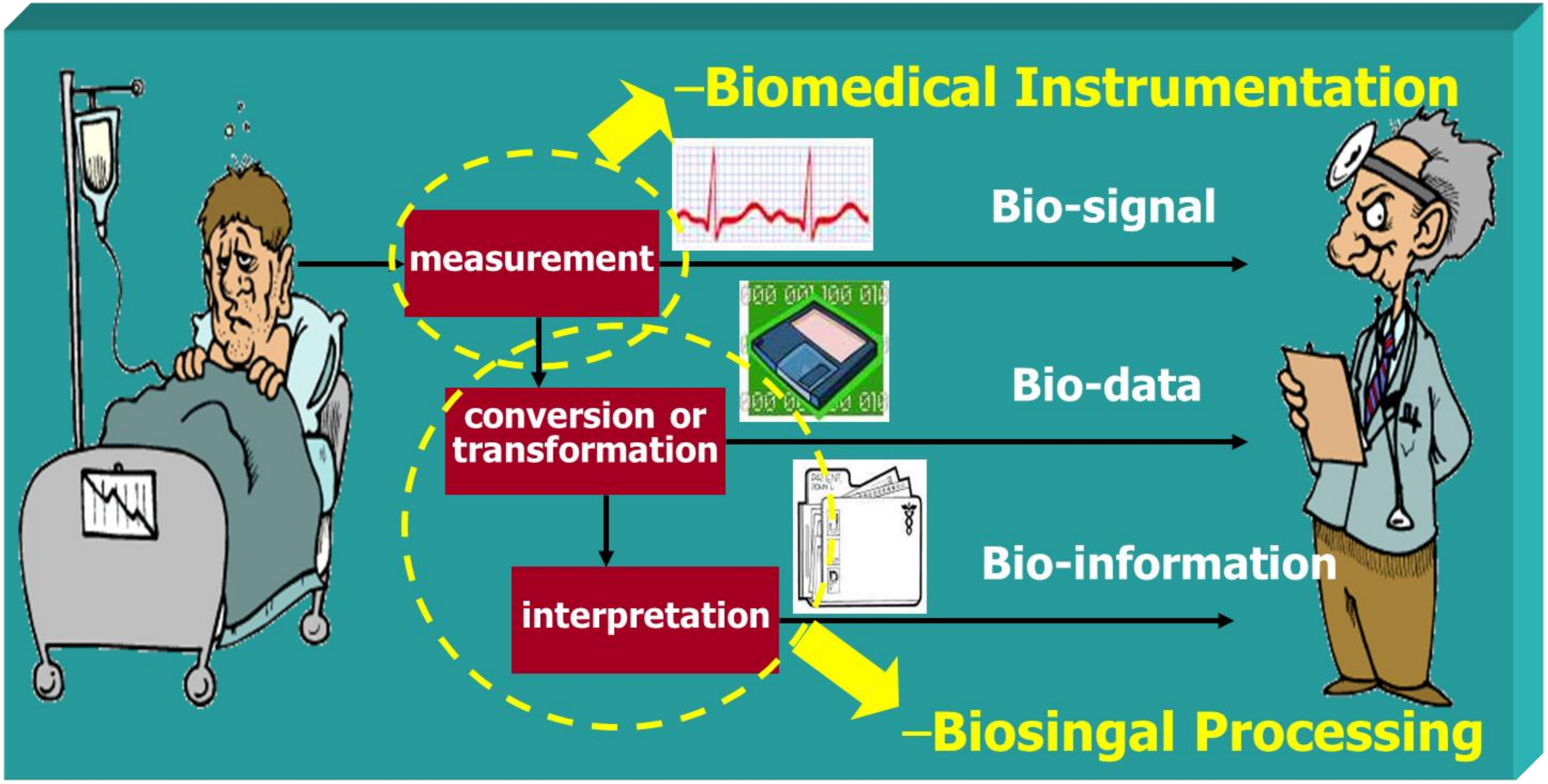
# 기초 컴퓨터 프로그래밍

- Python 기본: 파일 입출력 & Data Science Modules -

충남대학교 의과대학 의공학교실

구 윤 서 교수

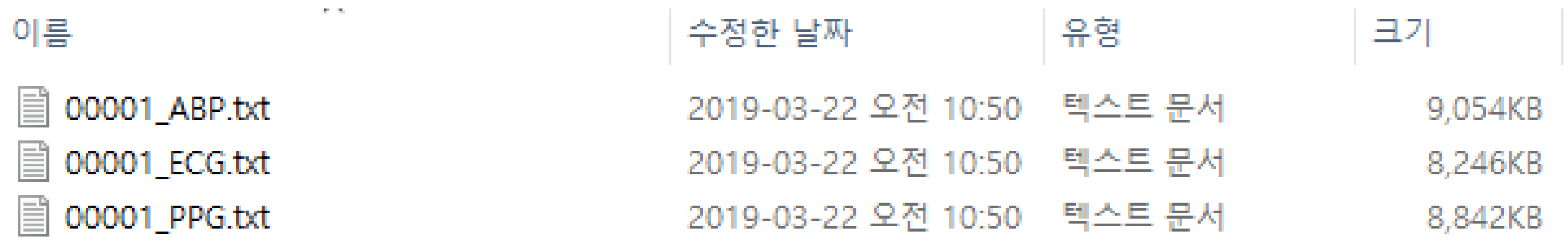
# 의료기기를 이용한 환자 진단 과정



## Patient Monitor (환자 감시 장치)

- Signal: ECG, PPG, ABP
- Index: Heart rate, SpO2, SBP/DBP, Temperature

