

파이썬 프로그래밍

26차시

람다 함수와
내장 함수 활용 1



! 학습개요

- ... 람다 함수(lambda function)
- ... 라이브러리
- ... 내장 함수

! 학습목표

- ... 람다 함수를 이해하고 구현할 수 있다.
- ... 라이브러리를 이해하고 표준 라이브러리와 써드 파티 라이브러리를 구별할 수 있다.
- ... 다양한 내장 함수를 활용할 수 있다.

Chapter 1.

람다 함수 (lambda function)

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

⚠ 람다 함수(lambda function)

+ 작고 이름이 없는(익명, anonymous) 함수

- 키워드 lambda 이후에 콤마로 구분된 인자 목록
- 람다 함수는 여러 개의 인자를 취할 수 있지만 표현식은 하나만 가능
- 키워드 return 없이 하나의 표현식 결과값이 반환

+ 람다 함수 구문과 예

lambda 인자 1, 인자 2, ... : 표현식(expression)

```
lambda x: x + 1  
lambda x, y: x + y  
lambda x, y: pow(x, y)
```

⚠ 람다 함수 구현과 호출

[코딩실습] 이름 없는 람다 함수 활용

난이도 응용

```
1. print((lambda x: x ** 3)(3))
2. print((lambda a, b: a % b)(10, 3))
3.
4. div = lambda a, b: a / b
5. print(div(10, 2))
```

결과

```
27
1
5.0
```

Chapter 2.

라이브러리

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

⚠ 라이브러리

필요한 기능을 담당하는 함수나 클래스를 모아놓은
모듈(modules)의 집합



⚠ 표준 라이브러리(standard library)

파이썬과 함께 설치돼 다른 추가적인 작업 없이 사용할 수 있는 라이브러리

+ 내장 함수(built-in function)

- 표준 라이브러리 중 **import** 없이 바로 함수 호출로 사용
 - 예: `print()` 함수



⚠ 써드 파티 라이브러리(third-party library)

+ 파이썬 패키지 색인 사이트(pypi.org)에서 제공

- 파이썬 프로그래밍 언어의 방대한 소프트웨어 저장소
- 매우 다양 라이브러리 제공
 - 교육 부문
 - 과학 및 수치 컴퓨팅
 - 빅데이터 및 머신 러닝
 - 웹 및 인터넷 개발
 - 그래픽
 - GUI
 - 게임 등



Chapter 3.

내장 함수

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

⚠ 산술 연산 관련 내장 함수

+ 절댓값 함수 `abs()`

```
>>> abs(-3)
3
>>> abs(-7.3876)
7.3876
```

+ 진수로 변환된 문자열을 반환하는 함수 `bin()`, `oct()`, `hex()`

```
>>> bin(20)
'0b10100'
>>> oct(20)
'0o24'
>>> hex(20)
'0x14'
```

```
>>>
>>> format(20, 'b')
'10100'
>>> format(20, 'o')
'24'
>>> format(20, 'x')
'14'
```

+ 내장 함수 `format()`

- 접두어 "0b", "0o", "0x"가 없이 표시

⚠ 산술 연산 관련 내장 함수

+ pow(), round(), divmod()

```
>>> pow(2, 3)
8
>>> pow(2.4, 3.1)
15.088805115741808
>>>
>>> round(3.141592)
3
>>> round(3.141592, 3)
3.142
>>>
>>> dm = divmod(10, 3)
>>> print(dm)
(3,1)
>>>
>>> dm = divmod(10.3, 3.1)
>>> print(dm)
(3.0, 1.0000000000000004)
```

+ min(), max(), sum()

```
>>> min(5, 1, 10)
1
>>> max(5, 1, 10)
10
>>> min([5, 1, 10])
1
>>> max((5, 1, 10))
10
>>>
>>> sum([5, 1, 10])
16
>>> sum([5, 1, 10], 5)
21
```

