KU KONKUK UNIVERSITY

부록

자연어처리연구실

내장 함수

- **import 필요 없이 python에서 바로 사용할 수 있는 함수**

함수	설명
abs	abs(x)는 숫자값을 입력값으로 받았을 때, 그 숫자의 절대값을 돌려주는 함수
chr	정수 형태의 아스키코드값을 입력으로 받아서 그에 해당하는 문자를 출력하는 함수
dir	객체가 가지고 있는 변수나 함수를 리스트 형태로 보여준다.
divmod	두 개의 숫자를 입력값으로 받았을 때 그 몫과 나머지를 튜플의 형태로 반환하는 함수
enumerate	입력값으로 시퀀스자료형(리스트, 터플, 문자열)을 입력으로 받아 첫번째로 그 순서값, 두번째로 그 순서값에 해당되는 시퀀스 자료형의 실제값을 반환
eval	입력값으로 실행가능한 문자열(1+2, 'hi' + 'a' 같은 것)을 입력으로 받아서 문자열을 실행한 결과값을 반환하는 함수
filter	함수와 리스트를 입력으로 받아서 리스트의 값이 하나씩 함수에 인수로 전달될 때, 참을 반환시키는 값만을 따로 모아서 리스트의 형태로 반환하는 함수이다.
hex	입력으로 정수값을 받아서 그 값을 십육진수값(hexadecimal)로 변환하여 돌려주는 함수
id	객체를 입력값으로 받아서 객체의 고유값(레퍼런스)을 반환하는 함수

Edited by Harksoo Kim

내장 함수

함수	설명
input	사용자 입력을 받는 함수이다.
int	string 형태의 숫자나 소수점 숫자 등을 정수의 형태로 반환시켜 돌려준다.
isinstance	입력값으로 인스턴스와 클래스 이름을 받아서 입력으로 받은 인스턴스가 그 클래스의 인스턴스인지를 판단
lambda	함수를 생성할 때 사용되는 예약어로 def와 동일하나 보통 한줄로 간결하게 만들어 사용할 때 사용한다. def를 쓸 정도로 복잡하지 않거나 def를 쓸 수 없는 곳에 쓰인다.
len	인수로 시퀀스 자료형(문자열, 리스트, 터플)을 입력받아 그 길이(요소의 개수)를 돌려주는 함수
list	인수로 시퀀스 자료형을 입력받아 그 요소를 똑같은 순서의 리스트로 만들어 돌려주는 함수
map	함수와 시퀀스 자료형(리스트, 터플, 문자열)을 입력으로 받아서 시퀀스 자료형의 각각의 요소가 함수의 입력으로 들어간 다음 나오는 출력값을 묶어서 리스트로 돌려주는 함수
max	시퀀스 자료형(문자열, 리스트, 터플)을 입력받아 그 최대값을 돌려주는 함수
min	시퀀스 자료형을 입력받아 그 최소값을 돌려주는 함수
type	type(object)은 인수로 객체를 입력받아 그 객체의 자료형이 무엇인지 알려주는 함수.

Edited by Harksoo Kim

내장 함수

함수	설명
oct	정수 형태의 숫자를 8진수 문자열로 바꾸어 돌려주는 함수
open	파일 이름과 읽기 방법을 입력받아 파일 객체를 돌려주는 함수이다.
ord	문자의 아스키 값을 돌려주는 함수이다.
pow(x,y)	x의 y승을 한 결과값을 돌려주는 함수이다.
range	인수로 정수값을 주어 그 숫자에 해당되는 범위의 값을 list 형태로 돌려주는 함수
repr	객체를 출력할 수 있는 문자열 형태로 변환하여 돌려주는 함수 이 변환된 값은 주로 eval 함수의 입력으로 쓰인다.
sorted	입력으로 받은 시퀀스 자료형을 소트한 후 그 결과를 리스트로 리턴하는 함수이다.
str	str(object)은 객체를 출력할 수 있는 문자열 형태로 변환하여 돌려주는 함수이다.
tuple	tuple(sequence)은 인수로 시퀀스 자료형을 입력받아 터플 형태의 자료로 바꾸어 돌려준다.
zip	zip 함수는 동일한 갯수의 요소값을 갖는 시퀀스 자료형을 묶어주는 역할.

