

파이썬 프로그래밍

9차시

문자열 관련 메소드



⚠ 학습개요

... 메소드 `str.replace(a, b)`

- 문자열 `str`에서 `a`가 나타나는 모든 부분을 `b`로 모두 바꾼 문자열을 반환

... 메소드 `count()`와 `join()`

- 부분 문자열 출현 횟수를 반환
- 문자열의 문자와 문자 사이에 원하는 문자열을 삽입

... 메소드 `find()`와 `index()`

- 문자열을 찾기

... 메소드 `split()`

- 문자열을 여러 문자열로 나눔

⚠ 학습개요

... 메소드 `center()`와 `strip()`

- 폭을 지정하고 중앙에 문자열 배치
- 문자열 앞뒤의 특정 문자들을 제거

... 메소드 `format()`

- 간결한 출력 처리

⚠ 학습목표

... 다양한 문자열 관련 메소드를 활용할 수 있다.

Chapter 1.

메소드

`str.replace(a, b)`

P Y T H O N P R O G R A M M I N G

⚠ 문자열 바꿔 반환하는 메소드 `replace()`

+ 메소드(method): 클래스에 소속된 함수

- 문자열 클래스 **str**에 속한 메소드
 - `str.replace(a, b)`처럼 호출

+ 메소드 `str.replace(a, b)`

- 문자열 **str**에서 **a**가 나타나는 모든 부분을 **b**로 모두 바꾼 문자열을 반환

```
>>> str = '자바는 인기 있는 언어 중 하나다.'
>>> str.replace('자바는', '파이썬은')
'파이썬은 인기 있는 언어 중 하나다.'
>>> str.replace(' ', '')
'자바는인기있는언어중하나다.'
```


⚠ 문자열 바꿔 반환하는 메소드 `replace(a, b, n)`

+ 메소드 `str.replace(old, new, count)`

- 문자열 `old`를 `new`로 대체하는데, 옵션인 `count`는 대체 횟수를 지정
- 옵션인 `count`가 없으면 모두 바꾸고, 있으면 앞에서부터 지정한 횟수만큼 바꿈

```
>>> str = '파이썬 파이썬 파이썬'
>>> str.replace('파이썬', 'Python!')
'Python! Python! Python!'
>>> str.replace('파이썬', 'Python!', 1)
'Python! 파이썬 파이썬'
>>> str.replace('파이썬', 'Python!', 2)
'Python! Python! 파이썬'
```

⚠ 문자열 바꿔 반환하는 메소드 `replace(a, b, n)`

+ 문자열은 수정될 수 없는 자료

- 문자열은 수정될 수 없다는(immutable) 특징
- `replace()` 등의 메소드 기능이 작용한 새로운 문자열을 반환

⚠ 소수 형태의 실수에서 모든 자릿수의 합 구하기

+ 메소드 `replace()` 사용

- 표준 입력으로 받은 실수 형태의 문자열에서 소수점을 제거

+ 각 자릿수를 참조해 모두 더하기

⚠ 소수 형태의 실수에서 모든 자릿수의 합 구하기

[코딩실습] 실수의 모든 자릿수 더하기

난이도 기본

```
1. value = input('실수(세 자리.두 자리로 345.78처럼)를 하나 입력하세요. >> ')
2. num = value.replace('.', '')
3. sum = 0
4. sum += int(num[0])
5. sum += int(num[1])
6. sum += int(num[2])
7. sum += int(num[3])
8. sum += int(num[4])
9. print('입력값:', value)
10. print('모든 자릿수 합:', sum)
```

결과

실수(세 자리.두 자리로 345.78처럼)를
하나 입력하세요. >> 345.67
입력값: 345.67
모든 자릿수 합: 25

실수(세 자리.두 자리로 345.78처럼)를
하나 입력하세요. >> 278.34
입력값: 278.34
모든 자릿수 합: 24

Chapter 2.

메소드 count()와 join()

P Y T H O N P R O G R A M M I N G

⚠ 함수 count()와 join()

✚ 메소드 count(): 부분 문자열 출현 횟수를 반환

✚ 메소드 join(): 문자열의 문자와 문자 사이에 원하는 문자열을 삽입

- '->'.join('12345')
- 문자열 '12345' 사이에 '->'를 삽입한 문자열을 반환

```
>>> str = '단순한 것이 복잡한 것보다 낫다.'
>>> str.count('복잡')
1
>>> str.count('것')
2
```

```
>>> num = '12345'
>>> '->'.join(num)
'1->2->3->4->5'
```



용어를 알아봅시다!

