Basic Python

Lecture 4. 리스트

BAE SEKANG(배세강)

Contents

1. List

- 리스트 생성
- 리스트 접근
- 리스트 수정
- 음수 인덱싱
- 2차원 리스트

2. Slicing

- 슬라이싱

3. List Methods

- 리스트 메소드

List

♣ 리스트 생성

- 리스트(List): 여러 아이템을 담을 수 있는 공간
- 대괄호 []를 활용하여 생성할 수 있다.
- 쉼표(,)를 활용해 아이템을 구분한다.

```
my_list = [10, 20, 30]

print(type(my_list))
```

>> <class 'list'>

List

♣ 리스트 접근

- 인덱스(index)를 활용하여 리스트에 접근한다.
- 첫 번째 원소의 index는 0이다. (1이 아님에 주의!)
- 리스트의 범위를 벗어난 접근을 할 수 없다.

```
my_list = [10, 20, 30]

print(my_list[0])
print(my_list[1])
print(my_list[3])
```

```
>> 10
>> 20
>> IndexError
```

List

♣ 리스트 수정

- index를 활용하여 해당 위치의 값을 변경한다.

```
my_list = [10, 20, 30]
print(my_list)
my_list[2] = 100
print(my_list)
```

>> [10, 20, 30]

>> [10, 20, 100]

List

♣ 리스트 수정

- del을 활용하여 값을 지울 수 있다.

```
my_list = [10, 20, 30]
print(my_list)

del my_list[1]
print(my_list)
```

>> [10, 20, 30]

>> [10, 30]

List

♣ len()

- list에 들어있는 원소의 개수를 반환한다.

```
my_list = [10, 20, 30]
a = len(my_list)

print(a)
```

>> 3

List

♣ 음수 인덱싱

- 음수 index를 활용하여 리스트의 뒤에서부터 접근할 수 있다.
- 리스트 맨 뒤의 첫 index는 -1이다.
- 리스트의 원소가 n개일 때, 유효한 index의 범위는 $-n \leq index \leq n-1$

```
my_list = [10, 20, 30]
print(my_list[-1])
my_list[-2] = 100
print(my_list)
```

>> 30

>> [10, 100, 30]

List

♣ 2차원 리스트

- 2차원 리스트: 여러 개의 1차원 리스트를 원소로 가진 리스트

```
my_list = [[1, 2], [3, 4], [5, 6, 7]]

print(my_list[0][1])

print(my_list[2])
```

>> 2

>> [5, 6, 7]

Q. 3차원, 4차원, …도 가능할까?

Slicing

♣ 슬라이싱(Slicing)

- list에서 연속된 구간을 잘라서 가져올 수 있다.
- my_list[start : end(포함 X) : step]

```
my_list = [10, 20, 30, 40, 50]
print(my_list[0:5])
print(my_list[1:5:1])
print(my_list[::2])
print(my_list[1:3])
print(my_list[::-1])
print(my_list[1:-2:])
```

```
>> [10, 20, 30, 40, 50]
>> [20, 30, 40, 50]
>> [10, 30, 50]
>> [20, 30]
>> [50, 40, 30, 20, 10]
>> [20, 30]
```

Slicing

♣ 슬라이싱(Slicing)

- 문자열도 슬라이싱이 가능하다.

```
my_string = "Hello World!"
print(my_string[2::3])
```

>> l r!

List Methods

♣ append()와 clear()

- append(): list 뒤쪽에 원소를 추가할 수 있다.
- clear(): list 원소를 모두 없앤다.

```
my_list = [10, 20]
my_list.append(30)
print(my_list)

my_list.clear()
print(my_list)
```

>> [10, 20, 30]

>> []

List Methods

Practice.

정수 a, b, c를 입력 받은 후, 리스트 [c, b, a]를 출력한다.

[입력]
10
15
25

[출력]
[25, 15, 10]

List Methods

♣ count()

- count(x): list에 들어있는 x의 개수를 반환한다.

```
my_list = [10, 20, 20, 30, 40]

print(my_list.count(20))
print(my_list.count(50))
```

```
>> 2
>> 0
```

List Methods

♣ index()

- index(x): 값 x의 index를 반환한다. x가 여러 개라면 가장 앞의 index를, x가 없으면 에러를 일으킨다.

```
my_list = [10, 20, 20, 30, 40]

print(my_list.index(20))
print(my_list.index(50))
```

>> 1
>> ValueError

List Methods

♣ insert()

- insert(i, x): index i 위치에 x를 끼워 넣는다.

```
my_list = [10, 20, 20, 30, 40]

my_list.insert(2, 50)
print(my_list)
my_list.insert(6, 60)
print(my_list)
my_list.insert(-2, 70)
print(my_list)
```

>> [10, 20, 50, 20, 30, 40]

>> [10, 20, 50, 20, 30, 40, 60]

>> [10, 20, 50, 20, 30, 70, 40, 60]

List Methods

♣ pop()

- pop(): list의 맨 뒤 원소를 꺼낸다.

```
my_list = [10, 20, 30]

value = my_list.pop()
print(value)
print(my_list)
```

```
>> 30
>> [10, 20]
```

List Methods

♣ pop()

- pop(index): 해당 인덱스에 위치한 원소를 꺼낸다.

```
my_list = [10, 20, 30]

value = my_list.pop(1)
print(value)
print(my_list)
```

```
>> 20
>> [10, 30]
```

List Methods

♣ sort()

- sort(): list를 오름차순으로 정렬한다.
- sort(reverse=True): list를 내림차순으로 정렬한다.

```
my_list = [10, 40, 20, 15, 50]
my_list.sort()
print(my_list)
my_list.sort(reverse=True)
print(my_list)
```

>> [10, 15, 20, 40, 50]

>> [50, 40, 20, 15, 10]

Q. sorted()는? ➔ 정렬된 새 리스트를 반환한다.

List Methods

♣ reverse()

- reverse(): list 원소의 순서를 반대로 뒤집는다.

```
my_list = [10, 40, 20, 15, 50]

my_list.reverse()
print(my_list)
```

>> [50, 15, 20, 40, 10]

Q. [::-1]과 다른 점은? ➔ [::-1]은 뒤집힌 새 리스트를 만든다.

List Methods

♣ extend()

- my_list.extend(new_list): new_list의 원소를 그대로 my_list에 넣는다.

```
my_list = [10, 20]

my_list.extend([30, 40])
print(my_list)
my_list.append([50, 60])
print(my_list)
```

>> [10, 20, 30, 40]

>> [10, 20, 30, 40, [50, 60]]

List Methods

♣ copy()

- copy(): list를 복사하여 반환한다.

```
my_list = [10, 20, 30]

new_list = my_list.copy()
print(new_list)
```

>> [10, 20, 30]

Q. new_list = my_list를 쓰면 안될까?

QUIZ

Problem 1.

정수 a, b를 순서대로 입력 받은 후 [a+b, a-b, a*b, a/b]를 출력하는 프로그램을 작성한다.

Problem 2.

my_list = [1, -1, 2, -2, 3, -3]일 때 print(my_list[:3:-1])의 출력 결과는

Problem 3.

append()와 extend()의 차이는?

THANK YOU!

Any Question?