
▣ Ch03 데이터 분석에 필요한 연장 챕기기

[03-1] 변하는 수, '변수' 이해하기

변수(Variable)

> 상수 값 대신에 이름으로 값을 대신해 사용하는 방식

○ Python 자료 구조(변수) 종류

[내장 자료형]

>> 스칼라(Scalar) 형 : int, float, str, bool 등

>> 비스칼라형 : list, tuple, dictionary 등

[외장 자료형] Pandas 자료형

>> 시리즈(Series) 형 : 1차원 나열형 자료

>> 데이터 프레임(Data Frame) 형 : 2차원 나열형 자료

In []:

[]스칼라(Scalar) 형

> 하나의 값 만으로 구성된 자료 구조

>> int, float, str, bool 등

```
In [166...]:  
## 스칼라(Scalar) 형 ##  
a, b, c, d = 3, 3.15, 'hello, world!', bool(True)  
print(type(a))  
print(type(b))  
print(type(c))  
print(type(d))  
  
<class 'int'>  
<class 'float'>  
<class 'str'>  
<class 'bool'>
```

<> 자료형의 메서드 사용하기

> Python에서는 자료구조가 Class 형태로 만들어져 관리가 되므로

> 자료형의 메서드를 사용할 수 있다.

```
In [167...]: c = 'hello, world!'
c.upper()           # 'HELLO, WORLD!'
```

```
Out[167]: 'HELLO, WORLD!'
```

```
In [128...]: c.capitalize()      # 'Hello, world!'
```

```
Out[128]: 'Hello, world!'
```

```
In [129...]: c.title()          # 'Hello, World!'
```

```
Out[129]: 'Hello, World!'
```

```
In [130...]: c.replace('!', '!!!!') # 'Hello, World!!!!'
```

```
Out[130]: 'hello, world!!!!'
```

```
In [131...]: c.find(',')        # 5
```

```
Out[131]: 5
```

```
In [132...]: c.split(',')     # ['Hello', 'World!']
```

```
Out[132]: ['Hello', 'World!']
```

```
In [ ]:
```

[] 비스칼라(Scalar) 형

> 여러 개의 값을 사용할 수 있도록 구성된 자료 구조

>> list, tuple, dictionary 등

```
In [2]: ## 리스트(List) 형 : []
## 생성 및 검색
a = [1, 2, 'a', 'b']    #리스트 생성 및 초기화
print(type(a))
a
```

```
<class 'list'>
```

```
Out[2]: [1, 2, 'a', 'b']
```

```
In [170...]: ## Tuple 형 : ()
a = (1, 2, 3, 4, 5)    #튜플의 생성 및 초기화
print(type(a))
a
```

```
<class 'tuple'>
```

```
Out[170]: (1, 2, 3, 4, 5)
```

```
In [171]: ## 딕셔너리(Dictionary) 형 : { }
## Key : Value로 생성
a = {'A':90, 'B':80, 'C':70, 'D':60}    #튜플의 생성 및 초기화
print(type(a))
a

<class 'dict'>
Out[171]: {'A': 90, 'B': 80, 'C': 70, 'D': 60}
```

In []:

[03-2] 마술 상자 같은 '함수' 이해하기

▣ 내장 함수

```
In [176]: ## 내장 함수
x = [1, 2, 3, 4]
print(sum(x))
print(max(x))
print(min(x))
```