Edited by Harksoo Kim

내장 함수 예

- **abs**

```
>>> abs(3)
3
>>> abs(-3)
3
>>> abs(1+2j)
2.23606797749979
```

- **chr**

```
>>> chr(97)
'a'
>>> chr(84)
'T'
```

- **dir**

```
>>> dir([1,2,3])
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__', '__delslice__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getslice__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__', '__iter__', '__le__', '__len__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__setslice__', '__sizeof__', '__str__', '__subclasshook__', 'append', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']
>>> dir({'1':'a'})
['__class__', '__cmp__', '__contains__', '__delattr__', '__delitem__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__', '__hash__', '__init__', '__iter__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'has_key', 'items', 'iteritems', 'iterkeys', 'itervalues', 'keys', 'pop', 'popitem', 'setdefault', 'update', 'values', 'viewitems', 'viewkeys', 'viewvalues']
```


내장 함수 예

- **divmod**

```
>>> divmod(7,3)
(2, 1)
>>> divmod(7.3,0.3)
(24.0, 0.10000000000000009)
```

- **eval**

```
>>> eval('1+2')
3
>>> eval("'hi' + 'a'")
'hia'
>>> eval('divmod(4,3)')
(1, 1)
```

- **enumerate**

```
>>> for i, name in enumerate(['body', 'foo', 'bar']):
        print i, name


0 body
1 foo
2 bar
```

- **filter**

```
>>> def positive(l):
        result = []
        for i in l:
                if i > 0:
                        result.append(i)
        return result

>>> print positive([1, -3, 2, 0, -5, 6])
[1, 2, 6]
```

```
>>> def positive(x):
        return x>0

>>> print list(filter(positive, [1, -3, 2, 0, -5, 6]))
[1, 2, 6]
```


내장 함수 예

- **hex**

```
>>> hex(234)
'0xea'
>>> hex(3)
'0x3'
```

- **id**

```
>>> a = 3
>>> id(3)
30257160L
>>> id(a)
30257160L
>>> b = a
>>> id(b)
30257160L
```

- **input**

```
>>> a = input()
"hello"
>>> a
'hello'
>>> b = input("Enter: ")
Enter: "hi"
>>> b
'hi'
```

```
>>> a = raw_input()
hi
>>> a
'hi'
>>> b = raw_input("Enter: ")
Enter: hello
>>> b
'hello'
```

- **int**

```
>>> int('3')
3
>>> int(3.4)
3
>>> int('11', 2)  # 11이라는 이진수 값
3
>>> int('1A', 16)  # 1A이라는 16진수 값
26
```

- **isinstance**

```
>>> class Person: pass

>>> a = Person()
>>> b = 3
>>> isinstance(a, Person)
True
>>> isinstance(b, Person)
False
```


내장 함수 예

- **lambda**

```
>>> lambda 인수1, 인수2, ...  : 인수를 이용한 표현식
```

```
>>> sum = lambda a,b: a+b
>>> sum(3,4)
7
```

=

```
>>> def sum(a, b):
        return a + b

>>> sum(3,4)
7
```

```
>>> l = [lambda a,b: a+b, lambda a,b: a*b]   # lambda 함수를 list에 삽입
>>> l
[<function <lambda> at 0x000000000290B3C8>, <function <lambda> at 0x000000000290B4A8>]

>>> l[0]
<function <lambda> at 0x000000000290B3C8>   # 위 list의 첫 번째 원소에 있는 lambda함수

>>> l[0](3,4)
7
```

sum

```
>>> l[1](3,4)
12
```

mul

내장 함수

- **len**

```
>>> len("python")
6
>>> len([1,2,3])
3
>>> len((1, 'a'))
2
```

- **list**

```
>>> list("python")
['p', 'y', 't', 'h', 'o', 'n']
>>> list((1,2,3))
[1, 2, 3]
>>> a = [1,2,3]
>>> b = list(a)
>>> b
[1, 2, 3]
```

```
>>> id(a)
42151048L
>>> id(b)
42132936L
```

a와 b의 id 값이 서로 다름.
a를 b에 복사하기 때문

내장 함수 예

- **map**

```
>>> def two_times(l):
        result = []
        for i in l:
            result.append(i*2)
        return result

>>> result = two_times([1,2,3,4])
>>> print result
[2, 4, 6, 8]
```

map 적용

```
>>> def two_times(x): return x*2

>>> list(map(two_times, [1,2,3,4]))
[2, 4, 6, 8]
```

