세상의 속도를
따라잡고 싶다면

점프 투 파이썬

박응용 지음(위키독스 운영자)

이지스퍼블리싱

02

파이썬 프로그래밍의 기초, 자료형

02-3 리스트 자료형

02-4 튜플 자료형

02-3 리스트 자료형

리스트(list)란?

- 자료형의 집합을 표현할 수 있는 자료형
  - 대괄호([ ])로 감싸고 각 요솟값은 쉼표(,)로 구분

```
리스트명 = [요소1, 요소2, 요소3, ...]
```

- 숫자와 문자열만으로 프로그래밍을 하기엔 부족한 점이 많음
  - 예) 1부터 10까지의 숫자 중 홀수 모음인 집합 {1, 3, 5, 7, 9}는 숫자나 문자열로 표현 불가능
  - 리스트로 해결 가능!

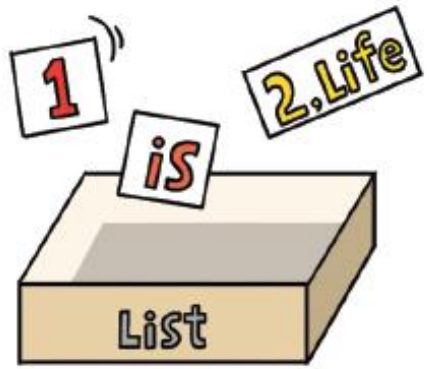
```
>>> odd = [1, 3, 5, 7, 9]
```

02-3 리스트 자료형

리스트(list)란?

- 리스트의 생김새
  - 리스트 안에는 어떠한 자료형도 포함 가능

```
>>> a = []
>>> b = [1, 2, 3]
>>> c = ['Life', 'is', 'too', 'short']
>>> d = [1, 2, 'Life', 'is']
>>> e = [1, 2, ['Life', 'is']]
```



02-3 리스트 자료형

리스트의 인덱싱

- 문자열과 같이 인덱싱 적용 가능

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

- 파이썬은 숫자를 0부터 세기 때문에 a[0]이 리스트 a의 첫 번째 요소

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

- 요솟값 간의 덧셈

```
>>> a[0] + a[2] <-- 1 + 3
4
```

- a[-1]은 리스트 a의 마지막 요솟값

```
>>> a[-1]
3
```

02-3 리스트 자료형

▪ 리스트의 인덱싱

- 리스트 내에 리스트가 있는 경우

```
>>> a = [1, 2, 3, ['a', 'b', 'c']]
```

- a[-1]은 마지막 요솟값인 리스트 ['a', 'b', 'c'] 반환

```
>>> a[0]
1
>>> a[-1]
['a', 'b', 'c']
>>> a[3]
['a', 'b', 'c']
```

- 리스트 a에 포함된 ['a', 'b', 'c'] 리스트에서 'a' 값을 인덱싱을 사용해 반환할 방법은?

```
>>> a[-1][0] <- ['a', 'b', 'c'][0]
'a'
```

  - a[-1]로 리스트 ['a', 'b', 'c']에 접근하고, [0]으로 요소 'a"에 접근

02-3 리스트 자료형

▪ 리스트의 슬라이싱

▪ 문자열과 같이 슬라이싱 적용 가능

```
>>> a = [1, 2, 3, 4, 5]
>>> a[0:2]
[1, 2]
```

```
>>> a = [1, 2, 3, 4, 5]
>>> b = a[:2] <- 처음부터 a[1]까지
>>> c = a[2:] <- a[2]부터 마지막까지
>>> b
[1, 2]
>>> c
[3, 4, 5]
```

02-3 리스트 자료형

리스트 연산하기

더하기(+)

- + 기호는 2개의 리스트를 합치는 기능
- 문자열에서 "abc" + "def" = "abcdef"가 되는 것과 같은 의미

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

반복하기(*)

- * 기호는 리스트의 반복을 의미
- 문자열에서 "abc" * 3 = "abcabcabc"가 되는 것과 같은 의미

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

02-3 리스트 자료형

리스트 연산하기

리스트 길이 구하기

- len( ) 함수 사용
- 문자열, 리스트 외에 앞으로 배울 튜플과 딕셔너리에서도 사용 가능한 내장 함수

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

02-3 리스트 자료형

▪ 리스트의 수정과 삭제

▪ 리스트의 값 수정하기

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

▪ del 함수를 사용해 리스트 요소 삭제하기

- del 키워드 사용

del 객체

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

```
>>> a = [1, 2, 3, 4, 5]
>>> del a[2:] ← a[2]부터 끝까지 삭제
>>> a
[1, 2]
```

※ 슬라이싱 기법 활용 가능

02-3 리스트 자료형

▪ 리스트 관련 함수

▪ 리스트에 요소 추가하기 – **append**

- 리스트의 맨 마지막에 요소 추가

```
>>> a = [1, 2, 3]
>>> a.append(4)  <- 리스트의 맨 마지막에 4를 추가
>>> a
[1, 2, 3, 4]
```

