

Python Basics

- Numpy, Dataframe, & MySQL

Iterable Data Types

- 한번에 하나씩 element를 반환할 수 있는 데이터 타입
- ' for' 문을 사용하여 element를 순회할 수 있는 데이터 타입
- Examples
 - range()
 - list
 - tuple
 - string
 - dictionary
 - set
 - numpy array

Numpy

- Homogeneous, multidimensional array
- Narray (or array)
 - ndim
 - shape
 - size
 - dtype
 - itemsize

ndarray attributes

```
import numpy as np
a = np.arange(15).reshape(3,5)
print(a)
print(a.shape)
print(a.ndim)
print(a.dtype.name)
print(a.itemsize)
print(a.size)
print(type(a))
```

ndarray 생성

```
import numpy as np  
a = np.array([[1,2,3], [4,5,6]])  
b = np.zeros((3,4))  
c = np.ones( (2,3,4), dtype = np.int64 )  
d = np.empty( (2,3) )  
e = np.arange(10,100, 10)
```

ndarray arithmetic operations

“elementwise” operations:

```
A = np.array( [20,30,40,50] )
```

```
B = np.arange(4)
```

```
C = A-B
```

```
D = B**2
```

```
print(A<35)
```

* 벡터화 연산 적용 가능: for loop 보다 빠른 실행

list vs ndarray

- list
 - 여러가지 타입의 원소
 - linked List 구현
 - 메모리 용량이 크고 속도가 느림
 - 벡터화 연산 불가

VS

- ndarray
 - 동일 타입의 원소
 - contiguous memory layout
 - 메모리 최적화, 계산 속도 향상
 - 벡터화 연산 가능

ndarray 벡터화 계산(vectorized computation) 효과

```
import time
```

```
a = [r for r in range(10000000)]  
b = [r for r in range(10000000)]  
c = []
```

```
start = time.time()  
for i in range(10000000):  
    c.append(a[i]*b[i])  
end = time.time()
```

```
print("elapsed time =", end-start)
```

vs

```
import numpy as np  
import time
```

```
a = np.arange(10000000)  
b = np.arange(10000000)
```

```
start = time.time()  
c = a*b  
end = time.time()
```

```
print("elapsed time =", end-start)
```


벡터화 연산 연습 문제

- 1000 이하의 정수를 백만개 생성해서, 리스트(배열)에 초기화해서 넣고,
- 리스트(배열)에서 500 보다 큰 수는 1로 바꾸고, 500 이하는 0으로 변환하는 새로운 리스트를 생성하라.
- 리스트와 배열로 이를 구현할 때, 반응시간을 비교하는 python 프로그램을 작성하라.

ndarray matrix-wise operations

```
A = np.array ( [[1,1],  
                [0,1]] )
```

```
B = np.array( [[2,0],  
               [3,4]] )
```

```
C = A*B           # elementwise product
```

```
D = A@ B          # matrix product
```

```
X = [1,2,3,4]
```

```
Y = [1,0,1,0]
```

```
F = np.inner(X,Y)    # inner product
```

Pandas & DataFrame

- Pandas: Python Data Analysis Library
- DataFrame: Pandas 가 제공하는 테이블 형태의 데이터 분석을 위한 자료구조
- ndarray, dict, list 등과 데이터 호환 및 변환 가능
- Excel/csv 포맷의 데이터파일을 쉽게 import 할 수 있다는 장점이 있음
 - read_excel(), read_csv()

DataFrame의 구조

The diagram illustrates the structure of a DataFrame using a table. The table has 4 columns and 4 rows. The first row contains headers: an empty cell, '중간시험', '기말시험', and '최종점수'. The subsequent three rows contain numerical data. Blue arrows point from the label 'index' to the first column of the data rows (0, 1, 2). Another set of blue arrows points from the label 'columns' to the three header columns ('중간시험', '기말시험', '최종점수').

	중간시험	기말시험	최종점수
0	35	30.1	92.2
1	34.12	30.1	91.95
2	34.3	29.75	91.15

Excel/CSV 파일 읽어들이기 예제

```
import pandas as pd
```

```
xls_file = 'score.xlsx'
```

```
df = pd.read_excel(xls_file)
```

```
for index, row in df.iterrows():  
    print(row)
```

mysql connection routines

```
import pymysql
```

```
from pymysql.constants.CLIENT import MULTI_STATEMENTS
```

```
def open_db(dbname='DS2022'):  
    conn = pymysql.connect(host='localhost',  
                           user='ds2022',  
                           passwd='ds2022',  
                           db=dbname,  
                           client_flag=MULTI_STATEMENTS)  
    cur = conn.cursor(pymysql.cursors.DictCursor)  
  
    return conn, cur
```

```
def close_db(conn, cur):  
    cur.close()  
    conn.close()
```

university.student table creation in MySQL

```
create table student (  
    sno int primary key,  
    sname varchar(30),  
    dept varchar(10),  
    enter_year int  
);
```

```
insert into student (sno, sname, dept, enter_year)  
values  
(100, 'Kim', 'CE', 2015),  
(200, 'Lee', 'CE', 2016),  
(300, 'Jang', 'MA', 2017),  
(400, 'Park', 'IE', 2018);
```

reading from MySQL table

```
from db_conn import *
```

```
conn, cur = open_db()
```

```
sql = "select * from student;"
```

```
cur.execute(sql)
```

```
row = cur.fetchone()
```

```
while row:
```

```
    print(row)
```

```
    row = cur.fetchone()
```

```
close_db(conn, cur)
```


실습: Kaggle Dataset → MySQL

- kaggle dataset 에서 dataset 을 다운로드
- dataset을 panda를 통해서 dataframe으로 읽어들이м
- mysql 의 table로 입력하는 python 코드
- table을 검색하는 python 코드