

데이터 구조

- 리스트, 튜플, 딕셔너리, 세트 -

7장. 리스트와 튜플

- 파이썬기초 -

리스트와 튜플선언

시퀀스 자료형 - 연속된 여러값들을 한 변수에 저장

리스트 = [값, 값, 값 ...]

수정, 추가 가능

```
>>> a = [1, 2, 3, 4, 5, 6, 7]
>>> a = [1, "apple", 3.14, False]
>>> type(a)
<class 'list'>
>>> a = [] #빈값생성
>>> a = list()
```

튜플 = (값, 값, 값)

수정, 추가 불가능

```
>>> b = (1, 2, 3, 4, 5, 6, 7)
>>> b = (1, "apple", 3.14, False)
>>> type(b)
<class 'tuple'>
>>> b = () #빈값생성
>>> b = tuple()
```

Range 함수

연속된 숫자를 생성하는 함수

range(끝)

```
>>> a = list(range(10))  
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
>>> a = list(range(1, 11))  
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
>>> a = tuple(range(10))  
(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
```

range(시작, 끝, 증가폭)

```
>>> a = list(range(-10, 10, 2))  
[-10, -8, -6, -4, -2, 0, 2, 4, 6, 8]
```

```
>>> a = list(range(5, 0, -1))  
[5, 4, 3, 2, 1]
```

리스트, 튜플 형변환

list(튜플)

```
>>> a = (1, 2, 3, 4, 5)
>>> list(a)
[1, 2, 3, 4, 5]
```

tuple(리스트)

```
>>> a = [1, 2, 3, 4, 5]
>>> tuple(a)
(1, 2, 3, 4, 5)
```

인덱스 접근 #1

기본접근 및 변경

```
>>> a = [1, 2, 3, 4, 5]
```

```
>>> a[2]
```

```
3
```

```
>>> a[-1]
```

```
5
```

```
>>> a[3] = 8
```

```
>>> a
```

```
[1, 2, 3, 8, 5]
```

슬라이스

```
>>> a = [1, 2, 3, 4, 5]
```

```
>>> a[0:4]
```

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

```
>>> a[3:5]
```

```
[4, 5]
```

```
>>> a[1:-2]
```

```
[2, 3]
```

인덱스 접근 #2

증가폭 변경

```
>>> a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> a[2: 8: 2]
[3, 5, 7]
```

끝 인덱스까지 가져오기

```
>>> a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> a[:3]
[1, 2, 3]
>>> a[3:]
[4, 5, 6, 7, 8, 9, 10]
```

슬라이스 요소할당

```
>>> a = [1, 2, 3, 4, 5]
>>> a[1:4] = ["a", "b", "c"]
>>> a
[1, a, b, c, 5]
>>> a[1:4] = ["d", "e"]
>>> a
[1, d, e, 5]
```

슬라이스 삭제

```
>>> a = [1, 2, 3, 4, 5]
>>> del a[1:4]
[1, 5]
```

리스트와 튜플 기능들 #1

특정값 있는지 확인 in

```
>>> a = [1, 2, 3, 4, 5]
```

```
>>> 1 in a
```

```
True
```

```
>>> 6 in a
```

```
False
```

연결하기 +

```
>>> a = [1, 2, 3, 4, 5]
```

```
>>> b = [6, 7, 8, 9]
```

```
>>> a + b
```

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

반복하기 *

```
>>> a = [1, 2, 3]
```

```
>>> a * 3
```

```
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

요소개수 구하기 len()

```
>>> a = [1, 2, 3]
```

```
>>> len(a)
```

```
3
```


리스트 기능들 #1

요수 추가하기 `append(값)`, `extend(값)`

```
>>> a = [1, 2, 3, 4, 5]
>>> a.append(6)
[1, 2, 3, 4, 5, 6]
>>> a.extend([7, 8])  = a + [7, 8]
[1, 2, 3, 4, 5, 6, 7, 8]
```

리스트 요소삭제 `pop(인덱스)`

```
>>> a = [1, 2, 3]
>>> a.pop(0)
[2, 3]
>>> a.pop()
[2]
```

특정 인덱스에 요소추가 `insert(인덱스, 값)`

```
>>> a = [1, 2, 3]
>>> a.insert(2, 100)
[1, 2, 100, 3]
```

리스트 특정값을 찾아삭제 `remove(값)`

```
>>> a = [100, 200, 300]
>>> a.remove(200)
[100, 300]
```

리스트 기능들 #2

특정값의 인덱스 구하기 index(값)

```
>>> a = [1, 2, 3, 4, 5]
>>> a.index(3)
2
```

특정값의 개수구하기 count(값)

```
>>> a = [1, 1, 2, 2, 2, 3, 3]
>>> a.count(2)
3
```

순서 뒤집기 reverse()

```
>>> a = [1, 2, 3]
>>> a.reverse()
[3, 2, 1]
```

정렬하기(오름차순) sort(), sort(reverse=False)