⚠ 함수 sorted()에서 정렬의 비교 키로 사용하는 키워드 인자 key

[코딩실습] 내장 함수 sorted()에서 키워드 인자 key 활용

난이도 응용

```
1. words = "The core of extensible programming is defining functions.".split()
2. # 각 항목 단어를 모두 소문자로 바꾼 항목을 키로 정렬
3. print(sorted(words, key = str.lower))
4. # 각 항목 단어의 첫자 1, 두 번째 문자를 키로 정렬
5. print(sorted(words, key = lambda word: word[1]))
6.
7. groupnumber = [('잔나비', 5), ('트와이스', 9), ('블랙핑크', 4), ('방탄소년단', 7)]
8. # 항목의 첫 번째 항목을 키로 정렬
9. print(sorted(groupnumber))
10. # 항목의 두 번째 항목인 그룹 인원수를 키로 정렬
11. print(sorted(groupnumber, key = lambda singer : singer[1]))
```

결과

```
['core', 'defining', 'extensible', 'functions.', 'is', 'of', 'programming', 'The']
['defining', 'of', 'The', 'core', 'programming', 'is', 'functions.', 'extensible']
[('방탄소년단', 7), ('블랙핑크', 4), ('잔나비', 5), ('트와이스', 9)]
[('블랙핑크', 4), ('잔나비', 5), ('방탄소년단', 7), ('트와이스', 9)]
```

⚠ 함수 sorted()에서 정렬의 비교 키로 사용하는 키워드 인자 key

[코딩실습] 내장 함수 sorted()에서 키워드 인자 key 활용

난이도 응용

```
1. pl = ['python', 'java', 'C++', 'Go', 'swift']
2. # 문자열 길이로 정렬
3. print(sorted(pl, key = len))
4. # 첨자 1(두번째 문자)로 정렬
5. print(sorted(pl, key = lambda x : x[1]))
6.
7. pl = [('python', 1990), ('java', 2000), ('C++', 1983), ('Go', 2009), ('swift',
    2014)]
8. # 첨자 0(언어 문자열)로 정렬
9. print(sorted(pl, key = lambda item : item[0]))
10. # 첨자 1(년도)로 정렬
11. print(sorted(pl, key = lambda item : item[1]))
```

결과

```
['Go', 'C++', 'java', 'swift', 'python']
['C++', 'java', 'Go', 'swift', 'python']
[('C++', 1983), ('Go', 2009), ('java', 2000), ('python', 1990), ('swift', 2014)]
[('C++', 1983), ('python', 1990), ('java', 2000), ('Go', 2009), ('swift', 2014)]
```


⚠ 항목의 논리를 검사하는 함수 `all()`, `any()`

+ 함수 `all()`(시퀀스)

- 시퀀스의 모든 항목이 참이거나 항목이 비어 있으면 `True`를 반환

```
>>> b = all([3 > 4, 'p' in 'py'])
>>> print(b)
False
>>> print(all('pyhon'))
True
>>> print(all(''))
True
>>> print(all({}))
True
```

```
>>> help(all)
Help on built-in function all in module builtins:

all(iterable, /)
    Return True if bool(x) is True for all values x in the iterable.

    If the iterable is empty, return True.
```

⚠ 항목의 논리를 검사하는 함수 `all()`, `any()`

+ 함수 `any()`(시퀀스)

- 시퀀스에서 **하나의 항목**이라도 참이거나 항목이 비어 있으면 `False`를 반환

```
>>> b = any([3 > 4, 'p' in 'py'])
>>> print(b)
True
>>> print(any(''))
False
>>> print(any({}))
False
```

```
>>> help(any)
Help on built-in function any in module builtins:

any(iterable, /)
    Return True if bool(x) is True for any x in the iterable.

    If the iterable is empty, return False.
```

⚠ 람다 함수(lambda function)

- ... 작고 이름이 없는(익명, anonymous) 함수
- ... 구현과 호출

⚠ 라이브러리

- ... 표준 라이브러리(standard library)
- ... 써드 파티 라이브러리(third-party library)

⚠ 내장 함수

... 산술 연산 내장 함수

- `abs()`, `min()`, `max()`, `sum()` 등

... 논리 내장 함수

- `all()`, `any()`