고려대학교 의료정보학과

KU MEDICINE IMDL Intelligent Medical Data Lab

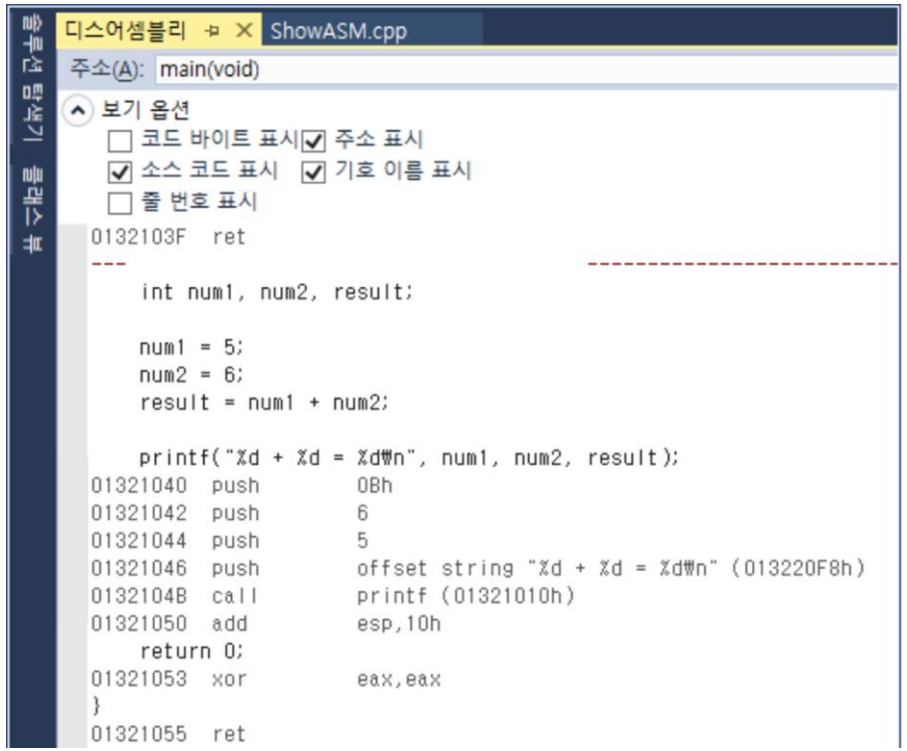
의료인공지능

파이썬 - 함수와 람다, 내장 함수, 클래스, 패키지

고려대학교 의료빅데이터연구소
채민수(minsuchae@korea.ac.kr)

1. 함수

- 함수란
  - 특정 작업을 수행하는 명령어들의 모음
  - 자주 하는 기능에서 사용되는 명령어의 모임



Reference : https://m.blog.naver.com/tipsware/221357889029

1. 함수

◦ 함수 정의

- def 키워드를 사용하여 함수명을 정의

```
def functionName():
    pass
```

- pass : 아무 작업도 하지 않음

- 소괄호() 사이에 파라미터를 추가하여 함수 호출하는 사용자에게 필요한 정보(parameter)를 받아 그에 대한 처리를 할 수 있음

```
def printUserName(name):
    print('Hello', name)
printUserName('채민수')
```

[실행결과]
Hello 채민수

1. 함수

- 함수 정의 실습
  - 두 개의 파라미터를 전달받아 그 연산결과의 합을 출력하여라

```
def printSum(num1, num2):
    print(num1+num2)
printSum(1,2)
```

  - 생각해보기

```
printSum(1,'2')
```

1. 함수

◦ 함수 결과 반환

- return : 값을 반환하거나 함수 종료

```
def functionNone():
    return
print(functionNone())
```

[실행결과]
None

- yield : 수행이 오래 걸리는 작업은 대기하기 보다 이를 효과적으로 활용하기 위함

```
def functionYield():
    yield 1
    yield 2
    yield 3
print(functionYield())
```

[실행결과]
<generator object functionYield at 0x7f3ca6d47450>

```
for v in functionYield():
    print(v)
```

[실행결과]
1
2
3

1. 함수

- 함수 값 반환 실습
  - 하나의 정수를 파라미터로 전달받아 FizzBuzz 출력
  - 3의 배수면 Fizz
  - 5의 배수면 Buzz
  - 15의 배수는 ?
  - 그 외는 주어진 정수 반환

