요약

변수

변수 = 값

```
a = 3
print(a) #3
```

숫자

정수, 실수

형변환: int( ), float( )

연산자: +, -, *, /, //, %, **

복합 대입연산자

```
a = 10
a += 1
print(a) #11
```

불

★True, False

조건문의 반환값으로 사용됨

<, <=, >, >=, ==, !=

and, or, not

문자열

‘ ’, “ ”

불변 시퀀스

문자열 선택

```
t = '가나다라마바사'
t[0]   # 가
t[1:3] #나다
t[0:5:2] #가다마
```

연산자: +, *

in, not in

len

format, lower, upper, strip, find, split, join, replace, …

리스트

[ ]

형변환: list( )

★모든 것의 시퀀스, 변경 가능

리스트 선택

```
t = [10, 20, 30, 40, 50]
t[0]   # 10
t[1:3] # [20, 30]
t[0:5:2] #[10, 30, 50]
```

연산자: +, *

in, not in

len

append, extend, insert,

pop, remove, count, sort

튜플

( )

형변환: tuple( )

★모든 것의 시퀀스, 변경 불가능

연산자: +, *

in, not in

len

튜플 선택

```
t = (10, 20, 30, 40, 50)
t[0]   # 10
t[1:3] # (20, 30)
t[0:5:2] #(10, 30, 50)
```

딕셔너리

{key:value}

dict()

★특징: key로 value를 찾음, 변경 가능

★dict[key], get

dict[key] = value, update

in, not in

len

items, keys, values

del, clear

셋

{요소}

set()

★특징: 중복을 허용하지 않음, 변경 가능, 순서 없음

add, update, remove

in, not in

len

intersection, union, difference

if

if 조건문 조건에 따라 코드를 실행하거나 실행하지 않게 만들고 싶을 때 사용

```
if 조건문:
    조건이 참일 때 실행할 문장
```

```
if 조건문:
    조건이 참일 때 실행할 문장
else:
    조건이 거짓일 때 실행할 문장
```

```
if 조건문1:
    실행문장1
elif 조건문2:
    실행문장2
else:
    실행문장3
```

while

while문 조건문이 참인 동안에 while문 아래의 문장이 반복되서 실행됨

```
while 조건문:
    조건문이 참일 때 실행할 문장1
    조건문이 참일 때 실행할 문장2
```

while

break 강제로 빠져나올 때

continue 현재 반복을 생략하고 다음 반복으로 넘어갈 때 사용

```
while True:
    x = input('단어를 입력하세요 ')
    if x == 'end':
        break
    print('입력:', x)
```

단어를 입력하세요

```
while True:
    x = input('정수입력하기(종료:q)')
    if x == 'q':
        break
    num = int(x)
    if num % 2 != 0:
        continue
    print(num, 'squared is', num*num)
```

정수입력하기(종료:q)

for

for 데이터의 첫번째 요소부터 마지막 요소까지 차례대로 변수에 대입되어 실행문장1, 실행문장2 등이 실행됨.

```
for 변수 in 데이터:
    실행문장1
    실행문장2
```

for

break 강제로 빠져나올 때

continue 현재 반복을 생략하고 다음 반복으로 넘어갈 때 사용

```
list_a = ['a', 'b', 'c', 'd', 'end', 'e', 'f']
for t in list_a:
    if t == 'end':
        break
    print(t)
```

```
a
b
c
d
```

```
scores = [10, 30, 120, 20, 60, 90, 110]
for s in scores:
    if s > 100:
        continue
    print(s,'점입니다. 100점을 위해서', 100-s,'점이 더 필요합니다.')
```

```
10 점입니다. 100점을 위해서 90 점이 더 필요합니다.
30 점입니다. 100점을 위해서 70 점이 더 필요합니다.
20 점입니다. 100점을 위해서 80 점이 더 필요합니다.
60 점입니다. 100점을 위해서 40 점이 더 필요합니다.
90 점입니다. 100점을 위해서 10 점이 더 필요합니다.
```

for

range(시작, 끝, 간격)

zip(iter1, iter2, …)

```
for i in range(2, 10):
    for j in range(1, 10):
        print('{:2}'.format(i*j), end=' ')
    print('')
```

```
names = ['YJ', 'SG', 'YM', 'JG', 'SJ', 'HD']
scores = [10, 20, 50, 20, 30, 40 ]
```

```
for n, s in zip(names, scores):
    print(n, ':', s)
```

함수

```
def func(text, n):
    for i in range(n):
        print(text)
func('hi', 5)
```

```
hi
hi
hi
hi
hi
```

```
def str_times(string, n):
    return string*n
```

```
print(str_times('안녕',3))
```

```
안녕안녕안녕
```

