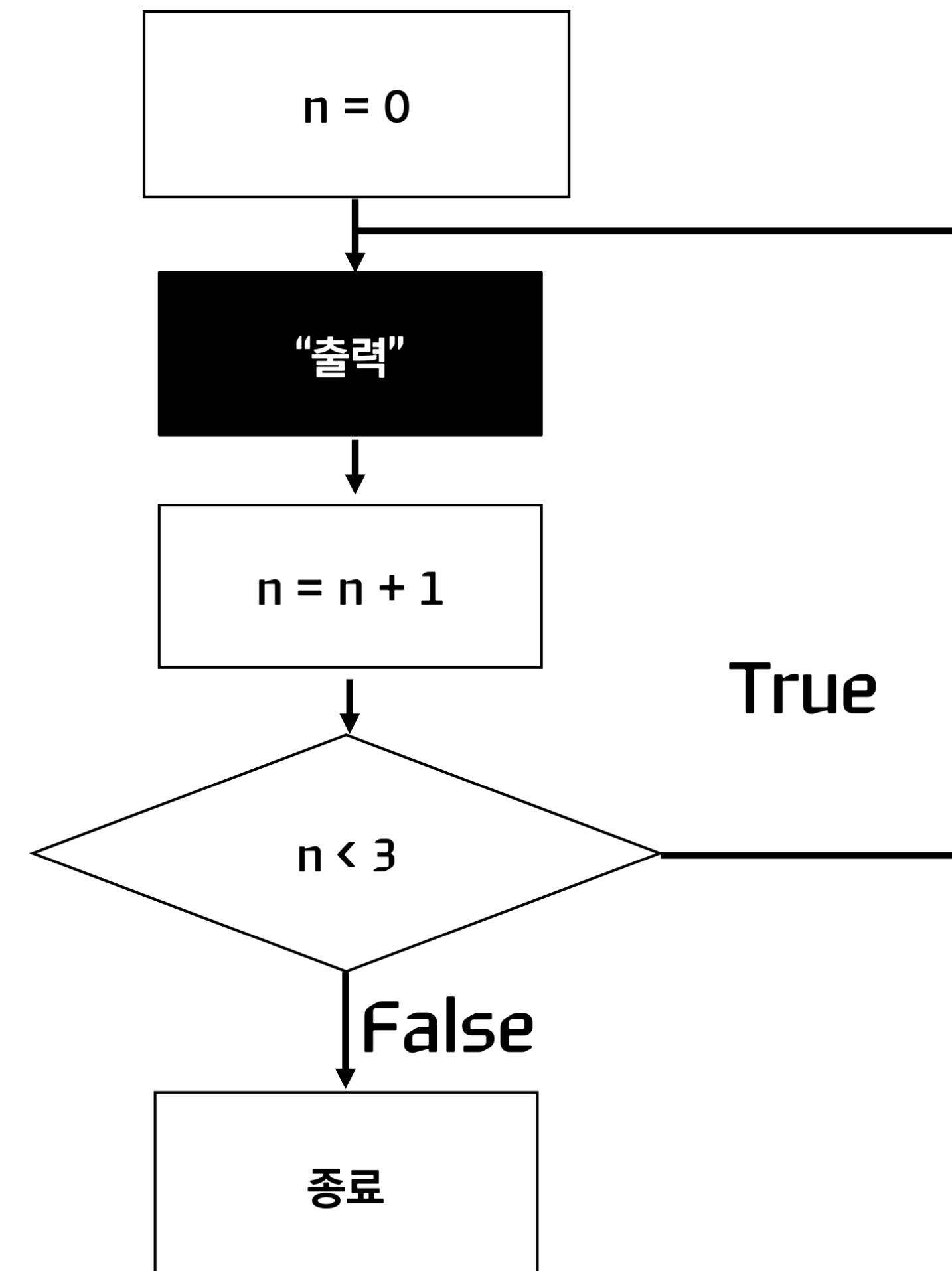
- 조건문 활용하기 (dust.py)
 - 미세먼지 코드 완성하기

3. 반복

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

반복

while



반복
while

while 에 해당하는 조건일 동안 계속 반복.

```
n = 0
while n < 3:
    print('출력')
    n = n+1
```

?

반복

while

while 에 해당하는 조건일 동안 계속 반복.

```
n = 0
while n < 3:
    print('출력')
    n = n + 1
```

3 < 3

False

출력
출력
출력

반복

while

while 에 해당하는 조건일 동안 계속 반복.

```
dust = [59, 24, 102, 45, 64]
```

```
n = 0
while n < 3:      3 < 3    False
    print(dust[n])
    n = n + 1
```

```
59
24
102
```

반복
for

for i in List:
 print('정해진 범위 안에서 계속')

반복
for
(리스트)

정해진 박스 내에서의 반복 시 사용
‘가지고 있는 모든 것을 꺼낸다’



```
dust = [59, 24, 102]
for i in dust: i = 102
    print(i)      print(102)
```

