2021학년도 2학기 언어와 컴퓨터

제6강 함수와 모듈

박수지

서울대학교 인문대학 언어학과

2021년 9월 27일 월요일

오늘의 목표

1. List comprehension으로 리스트를 만들 수 있다.
2. `str.join()` 메서드로 문자열을 결합할 수 있다.
3. `import` 문을 사용하여 모듈이나 객체를 가져올 수 있다.
4. `def` 문을 사용하여 함수를 정의할 수 있다.
5. `return` 키워드의 작동 방식을 이해할 수 있다.
6. `try-except` 문으로 예외를 처리할 수 있다.
7. `raise` 키워드로 예외를 일으킬 수 있다.

List comprehension: 기본 사용법

```
1 [x for x in range(1, 11)]
```

특정한 조건을 만족시키는 항목만 뽑아서 리스트 만들기

```
1 [x for x in range(1, 11) if (10 % x) == 0]
```

열의 모든 항목에 대해 같은 작업을 반복하여 리스트 만들기

```
1 [2 * x + 1 for x in range(1, 11)]
```

https://docs.python.org/ko/3/tutorial/datastructures.html#list-comprehensions

형식

```
<구분자>.join(<문자열의 열>)
```

기본 사용법

```
'**'.join('배고파')
' '.join(['집에', '가고', '싶어'])
```

https://docs.python.org/ko/3/library/stdtypes.html#str.join

숫자나 글자만 뽑아서 연결하기

```
1 word = '배고파(...)'
2 ''.join(char for char in word if char.isalnum())
```

구구단 2단 만들기

```
1 print(' '.join(str(2*x) for x in range(1, 10)))
```

용어집

https://docs.python.org/ko/3/glossary.html

모듈 파이썬 코드의 조직화 단위를 담당하는 객체. 모듈은 임의의 파이썬 객체들을 담는 이름 공간을 갖습니다.

- IDLE의 “Run module”

임포팅 한 모듈의 파이썬 코드가 다른 모듈의 파이썬 코드에서 사용될 수 있도록 하는 절차.

import 문

https://docs.python.org/ko/3/reference/simple_stmts.html#import

모듈에서 객체 가져오기

```
1 from <모듈> import <이름>
```

모듈 자체를 가져오기

```
1 import <모듈>
```

sin, pi 가져오기

```
>>> from math import sin, pi
>>> sin(pi / 6)
0.49999999999999994
```

math 모듈 가져오기

```
>>> import math
>>> math.sin(math.pi / 6)
0.49999999999999994
```

이제는 0.5가 아니라도 놀라지 않아요.

내장 모듈 사용 예시

`timeit` 모듈의 `timeit` 함수로 코드 실행 시간 측정하기

```
>>> from timeit import timeit
>>> timeit("'ab'.lower()")
0.0951438939991931
>>> timeit("'a'.lower(); 'b'.lower()")
0.19719874600195908
```

길이 2인 문자열 한 개를 소문자로 바꾸는 것이 길이 1인 문자열 두 개를 소문자로 바꾸는 것보다 빠르다.

모듈의 이름 공간

random 모듈에 어떤 객체가 있는지 찾기

```
>>> import random
>>> dir(random)
['BPF', 'LOG4', 'NV_MAGICCONST', 'RECIP_BPF', 'Random', 'SG_MAGICC
ONST', 'SystemRandom', 'TWOPI', '_BuiltinMethodType', '_MethodType
', '_Sequence', '_Set', '__all__', '__builtins__', '__cached__', '
__doc__', '__file__', '__loader__', '__name__', '__package__', '__
spec__', '_acos', '_bisect', '_ceil', '_cos', '_e', '_exp', '_inst
', '_itertools', '_log', '_os', '_pi', '_random', '_sha512', '_sin
', '_sqrt', '_test', '_test_generator', '_urandom', '_warn', 'beta
variate', 'choice', 'choices', 'expovariate', 'gammavariate', 'gau
ss', 'getrandbits', 'getstate', 'lognormvariate', 'normalvariate',
 'paretovariate', 'randint', 'random', 'randrange', 'sample', 'see
d', 'setstate', 'shuffle', 'triangular', 'uniform', 'vonmisesvaria
te', 'weibullvariate']
```

용어집

https://docs.python.org/ko/3/glossary.html

함수 호출자에게 어떤 값을 돌려주는 일련의 문장들. 없거나 그 이상의 인자 가 전달될 수 있는데, 바디의 실행에 사용될 수 있습니다. 매개변수와 메서드와 함수 정의 섹션도 보세요.

매개변수 함수 (또는 메서드) 정의에서 함수가 받을 수 있는 인자 (또는 어떤 경우 인자들)를 지정하는 이름 붙은 엔티티.

인자 함수를 호출할 때 함수 (또는 메서드) 로 전달되는 값.

메서드 클래스 바디 안에서 정의되는 함수. 그 클래스의 인스턴스의 어트리뷰트로서 호출되면, 그 메서드는 첫 번째 인자 (보통 self 라고 불린다) 로 인스턴스 객체를 받습니다. 함수와 중첩된 스코프를 보세요.