함수

지역변수, 전역변수

```
x = 100
def f():
    x = 200
f()
print(x)
```

```
100
```

Numpy

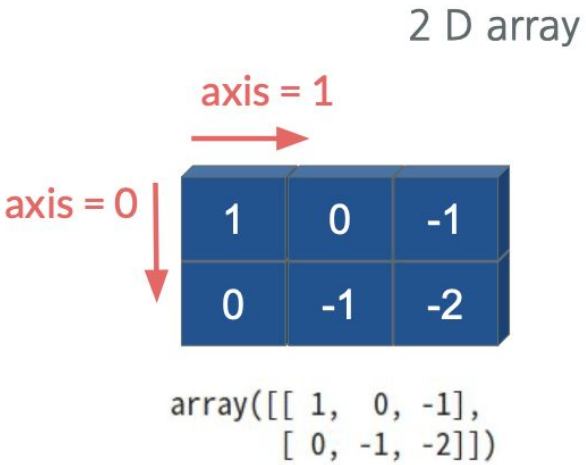
수치계산용 라이브러리

ndarray

수학, 통계 함수, 선형대수, …

Numpy

★ndarray.shape



- size: 6
- shape: (2, 3)
- ndim: 2

Numpy

ndarray 생성

- np.array()
- np.zeros, ones, full, …
- np.random.rand, randn, randint, ...

ndarray shape

- flatten()
- ★ reshape()
- T, transpose()

Numpy

인덱싱, 슬라이싱

```
arr2d = np.arange(20).reshape(2, -1)
arr2d
```

```
array([[ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14, 15, 16, 17, 18, 19]])
```

```
arr2d[1][0]
```

```
10
```

```
arr2d[1, 0]
```

```
10
```

2. slicing

```
arr2d
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19]])
```

```
arr2d[:3][:2]
```

```
array([[0, 1, 2, 3, 4],
       [5, 6, 7, 8, 9]])
```

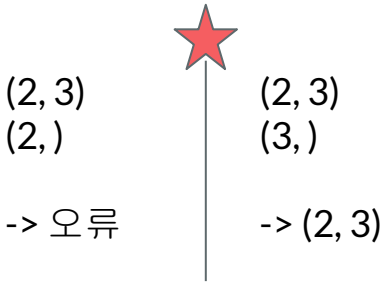
```
arr2d[:3, :2]
```

```
array([[ 0,  1],
       [ 5,  6],
       [10, 11]])
```

Numpy

브로드 캐스팅 규칙

1. 차원이 작은 쪽의 왼쪽이 1로 채워짐
2. 차원은 같은데 shape이 다르고, 그게 1이면 브로드 캐스팅 일어남
3. 차원은 같은데 shape이 다르고, 그 값이 다르면 오류



참고: 브로드캐스팅 규칙

Numpy

유니버셜 함수

abs, sqrt, exp, sign, …

add, subtract, power, …

통계 메서드

sum, mean, std, var, …

Numpy

any

all

where(조건, T일때, F일때)

sort

선형대수 함수

Pandas

★표 형식의 데이터

데이터 전처리, 분석용

★ Series Data Frame

	Date	kospi	kosdaq	gold_fut_132030	Bond_273130
0	2020. 1. 2 오후 3:30:00	2175.17	674.02	10845	108215
1	2020. 1. 3 오후 3:30:00	2176.46	669.93	11000	108565
2	2020. 1. 6 오후 3:30:00	2155.07	655.31	11245	108745
3	2020. 1. 7 오후 3:30:00	2175.54	663.44	11180	108400
4	2020. 1. 8 오후 3:30:00	2151.31	640.94	11360	108270
5	2020. 1. 9 오후 3:30:00	2186.45	666.09	11055	107980
6	2020. 1. 10 오후 3:30:00	2206.39	673.03	11035	107760
7	2020. 1. 13 오후 3:30:00	2229.26	679.22	11080	107695
8	2020. 1. 14 오후 3:30:00	2238.88	678.71	10975	107860

Pandas

★데이터 읽고, 쓰기

read_csv/ to_csv

read_json/ to_json

series indexing/ slicing

s[i]

s[label]

s.iloc[i]

s.loc[label]

Pandas

DataFrame indexing/slicing

df[열]

df[행:행]

★df.loc[행, 열] : 라벨

★df.iloc[행, 열] : 정수 위치

Pandas

산술연산

add, sub, mul, div, pow, …

통계

describe, count, min, max, sum, mean, var, std, …

unique, value_counts

Pandas

```
df
```

	0	1	2	3	4
0	0	1	2	3	4
1	5	6	7	8	9

```
df.mean(axis=0)
```

```
0    2.5
1    3.5
2    4.5
3    5.5
4    6.5
dtype: float64
```

```
df.mean(axis=1)
```

```
0    2.0
1    7.0
dtype: float64
```

Pandas

Series		DataFrame	
★apply(함수)	원소별	applymap(함수)	원소별
map(함수)	원소별	apply(함수)	축별로

Pandas

삭제: drop

병합: concat, merge

isnull, notnull

dropna, fillna

duplicated

drop_duplicated

Pandas

replace

cut, qcut

get_dummies

★groupby

.get_group

.agg

.filter

Python

- 자료구조의 특징

(int, float, bool, string,

list, tuple, dictionary, set)

- if, for, while 사용법
- function, class 사용법

Numpy

- numpy 특징
- shape
- indexing/ slicing
- 산술연산
  브로드캐스팅

Pandas

- pandas 특징
- pandas 자료구조
- loc, iloc
- axis =0, 1
- Series.apply
- groupby