59 24 102

while 조건:
조건이 True 인 동안
반복적으로 실행 되기에
종료조건이 반드시 필요

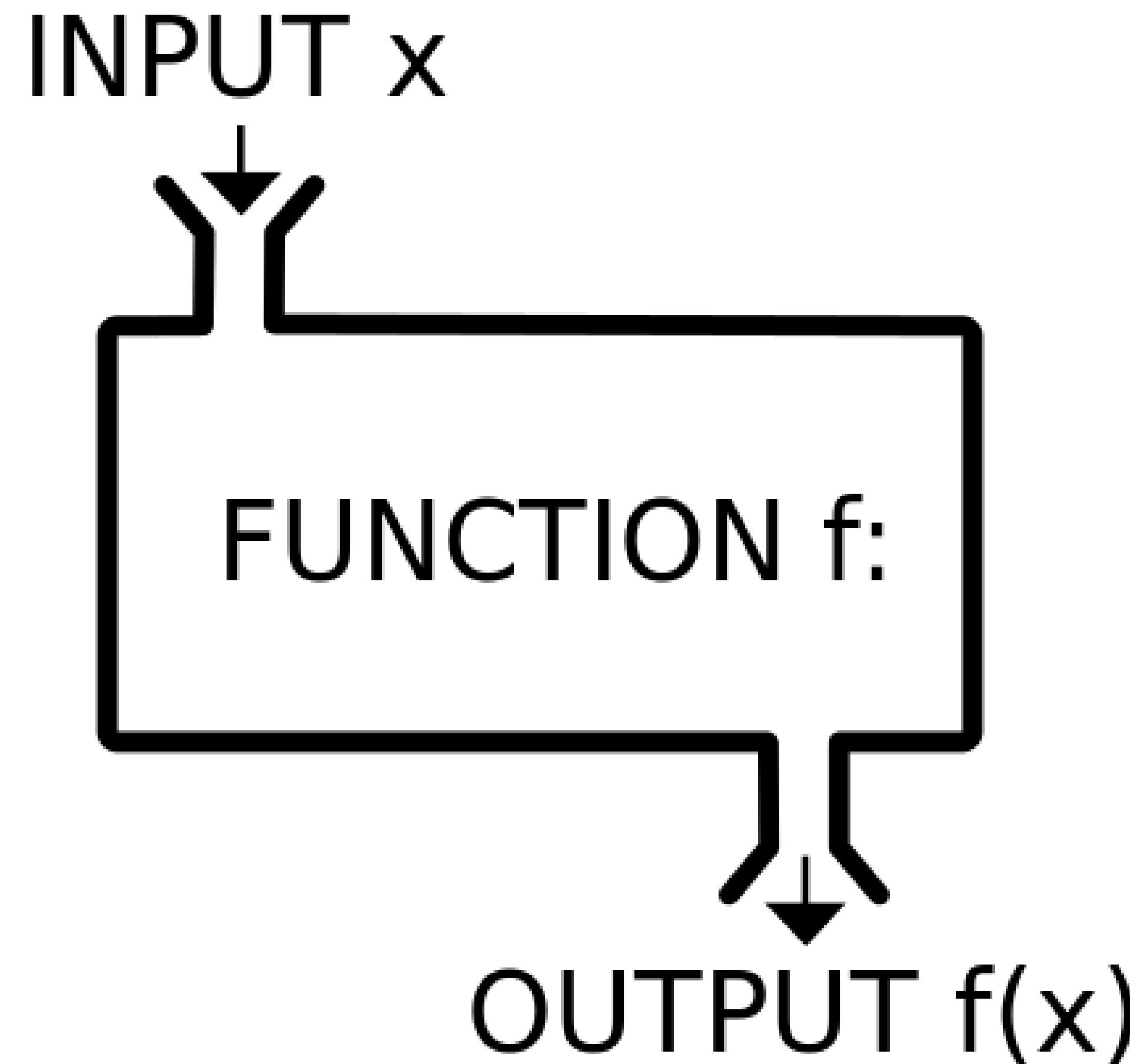
VS

for i in dust:
정해진 범위를 반복하기에
종료조건이 필요 없음

실습 5

- 반복문 활용하기 (`hello.py` 수정)
 - 여러번 인사하기

함수
Function



특정한 용도의 동작하는 코드를 한 곳에 모아 놓은 것

Excel 함수(function)

- 1) sum()
- 2) average()
- 3) count()

