“DS/AI 프로그래밍”

2주차
python 기초

J.-K.- Seo
데이터과학원

Numbers

```
>>> 17 / 3  # classic division returns a float
5.666666666666667
>>>
>>> 17 // 3  # floor division discards the fractional part
5
>>> 17 % 3  # the % operator returns the remainder of the division
2
>>> 5 * 3 + 2  # floored quotient * divisor + remainder
17
```

```
>>> 5 ** 2  # 5 squared
25
>>> 2 ** 7  # 2 to the power of 7
128
```

```
>>> '"Isn\'t," they said.'
'"Isn\'t," they said.'
>>> print('"Isn\'t," they said.')
"Isn't," they said.
>>> s = 'First line.\nSecond line.'  # \n means newline
>>> s  # without print(), \n is included in the output
'First line.\nSecond line.'
>>> print(s)  # with print(), \n produces a new line
First line.
Second line.
```

Lists

```
>>> squares = [1, 4, 9, 16, 25]
>>> squares
[1, 4, 9, 16, 25]

>>> squares[0]  # indexing returns the item
1
>>> squares[-1]
25
>>> squares[-3:]  # slicing returns a new list
[9, 16, 25]
```

While

```
>>> while a < 10:
...     print(a)
...     a, b = b, a+b
...
```

IF

```
>>> x = int(input("Please enter an integer: "))
Please enter an integer: 42
>>> if x < 0:
...     x = 0
...     print('Negative changed to zero')
... elif x == 0:
...     print('Zero')
... elif x == 1:
...     print('Single')
... else:
...     print('More')
...
```

For

```
>>> # Measure some strings:
... words = ['cat', 'window', 'defenestrate']
>>> for w in words:
...     print(w, len(w))
...
cat 3
window 6
defenestrate 12
```

range

```
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
```

break

```
>>> for n in range(2, 10):
...     for x in range(2, n):
...         if n % x == 0:
...             print(n, 'equals', x, '*', n//x)
...             break
...     else:
...         # loop fell through without finding a
factor
...         print(n, 'is a prime number')
...
```

def

```
>>> def fib(n):    # write Fibonacci series up to n
...     """Print a Fibonacci series up to n."""
...     a, b = 0, 1
...     while a < n:
...         print(a, end=' ')
...         a, b = b, a+b
...     print()
...
```

Lambda

```
>>> def make_incrementor(n):
...     return lambda x: x + n
...
>>> f = make_incrementor(42)
>>> f(0)
42
>>> f(1)
43
```

Class

- 다른 언어와 동일하게 python에서도 클래스를 이용하여 보다 편리하게 함수를 다룰 수 있다. 참고로 클래스는 데이터나 처리의 정의 등을 하나로 정리해둔 모형과 같은 것이다.

  출처: https://engineer-mole.tistory.com/190 [매일 꾸준히, 더 깊이]

```
def f(self):
    return 'hello world'
```

```
class MyClass:
    i = 12345
    def f(self):
        return 'hello world'
```

인스턴스 생성 → 클래스 내의 함수(메소드) 호출 가능

```
x = MyClass()
>>> x.f()
```

```
class Dog:
    def __init__(self, name):
        self.name = name
        self.tricks = [] # creates a new empty list for each dog
    def add_trick(self, trick):
        self.tricks.append(trick)
```

```
>>> class Tesla:
...     #creating a class variable and making it a global variable
...     global speed
...     speed = 60
...     print("Acessing speed variable within the class:", speed)
...     def __init__(self, speed):
...         self.speed = speed
...     def display_speed(self):
...         print("Speed of the Tesla is:", self.speed)
...
```

```
class Bag:
    def __init__(self):
        self.data = []

    def add(self, x):
        self.data.append(x)

    def addtwice(self, x):
        self.add(x)
        self.add(x)
```

```
# Function defined outside the class
def f1(self, x, y):
    return min(x, x+y)

class C:
    f = f1

    def g(self):
        return 'hello world'

    h = g
```

The End

References

- https://docs.python.org/3/tutorial/
- https://docs.python.org/3/tutorial/introduction.html#numbers
- https://docs.python.org/3/tutorial/controlflow.html#match-statements
- https://docs.python.org/3/tutorial/classes.html#class-and-instance-variables
- https://engineer-mole.tistory.com/190
- https://www.delftstack.com/ko/howto/python/define-a-class-global-variable-in-python/