여기에 lambda 적용

```
>>> list(map(lambda a: a*2, [1,2,3,4]))
[2, 4, 6, 8]
```

[map을 이용해서 리스트의 값을 1씩 추가하는 예]

```
>>> def plus_one(x):
        return x + 1

>>> print list(map(plus_one, [1,2,3,4,5,6]))
[2, 3, 4, 5, 6, 7]
```


내장 함수 예

- **max**

```
>>> max([1,2,3])
3
>>> max("python")
'y'
```

- **min**

```
>>> min([1,2,3])
1
>>> min("python")
'h'
```

- **oct**

```
>>> oct(34)
'042'
>>> oct(123456)
'0361100'
```

- **open**

모드	설명
w	쓰기 모드로 파일 열기
r	읽기 모드로 파일 열기
a	추가 모드로 파일 열기
b	바이너리 모드로 파일 열기

```
>>> fwrite = open("write_mode.txt", 'w')
>>> fread = open("write_mode.txt", 'r')
>>> fread2 = open("write_mode.txt")
>>> f = open("write_mode.txt", 'rb')
>>> fappend = open("write_mode.txt", 'a')
```


내장 함수 예

- **ord**

```
>>> ord('a')
97
>>> ord('0')
48
```

- **pow**

```
>>> pow(2,4)
16
>>> pow(9,4)
6561
```

- **range**

```
>>> list(range(5))
[0, 1, 2, 3, 4]
>>> list(range(5, 10))
[5, 6, 7, 8, 9]
>>> list(range(5, 10,2))
[5, 7, 9]
>>> list(range(0, -10, -1))
[0, -1, -2, -3, -4, -5, -6, -7, -8, -9]
```

- **repr**

```
>>> repr("hi".upper())
"'HI'"
>>> eval(repr("hi".upper()))
'HI'
>>> eval(str("hi".upper()))

Traceback (most recent call last):
  File "<pyshell#150>", line 1, in <module>
    eval(str("hi".upper()))
  File "<string>", line 1, in <module>
NameError: name 'HI' is not defined
```

- **sorted**

```
>>> sorted([3,1,2])
[1, 2, 3]
>>> sorted(['a','c','b'])
['a', 'b', 'c']
>>> sorted("zero")
['e', 'o', 'r', 'z']
>>> sorted((3,2,1))
[1, 2, 3]
```

내장 함수 예

- **str**

```
>>> str(3)
'3'
>>> str('hi')
'hi'
>>> str('hi'.upper())
'HI'
```

- **tuple**

```
>>> tuple("abc")
('a', 'b', 'c')
>>> tuple([1,2,3])
(1, 2, 3)
>>> tuple((1,2,3))
(1, 2, 3)
```

- **type**

```
>>> type("abc")
<type 'str'>
>>> type([])
<type 'list'>
>>> type(open("test", 'w'))
<type 'file'>
```

- **zip**

```
>>> list(zip([1,2,3], [4,5,6]))
[(1, 4), (2, 5), (3, 6)]
>>> list(zip([1,2,3], [4,5,6], [7,8,9]))
[(1, 4, 7), (2, 5, 8), (3, 6, 9)]
>>> list(zip("abc", "dgf"))
[('a', 'd'), ('b', 'g'), ('c', 'f')]
>>> list(zip("abc", "dgfa"))
[('a', 'd'), ('b', 'g'), ('c', 'f')]
>>> list(zip("abcfffff", "dgfa"))
[('a', 'd'), ('b', 'g'), ('c', 'f'), ('f', 'a')]
```