```
>>> a = [3, 2, 1, 4]
>>> a.sort()
[1, 2, 3, 4]
```

정렬하기(내림차순) sort(reverse=True)

```
>>> a = [3, 2, 1, 4]
>>> a.sort(reverse=True)
[4, 3, 2, 1]
```

튜플 기능

두 변수 간 값 바꾸기

```
>>> a = 1
```

```
>>> b = 2
```

```
>>> temp = a
```

스왑한다고 말함

```
>>> a = b
```

```
>>> b = temp
```

// a=2, b=1

튜플 사용하기

```
>>> a, b = b, a
```

// a=2, b=1

다차원 리스트와 튜플

리스트 = [[값, 값], [값, 값] ...]

```
>>> a = [[1, 2], [3, 4], [5, 6]]
```

튜플 = ((값, 값), (값, 값) ...)

```
>>> a = ((1, 2), (3, 4), (5, 6))
```

접근시 리스트[세로인덱스][가로인덱스]

```
>>> a[0][1]
```

```
2
```

```
>>> a[2][0]
```

```
5
```

이차원 리스트 예

H22

	A	B	C	D	E	F	G	H	I	J	K	L
1	우리 도서관 4월 대출 현황							₩	₩	₩	₩	
2												
3												
4	도서번호	도서명	저자	이용형태	대출자	대출일자	대출횟수 (누적)	분류	반납일자			
5	1-A01	여울물 소리	황석영	도서관 방문	김민선	2013.4.2	12회	소설	2013.4.7			
6	2-B01	사람외전	이외수	차량 도서관	이영미	2013.4.2	24회	에세이	2013.4.2			
7	1-A32	7년후	기욤 뫼리	도서관 방문	강영수	2013.4.3	8회	소설	2013.4.8			
8	2-B33	풀이	김철규	도우미 대출	정민욱	2013.4.4	21회	에세이	2013.4.11			
9	3-C21	먼 곳	문태준	차량 도서관	송광호	2013.4.5	18회	시집	2013.4.5			
10	2-B22	감동의 습관	송정림	차량 도서관	김영자	2013.4.5	16회	에세이	2013.4.5			
11	1-A23	깊은 상처	노이하우스	도서관 방문	박대광	2013.4.6	7회	소설	2013.4.11			
12	1-A82	파이 이야기	마텔	도우미 대출	성민아	2013.4.7	13회	소설	2013.4.14			
13	도서관 방문의 평균 대출횟수			9	최대 대출횟수(누적)			24회				
14	차량 도서관 이용 횟수			3	도서명			여울물 소리	저자	황석영		

문제)
도서번호의 첫글자가 1이면 "소설" 2이면 "에세이" 3이면 "시집"

회원 정보

이름	연락처	나이
1 홍경수	0107773333	31

이름

연락처

나이

split, join 함수

split 함수

문자를 리스트로

```
>>> fruit = "사과,배,옥수수,당근"  
>>> fruit_list = fruit.split(",")  
>>> print(fruit_list)
```

`['사과', '배', '옥수수', '당근']`

join 함수

리스트를 문자로

```
>>> fruit_list = ['사과', '배', '옥수수', '당근']  
>>> fruit = "".join(fruit_list)  
>>> print(fruit)
```

`사과배옥수수당근`

```
>>> fruit = " ".join(fruit_list)  
>>> print(fruit)
```

`사과 배 옥수수 당근`

```
>>> fruit = ", ".join(fruit_list)  
>>> print(fruit)
```

`사과,배,옥수수,당근`

리스트로서 문자열

문자열은 각 문자의 리스트와 비슷하게 동작

```
>>> text = "Python is awesome"
>>> print(text[7])
>>> print(text[2: 8: 2])
>>> print(text[:3])
>>> print(text[3:])
```

```
>>> text[1] = "y"
>>> text[1:4] = ["a", "b", "c"]
>>> del text[1:4]
```



```
>>> text_list = list(text)
>>> text_list[1] = "y"
>>> text_list[1:4] = ["a", "b", "c"]
>>> del text_list[1:4]
```

실습01

range 함수를 이용하여 사용자가 입력한 수까지 2의 배수값을 넣은 리스트를 만들고 리스트의 맨 마지막에 사용자가 입력한 추가해주세요

실행 :

숫자를 입력해주세요 : 10

[2, 4, 6, 8, 10, 10]

숫자를 입력해주세요 : 20

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 20]

실습02

다음의 주민 등록 번호에서 성별과 나이(태어난 해 1살 기준)를 가져오세요

실행 :

주민등록번호를 입력하세요 : 051110-2063423

성별 : 여자

나이 : 17세

주민등록번호를 입력하세요 : 930326-1068551

성별 : 남자

나이 : 29세

8장. 딥서너리

- 파이썬기초 -

딕셔너리

키와 값으로 이루어진 값들을 저장하는 자료형

딕셔너리 = {키1: 값1, 키2: 값2}

score = {"name": "Tom", "math": 80, "english": 70}