## Data Output Files

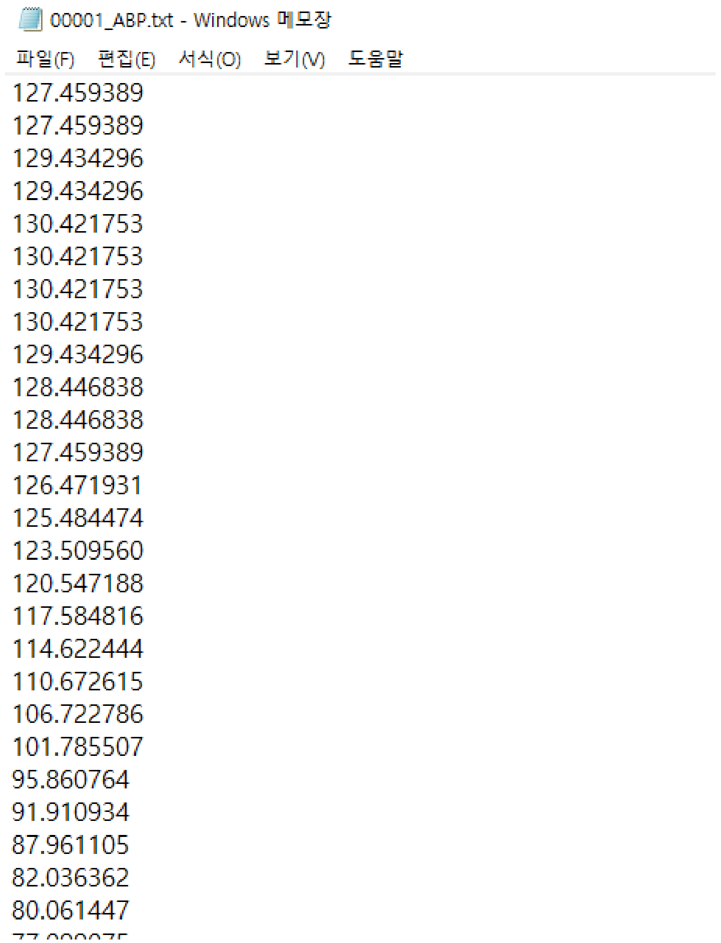


| 이름 | 수정한 날짜 | 유형 | 크기 |
|---|---|---|---|
| 00001_ABP.txt | 2019-03-22 오전 10:50 | 텍스트 문서 | 9,054KB |
| 00001_ECG.txt | 2019-03-22 오전 10:50 | 텍스트 문서 | 8,246KB |
| 00001_PPG.txt | 2019-03-22 오전 10:50 | 텍스트 문서 | 8,842KB |