10
4
1

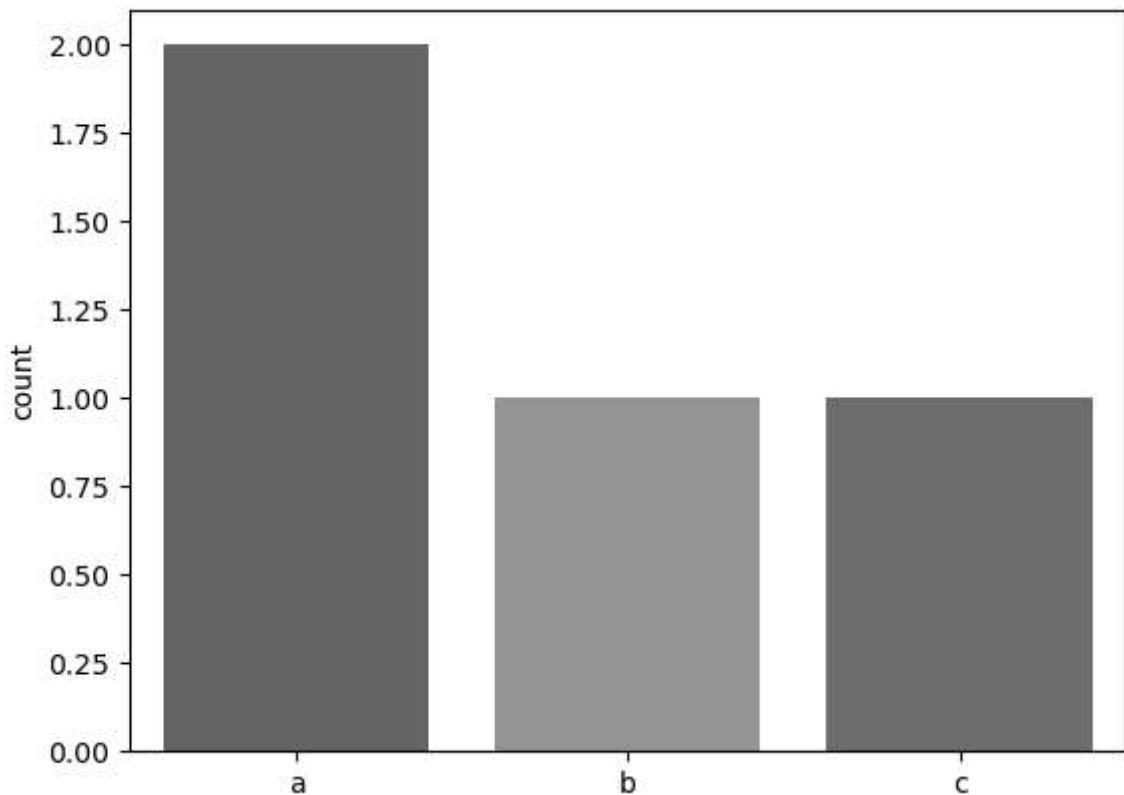
▣ 외장 함수

> 패키지 임포트 후 사용

```
In [3]: ## 외장 함수
import seaborn as sns

var = ['a', 'a', 'b', 'c']
sns.countplot(x = var)

Out[3]: <AxesSubplot:ylabel='count'>
```



[] lamda 함수

>> 함수 객체를 반환하는 함수

>> return문이 필요없는 함수

>> 간단한 함수를 선언하면서, 바로 실행도도록 하는 용도로도 사용

```
In [184]: ## [lamda 함수 선언]
multi=lamda x,y:x*y #선언: lamda 함수를 선언하여 함수 객체로 반환
multi(3,4)           #실행: 함수 객체를 통해 lamda 함수 실행
```

Out[184]: 12

```
In [185]: ## [lamda 함수 선언] 매개변수에 기본값 사용
multi=lamda x=3,y=4:x*y #매개변수에 기본값 사용
multi()
```

Out[185]: 12

```
In [186]: ## [lamda 함수 선언] 선언과 동시에 실행
(lamda x,y:x*y)(3,4) #선언과 동시에 실행
```

Out[186]: 12

```
In [181]: # 매개변수 x, y를 사용하여 x+y를 반환하는 lamda 함수 선언
lambda x, y:x+y #lamda 함수는 실행 후 함수 객체를 반환

# 선언된 lamda 함수 바로 사용
(lambda x, y:x+y)(3, 4) #반환된 함수 객체를 사용하여 (3, 4)를 실행
```

Out[181]: 7

```
In [190...]: ## [Data Frame] 생성
import pandas as pd
long_name_df = pd.DataFrame({'Eng' : [87, 79, 80],
                             'Mat' : [80, 90, 80]},
                             index = ['Kims', 'Lees', 'Parks']) #index 강제 부여
long_name_df
```

Out[190]:

	Eng	Mat
Kims	87	80
Lees	79	90
Parks	80	80

lambda 함수 활용

```
In [192...]: ## lambda 함수 활용
#long_name_df 객체에서 lambda 함수가 생성되었으므로 매개변수 x는 객체명 long_name_df 대신 x로 사용 가능
long_name_df.assign(avg = lambda x: (x['Eng'] + x['Mat'])/2) # long_name_df 대신 x로 사용 가능
```

Out[192]:

