



..... Data Processing Based Python

SECT4. 데이터형 (Data Type)

Technical Competency for BigData Course

김진수



- ◆ 숫자형, Numeric Type
- ◆ 논리형, Boolean Type
- ◆ 문자열형, String Type
- ◆ 데이터형 에러, Data Type Error
- ◆ 데이터형 변환, Data Type Converting
- ◆ 변할 수 없는(Immutable) vs. 변할 수 있는(Mutable)



❖ 데이터형이란?

- 프로그래밍 언어를 사용하여 데이터의 공통된 특징과 용도에 따라 분류하여 정의한 것

❖ 파이썬의 기본적인 데이터형 3가지

- 숫자형, Numeric Type
- 논리형, Boolean Type
- 문자열형, String Type

A person is shown from the chest up, standing in a server room. They have their hands clasped in front of their face. A large, glowing red eye graphic is superimposed over the left side of the image, with the text 'FACE RECOGNITION' and 'AI' visible. The server racks in the background are illuminated with blue light.

- 데이터를 더하고, 빼고, 곱하고, 나눌 수 있는 데이터형

A screenshot of a Windows-style application window titled "Python 3.4.4 Shell". The window has a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area shows a command prompt where several ">>>" prompts are visible. Two lines of output are shown, each preceded by "=====" and followed by "=====":
1. "RESTART: Shell"
2. "RESTART: C:/Python34/sector_04.py"
Below the second restart line, a series of arithmetic operations are performed and displayed:
`x = 34`
`y = 5`
`x + y = 39`
`x - y = 29`
`x * y = 170`
`x / y = 6.8`
`x // y = 6`
`x % y = 4`
`-x = -34`
`+x = 34`
`x ** y = 45435424`
Following these calculations, there are more ">>>" prompts. At the bottom right of the window, the status bar displays "Ln: 148 Col: 0".



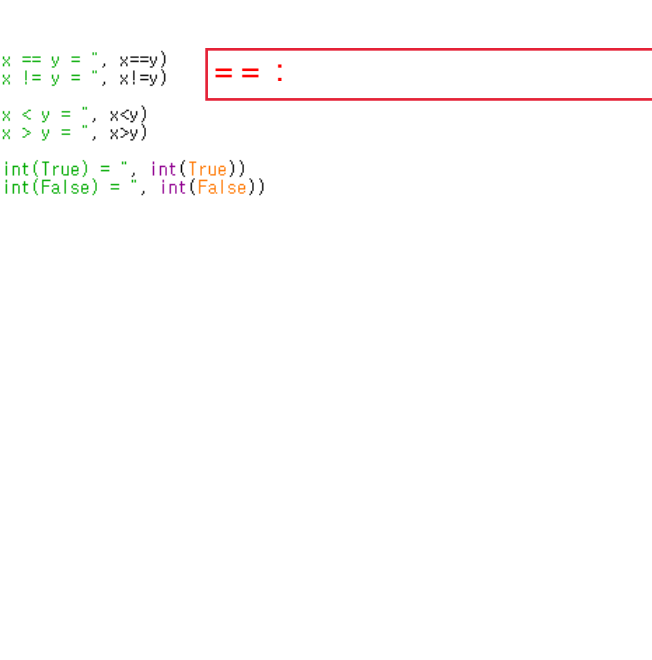
❖ 연산 기호 정리

Operation	Result	Priority	Full documentation
$x + y$	sum of x and y		
$x - y$	difference of x and y		
$x * y$	product of x and y		
x / y	quotient of x and y		
$x // y$	floored quotient of x and y		
$x \% y$	remainder of x / y		
$-x$	x negated		
$+x$	x unchanged		
<code>abs(x)</code>	absolute value or magnitude of x		abs()
<code>int(x)</code>	x converted to integer		int()
<code>float(x)</code>	x converted to floating point		float()
<code>complex(re, im)</code>	a complex number with real part re , imaginary part i m . im defaults to zero.		complex()
<code>c.conjugate()</code>	conjugate of the complex number c		
<code>divmod(x, y)</code>	the pair $(x // y, x \% y)$		divmod()
<code>pow(x, y)</code>	x to the power y		pow()
$x ** y$	x to the power y		

A person in a dark jacket is shown from the chest up, holding a glowing, translucent sphere in their hands. The background is a server room with rows of server racks and blue lighting. In the upper left corner, there is a circular graphic with a red and white target-like pattern and some text that is partially obscured.

◆ 논리형

- 가 .



The screenshot shows a Python IDE window titled "sector_04.py - C:/Python34/sector_04.py (3.4.4)". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code in the editor is as follows:

```
x = 4
y = 9

print("x == y = ", x==y)
print("x != y = ", x!=y)

print("x < y = ", x<y)
print("x > y = ", x>y)

print("int(True) = ", int(True))
print("int(False) = ", int(False))
```

A red rectangular box highlights the text "==" in the first print statement, indicating the equality comparison operator.