- 측정 data를 주로 Text file 혹은 Excel file (ex. csv(comma separate values)) 형태로 PC에 출력

- 이러한 data 파일을 코드에서 불러와서 처리하거나 결과를 저장하여 새로운 파일을 생성하기 위해 파일 입출력 방법 숙지 필요

## Data Output Files


00001_ABP.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말

```
127.459389
127.459389
129.434296
129.434296
130.421753
130.421753
130.421753
130.421753
129.434296
128.446838
128.446838
127.459389
126.471931
125.484474
123.509560
120.547188
117.584816
114.622444
110.672615
106.722786
101.785507
95.860764
91.910934
87.961105
82.036362
80.061447
```

- 실제 Analog values로 변환하기 위해서는 Hardware specification 정보 (ADC range, bit 수 등)가 필요

## File Open & Close

- Open 함수: File handling을 위해 open 함수를 사용하여 접근
  - 접근 모드 ‘r’: read
  - 접근 모드 ‘w’: write
  - 접근 모드 ‘a’ : append (파일에 새로운 내용 추가)

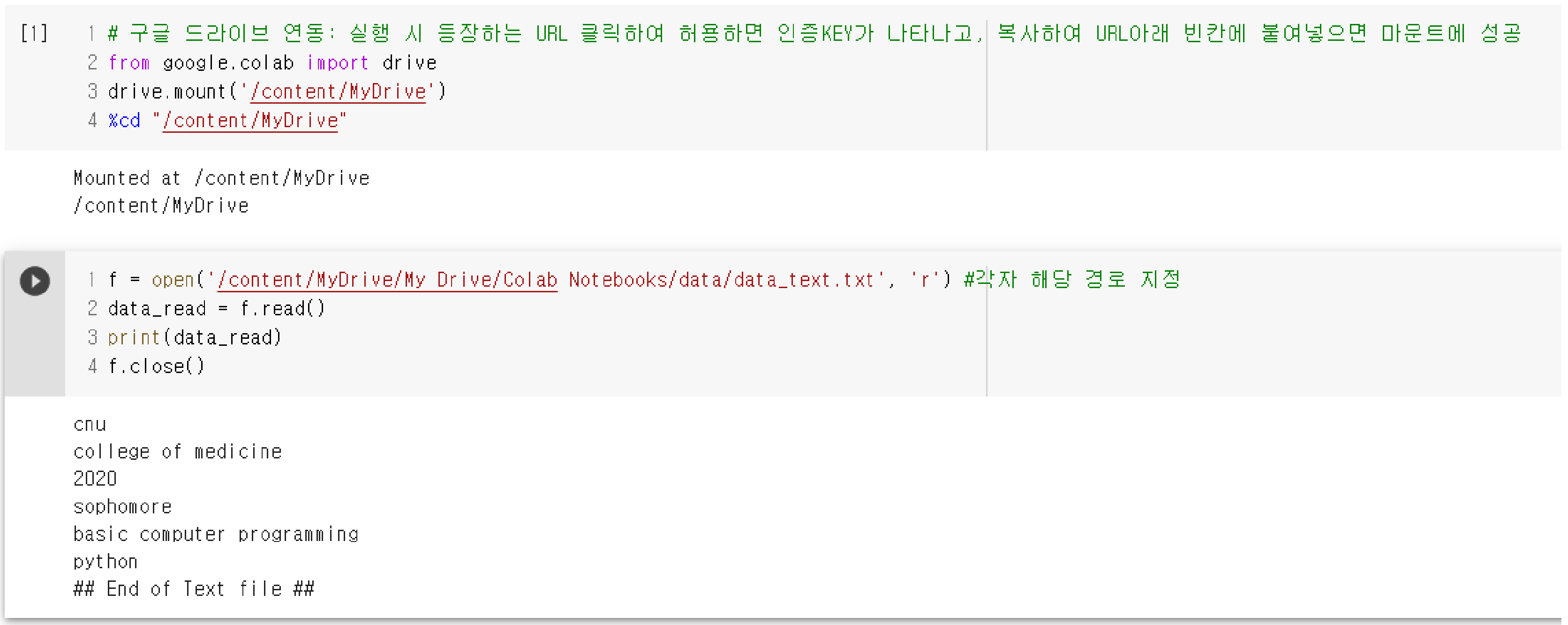
- Close 함수: File 처리 후 resource 해제

```
f = open(‘파일 이름’, ‘접근 모드’)


f.close()
```