	Eng	Mat	avg
Kims	87	80	83.5
Lees	79	90	84.5
Parks	80	80	80.0

[03-3] 함수 꾸러미, '패키지' 이해하기

▣ 함수, 모듈, 패키지

- > **함수(Function)** : 단위 기능을 수행하는 명령의 집합
- > **모듈(Module)** : 함수의 모음으로 함수보다는 좀 더 큰 단위의 작업을 수행
- > **패키지(Package)** : 관련이 깊은 모듈들을 하나로 모아놓은 것

<> 패키지명.모듈명.함수명() 으로 함수 사용하기

```
In [ ]: # sklearn 패키지의 metrics 모듈 로드하기
import sklearn.metrics
```

```
In [ ]: # sklearn 패키지 metrics 모듈의 accuracy_score() 사용하기
sklearn.metrics.accuracy_score()
```

<> 모듈명.함수명() 으로 함수 사용하기

```
In [ ]: # sklearn 패키지의 metrics 모듈 로드하기
from sklearn import metrics
metrics.accuracy_score()
```

<> 함수명()으로 함수 사용하기

```
In [ ]: # sklearn 패키지 metrics 모듈의 accuracy_score() 로드하기  
from sklearn.metrics import accuracy_score  
accuracy_score()
```

[연습] Turtle 모듈 활용

```
In [1]: import turtle as t  
  
t.shape('turtle')  
t.forward(100)  
t.left(90)  
  
t.exitonclick() # 실행 창을 닫지 않도록
```

[Coding] Turtle 모듈 활용 n각형 그리기

>> **import turtle** 모듈을 활용

>> 키보드로 **turtle**이 그릴 n각형의 n을 입력받는다.

>> 함수 호출을 통해 원하는 n각형을 그린다.

>> n이 2이하가 입력되면 작업을 종료한다.

```
In [ ]:
```

[Coding] 숫자 야구 게임

랜덤한 3자리 숫자 맞추기 게임을 완성하시오.

> 랜덤한 3자리 정수 발생, 단 각 자리 수의 값은 같으면 안됨

> 게이머는 추측하는 3자리 값을 키보드로 입력 (맞출 때까지)

> 각 위치(digit)에서 값이 같으면 Strike, 값은 존재하나 위치가 다르면 Ball로 처리

> 매회 ">>%d Strike, %d Ball"로 결과 제공

> 게임 종료 조건 : 3 Strike 또는 입력 값 00

```
In [ ]:
```

패키지 설치하기

```
In [1]: pip install pydataset
```

```
Requirement already satisfied: pydataset in c:\Users\Wadmin\Anaconda3\lib\site-packages (0.2.0)
Requirement already satisfied: pandas in c:\Users\Wadmin\Anaconda3\lib\site-packages (from pydataset) (1.4.4)
Requirement already satisfied: pytz>=2020.1 in c:\Users\Wadmin\Anaconda3\lib\site-packages (from pandas->pydataset) (2022.1)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\Users\Wadmin\Anaconda3\lib\site-packages (from pandas->pydataset) (2.8.2)
Requirement already satisfied: numpy>=1.18.5 in c:\Users\Wadmin\Anaconda3\lib\site-packages (from pandas->pydataset) (1.21.5)
Requirement already satisfied: six>=1.5 in c:\Users\Wadmin\Anaconda3\lib\site-packages (from python-dateutil>=2.8.1->pandas->pydataset) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

<> 패키지 함수 사용하기

```
In [2]: import pydataset
pydataset.data()
```

```
Out[2]:
```

	dataset_id	title
0	AirPassengers	Monthly Airline Passenger Numbers 1949-1960
1	Bjsales	Sales Data with Leading Indicator
2	BOD	Biochemical Oxygen Demand
3	Formaldehyde	Determination of Formaldehyde
4	HairEyeColor	Hair and Eye Color of Statistics Students
...
752	VerbAgg	Verbal Aggression item responses
753	cake	Breakage Angle of Chocolate Cakes
754	cbpp	Contagious bovine pleuropneumonia
755	grouseticks	Data on red grouse ticks from Elston et al. 2001
756	sleepstudy	Reaction times in a sleep deprivation study

757 rows × 2 columns

```
In [3]: df = pydataset.data('mtcars') # mtcars 데이터를 df 데이터 프레임 df에 할당
df # df 출력
```

Out[3]:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

In []: