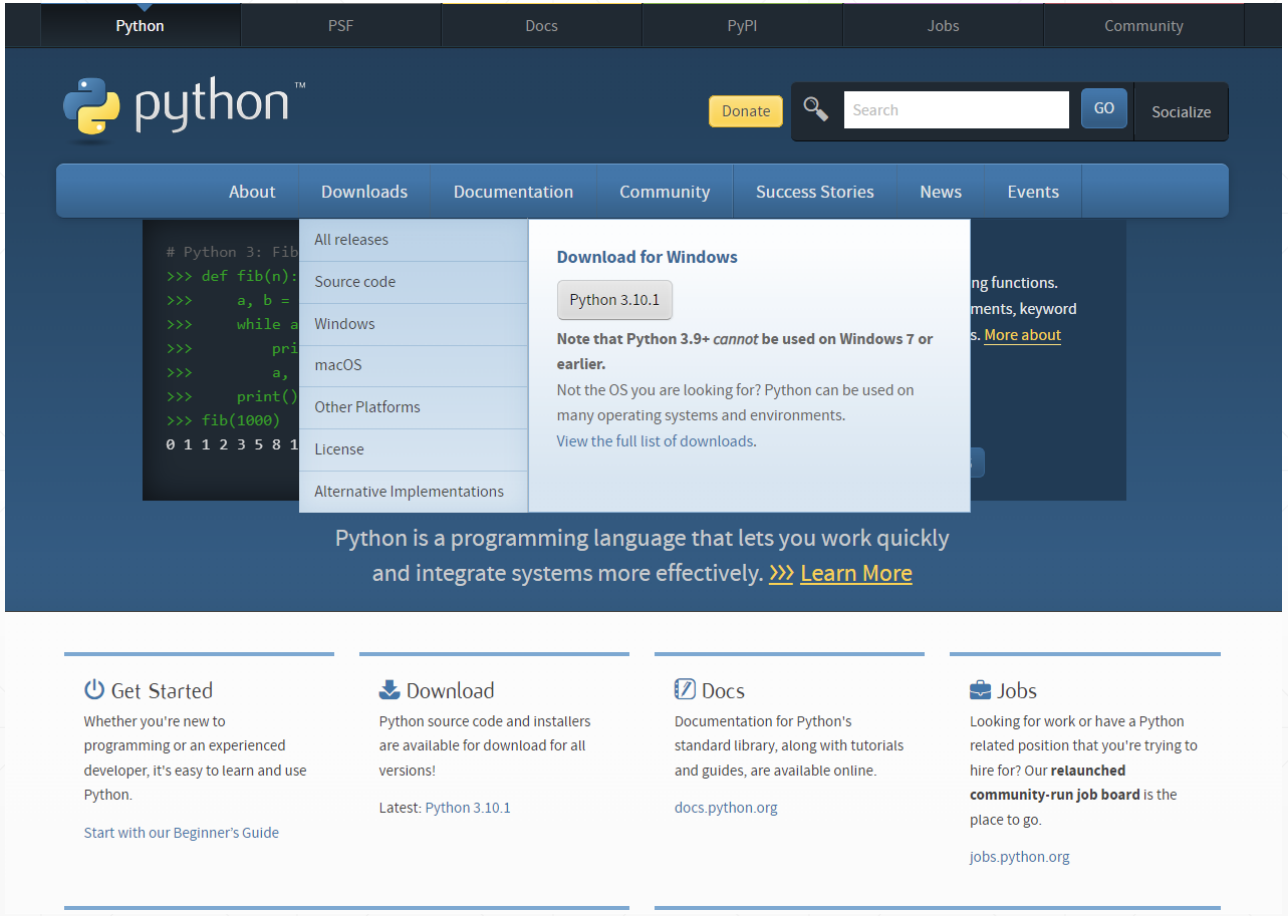
파이썬 프로그래밍 기초

빅데이터 분석

파이썬 설치하기



www.python.org

데이터 분석을 위한 주요 라이브러리

① numpy

numpy 임포트	`>>> import numpy as np`

리스트를 이용하여 numpy 생성	`>>> ar1 = np.array([1,2,3,4,5])`
	`>>> ar1`
	`>>> type(ar1)`
	`>>> ar2 = np.array([[10,20,30],[40,50,60]])`
	`>>> ar2`

값의 범위를 지정하여 numpy 생성	`>>> ar3 = np.arange(1,11,2)` `>>> ar3`
구조를 지정하여 numpy 생성	`>>> ar4 = np.array([1,2,3,4,5,6]).reshape((3,2))` `>>> ar4`
초기값과 구조를 지정하여 numpy 생성	`>>> ar5 = np.zeros((2,3))` `>>> ar5`

Numpy 슬라이싱	`>>> ar6 = ar2[0:2,0:2]`
	`>>> ar6`
	`>>> ar7 = ar2[0,:]`
	`>>> ar7`

numpy는 수치 데이터를 다루기 위한 라이브러리로 다차원 배열 자료구조인 ndarray를 지원하며 선형대수계산 등의 행렬 연산에 주로 사용된다.