# File Read

- Read 함수: Text file의 내용을 문자열(str)로 반환

```python
f = open('data_text.txt', 'r')
data_read = f.read()
print(data_read)
f.close()
```



```
[1]
# 구글 드라이브 연동: 실행 시 등장하는 URL 클릭하여 허용하면 인증KEY가 나타나고, 복사하여 URL아래 빈칸에 붙여넣으면 마운트에 성공
from google.colab import drive
drive.mount('/content/MyDrive')
%cd "/content/MyDrive"
```

```
Mounted at /content/MyDrive
/content/MyDrive
```

```python
f = open('/content/MyDrive/My Drive/Colab Notebooks/data/data_text.txt', 'r') #각자 해당 경로 지정
data_read = f.read()
print(data_read)
f.close()
```

```
cnu
college of medicine
2020
sophomore
basic computer programming
python
## End of Text file ##
```

## File Read

- File 객체와 같이 IO를 다룰 때 open() 후에는 close() 필요

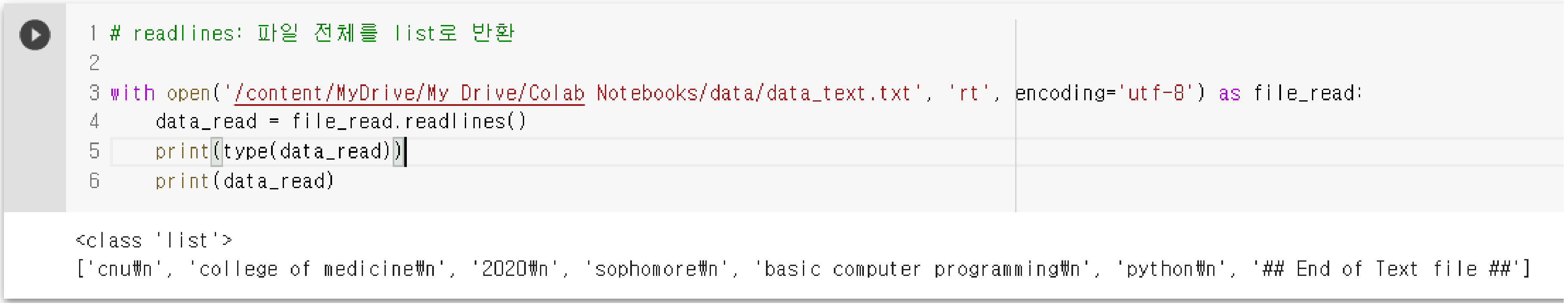
- Python에서는 이런 context 관리를 해주는 구문 존재 → with문 (with의 사용을 권장)

- close()를 알아서 해주어 사용이 편리

# File Read

- readlines: 파일 전체를 list로 반환

```
with open('data_text.txt', 'r') as file_read:
    data_read = file_read.readlines()
    print(type(data_read))
    print(data_read)
```



```
# readlines: 파일 전체를 list로 반환

with open('/content/MyDrive/My Drive/Colab Notebooks/data/data_text.txt', 'rt', encoding='utf-8') as file_read:
    data_read = file_read.readlines()
    print(type(data_read))
    print(data_read)
```

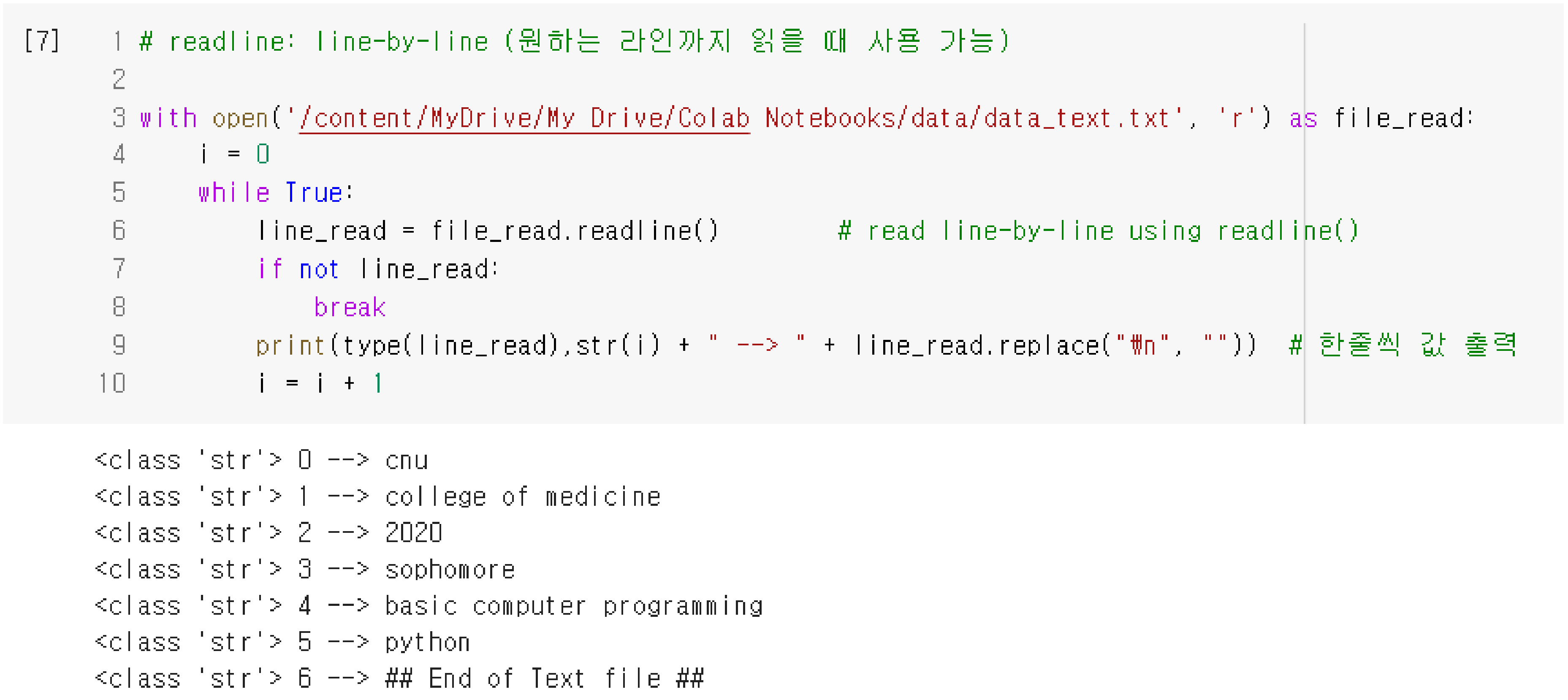
```
<class 'list'>
['cnu\n', 'college of medicine\n', '2020\n', 'sophomore\n', 'basic computer programming\n', 'python\n', '## End of Text file ##']
```

## File Read

- readline: line-by-line (원하는 라인까지 읽을 때 사용 가능)을 문자열(str)로 반환

```python
with open( 'data_text.txt', 'r') as file_read:
    i = 0
    while True:
        line_read = file_read.readline() # read line-by-line
        if not line_read:
            break
        print(type(line_read), str(i) + " --> " + line_read.replace("\n", ""))
# 한줄씩 출력
        i = i + 1
```