1. 함수

◦ 함수 값 반환 실습

- 하나의 정수를 파라미터로 전달받아 FizzBuzz 출력

```
def FizzBuzz(num):
    if num % 15 == 0:
        return 'FizzBuzz'
    elif num % 3 == 0:
        return 'Fizz'
    elif num % 5 == 0:
        return 'Buzz'
    else:
        return num

for i in range(1, 16):
    print(FizzBuzz(i))
```

[실행결과]

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
```

1. 함수

◦ 함수에서 2개 이상 값 반환

- Tuple 을 통해 구현되어 있음

```
def minmax(num1,num2):
    return min(num1,num2), max(num1,num2)
print(minmax(1,3))
```

[실행결과]
(1, 3)

1. 함수

- 디폴트 파라미터(Default parameters)
  - 함수 정의 시 파라미터에 초기값 지정

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

Print *objects* to the text stream *file*, separated by *sep* and followed by *end*. *sep*, *end*, *file*, and *flush*, if present, must be given as keyword arguments.

All non-keyword arguments are converted to strings like `str()` does and written to the stream, separated by *sep* and followed by *end*. Both *sep* and *end* must be strings; they can also be `None`, which means to use the default values. If no *objects* are given, `print()` will just write *end*.

The *file* argument must be an object with a `write(string)` method; if it is not present or `None`, `sys.stdout` will be used. Since printed arguments are converted to text strings, `print()` cannot be used with binary mode file objects. For these, use `file.write(...)` instead.

Whether the output is buffered is usually determined by *file*, but if the *flush* keyword argument is true, the stream is forcibly flushed.

Changed in version 3.3: Added the *flush* keyword argument.

Reference : https://docs.python.org/3/library/functions.html#print

1. 함수

◦ 특수한 파라미터

- *args : Tuple 기반의 가변인자
- **kwargs : Dictionary 기반의 가변인자

`arguments` ¶

An ordered, mutable mapping (`collections.OrderedDict`) of parameters' names to arguments' values. Contains only explicitly bound arguments. Changes in `arguments` will reflect in `args` and `kwargs`.

Should be used in conjunction with `Signature.parameters` for any argument processing purposes.

> **Note:** Arguments for which `Signature.bind()` or `Signature.bind_partial()` relied on a default value are skipped. However, if needed, use `BoundArguments.apply_defaults()` to add them.

`args`

A tuple of positional arguments values. Dynamically computed from the `arguments` attribute.

`kwargs`

A dict of keyword arguments values. Dynamically computed from the `arguments` attribute.

Reference : https://docs.python.org/3.7/library/inspect.html?#inspect.BoundArguments.arguments

1. 함수

◦ 자료형에 따른 구분하여 처리

- isinstance 함수를 통하여 처리

```
def isPrimenumber(number):
    def _isPrime(number):
        if number <= 1:
            return False
        for i in range(2, number):
            if number % i == 0:
                return False
        else:
            return True

    if isinstance(number,int):
        return _isPrime(number)
    elif isinstance(number,list) or isinstance(number,tuple):
        results = []
        for v in number:
            if isinstance(v,int):
                results.append(_isPrime(v))
        return results
    else:
        return None
```

```
print(isPrimenumber(1))
```

[실행결과]
False

```
print(isPrimenumber([1,2,3,4,5]))
```

[실행결과]
[False, True, True, False, True]

1. 함수

- 자료형에 따른 구분하여 처리

```
def fit(self, X, y, sample_weight=None, check_input=True):

    random_state = check_random_state(self.random_state)

    check_scalar(
        self.ccp_alpha,
        name="ccp_alpha",
        target_type=numbers.Real,
        min_val=0.0,
    )

    if check_input:
        # Need to validate separately here.
        # We can't pass multi_output=True because that would allow y to be
        # csr.
        check_X_params = dict(dtype=DTYPE, accept_sparse="csc")
        check_y_params = dict(ensure_2d=False, dtype=None)
        X, y = self._validate_data(
            X, y, validate_separately=(check_X_params, check_y_params)
        )
        if issparse(X):
            X.sort_indices()

            if X.indices.dtype != np.intc or X.indptr.dtype != np.intc:
                raise ValueError(
                    "No support for np.int64 index based sparse matrices"
                )

    if self.criterion == "poisson":
        if np.any(y < 0):
            raise ValueError(
                "Some value(s) of y are negative which is"
                " not allowed for Poisson regression."
            )
        if np.sum(y) <= 0:
            raise ValueError(
                "Sum of y is not positive which is "
                "necessary for Poisson regression."
            )
```

Reference : https://github.com/scikit-learn/scikit-learn/blob/36958fb24/sklearn/tree/_classes.py#L155

1. 람다

◦ 스스로 해보기

- 다음 수식을 통해 피보나치 수열을 함수로 구현하라. 파라미터는 구하려는 항이며, 항은 0부터 시작함

$$y(x) = \begin{cases} 1 & x = 0 \\ 1 & x = 1 \\ y(x-2) + y(x-1) & else \end{cases}$$