Python 함수

- 1) Built-in Functions (내장함수)
- 2) Non-built-in functions

Python 내장함수

- 1) `print('hi')`
- 2) `abs(-3)` => 3
- 3) `len('hi')` => 2

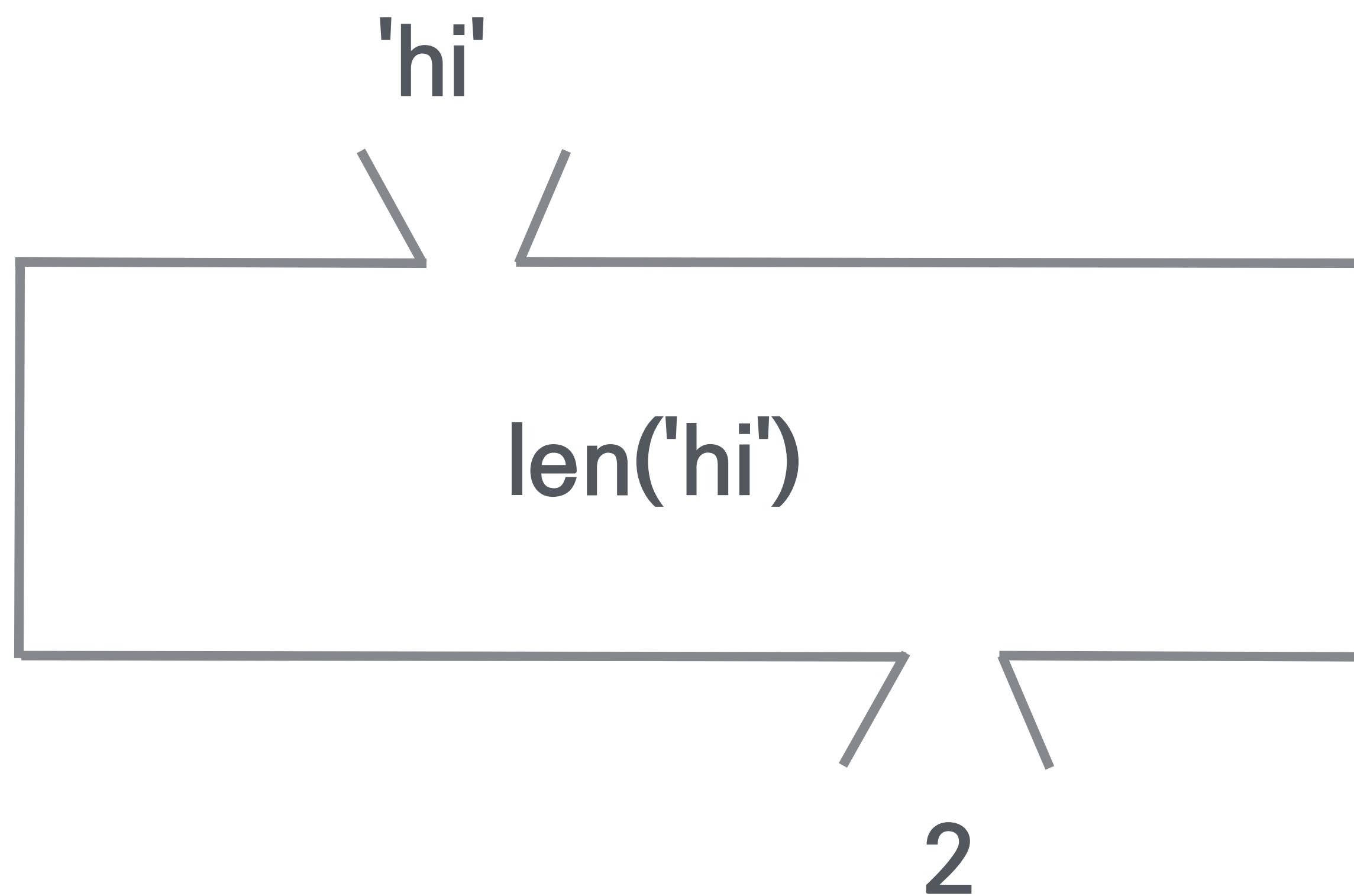
2. 내장 함수

파이썬 인터프리터에는 항상 사용할 수 있는 많은 함수와 형이 내장되어 있습니다. 여기에서 알파벳 순으로 나열합니다.

내장 함수				
<code>abs()</code>	<code>dict()</code>	<code>help()</code>	<code>min()</code>	<code>setattr()</code>
<code>all()</code>	<code>dir()</code>	<code>hex()</code>	<code>next()</code>	<code>slice()</code>
<code>any()</code>	<code>divmod()</code>	<code>id()</code>	<code>object()</code>	<code>sorted()</code>
<code>ascii()</code>	<code>enumerate()</code>	<code>input()</code>	<code>oct()</code>	<code>staticmethod()</code>
<code>bin()</code>	<code>eval()</code>	<code>int()</code>	<code>open()</code>	<code>str()</code>
<code>bool()</code>	<code>exec()</code>	<code>isinstance()</code>	<code>ord()</code>	<code>sum()</code>
<code>bytearray()</code>	<code>filter()</code>	<code>issubclass()</code>	<code>pow()</code>	<code>super()</code>
<code>bytes()</code>	<code>float()</code>	<code>iter()</code>	<code>print()</code>	<code>tuple()</code>
<code>callable()</code>	<code>format()</code>	<code>len()</code>	<code>property()</code>	<code>type()</code>
<code>chr()</code>	<code>frozenset()</code>	<code>list()</code>	<code>range()</code>	<code>vars()</code>
<code>classmethod()</code>	<code>getattr()</code>	<code>locals()</code>	<code>repr()</code>	<code>zip()</code>
<code>compile()</code>	<code>globals()</code>	<code>map()</code>	<code>reversed()</code>	<code>__import__()</code>
<code>complex()</code>	<code>hasattr()</code>	<code>max()</code>	<code>round()</code>	
<code>delattr()</code>	<code>hash()</code>	<code>memoryview()</code>	<code>set()</code>	