# File Read

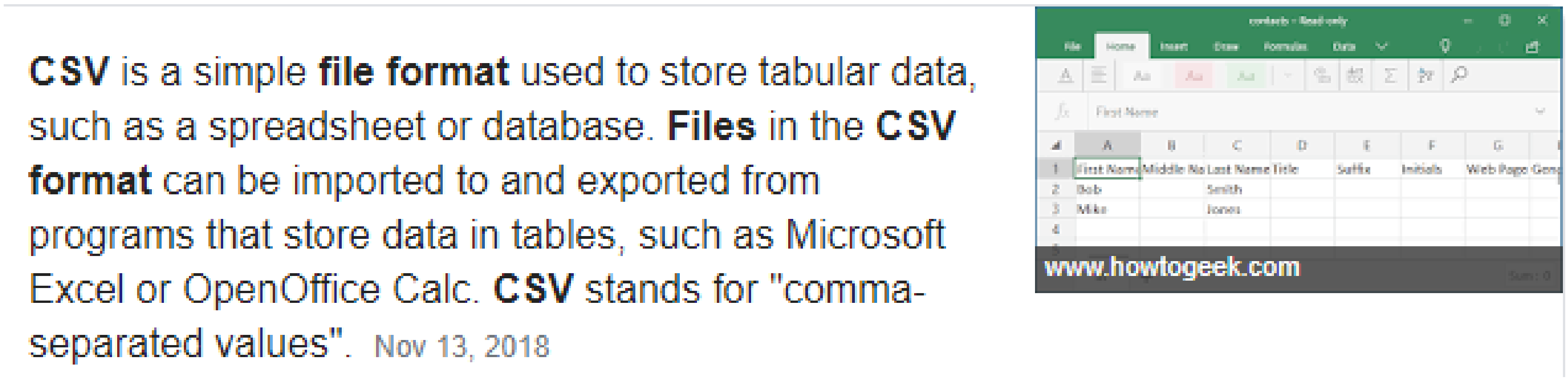


```
[7]
# readline: line-by-line (원하는 라인까지 읽을 때 사용 가능)

with open('/content/MyDrive/My Drive/Colab Notebooks/data/data_text.txt', 'r') as file_read:
    i = 0
    while True:
        line_read = file_read.readline()        # read line-by-line using readline()
        if not line_read:
            break
        print(type(line_read),str(i) + " --> " + line_read.replace("₩n", ""))  # 한줄씩 값 출력
        i = i + 1
```

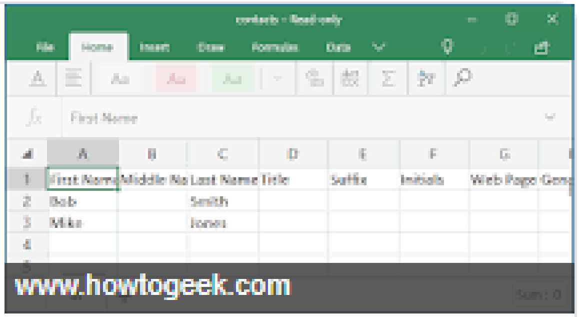
```
<class 'str'> 0 --> cnu
<class 'str'> 1 --> college of medicine
<class 'str'> 2 --> 2020
<class 'str'> 3 --> sophomore
<class 'str'> 4 --> basic computer programming
<class 'str'> 5 --> python
<class 'str'> 6 --> ## End of Text file ##
```

# File Read

- CSV file: Comma Separate Values



**CSV** is a simple **file format** used to store tabular data, such as a spreadsheet or database. **Files** in the **CSV format** can be imported to and exported from programs that store data in tables, such as Microsoft Excel or OpenOffice Calc. **CSV** stands for "comma-separated values". Nov 13, 2018

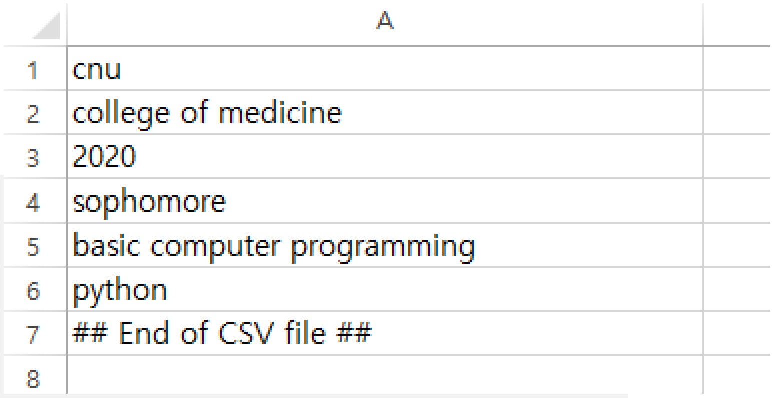


| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | First Name | Middle Na | Last Name | Title | Suffix | Initials | Web Page |
| 2 | Bob | | Smith | | | | |
| 3 | Mike | | Jones | | | | |
| 4 | | | | | | | |

