

Python 과학 프로그래밍 기초

시퀀스형 데이터 (2)

한림대학교 소프트웨어융합대학
박섭형

2021년 1학기

리스트 메소드

- append
- extend
- insert
- pop
- remove
- clear
- sort
- reverse
- copy

Python 숫자형 객체는 모두 immutable이다

```
[1]: a = b = c = 3
      b = 2
      print(a, b, c)
```

3 2 3

```
[2]: a = b = c = "test string"
      b = "another string"
      print(a)
      print(b)
      print(c)
```

```
test string
another string
test string
```

```
[3]: a = 2
      print(id(a))
      a = a + 3
      print(id(a))
```

```
140724417996608
140724417996704
```

```
[4]: b = a
      print(id(a))
      print(id(b))
      print(a, b)
```

```
140724417996704
140724417996704
5 5
```

```
[5]: b = 2
      print(id(a))
      print(id(b))
      print(a, b)
```

```
140724417996704
140724417996608
5 2
```

리스트의 Assignment vs Shallow Copy vs Deep Copy

리스트의 Assignment

```
[6]: a = [1, 2, 3, 4]
      b = a
      print(id(a))
```

```
print(id(b))
print(a, b)
print(a == b, a is b)
```

2223627720896

2223627720896

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

True True

```
[7]: b[1] = 10
print(id(a))
print(id(b))
print(a, b)
print(a == b, a is b)
```

2223627720896

2223627720896

[1, 10, 3, 4] [1, 10, 3, 4]

True True

```
[8]: a = [1, 2, 3, 4]
b = a
c = a.copy()
b[1] = 10
print(id(a))
print(id(b))
print(id(c))
print(a, b, c)
print(a == b, a is b, a is c)
```

2223627702080

2223627702080

2223627701568

[1, 10, 3, 4] [1, 10, 3, 4] [1, 2, 3, 4]

True True False

Deepcopy

```
[9]: from copy import deepcopy
a = [[1,2],[3,4]]
b = a
c = a.copy()
d = deepcopy(a)
```

```
[10]: print(id(a))
print(id(b))
print(id(c))
print(id(d))
```

```
2223627982272
2223627982272
2223627747264
2223627545088
```

```
[11]: print(id(a[0]), id(a[1]))
print(id(b[0]), id(b[1]))
print(id(c[0]), id(c[1]))
print(id(d[0]), id(d[1]))
```

```
2223627720256 2223626965504
2223627720256 2223626965504
2223627720256 2223626965504
2223627702080 2223627701824
```

```
[12]: print(a)
print(b)
print(c)
print(d)
```

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

```
[13]: b[1][1] = 10
print(a)
print(b)
print(c)
print(d)
```

```
[[1, 2], [3, 10]]
[[1, 2], [3, 10]]
[[1, 2], [3, 10]]
[[1, 2], [3, 4]]
```

시퀀스 자료형 관련 메소드

- len()
- min()
- max()
- sum()
- sorted()
- reversed()
- any()
- all()
- enumerate()

Len(): 시퀀스 자료형의 원소 길이 구하기

len(object): 컨테이너 내의 항목의 개수 반환

```
[14]: l = ['a', 'b', 'c', 'd', 'e', 'f', 'g']
t = (0, 2, 4, 6, 8, 10, 12, 14)
s = "abcdefg"
r = range(7)
print(len(l), len(t), len(s), len(r))
```

7 8 7 7

시퀀스 자료형의 최댓값, 최솟값 구하기

min(iterable): iterable 내의 최솟값을 반환
max(iterable): iterable 내의 최댓값을 반환

```
[15]: print(min(l), max(l))
      print(min(t), max(t))
      print(min(s), max(s))
      print(min(r), max(r))
```

```
a g
0 14
a g
0 6
```

숫자를 포함한 시퀀스 자료형의 합계 구하기

`sum(iterable, start=0)`: iterable의 합과 start의 합을 반환

```
[16]: l = [1, 5, 8, 3, 2, 4]
      t = (0, 2, 4, 6, 8, 10, 12, 14)
      r = range(7)
      sum(l), sum(t), sum(r)
```

```
[16]: (23, 56, 21)
```

```
[17]: sum(l, 10)
```

```
[17]: 33
```

Sorted: 시퀀스 자료형의 정렬

```
[18]: ls = sorted(l)
      print(l, ls)
```

```
[1, 5, 8, 3, 2, 4] [1, 2, 3, 4, 5, 8]
```

```
[19]: s = "Try it!"
      ss = sorted(s)
      print(s, ss)
```

```
Try it! [' ', '!', 'T', 'i', 'r', 't', 'y']
```

```
[20]: t = (1, 5, 10, 5, 6)
      st = sorted(t)
      print(t, st)
```

(1, 5, 10, 5, 6) [1, 5, 5, 6, 10]

```
[21]: t = (1, 5, 'a')
      st = sorted(t)
      print(t, st)
```

```
-----
TypeError                                Traceback (most recent call last)
<ipython-input-21-89d678f6ec63> in <module>
      1 t = (1, 5, 'a')
----> 2 st = sorted(t)
      3 print(t, st)

TypeError: '<' not supported between instances of 'str' and 'int'
```

Reversed:

reversed(sequence): sequence의 모든 값을 반대 순서로 만든 iterator를 반환

```
[22]: a = [1, 3, 7, 4]
      b = reversed(a)
      print(b)
      print(list(b))
```

<list_reverseiterator object at 0x00000205BA9068E0>
[4, 7, 3, 1]

```
[23]: a = [1, 3, 7, 4]
      for n in reversed(a):
          print(n, end=", ")
```

4, 7, 3, 1,

Any:

`any(iterable)`: iterable 내의 어떤 x가 True이면 True를 반환, 그렇지 않으면 False를 반환

```
[24]: a = [1, 5, 0, 3]
      any(a)
```

[24]: True

```
[25]: any([], {})
```

[25]: False

All:

`all(iterable)`:

- iterable 내의 모든 x가 True이면 True를 반환
- 그렇지 않으면 False를 반환

```
[26]: a = [1, 5, 0, 3]
      all(a)
```

[26]: False

```
[27]: all("abc")
```

[27]: True

```
[28]: all([]), all(""), all(tuple())
```

[28]: (True, True, True)

Enumerate:

`enumerate(iterable, start=0)`:

start로 시작하는 카운트와 iterable 내의 value의 쌍을 가진 enumerate 객체를 반환

```
[29]: a = [1, 5, 0, 3]
      for n, v in enumerate(a):
          print(n, v)
```


0 1

1 5

2 0

3 3

```
[30]: a = [1, 5, 0, 3]
      for n, v in enumerate(a, 10):
          print(n, v)
```

10 1

11 5

12 0

13 3