Ln: 12 Col: 1

[illegible]

비교연산자와 논리연산자



❖ 비교연산자

비교연산자	의미
<	strictly less than
<=	less than or equal
>	strictly greater than
>=	greater than or equal
==	equal
!=	not equal
is	object identity : (ex) :
is not	negated object identity



❖ 논리연산자

논리연산자	결과	참고사항
<code>x or y</code>	if <code>x</code> is false, then <code>y</code> , else <code>x</code>	<code>x</code> 가 거짓인 경우에만 <code>y</code> 수행
<code>x and y</code>	if <code>x</code> is false, then <code>x</code> , else <code>y</code>	<code>x</code> 가 참인 경우에만 <code>y</code> 수행
<code>not x</code>	if <code>x</code> is false, then True, else False	논리연산자가 아닌 연산자에 비해 우선순위가 낮음. 다른 연산자와 함께 사용하는 경우 주의바람



❖ 연산자 우선순위, Operators Precedence

- 가장 높은 우선 순위에서 가장 낮은 모든 연산자를 보여준다.

Operator	Description
**	지수 (전원으로 인상)
~ + - ~():	complement, 단항 플러스와 마이너스 (마지막 두의 메서드 이름은 + @이며, - @)
* / % //	곱하기, 나누기, 나머지, 몫
+ -	덧셈과 뺄셈
>> <<	좌우 비트 시프트
&	비트 'AND'
^	비트 전용 'OR'와 정기적 인 'OR'
<= < > >=	비교 연산자
<> == !=	평등 연산자 <> (3. .)
= %= /= //= -= += *= **=	할당 연산자
is is not	식별 연산자
in not in	멤버 연산자
not or and	논리 연산자



❖ 문자열형

- 데이터가 여러 문자로 구성되어 있고 다른 문자와 연결될 수 있으며 데이터에 포함된 문자열의 길이를 확인할 수 있는 데이터형
- 파이썬에서 문자열 데이터형은 str
- 입력하고 하는 문자열을 홑따옴표(')로 감싸거나 쌍따옴표(")로 감싸면 문자열로 인식
- 여러줄인 경우에는 세 개의 홑따옴표('')나 세 개의 (""")로 감싸면 된다.

문자열형 관련 유용한 함수



❖ 실습 예제

```
test = '파이썬 프로그래밍 재미있다!' # 문자열을 변수에 저장

result = test.startswith('파이썬') # 문자열이 '파이썬'으로 시작하는지 확인
print(result)
result = test.endswith('!') # 문자열이 '!'로 끝나는지 확인
print(result)
result = test.endswith('어려워요!') # 문자열이 '어려워요!'로 끝나는지 확인
print(result)
result = test.replace('파이썬', 'Python') # 문자열중 '파이썬'을 'Python'으로 변경
print(result)
```

```
True
True
False
Python 프로그래밍 재미있다!
```

```
test = 'Python Programming is Interesting!'

result = test.upper() # 문자열을 모두 대문자로 변경
print(result)
result = test.lower() # 문자열을 모두 소문자로 변경
print(result)
result = '/'.join(test) # 문자열의 각 문자 사이에 '/' 문자 집어 넣기
print(result)
```

```
PYTHON PROGRAMMING IS INTERESTING!
python programming is interesting!
P/y/t/h/o/n/ /P/r/o/g/r/a/m/m/i/n/g/ /i/s/ /I/n/t/e/r/e/s/t/i/n/g/!
```

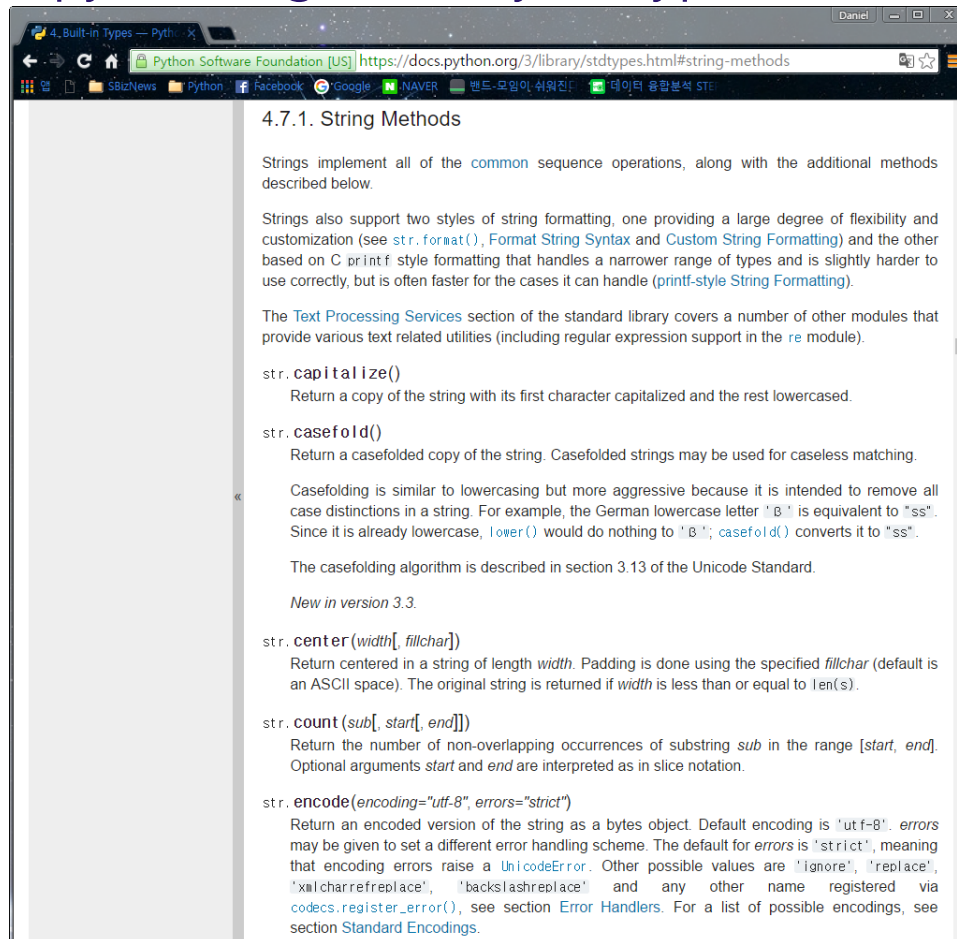
str (.)

파이썬이 제공하는 문자열 함수



❖ 파이썬의 문자열 함수

- <https://docs.python.org/3/library/stdtypes.html#string-methods>



데이터형 에러, Data Type Error



- ❖ 파이썬의 기본 내장 함수인 `type()`로 데이터형 확인 가능하다.
- ❖ 데이터형을 섞어 쓰다 보면 예상치 못한 에러를 볼 수 있다.
- ❖ 에러 메시지 분석

```
data_exam.py - C:/Python34/data_exam.py (3.4.4)
File Edit Format Run Options Window Help
num = 69
str = '이십육'

sum = num + str
print(sum)
Ln: 6 Col: 0
```



```
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
>>>
>>>
>>>
===== RESTART: C:/Python34/data_exam.py =====
Traceback (most recent call last):
  File "C:/Python34/data_exam.py", line 4, in <module>
    sum = num + str
TypeError: unsupported operand type(s) for +: 'int' and 'str'
>>>
Ln: 25 Col: 4
```



❖ 파이썬의 데이터 선언은 동적 타이핑(Dynamic Typing)

- 파이썬에서는 변수를 선언할 때 이 변수의 데이터형이 무엇인지 표기하지 않는다.
- 자바나 C++과 같은 언어를 사용했다면 반드시 변수명 앞에 데이터형을 표기해야 한다.

❖ 동적타이핑의 장점

- 변수의 사용을 자유롭게 하며
- 개발자에게 프로그램 작성의 자유도를 높여준다.

❖ 동적타이핑의 단점

- 프로그램 실행 시 데이터형 오류가 발생하는 문제
- 이런 오류는 프로그램을 실행하기 전에는 확인할 수 있는 방법이 없다.



❖ 실습 예제 : 문자열형 → 숫자형

```
num_data = 350  
str_data = '350'  
  
sum = int(str_data) + num_data  
print('합계는? ', str(sum))
```

합계는? 700

Immutable vs Mutable



- ❖ **Immutable** : 변할 수 없는 데이터
- ❖ **Mutable** : 변할 수 있는 데이터

변할 수 없는(Immutable)	변할 수 있는(Mutable)
숫자형(numbers)	리스트형(list)
문자열형(string)	사전형(dict)
튜플형(tuple)	집합형(set)
불변집합형(frozenset)	바이트배열형(byte array)
바이트형(bytes)	

Immutable vs Mutable 예제



❖ 실습 예제

```
# Immutable 예제
hello = '안녕하세요!' # hello 문자열형 변수 선언

print(hello) # hello 값 확인
print(id(hello)) # hello 객체 식별자 확인

hello = '반갑습니다!' # hello 값 변경

print(hello) # hello 값 확인
print(id(hello)) # hello 객체 식별자 확인
```

```
안녕하세요!
2185373891640
반갑습니다!
2185373892520
```

```
# Mutable 예제
hello_list = ['안녕하세요!'] # 리스트형 선언

print(hello_list) # 리스트 값 확인
print(id(hello_list)) # 리스트 객체 식별자 확인

hello_list[0] = '반갑습니다!' # 리스트 첫번째 항목 값 변경하기

print(hello_list) # 리스트 값 확인
print(id(hello_list)) # 리스트 객체 식별자 확인
```

```
['안녕하세요!']
2185371662600
['반갑습니다!']
2185371662600
```

가 .



❖ 데이터형의 개념

❖ 파이썬의 기본적인 데이터형

가

.

- 숫자형, Numeric Type
- 논리형, Boolean Type
- 문자열형, String Type

❖ 데이터형의 차이로 인해 발생한 에러 메시지 출력 사례

❖ 에러 메시지를 읽는 방법

❖ 에러를 해결하기 위한 데이터형 변환 방법

❖ 변환 수 없는 데이터형과 변환 수 있는 데이터형 구분

- 변환 수 없는 (Immutable)
- 변환 수 있는 (Mutable)