```
def fib(num):




print(fib(2))
```

```
[실행결과]
3
```

2. 람다

◦ 람다란

- 주어진 수학 식을 계산하는 연산

- 1회성 함수를 만들거나 함축적으로 표현하려고 할 때 사용

- 람다 표현 방식

➢ lambda 인수: 반환값

```
my_func = lambda x, y : x*y
print(my_func(2,3))
```

[실행결과]
6

2. 람다

◦ 람다 실습

- 람다를 통한 리스트 내 덧셈 연산

```
import functools
print(functools.reduce(lambda x,y: x+y, [1,2,3,4,5]))
```

[실행결과]
10

- 람다를 통한 리스트 내 곱셈 연산

```
import functools
print(functools.reduce(lambda x,y: x*y, [1,2,3,4,5]))
```

[실행결과]
120

2. 람다

◦ 리스트 함축

- my_list = [value for_expression]
- 기존 코드 방식을 통해 리스트에 1부터 5까지 추가

```
a_list = []
for i in range(1,6):
    a_list.append(i)
print(a_list)
```

```
[실행결과]
[1, 2, 3, 4, 5]
```

- 람다를 통한 리스트에 1부터 5까지 추가

```
b_list = [v for v in range(1,6)]
print(b_list)
```

```
[실행결과]
[1, 2, 3, 4, 5]
```

2. 람다

◦ 리스트 함축

- 조건문을 통한 함축

```
my_list = [1,2,3,4,5,6,7,8,9,10]
new_list = [i for i in my_list if i%2==0]
print(new_list)
```

[실행결과]
[2, 4, 6, 8, 10]

- 조건문에 else 추가하여 함축

```
new_list2 = [i*i if i%2==0 else i for i in my_list]
print(new_list2)
```

[실행결과]
[1, 4, 3, 16, 5, 36, 7, 64, 9, 100]

2. 람다

◦ 스스로 해보기

- 람다를 활용하여 리스트의 최소값을 반환하도록 하여라

```
import functools
print(functools.reduce(lambda x,y:                    , [1,2,3,4,5,7,8,9,10]))
```

```
[실행결과]
1
```

- 람다를 통해 1부터 10까지 순회하며 홀수는 곱하기 3한 값을, 짝수는 곱하기 5한 값을 저장하여라

```
result = [                        v for v in range(1,11)]
print(result)
```

```
[실행결과]
[3,10,9,20,15,30,21,40,27,50]
```

3. 내장 함수

Built-in Functions

A	E	L	R
abs()	enumerate()	len()	range()
aiter()	eval()	list()	repr()
all()	exec()	locals()	reversed()
any()			round()
anext()	F	M	
ascii()	filter()	map()	S
	float()	max()	set()
B	format()	memoryview()	setattr()
bin()	frozenset()	min()	slice()
bool()			sorted()
breakpoint()	G	N	staticmethod()
bytearray()	getattr()	next()	str()
bytes()	globals()		sum()
		O	super()
C	H	object()	
callable()	hasattr()	oct()	T
chr()	hash()	open()	tuple()
classmethod()	help()	ord()	type()
compile()	hex()		
complex()		P	V
	I	pow()	vars()
D	id()	print()	
delattr()	input()	property()	Z
dict()	int()		zip()
dir()	isinstance()		
divmod()	issubclass()		_
	iter()		__import__()

Reference : https://docs.python.org/3/library/functions.html

3. 내장 함수

- print
  - 기본적인 I/O 기능으로 콘솔에 출력 (API 통신을 통해 구현)

- input
  - 기본적인 I/O 기능으로 키보드로부터 입력 (API 통신을 통해 구현)

- len
  - 리스트, 문자열 등의 길이를 반환

3. 내장 함수

- int
  - 주어진 파라미터를 정수형으로 형변환을 수행

- float
  - 주어진 파라미터를 실수형으로 형변환을 수행

- range
  - 튜플로 주어진 begin부터 end 미만까지 값을 반환

4. 클래스

◦ 클래스란

- 사용자 정의 타입
- 데이터와 관련된 함수를 묶음
- 클래스를 통해 객체를 생성

- class 키워드로 정의

```
class ClassName:
    pass
c = ClassName()
print(c)
```

[실행결과]

