Libraries for Data Analysis (Pandas, Numpy, Matplotlib)

In [3]:
```
# 라이브러리 설치
!pip install pandas numpy matplotlib
```

```
Collecting pandas
  Downloading pandas-1.5.2-cp38-cp38-macosx_10_9_x86_64.whl (11.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.9/11.9 MB 3.7 MB/s eta 0:00:00
00:01:00:01
Collecting numpy
  Downloading numpy-1.24.1-cp38-cp38-macosx_10_9_x86_64.whl (19.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19.8/19.8 MB 3.2 MB/s eta 0:00:00
00:01:00:01
Collecting matplotlib
  Downloading matplotlib-3.6.3-cp38-cp38-macosx_10_12_x86_64.whl (7.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.3/7.3 MB 2.5 MB/s eta 0:00:00
00:01:00:01
Collecting pytz>=2020.1
  Downloading pytz-2022.7.1-py2.py3-none-any.whl (499 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 499.4/499.4 kB 2.5 MB/s eta 0:00:00
00:01:00:01
Requirement already satisfied: python-dateutil>=2.8.1 in ./opt/anaconda3/envs/pybook/lib/python3.8/site-packages (from pandas) (2.8.2)
Collecting pillow>=6.2.0
  Downloading Pillow-9.4.0-2-cp38-cp38-macosx_10_10_x86_64.whl (3.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.3/3.3 MB 2.8 MB/s eta 0:00:00
00:01:00:01
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.4.4-cp38-cp38-macosx_10_9_x86_64.whl (65 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 65.4/65.4 kB 984.9 kB/s eta 0:00:00
0:00:01

Collecting pyparsing>=2.2.1
  Using cached pyparsing-3.0.9-py3-none-any.whl (98 kB)
Collecting fonttools>=4.22.0
  Downloading fonttools-4.38.0-py3-none-any.whl (965 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 965.4/965.4 kB 2.2 MB/s eta 0:00:00
00:01:00:01
Collecting contourpy>=1.0.1
  Downloading contourpy-1.0.7-cp38-cp38-macosx_10_9_x86_64.whl (243 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 244.0/244.0 kB 2.8 MB/s eta 0:00:0
0:00:01
Requirement already satisfied: packaging>=20.0 in ./opt/anaconda3/envs/pybook/lib/python3.8/site-packages (from matplotlib) (23.0)
Collecting cycler>=0.10
  Using cached cycler-0.11.0-py3-none-any.whl (6.4 kB)
Requirement already satisfied: six>=1.5 in ./opt/anaconda3/envs/pybook/lib/python3.8/site-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Installing collected packages: pytz, pyparsing, pillow, numpy, kiwisolver, fonttools, cycler, pandas, contourpy, matplotlib
Successfully installed contourpy-1.0.7 cycler-0.11.0 fonttools-4.38.0 kiwisolver-1.4.4 matplotlib-3.6.3 numpy-1.24.1 pandas-1.5.2 pillow-9.4.0 pyparsing-3.0.9 pytz-2022.7.1
```

판다스란

- 파이썬에서 가장 널리 사용되는 데이터 분석 라이브러리로
- 데이터 프레임이라는 자료구조를 사용한다
- 데이터 프레임은 엑셀의 스프레드시트와 유사한 형태이며 파이썬으로 데이터를 쉽게 처리할 수 있도록 한다

In [6]:
```
# 판다스 라이브러리 불러오기
import pandas as pd
```

데이터 프레임 생성

In [7]:
```
# 판다스의 데이터 프레임 생성
names = ['Bob', 'Jessica', 'Mary', 'John', 'Mel']
births = [968, 155, 77, 578, 973]
custom = [1, 5, 25, 13, 23232]

BabyDataSet = list(zip(names, births))
df = pd.DataFrame(data = BabyDataSet, columns = ['Names', 'Births'])

# 데이터 프레임 상단 부분 출력
df.head()
```

Out[7]:

	Names	Births
0	Bob	968
1	Jessica	155
2	Mary	77
3	John	578
4	Mel	973

In [8]:
```
BabyDataSet
```

Out[8]:
```
[('Bob', 968), ('Jessica', 155), ('Mary', 77), ('John', 578), ('Mel', 973)]
```