+ 함수와 메소드

- 함수(function)는 특정 작업을 수행하는 독립된 '코드 모임'이다.
함수 len()을 호출할 때는 len('function')이라고 호출한다.
반면, **메소드는 클래스에 포함돼 있는 함수**를 말한다.
메소드 호출을 살펴보면 'object'.count('o')처럼 str 객체 'object'에 점(.)을 붙인 후 메소드 count('o')라는 이름으로 호출한다.
즉, 함수는 독립적으로 직접 호출하지만 메소드는 객체를 통해 호출한다.



용어를 알아봅시다!

함수(function)

특정한 업무를 수행하는
프로그램 단위

`len('python')`



독립적

메소드(method)

`'python'.count('th')`



객체 또는 클래스에 소속

[그림9-5] 함수와 메소드

Chapter 3.

메소드 find()와 index()

P Y T H O N P R O G R A M M I N G

⚠ 문자열을 찾는 메소드 `find('sub')`와 `index('sub')`

+ 클래스 `str`에서 부분 문자열 `sub`가 맨 처음에 위치한 첨자를 반환

- 메소드 `str.find('sub')`: 없으면 `-1`을 반환
- 메소드 `str.index('sub')`: 없으면 `ValueError`를 발생
- 역순으로 검색: `rfind()`, `rindex()`

⚠ 문자열을 찾는 메소드 find()와 index()

```
>>> str = '자바 C 파이썬 코틀린'
>>> str.find('자바')
0
>>> str.index('자바')
0
>>> str.find('파이')
5
>>> str.find('파이썬')
5
```

```
>>> str.find('c++')
-1
>>> str.index('c++')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: substring not found
```

⚠ 문자열을 찾는 메소드 find()와 index()

[코딩실습] 문자열에 두 단어의 순서 교환과 역순 출력

난이도 기본

```
1. str = input('2개의 단어를 빈 공간으로 구분해 입력하세요. >> ')
2. pos = str.find(' ')
3. preWord = str[:pos]
4. postWord = str[pos+1:]
5. print(preWord, postWord)
6. print(preWord[::-1], postWord[::-1])
```

결과

2개의 단어를 빈 공간으로 구분해 입력하세요. >> 사과 복숭아
사과 복숭아
과사 아송복

Chapter 4.

메소드 `split()`

P Y T H O N P R O G R A M M I N G

⚠ 문자열을 여러 문자열로 나누는 메소드 split()

+ 문자열 str에서 공백을 기준으로 문자열을 분리

- 리스트 [항목1, 항목2, ...] 반환
 - 리스트란 항목의 나열인 파이썬 자료형
- 공백은 **whitespace**라고도 부르며, 공백, 탭, 엔터키(뉴라인) 등

```
>>> '사과 배 복숭아 딸기 포도'.split()
['사과', '배', '복숭아', '딸기', '포도']
>>> '데스크톱 1000000 노트북 1800000 스마트폰 1200000'.split()
['데스크톱', '1000000', '노트북', '1800000', '스마트폰', '1200000']
```


⚠ 문자열을 여러 문자열로 나누는 메소드 split()

+ str.split(',')

- 괄호 안에 특정한 문자열 값이 있을 경우
 - 이 부분 문자열 값을 구분자를 이용해 문자열을 나눠 줌

```
>>> '데스크톱 1000000, 노트북 1800000, 스마트폰 1200000'.split(',')  
['데스크톱 1000000', ' 노트북 1800000', ' 스마트폰 1200000']  
>>> '데스크톱 1000000, 노트북 1800000, 스마트폰 1200000'.split(' , ')  
['데스크톱 1000000', '노트북 1800000', '스마트폰 1200000']
```

⚠ 4개의 수를 입력 받아 합, 평균값, 최댓값, 최솟값을 출력

+ 표준 입력 `input()`에서 한 번에 여러 값을 입력

```
>>> m, n = '100 200'.split()
>>> m, n
('100', '200')
```