```
<__main__.ClassName object at 0x7fa02316b450>
```

4. 클래스

◦ 클래스 생성자

- def __init__ 함수로 정의

- 첫번째 파라미터는 self 로 고정

- 이 후 파라미터는 함수의 파라미터

4. 클래스

◦ 클래스 생성자 실습

```
class Counter:
    def __init__(self, start=1):
        self.now = start

    def current(self):
        return self.now

    def next(self):
        self.now = self.now + 1
c1 = Counter(11)
c1.next()
print(c1.current())
```

[실행결과]
12

4. 클래스

◦ 클래스 내 변수

- 객체 내 변수는 동일한 클래스의 다른 객체의 값과 구분

- 클래스 내 지역변수를 저장하기 위해선 객체를 가르키는 self 변수 사용

- 접근제어문이 없으므로 클래스 외부에서 객체 변수 접근이 가능함

4. 클래스

◦ 클래스 상속

- 기존의 클래스를 기능을 확장하여서 사용

- 다중 상속도 지원

```
class ClassNameChild(ClassName):
    def __init__(self):
        ClassName.__init__(self)
child = ClassNameChild()
print(child)
```

[실행결과]
<__main__.ClassNameChild object at 0x7fa023c188d0>

4. 클래스

◦ 클래스 상속

- 기존의 클래스를 기능을 확장하여서 사용

- 다중 상속도 지원

```
class ClassNameChild(ClassName):
    def __init__(self):
        ClassName.__init__(self)
child = ClassNameChild()
print(child)
```

[실행결과]
<__main__.ClassNameChild object at 0x7fa023c188d0>

4. 클래스

◦ 클래스 상속

```
1896  @keras_export("keras.callbacks.EarlyStopping")
1897  class EarlyStopping(Callback):
1898      """Stop training when a monitored metric has stopped improving.
1899
1900      Assuming the goal of a training is to minimize the loss. With this, the
1901      metric to be monitored would be `'loss'`, and mode would be `'min'`. A
1902      `model.fit()` training loop will check at end of every epoch whether
1903      the loss is no longer decreasing, considering the `min_delta` and
1904      `patience` if applicable. Once it's found no longer decreasing,
1905      `model.stop_training` is marked True and the training terminates.
1906
1907      The quantity to be monitored needs to be available in `logs` dict.
1908      To make it so, pass the loss or metrics at `model.compile()`.
```

Reference : https://github.com/keras-team/keras/blob/v2.10.0/keras/callbacks.py#L1896-L2061

4. 클래스

- 매직 메소드
  - 비교 연산자
    - __eq__, __gt__, __lt__
  - 산술 연산자
    - __add__, __sub__, __mult__, __truediv__, __mod__, __pow__
  - 리플렉션 메소드
    - __radd__, __rsub__, __rmul__
  - 단항 산술 연산자
    - __pos__, __neg__, __abs__, __round__, floor__, __ceil__, __trunc__
  - 비트 연산자
    - __and__, __or__, __lshift__

4. 클래스

- 매직 메소드
  - 복합 연산자
    - __iadd__, __isub__, __imul__
  - 변환 메소드
    - __int__, __float__, __complex__, __hex__, __oct__, __index__, __bool__
  - 컬렉션 관련 메소드
    - __len__, __getitem__, __setitem__, __delitem__, __contains__, __iter__, __next__
  - 객체 표현
    - __format__, __str__, __repr__

4. 클래스

◦ 스스로 해보기

- 학생의 학번, 이름, 점수를 저장하도록 한다. 단, 점수는 Dictionary 자료형을 사용하여 통해 국어점수, 수학점수, 영어점수를 저장하도록 한다. 또한, __str__ 매직 메소드를 활용하여 학생의 학번과 이름을 출력하도록 하여라.

```
class Student:
    pass
s1 = Student('20220000','아무개',{'국어':80, '영어':90, '수학':100})
print(s1.name,'평균은 ',s1.average(),'입니다.')
print(s1)
```

```
[실행결과]
아무개 평균은 90 입니다.
학번: 20220000, 이름 : 아무개
```

5. 패키지

◦ 개발자가 만들어놓은 모듈

- 특정 기능을 수행하기 위해 최소한의 라이브러리 배포 단위
  - ➢ csv : csv 파일처리를 하기 위한 라이브러리
  - ➢ imblearn : 데이터 불균형을 해결하기 위한 오버샘플링과, 언더샘플링을 제공하는 라이브러리. https://imbalanced-learn.org/stable/

- pip 명령어를 통하여 설치 가능