내장함수 예시

len()



Python Module

- random

- 1) 오늘 점심 추천
- 2) 로또 번호 추첨

모듈 활용

1. 함수가 포함된 코드를 불러온다. (`import`)
2. 함수를 사용한다.

```
import random
```

- 1) random.choice()
- 2) random.sample()

random.choice(리스트)

리스트에서 임의로(randomly) 하나의 요소를 선택(choice)

ex) random.choice(menu)

예시 random.choice()

```
menu = ['롯데리아', '김밥천국', '홍콩반점']
```

```
random.choice(menu)
```

홍콩반점

예시

random.choice()

1. 함수가 포함된 코드를 불러온다.
`import random`
2. 점심메뉴 리스트를 만든다.
`menu = ['롯데리아', '김밥천국', '홍콩반점']`
3. 함수를 통해 얻은 결과 값을 저장한다.
`choice = random.choice(menu)`
4. 출력해본다.
`print(choice)`

실습 6

- random 모듈 활용하기 (lunch.py)
 - 점심메뉴 코드 업그레이드

random.sample(리스트, 개수)

리스트에서 특정 수의 요소를
임의적으로(randomly) 비복원추출(sample)

ex) random.sample(numbers, 6)

예시 random.sample()

numbers = [1, 2, 3, 4, ..., 44, 45]

random.sample(numbers, 6)

[13, 22, 6, 37, 1, 45]

실습 7

- random 활용하기 (lotto.py)
 - 로또 추첨

〈작성방법〉

1. random 친구를 불러 온다.
2. 1부터 45 까지의 숫자를 저장한다.
3. random 모듈의 sample 함수를 통해 6개의 숫자를 무작위로 뽑아 새로운 박스에 저장한다.
4. 새로운 박스를 출력한다.

챗봇으로 옮겨보자

Summary

프로그래밍 언어 : 3형식

1. 저장
2. 조건
3. 반복

python 문법 : 저장

1. 저장
2. 조건
3. 반복

저장	=
박스 내용	'글자', 숫자, True/False
박스 종류	변수, [리스트], {딕셔너리}

python 문법 : 조건

1. 저장
2. 조건
3. 반복

`if` 비교문:

실행문

`elif` 비교문:

실행문

`else`:

실행문

python 문법 : 반복

1. 저장
2. 조건
3. 반복

`while(True):`
조건 True 인 동안 반복
종료조건이 필요함.

`for i in dust:`

박스에 담긴 만큼 반복
종료조건 필요없음.

session 2.

Python 문법

맛보기

```
# 숫자
number = 3
print(type(number))

# 문자
string = '문자열'
print(type(string))

# boolean
boolean = True
print(type(boolean))

# 형변환
string_number = '3'
print(string_number + 5)
print(int(string_number) + 5)
```

Data type

```
# f-string / string interpolation
name = '홍길동'
print(f'{name}입니다. 반갑습니다.')
```

f-string

```
# 리스트 선언  
my_list = ['java', 'django']  
  
# 리스트 원소 접근  
print(my_list[0])  
  
# 리스트 원소 변경  
my_list[0] = 'python'  
  
# 리스트 원소 접근  
print(my_list[0])  
  
# 리스트 길이  
print(len(my_list))
```

List

```
# 딕셔너리 선언
my_home = {
    'location': 'seoul',
    'area-code': '02',
}

# 딕셔너리 원소 접근
print(my_home['location'])
print(my_home['name'])
print(my_home.get('location'))
print(my_home.get('name'))

# 딕셔너리 원소 변경
my_home['location'] = 'gumi'
print(my_home['location'])
```

Dictionary

```
# 산술 연산자  
print(3 + 5)  
print(5 - 3)  
print(100 * 5)  
print(100 / 3)  
print(100 // 3)  
print(100 % 3)  
print(2 ** 5)
```

```
# 비교 연산자  
print(5 == 5)  
print(3 == '3')  
print(3 != 5)  
print(3 >= 3)  
print(5 < 4)
```

기초 연산자

실습

- 리스트
- 딕셔너리
- 조건문
- 반복문

session 3.

프로그래밍을 통해
정보 수집하기

session 3.