함수 호출 예시

```
>>> int('1111', base=2)
15
```

함수	int
함수 호출	int(x='1111', base=2)
매개변수	x, base
인자	'1111', 2
반환값	15

주의

매개변수나 반환값이 없는 함수도 있다.

def 문

형식

```
def <함수명>(<매개변수>):
    <함수를 호출할 때 실행할 스위트>
    (return <반환할 표현식>)
```

예시

```
>>> def get_square_area(side):
...     return side ** 2
...
>>> get_square_area(5)
25
```

매개변수가 0개인 함수

```
1 def get_five():
2     return 5
3
4 a = get_five()
5 print(a)
```

반환값이 없는 함수

```
1 def hungry(x):
2     print('배고파!')
3
4 a = hungry(3)
5 print(a)
```

기존 프로그램을 함수로 고쳐 쓰기

영어 부정관사 고르기

기존 프로그램

```
# 입력
noun = input('명사를 입력하세요.')

# 계산
if noun.startswith(tuple('aeiou')):
    particle = 'an'
else:
    particle = 'a'

# 출력
print('{} {}'.format(particle, noun))
```

def 문

```
def get_particle(noun):
    '''영어 단어에 맞는 부정관사 반환'''
    if noun.startswith(tuple('aeiou')
        ):
        particle = 'an'
    else:
        particle = 'a'

    return particle
```

기존 프로그램을 함수로 고쳐 쓰기

문장 부호 제거하기

기존 프로그램

```
# 입력
text = input('텍스트를 입력하세요. ')

# 초기화
word = ''

# 계산
for char in text:
    if char.isalnum():
        word += char

# 출력
print(word)
```

def 문

```
def normalize(text):
    '''문장 부호 제거'''
    word = ''
    for char in text:
        if char.isalnum():
            word += char

    return word
```

기존 프로그램을 함수로 고쳐 쓰기

지키면 좋은 것들

- 한 가지 함수는 한 가지 작업만 처리하도록 한다.
  예시 부정관사를 붙이는 함수와 복수형을 만드는 함수는 따로 만들자.
- 함수의 이름은 snake_case로 짓는다.
  비교 camelCase
- def 문 스위트 첫 줄에 독스트링을 작성한다.
  효과 help() 함수로 도움말을 볼 수 있다.

의문

반환값을 여러 개 가질 수 있는가?

```
1 def example0():
2     return 0
3     return 1
4
5 print(example0())
```

return의 역할

실행을 멈추고 함수가 호출된 곳으로 돌아간다.

```python
def example1():
    print('출력됩니다.')
    return
    print('출력될까요?')

example1()
```

```python
def example2():
    for i in range(3):
        print(f'{i}번째 출력')
        return

example2()
```

if 절 내의 return

이후의 내용은 else 절 내에 있는 것과 같아진다.

```
1  def example3(n):
2      if n % 2:
3          print('홀수입니다.')
4          return
5      else:
6          print('짝수입니다.')
7      print('언제 출력될까요?')
8
9  example3(2)
10 example3(3)
```

```
1  def example4(n):
2      if n % 2:
3          return '홀'
4      return '짝'
5
6  print(example4(4))
7  print(example4(5))
```

사실 가장 간단한 홀짝 판정

```
1  def isodd(n):
2      return bool(n % 2)
```

예외 처리 구문

try/except

형식

```
try:
    <예외가 발생할 가능성이 있는 코드>
except:
    <예외가 발생했을 때 실행할 코드>
```

예외 처리 구문

예시: 예외 유형별로 처리하기

```
1  def myint(n):
2      try:
3          # n의 자료형과 값에 따라 예외가 발생할 수 있는 코드
4          return int(n)
5      except ValueError: # int('2.0')
6          print('올바른 정수 문자열을 입력하세요.')
7      except TypeError: # int([2])
8          print('올바른 자료형을 입력하세요.')
9      finally:
10         print('수고하셨습니다.')
```

예외 처리 구문

예시: 모든 예외를 한꺼번에 처리하기

```
1 def myint(n):
2     try:
3         return int(n)
4     except: # 임의의 예외
5         print('어쨌든 무엇인가가 잘못되었습니다.')
```

실행

```
>>> myint(2)
2
>>> myint('2.0')
어쨌든 무엇인가가 잘못되었습니다.
>>> myint([2])
어쨌든 무엇인가가 잘못되었습니다.
```

예외 발생 구문

try/except

형식

```
1 raise <예외 유형>(예외 메시지)
```

예외 유형 KeyError, TypeError, ValueError, ZeroDivisionError, ...

예외 발생 구문

예시: 홀수가 아니면 에러를 일으키는 함수

```
1  def odd_only(n):
2      if not isinstance(n, int):
3          raise TypeError('정수를 주세요!!')
4      elif n % 2:
5          print('좋아요!!')
6      else:
7          raise ValueError('홀수를 주세요!!')
```

`isinstance(5, int)` 5의 자료형이 int인지 판정하는 함수 호출

예외 발생 구문

예시: 홀수가 아니면 에러를 일으키는 함수

실행

```
>>> odd_only(3)
좋아요!!
>>> odd_only(3.0)
...
TypeError: 정수를 주세요!!
>>> odd_only(4)
...
ValueError: 홀수를 주세요!!
```

요약

List comprehension

`[<표현식> for <반복자> in <열> (if <조건>)]`

`str.join()`

`<구분자>.join(<열>)`

모듈에서 객체 가져오기

`from <모듈> import <객체>`

함수 정의하기

`def` 문

예외 처리하기

`try-except` 문

더 생각해 볼 것

- `zip()` 함수에 대하여 알아보자.

다음 시간에 할 것

- 정규표현식(regular expressions)
- SLP3 Ch.2 읽어 오기
  `https://web.stanford.edu/~jurafsky/slp3/2.pdf`