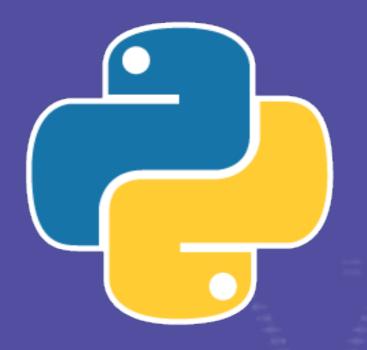
핵심파이썬 기초프로그래밍

파이썬에서 자료를 담는 여러가지 방식



이호준 선생님

리스트활용

1) list.append(d)

자료 d를 리스트 마지막 원소 뒤에 추가 오직 한 개의 자료만 넣을 수 있다

```
a = []
b = ['a', 'b', 'c']
a.append(10)
b.append('d')
print(a, b) # [10] ['a', 'b', 'c', 'd']
```

2) list.insert(i, d)

인덱스 i에 자료 d를 추가 오직 한 개의 자료만 넣을 수 있다

```
c = [1, 2, 4, 5]
c.insert(2, 3)
print(c) # [1, 2, 3, 4, 5]
```

3) list.remove(d)

처음 나오는 자료 d를 제거

```
d = [3, 1, 2, 3]
d.remove(3)
print(d) # [1, 2, 3]
```

4) list.sort()

리스트를 정렬

숫자형은 오름차순, 문자열은 사전순

```
e = [6, 2, 4, 1]
f = ['carrot', 'apple', 'banana']
e.sort()
f.sort()
print(e, f) # [1, 2, 4, 6] ['apple',
'banana', 'carrot]
```

시퀀스 자료형

시퀀스 자료형

순서가 있는 자료형! 리스트, 문자열 등이 이에 속함

```
a = "Once" #문자열
b = ['T', 'W', 'I', 'C', 'E'] #리스트
c = (1, 2, 3, 4, 5) #튜플
```

- 1) 원소 간의 순서가 존재
- → 인덱싱/슬라이싱이 가능하다

```
a = "once"
b = ['t', 'w', 'i', 'c', 'e']
print(a[1]) # n
print(b[2:4]) # ['i', 'c']
```

인덱싱/슬라이싱을 할 때 음수를 넣거나, 자리를 비우는 것이 가능!

```
a = "once"
b = ['t', 'w', 'i', 'c', 'e']
print(a[-1]) # e | 뒤에서 1번째 원소
print(b[:3]) # ['t', 'w', 'i'] | 처음~3번째 슬라이싱
```

2) 멤버 조회

in 연산자로 시퀀스 안에 원소가 있는지 확인 가능

```
a = "once"
b = ['t', 'w', 'i', 'c', 'e']
print('o' in a) # True
print('b' in c) # False
```

3) 길이 확인

len 연산자로 시퀀스 안에 원소가 있는지 확인 가능

```
a = "once"
b = ['t', 'w', 'i', 'c', 'e']
print(len(a)) # 4
print(len(b)) # 5
```

4) 연결 연산

+ 연산자로 같은 시퀀스 두개를 이어 붙일 수 있다

```
c = ['t', 'w', 'i'] + ['c', 'e']
print(c) #['t', 'w', 'i', 'c', 'e']
```

5) 반복 연산

* 연산자로 시퀀스를 반복할 수 있다

```
d = "shy" * 3
print(d) #shyshyshy
```

Dictionary(딕셔너리)

Dictionary?

Dictionary → 사전 짝꿍이 있는 자료형!

dictionary

noun ● UK (1) /'dɪk·ʃən·ər·i/ US (1) /'dɪk·ʃəˌner·i/ PLURAL dictionaries

a book that contains a list of words in alphabetical order with their meanings explained and sometimes written in another language

사전

Use your dictionaries to look up any words you don't understand.

성	이름
olnilol	
이메일	
비밀번호	
비밀번호 확인	

Dictionary(딕셔너리)

{}-중괄호로 묶어서 표현

```
dict_zero = {}
person = {'name':'Michael', 'age':10}
```

Dictionary(딕셔너리)

짝꿍이 있는 자료형

{key: value}의 형식: key를 알면 value를 알 수 있음

```
dict_zero = {}
person = {'name':'Michael', 'age':10}
```

Key

열쇠처럼 자료를 꺼낼 수 있는 도구

```
dict_zero = {}
person = { 'name': 'Michael', 'age':10}
```

Value

Dictionary에서 Key로 꺼낸 자료

```
dict_zero = {}
person = { 'name': 'Michael', 'age':10}
```

Dictionary[key]

Dictionary에서 자료를 꺼내기

```
person = {'name':'Michael', 'age':10}
print(person['name']) # Michael
print(person['age']) # 10
     Dictionary key
```

Dictionary[key]

Dictionary에서 자료를 추가하기

```
person = {'name':'Michael', 'age':10}
person['hometown'] = Seoul
Dictionary key
                    value
```

del

del 함수로 Dictionary의 원소 삭제

```
person = {'name':'Michael', 'age':10}
del person['age']
   Dictionary key
print(person) # {'name':'Michael'}
```

Dictionary의 특징

Key는 변할 수 없는 자료형

→ 리스트는 안되고, 튜플은 된다!

```
datas = {[1, 2, 3]: 'Alphabet'} # Error

datas = {(1, 2, 3): 'Number'} # OK
```

/* elice */

문의및연락처

academy.elice.io contact@elice.io facebook.com/elice.io medium.com/elice