```
>>> score = {"name": "tom", "math": 80, "english": 70}
```

```
>>> score["name"]
```

```
tom
```

```
>>> score["name"] = "michael"
```

```
>>> score["name"]
```

```
michael
```

```
>>> type(score)
```

```
dict
```

dict

딕셔너리 = dict(키1=값1, 키2=값2)

score = dict(name="Tom", math=80, english=70)

```
>>> score = dict(name="tom", math=80, english=70)
```

```
>>> score["name"]
```

```
tom
```

```
>>> score = dict() #비어있는 딕셔너리
```

```
>>> score = {} #비어있는 딕셔너리
```


딕셔너리 기능 #1

키가 있는지 확인

```
>>> score = {"name": "tom", "math": 80, "english": 70}
>>> "math" in score
True
>>> "age" in score
False
```

키의개수

```
>>> len(score)
3
```

키-값 쌍 추가하기.setdefault(키, 값)

```
>>> score.setdefault("age", 20)    score["age"] = 20
{"name": "tom", "math": 80, "english": 70, "age": 20}
```

키-값 수정하기 update({키: 값})

```
>>> score.update({"math": 90})
{"name": "tom", "math": 90, "english": 70, "age": 20}
```

딕셔너리 기능 #2

키로 딕셔너리 항목삭제 pop(키,기본값)

```
>>> score = {"name": "tom", "math": 80, "english": 70}
>>> score.pop("name") #삭제된 키의 값 반환
tom
```

```
>>> score
{"math":80, "english":70}
```

```
>>> score.pop("age", 0)
0
```

모든 값 삭제 clear()

```
>>> score.clear()
>>> score
{}
```

모든 키, 값 가져오기

```
>>> score.keys()
dict_keys(["math", "english"])
>>> score.values()
dict_values([80, 70])
>>> score.items()
dict_items([("math",80), ("english",70)])
```

실습01

(1) 이름, 나이, 연락처를 입력받아 딕셔너리를 만들어 출력해주세요

실행 :

이름을 입력해주세요 : 홍길동

나이를 입력해주세요 : 27

연락처를 입력해주세요 : 010-3023-1223

{'이름': '홍길동', '나이': '27', '연락처': '010-3023-1223'}

(2) 두사람의 이름, 나이, 연락처를 입력받아 각각 딕셔너리를 만들어 리스트에 넣어주세요

실행 :

이름을 입력해주세요 : 홍길동

나이를 입력해주세요 : 27

연락처를 입력해주세요 : 010-3023-1223

이름을 입력해주세요 : 이몽룡

나이를 입력해주세요 : 30

연락처를 입력해주세요 : 010-3030-4434

[{'이름': '홍길동', '나이': '27', '연락처': '010-3023-1223'}, {'이름': '이몽룡', '나이': '30', '연락처': '010-3030-4434'}]

9장. 세트

- 파이썬기초 -

세트

중복을 허용하지 않은 값들을 저장하는 자료형 - 순서가 없다

세트 = {값1, 값2, 값3, 값4}

`animal = {"dog", "cat", "monkey", "horse"}`

```
>>> animal = {"dog", "cat", "monkey", "horse"}
```

```
>>> type(animal)
```

```
<class 'set'>
```

세트의 기능

세트에 특정값 확인

```
>>> animal = {"dog", "cat", "monkey", "horse"}  
>>> "cat" in animal  
True
```

set을 사용하여 세트 만들기

```
>>> a = set("animal")  
>>> a  
{“a”, “n”, “i”, “m”, “a”, “l”}
```

```
>>> b = set(range(5))  
>>> b  
{0, 1, 2, 3, 4}
```


집합 연산 #1

합집합 |, set.union

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a | b
{1, 2, 3, 4, 5}
>>> set.union(a, b)
{1, 2, 3, 4, 5}
```

교집합 &, set.intersection

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a & b
{3}
>>> set.intersection(a, b)
{3}
```

집합 연산 #2

차집합 -, set.difference

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a - b
{1, 2}
>>> set.difference(a, b)
{1, 2}
```

대칭차집합 ^, set.symmetric_difference

```
>>> a = {1, 2, 3}
>>> b = {3, 4, 5}
>>> a ^ b
{1, 2, 4, 5}
>>> set.symmetric_difference(a, b)
{1, 2, 4, 5}
```

세트 조작하기

추가하기 add(요소)

```
>>> a = {1, 2, 3, 4}
>>> a.add(5)
>>> a
{1, 2, 3, 4, 5}
```

삭제하기 remove(요소), discard(요소)

```
>>> a = {1, 2, 3, 4}
>>> a.remove(1)
>>> a
{2, 3, 4}
>>> a.discard(2)
>>> a
{3, 4}
```

remove는 없으면 error를 표시하고, discard는 없으면 그냥 지나감