- 어떤 자료형도 추가 가능

```
>>> a.append([5, 6])  <- 리스트의 맨 마지막에 [5, 6]을 추가
>>> a
[1, 2, 3, 4, [5, 6]]
```

▪ 리스트 정렬 – **sort**

- 리스트의 요소를 순서대로 정렬

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

- 문자의 경우 알파벳 순서로 정렬 가능

```
>>> a = ['a', 'c', 'b']
>>> a.sort()
>>> a
['a', 'b', 'c']
```

02-3 리스트 자료형

▪ 리스트 관련 함수

- 리스트 뒤집기 - **reverse**
  - 리스트를 역순으로 뒤집어 줌
  - 요소를 역순으로 정렬하는 것이 아닌, 현재의 리스트 그대로 뒤집음

```
>>> a = ['a', 'c', 'b']
>>> a.reverse()
>>> a
['b', 'c', 'a']
```

- 인덱스 반환 – **index**
  - 요소를 검색하여 위치 값 반환

```
>>> a = [1, 2, 3]
>>> a.index(3) <- 3은 리스트 a의 세 번째(a[2]) 요소
2
>>> a.index(1) <- 1은 리스트 a의 첫 번째(a[0]) 요소
0
```

  - 값이 존재하지 않으면, 값 오류 발생

```
>>> a.index(0)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: 0 is not in list
```

02-3 리스트 자료형

▪ 리스트 관련 함수

▪ 리스트에 요소 삽입 - **insert**

- insert(a, b)
  - a번째 위치에 b를 삽입하는 함수

```
>>> a = [1, 2, 3]
>>> a.insert(0, 4)  <- a[0] 위치에 4 삽입
>>> a
[4, 1, 2, 3]
```

```
>>> a.insert(3, 5)  <- a[3] 위치에 5 삽입
>>> a
[4, 1, 2, 5, 3]
```

▪ 리스트 요소 제거 – **remove**

- remove(x)
  - 리스트에서 첫 번째로 나오는 x를 삭제

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

- 값이 여러 개인 경우 첫 번째 것만 삭제

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

02-3 리스트 자료형

리스트 관련 함수

- 리스트 요소 끄집어 내기 - **pop**
  - 리스트의 맨 마지막 요소를 돌려주고 해당 요소 삭제
  - pop(x)
    - 리스트의 x번째 요소를 돌려주고 해당 요소 삭제

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

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

- 리스트에 포함된 요소 x의 개수 세기 – **count**
  - 리스트에 포함된 요소의 개수 반환
  - count(x)
    - 리스트 안에 x가 몇 개 있는지 조사하여 그 개수를 돌려주는 함수

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

02-3 리스트 자료형

▪ 리스트 관련 함수

- 리스트 확장 - **extend**
  - 리스트에 리스트를 더하는 함수
  - extend(x)
    - x에는 리스트만 올 수 있음

a.extend([4, 5])

=

a += [4, 5]

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

02-4 튜플 자료형

튜플(tuple)이란?

- 리스트와 유사한 자료형

리스트	튜플
[ ]로 둘러쌈	( )로 둘러쌈
요솟값 생성 / 삭제 / 수정 가능	요솟값 변경 불가능

```
t1 = ()
t2 = (1,)
t3 = (1, 2, 3)
t4 = 1, 2, 3
t5 = ('a', 'b', ('ab', 'cd'))
```

- 튜플은 1개의 요소만을 가질 때는 요소 뒤에 쉼표(,)를 반드시 붙여야 함 (예) t2 = (1,)
- 소괄호( )를 생략해도 무방함 (예) t4 = 1, 2, 3
- 프로그램이 실행되는 동안 값을 유지해야 한다면 튜플을, 수시로 값을 변경해야 하면 리스트 사용

02-4 튜플 자료형

- **튜플의 요솟값을 지우거나 변경하려고 하면 어떻게 해야 할까?**

1. 튜플 요솟값을 삭제하려 할 때

```
>>> t1 = (1, 2, 'a', 'b')
>>> del t1[0]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object doesn't support item deletion
```

2. 튜플 요솟값을 변경하려 할 때

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[0] = 'c'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

02-4 튜플 자료형

튜플 다루기

인덱싱하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[0]
1
>>> t1[3]
'b'
```

슬라이싱하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[1:]
(2, 'a', 'b')
```

튜플 더하기와 곱하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> t2 = (3, 4)
>>> t3 = t1 + t2
>>> t3
(1, 2, 'a', 'b', 3, 4)
```

```
>>> t2 = (3, 4)
>>> t3 = t2 * 3
>>> t3
(3, 4, 3, 4, 3, 4)
```

튜플 길이 구하기

```
>>> t1 = (1, 2, 'a', 'b')
>>> len(t1)
4
```

감사합니다.