데이터 프레임 기본 정보 출력

In [9]:
```
# 데이터 프레임 열 타입 정보
df.dtypes
```

Out[9]:
```
Names     object
Births     int64
dtype: object
```

In [10]:
```
# 데이터 프레임 인덱스 정보
df.index
```

Out[10]:
```
RangeIndex(start=0, stop=5, step=1)
```

In [11]:
```
# 데이터 프레임 열 형태 정보
df.columns
```

Out[11]:
```
Index(['Names', 'Births'], dtype='object')
```

In [13]:
```
# 조건을 추가하여 데이터 선택하기
df[df['Births'] > 100]
```

Out[13]:

	Names	Births
0	Bob	968
1	Jessica	155
3	John	578
4	Mel	973

넘파이란

- Numerical Python의 줄임말로
- 수치 계산을 위해 만들어진 파이썬 라이브러리
- 넘파이의 자료구조는 pandas, matplotlib 라이브러리의 기본 데이터 타입으로 사용되기도 한다
- 배열(array)개념으로 변수를 사용하며
- 벡터, 행렬 등의 연산을 쉽고 빠르게 수행한다

In [15]:
```
# 넘파이 라이브러리 불러오기
import numpy as np
```

In [17]:
```
# 넘파이 배열 생성
arr1 = np.arange(15).reshape(3, 5)
arr1
```

Out[17]:
```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])
```

note: 이 배열은 넘파이 배열이며, 파이썬의 기본 자료구조와는 다른 데이터 타입이다.

In [21]:
```
# 0으로 채워진 넘파이 배열 생성
arr2 = np.zeros((3, 4))
arr2
```

Out[21]:
```
array([[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]])
```

In [22]:
```
# 1로 채워진 넘파이 배열 생성
arr3 = np.ones((3, 4))
arr3
```

Out[22]:
```
array([[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]])
```

In [25]:
```
# 넘파이 배열 생성2
arr4 = np.array([
    [1, 2, 3],
    [4, 5, 6]
], dtype = np.float64)

arr4
```

Out[25]:
```
array([[1., 2., 3.],
       [4., 5., 6.]])
```

In [28]:
```
# 넘파이 행렬 연산
arr5 = np.dot(arr4, arr1)
arr5
```

Out[28]:
```
array([[ 40.,  46.,  52.,  58.,  64.],
       [ 85., 100., 115., 130., 145.]])
```

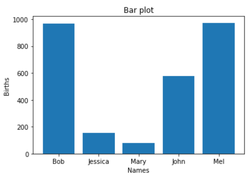
Matplotlib이란

- 기본적인 데이터 시각화 라이브러리
- %matplotlib inline' 선언으로 현재 실행중인 주피터 노트북에서 그래프를 출력할 수 있다

In [29]:
```
# matplotlib 라이브러리 불러오기
%matplotlib inline
import matplotlib.pyplot as plt
```

In [31]:
```
y = df['Births']
x = df['Names']
```

In [32]:
```
# [1] bar plot 막대 그래프
plt.bar(x, y) # 막대 그래프 객체 생성
plt.xlabel('Names') # 엑스축 제목
plt.ylabel('Births') # 와이축 제목
plt.title('Bar plot') # 그래프 제목
plt.show() # 그래프 출력
```



In [33]:
```
# [2] scatter plot 산점도 그래프
# 랜덤 추출 시드 고정
np.random.seed(19920613)

# 산점도 데이터 생성
x = np.arange(0.0, 100.0, 5.0) # 5간격으로 0부터 100까지 숫자 생성
y = (x * 1.5) + np.random.rand(20) * 50 # random.rand(): 넘파이 배열 타입의 난수 생성
# rand()에 숫자를 넣어주면 넣은 숫자의 길이만큼 랜덤한 숫자를 생성하고 그것을 1차원 array로 반환한다.

# 산점도 데이터 출력
plt.scatter(x, y, c="b", alpha=0.5, label="scatter point") # 산점도 그래프 객체 생성
# c: color, b: blue
# alpha: The alpha blending value, between 0 (transparent) and 1 (opaque).
# label: legend name
plt.xlabel("X") # 엑스축 제목
plt.ylabel("Y") # 와이축 제목
plt.legend(loc='upper left') # 범례 위치
plt.title('Scatter plot') # 그래프 제목
plt.show() # 그래프 출력
```