⚠ 4개의 수를 입력 받아 합, 평균값, 최댓값, 최솟값을 출력

+ 최댓값과 최솟값은 함수 `max()`와 `min()`을 사용

[코딩실습] 4개의 수를 입력받아 합, 평균값, 최댓값, 최솟값을 출력

난이도 기본

```
1. m, n, x, y = input('4개의 수 입력 >> ').split()
2. a, b, c, d = float(m), float(n), float(x), float(y)
3. print('입력값: ', a, b, c, d)
4. sum = a + b + c + d
5. print('합: ', sum, '평균: ', sum / 4)
6. print('최대: ', max(a, b, c, d), '최소: ', min(a, b, c, d))
```

결과

```
4개의 수 입력 >> 3.7 5.8 9 2.5
입력값: 3.7 5.8 9.0 2.5
합: 21.0   평균값 5.25
최댓값: 9.0   최솟값: 2.5
```

Chapter 4.

메소드

center()와 strip()

P Y T H O N P R O G R A M M I N G

⚠ 메소드 center()와 strip()

+ 폭을 지정하고 중앙에 문자열 배치하는 메소드 center()

```
'파이썬 강좌'.center(30, '*')
'파이썬 강좌'.center(30)
'파이썬 강좌'.center(30, '=')
```

```
***** 파이썬 강좌 *****
'          파이썬 강좌          '
'===== 파이썬 강좌 ====='
```

⚠ 메소드 center()와 strip()

+ 문자열 앞뒤의 특정 문자들을 제거하는 strip() 메소드

- 인자의 문자열은 제거되어야 할 모든 '문자의 조합'을 의미

```
'python'.lstrip()
'python'.rstrip()
'python'.strip()
***python---'.strip('* -')
```

```
'python '
' python'
'python'
'python'
```


Chapter 5.

메소드 format()

P Y T H O N P R O G R A M M I N G

⚠ 문자열의 format() 메소드를 이용해 간결한 출력 처리

+ str.format(인자들)

- 문자열 str 중간 중간에 변수나 상수를 함께 출력

```
>>> '3 + 4 = 7'
'3 + 4 = 7'
>>> str = '{} + {} = {}'.format(3, 4, 3 + 4)
>>> print(str)
3 + 4 = 7
```

```
>>> a, b = 10, 4
>>> print('{} * {} = {}'.format(a, b, a * b))
10 * 4 = 40
>>> print('{0} / {1} = {2}'.format(a, b, a / b))
10 / 4 = 2.5
>>> print('{1} / {0} = {2}'.format(a, b, b / a))
4 / 10 = 0.4
```

⚠ 문자열의 format() 메소드를 이용해 간결한 출력 처리

```
>>> a, b = 10, 3
>>> print('{0:d} / {1:d} = {2:f}'.format(a, b, a / b))
10 / 3 = 3.333333
>>> print('{0:5d} / {1:5d} = {2:10.3f}'.format(a, b, a / b))
10 /      3 =      3.333
```

⚠ C 언어의 포매팅 스타일인 %d와 %f 등으로 출력

+ C의 printf()에서 사용하는 형식 지정자 스타일도 지원

- %d, %x, %o, %f, %c, %s 등을 사용해 %로 이어지는 뒤의 상수나 변수를 순서대로 10진수, 16진수, 8진수, 소수점, 문자, 문자열로 출력

```
>>> '%d - %x = %o' % (30, 20, 30 - 20)
'30 - 14 = 12'
>>> print('%d ** %x = %o' % (3, 2, 3 ** 2))
3 ** 2 = 11
>>> print('%10.2f' % 2.718281)
      2.72
>>> print('%10c' % 'p')
      p
>>> print('%10s' % 'python')
python
>>> print('%d%%' % 99 )
99%
```

⚠ 메소드 `str.replace(a, b)`

... 문자열 `str`에서 `a`가 나타나는 모든 부분을 `b`로 모두 바꾼 문자열을 반환

⚠ 메소드 `count()`

... 부분 문자열 출현 횟수를 반환

⚠ 메소드 join()

... 문자열의 문자와 문자 사이에 원하는 문자열을 삽입

⚠ 메소드 find()와 index()

... 문자열을 찾기

SUMMARY



⚠ 메소드 `split()`

... 문자열을 여러 문자열로 나눔

⚠ 메소드 `center()`

... 폭을 지정하고 중앙에 문자열 배치



⚠ 메소드 `strip()`

... 문자열 앞뒤의 특정 문자들을 제거

⚠ 메소드 `format()`

... 간결한 출력 처리