데이터 분석을 위한 주요 라이브러리 (cont'd)

① numpy

numpy 사칙 연산	>>> ar8 = ar1 + 10
	>>> ar8
	>>> ar1 + ar8
	>>> ar8 - ar1
	>>> ar1 * 2
	>>> ar1 / 2

Numpy 행렬곱 연산	>>> ar9 = np.dot(ar2,ar4) >>> ar9

데이터 분석을 위한 주요 라이브러리 (cont'd)

② pandas

pandas 임포트	`>>> import pandas as pd`

Series 생성:
pd.Series()
	`>>> data1 = [10,20,30,40,50]`
	`>>> data1`
	`>>> data2 = ['1반', '2반', '3반', '4반', '5반']`
	`>>> data2`

리스트를 이용하여 Series 생성	`>>> sr1 = pd.Series(data1)`
	`>>> sr1`
	`>>> sr2 = pd.Series(data2)`
	`>>> sr2`

값을 이용하여 Series 생성	`>>> sr3 = pd.Series([101,102,103,104,105])`
	`>>> sr3`
	`>>> sr4 = pd.Series(['월', '화', '수', '목', '금'])`
	`>>> sr4`

pandas는 데이터 분석에서 자주 사용하는 테이블 형태를 다룰 수 있는 라이브러리다. pandas는 1차원 자료구조인 Series, 2차원 자료구조인 DataFrame, 3차원 자료구조인 Panel을 지원하는데 그 중에서 Series와 DataFrame을 많이 사용한다.

데이터 분석을 위한 주요 라이브러리 (cont'd)

② pandas

인덱스를 지정하여 Series 생성	`>>> sr5 = pd.Series(data1,`
	`  index=[1000,1001,1002,1003,1004])`
	`>>> sr5`
	`>>> sr6 = pd.Series(data1,index=data2)`
	`>>> sr6`
	`>>> sr7 = pd.Series(data2,index=data1)`
	`>>> sr7`
	`>>> sr8 = pd.Series(data2,index=sr4)`
	`>>> sr8`

Series 인덱싱	`>>> sr8[2]`
	`>>> sr8['수']`
	`>>> sr8[-1]`

Series 슬라이싱	`>>> sr8[0:4]`
Series 인덱스 구하기: index	`>>> sr8.index`
Series 값 구하기: values	`>>> sr8.values`

데이터 분석을 위한 주요 라이브러리 (cont'd)

② pandas

Series 원소가 숫자이면 덧셈 수행	`>>> sr1 + sr3`
Series 원소가 문자열이면 문자열 연결 수행	`>>> sr4 + sr2`

데이터 분석을 위한 주요 라이브러리 (cont'd)

② pandas

pd.DataFrame()	>>> data_dic = {'year':[2018,2019,202],
	'sales':[350,480,1099]}
	>>> data_dic

딕셔너리를 이용하여 DataFrame 생성	>>> df1 = pd.DataFrame(data_dic) >>> df1

리스트를 이용하여 DataFrame 생성1	>>> df2 = pd.DataFrame([[89.2,92.5,90.8],
	[92.8,89.9,95.2]],
	index=['중간고사','기말고사'],
	columns=data2[0:3])
	>>> df2

리스트를 이용하여 DataFrame 생성2	>>> data_df = [['20201101', 'Hong', '90', '95'],
	['20201102', 'Kim', '93', '94'],
	['20201103', 'Lee', '87', '97']]
	>>> df3 = pd.DataFrame(data_df)
	>>> df3

데이터 분석을 위한 주요 라이브러리 (cont'd)

② pandas

DataFrame 열 이름 설정	>>> df3.columns = ['학번', '이름', '중간고사', '기말고사'] >>> df3

DataFrame 조회	>>> df3.head(2)
	>>> df3.tail(2)
	>>> df3['이름']

CSV 파일로 저장	>>> df3.to_csv('C:/Users/.../score.csv', header='False')
CSV 파일을 DataFrame으로 불러오기	>>> df4 = pd.read_csv('C:/Users/.../score.csv', encoding='utf-8', index_col=0, engine='python') >>> df4

데이터 분석을 위한 주요 라이브러리 (cont'd)

③ matplotlib

matplotlib 임포트	`>>> import matplotlib`
pyplot 모듈 임포트하기	`>>> import matplotlib.pyplot as plt`

데이터 준비	`>>> x = [2016, 2017, 2018, 2019, 2020]` `>>> y = [350, 410, 520, 695, 543]`
x축과 y축 데이터를 지정하여 라인플롯 생성	`>>> plt.plot(x,y)`
차트 제목 설정	`>>> plt.title('Annual sales')`
축 레이블 설정	`>>> plt.xlabel('years')` `>>> plt.ylabel('sales')`
라인플롯 표시	`>>> plt.show()`

Matplotlib은 라인플롯 차트, 바 차트, 파이 차트, 히스토그램, 산점도 등의 다양한 차트 그리기를 지원하는 라이브러리다.
데이터 탐색이나 분석 결과를 시각화하기 위해 많이 사용한다.

데이터 분석을 위한 주요 라이브러리 (cont'd)

③ matplotlib

데이터 준비	`>>> y1 = [350, 410, 520, 695]`
	`>>> y2 = [200, 250, 385, 350]`
	`>>> x = range(len(y1))`

x축과 y축 데이터를 지정하여 바 차트 생성	`>>> plt.bar(x, y1, width=0.7, color="blue")`
	`>>> plt.bar(x, y2, width=0.7, color="red",`
	`bottom=y1)`

차트 제목 설정	`>>> plt.title('Quarterly sales')`
축 레이블 설정	`>>> plt.xlabel('Quarters')` `>>> plt.ylabel('sales')`
눈금 이름 리스트 생성	`>>> xLabel = ['first', 'second', 'third',` `'fourth']`
바 차트의 x축 눈금 이름 설정	`>>> plt.xticks(x, xLabel, fontsize=10)`
범례설정	`>>> plt.legend(['chairs', 'desks'])`
바 차트 표시	`>>> plt.show()`