웹 페이지 크롤링

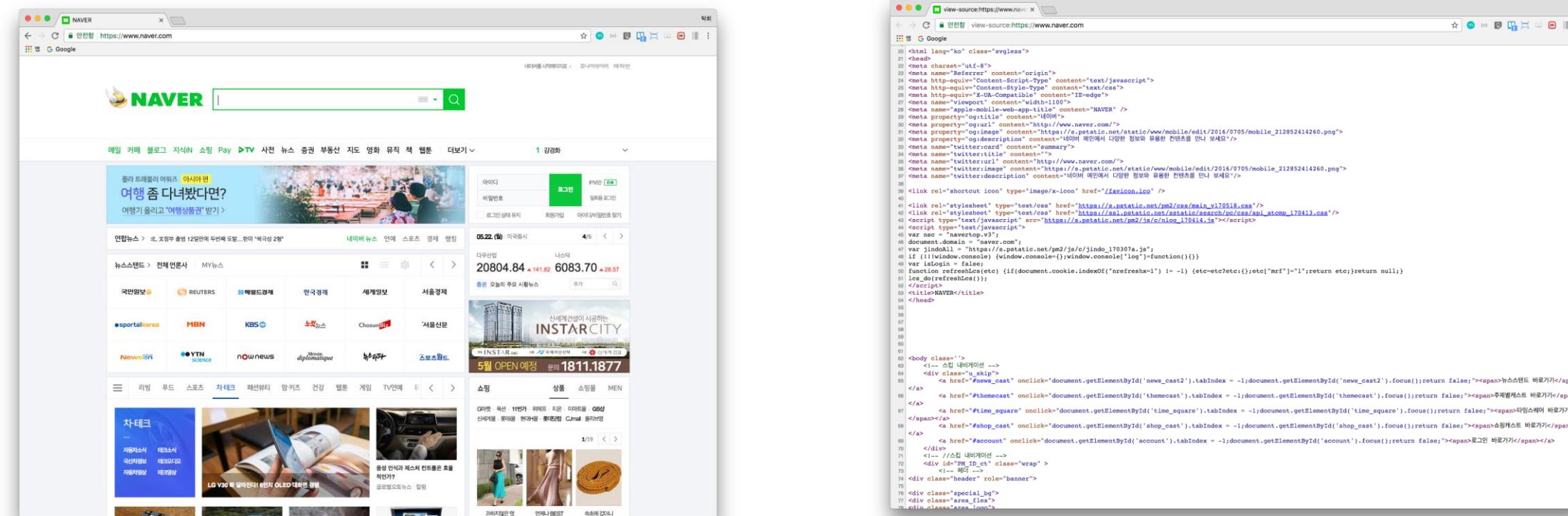
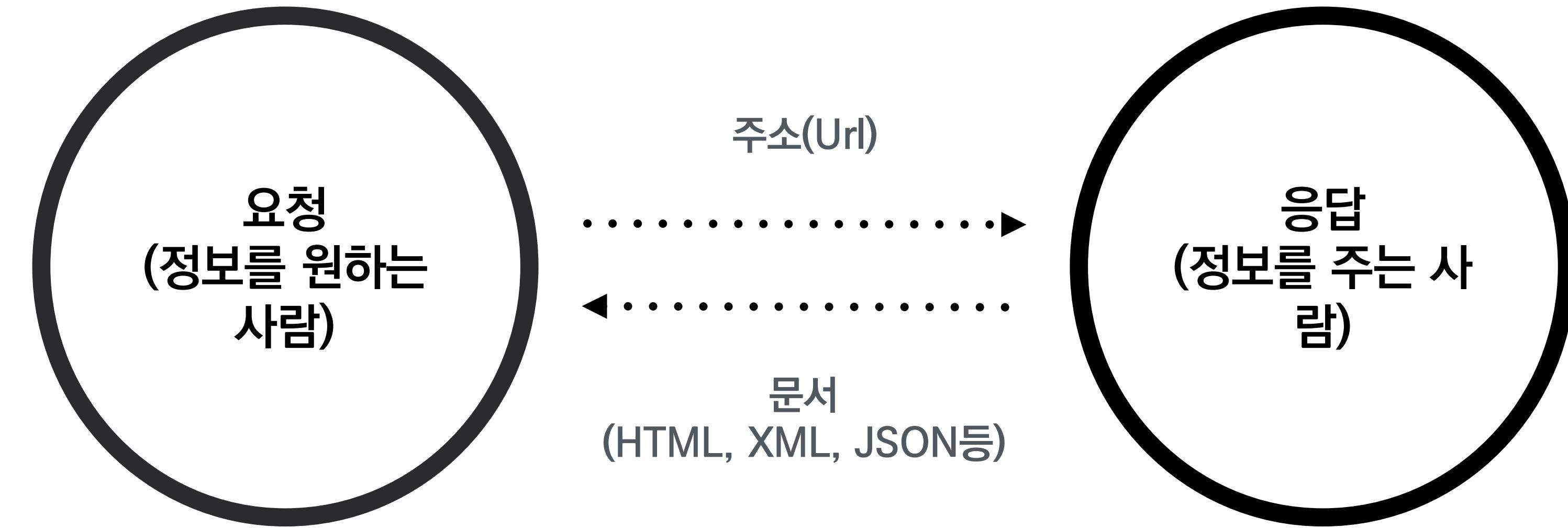
웹 크롤링이란?

Web crawling

조직적, 자동화된 방법으로 웹을 탐색하는 것.
검색 엔진과 같은 여러 사이트에서는 데이터의 최신 상태
유지를 위해 웹 크롤링을 함 (위키백과)

1. Naver에 들어간다.
2. 정보를 검색한다.
3. 원하는 정보만 복사한다.
4. 내 컴퓨터에 저장한다.

[‘<https://www.naver.com>’](https://www.naver.com)



네이버 통합검색

search.naver.com/search.naver?sm=tab_hty.top&where=nexearch&query=삼성전자

N 삼성전자

로그인

통합 뉴스 실시간검색 지도 이미지 VIEW 지식iN 쇼핑 지식백과 ⋮ 검색옵션 ⋮

공유

S www.samsung.com

삼성전자

나의 정보 · 매장 찾기 · 서비스센터 찾기 · 뉴스룸 · 삼성닷컴 FAQ · 기획전
갤럭시, TV, PC, 가전, 액세서리, 소모품 제품정보, 온라인 스토어 제품판매, 고객지원 정보 제공.

블로그 네이버TV 블로그 트위터 유튜브 인스타그램 페이스북

연관 검색어

신고 ×

삼성전자우 현대자동차 lg전자 삼성
sk하이닉스 카카오 코스피 엘지전자
현대차 셀트리온

https://search.naver.com/search.naver?sm=tab_hty.top&where=nexearch&query=삼성전자

크롬이 주소에 대한 요청을 보내고,
그 응답 결과를 웹 화면으로 예쁘게 보여준다.

파이썬이 주소에 대한 요청을 보낸다.

```
$ pip install requests
```

import requests

- 1) requests.get(주소)
- 2) requests.get(주소).text
- 3) requests.get(주소).status_code

requests.get('주소')

‘주소’에 요청(request) 보내서, 정보를 받아줘(get)

ex) requests.get('http://naver.com')

```
requests.get('주소').text
```

‘주소’에 요청(request) 보내서, 정보 받아서(get),
글(text)만 뽑아줘

ex) requests.get(<http://naver.com>).text

```
requests.get('주소').status_code
```

‘주소’에 요청(request) 보내서, 정보 받아서(get),

상태(status_code)만 뽑아줘

ex) `requests.get('http://naver.com').status_code`

정보 스크랩

1. 정보가 있는 사이트 URL을 확인한다.
2. URL로 요청을 보낸다.
3. 응답 결과에서 원하는 정보를 찾는다.

정보 스크랩

1단계

- 원하는 정보가 있는 주소로 요청을 보내,
응답을 저장한다.

```
import requests
```

```
response = requests.get(url).text
```

- 정보를 출력하여 확인한다.

```
print(response)
```

파이썬이
그 응답 결과를 예쁘게 관리한다.

```
$ pip install beautifulsoup4
```

```
from bs4 import BeautifulSoup
```

- 1) BeautifulSoup(문서)
- 2) BeautifulSoup.select(경로)
- 3) BeautifulSoup.select_one(경로)

BeautifulSoup(문서)

받은 문서를 예쁘게(보기 좋게, 검색하기 좋게) 만들어줘

ex) data = BeautifulSoup(response)

.select(selector)

문서 안에 있는 특정 내용을 뽑아줘(select)

ex) data.select(response)

.select_one(selector)

문서 안에 있는 특정 내용을 하나만 뽑아줘(select_one)

ex) data.select_one('정보경로')

파이썬이 주소에 대한 요청을 보내고(requests),
그 응답 결과를 예쁘게 관리한다. (bs4)

원하는 값(KOSPI)은 어떻게 찾을까?

HTML 문서에서 원하는 위치를 선택하자.
selector!

국내증시 : 네이버 금융

finance.naver.com/sise/

NAVER 금융

금융 홈 국내증시 해외증시 시장지표 펀드 투자전략 뉴스 MY금융 추천종목

[금융감독원] 전자공시시스템 [한국거래소] 공매도 종합 포털 [한국은행] 100대 통계지표 [네이버뉴스] 글로벌 경제 리포트

국내증시

주요시세정보
코스피 | 코스닥 | 선물
코스피200 | 코넥스

시가총액 | 배당
업종 | 테마 | 그룹사
ETF

상승 | 보합 | 하락
상한가 | 하한가
급등 | 급락

거래상위 | 급증 | 급감

투자자별매매동향
외국인매매 | 기관매매
프로그램매매동향
증시자금동향

신규상장
외국인보유
장외시세

IPO

투자자보호
관리종목
거래정지종목
시장경보종목

조건검색
골든크로스 | 캡상승

증목명·펀드명·환율명·원자재명 입력

통합검색

로그인

인기 검색 종목 더보기
1. SK하이닉스 84,700 ▲
2. 신라젠 60,200 ▲
3. 셀트리온 178,300 ▲
4. 내츄럴엔도.. 28,250 ▲
5. 한국항공우.. 56,700 ▼
6. 코오롱글로.. 29,100 ▲
7. LG디스플레.. 29,700 ▼
8. 카카오 151,000 ▲
9. NAVER 799,000 ▼
10. 텔콘 7,230 ▲

코스피 2,490.05 ▲ 0.51 +0.02% 코스닥 675.86 ▲ 2.91 +0.43% 코스피200 329.45 ▲ 0.56 +0.17%

코스피 개인 +2,042억 외국인 +3,156억 기관 -6,174억

설시간 2017.10.23 장마감

등락 종목 ↑ 323 ▲ 74 ↓ 469 0 프로그램 매매동향 차익 -5,803억 비차익 +179억 전체 -5,623억

주요 해외지수

다우산업 23,328.63 ▲
나스닥 6,629.05 ▲
홍콩H 11,491.07 ▼
상해종합 3,380.70 ▲
니케이225 21,696.65 ▲

최근조회종목 MY STOCK

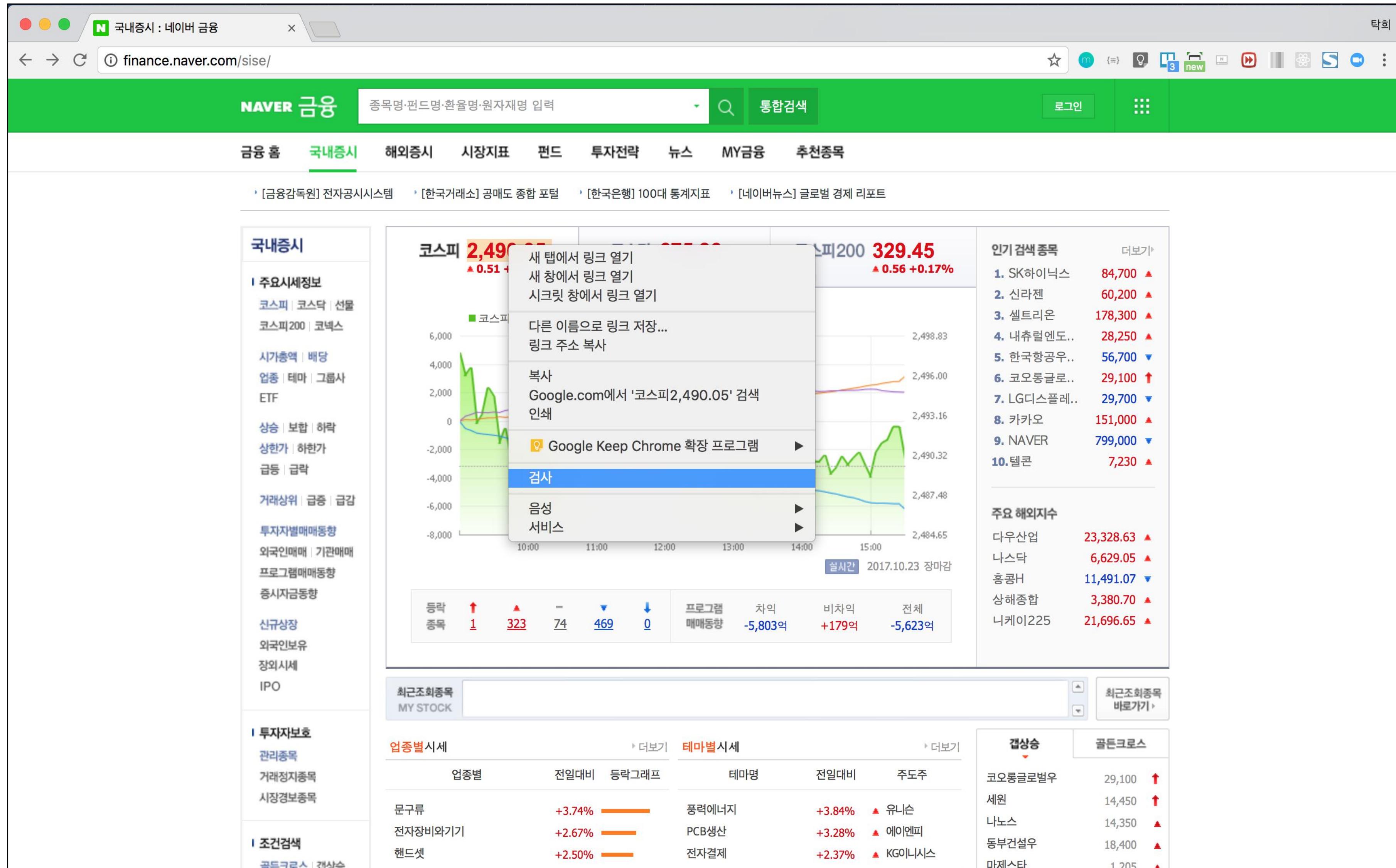
업종별 시세 더보기 테마별 시세 더보기

업종별	전일대비	등락그래프	테마명	전일대비	주도주
문구류	+3.74%		풍력에너지	+3.84%	▲ 유니슨
전자장비와기기	+2.67%		PCB생산	+3.28%	▲ 에이엔피
핸드셋	+2.50%		전자결제	+2.37%	▲ KG이니시스

캡상승
골든크로스

	캡상승	골든크로스
코오롱글로벌우	29,100 ▲	
세원	14,450 ▲	
나노스	14,350 ▲	
동부건설우	18,400 ▲	
마제스타	1,205 ▲	

원하는 정보를 선택 후 우클릭, ‘검사’ 선택



개발자 도구(우측)에서 해당 정보 선택

국내증시 : 네이버 금융

finance.naver.com/sise/

NAVER 금융

금융 홈 국내증시 해외증시 시장지표 펀드 투자전략 뉴스 MY금융 추천종목

[금융감독원] 전자공시시스템 [한국거래소] 공매도 종합 포털 [한국은행] 100대 통계지표 [네이버뉴스] 글로벌 경제 리포트

국내증시

| 주요시세정보
코스피 | 코스닥 | 선물
코스피200 | 코넥스

시가총액 | 배당
업종 | 테마 | 그룹사
ETF

상승 | 보합 | 하락
상한가 | 하한가
급등 | 급락

거래상위 | 급증 | 급감

투자자별매매동향
외국인매매 | 기관매매
프로그램매매동향
증시자금동향

신규상장
외국인보유
장외시세
IPO

| 투자보호
관리종목
거래정지종목
시장경보종목

| 조건검색
골든크로스 | 캡상승

코스피 **2,490.05**
▲ 0.51 +0.02%
코스닥 **675.86**
▲ 2.91 +0.43%
코스피200 **329.45**
▲ 0.56 +0.17%

인기 검색 종목 더보기
1. SK하이닉스 84,700 ▲
2. 신라젠 60,200 ▲
3. 셀트리온 178,300 ▲
4. 내츄럴엔도.. 28,250 ▲
5. 한국항공우.. 56,700 ▼
6. 코오롱글로.. 29,100 ▲
7. LG디스플레.. 29,700 ▼
8. 카카오 151,000 ▲
9. NAVER 799,000 ▼
10. 텔콘 7,230 ▲

주요 해외지수
다우산업 23,328.63 ▲
나스닥 6,629.05 ▲
홍콩H 11,491.07 ▼
상해종합 3,380.70 ▲
니케이225 21,696.65 ▲

최근조회종목
MY STOCK

업종별시세 더보기 테마별시세 더보기
업종별 전일대비 등락그래프 테마명 전일대비 주도주
문구류 +3.74% 풍력에너지 +3.84% ▲ 유니슨
전자장비와기기 +2.67% PCB생산 +3.28% ▲ 에이엔피
핸드셋 +2.50% 전자결제 +2.37% ▲ KG이니시스

캡상승 골든크로스
코오롱글로벌우 29,100 ▲
세원 14,450 ▲
나노스 14,350 ▲
동부건설우 18,400 ▲
마제스타 1,205 ▲

Elements Audits Console Sources Network

Elements Audit Console Sources Network

Copy outerHTML Copy selector Copy XPath Cut element Copy element Paste element

Styles Event Filter element.style { position: absolute; top: 12px; left: 83px; display: inline-block; height: 19px; font-family: arial; font-weight: bold; font-size: 21px; color: #e00400; }

Properties

Break on... Scroll into view

Console What's New

우클릭 후, 복사(copy) → CSS selector 복사(copy selector)

선택자는
HTML의 특정 부분을 가져오기 위해
사용한 것.

선택자는
HTML의 특정 부분을 가져오기 위해
사용한 것.
그리고 BeautifulSoup이 구조화 해준 것!

크롤링 정리하기.

1. Naver에 들어간다.
2. 정보를 검색한다.
3. 원하는 정보만 복사한다.
4. 내 컴퓨터에 저장한다.

정보 스크랩

1단계

- 원하는 정보가 있는 주소로 요청을 보내,
응답을 저장한다.

```
import requests
```

```
response = requests.get(url).text
```

- 정보를 출력하여 확인한다.

```
print(response)
```

정보 스크랩

2단계

1. 정보를 조작하기 편하게 바꾸고,
`from bs4 import BeautifulSoup`
`data = BeautifulSoup(response)`
2. 바꾼 정보 중 원하는 것만 뽑아서,
`kospi = data.select_one('selector 경로')`
3. 출력한다.
`print(kospi.text)`

```
import requests
from bs4 import BeautifulSoup

url = 'https://finance.naver.com/sise/'
response = requests.get(url).text
data = BeautifulSoup(response, 'html.parser')
kospi = data.select_one('#KOSPI_now')
result = kospi.text
```

naver_kospi.py

실습. 환율 정보 수집하기 (naver_exchange.py)

<https://finance.naver.com/marketindex>

The screenshot shows the Naver Finance Market Index page. At the top, there's a search bar and a navigation menu with tabs like 'NAVER 금융', '시장지표' (selected), '펀드', '리서치', '뉴스', and 'MY'. Below the menu, there are two main sections: '환전 고시 환율' and '국제 시장 환율'. The '환전 고시 환율' section displays rates for USD, JPY, EUR, and CNY with small line charts. The '국제 시장 환율' section displays rates for oil prices (WTI, Brent) and various currencies (JPY, EUR, CNY) with larger line charts. At the bottom, there's a '주요뉴스' (Top News) section and a '국내 시장 금리' (Domestic Market Interest Rates) section.