# File Read

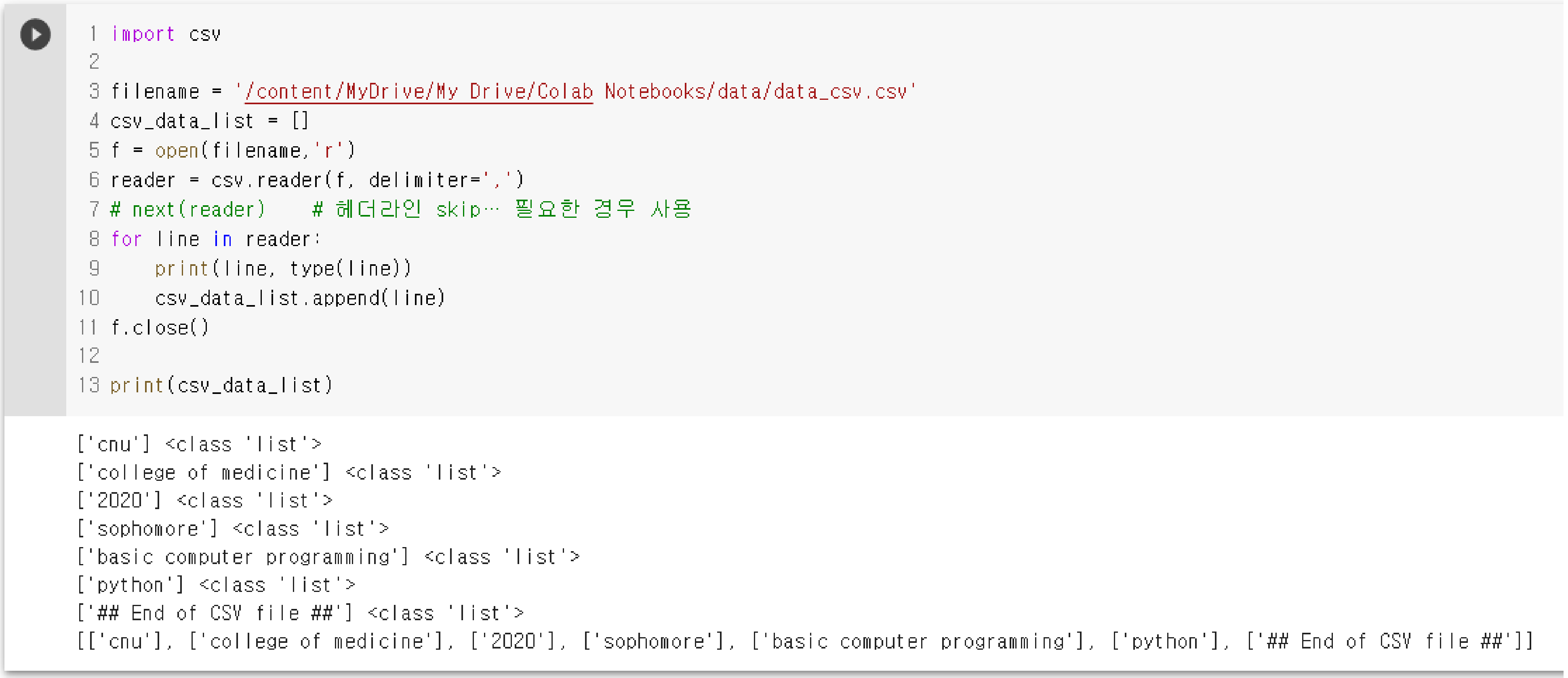
- Python의 built-in package인 CSV package를 사용하여 읽기 가능
- csv.reader

```python
import csv

filename = 'data_csv.csv'
csv_data_list = []
f = open(filename,'r')
reader = csv.reader(f, delimiter=',')

for line in reader:
    print(line, type(line))
    csv_data_list.append(line)
f.close()

print(csv_data_list)
```



|   | A |
|---|---|
| 1 | cnu |
| 2 | college of medicine |
| 3 | 2020 |
| 4 | sophomore |
| 5 | basic computer programming |
| 6 | python |
| 7 | ## End of CSV file ## |
| 8 |  |

# File Read



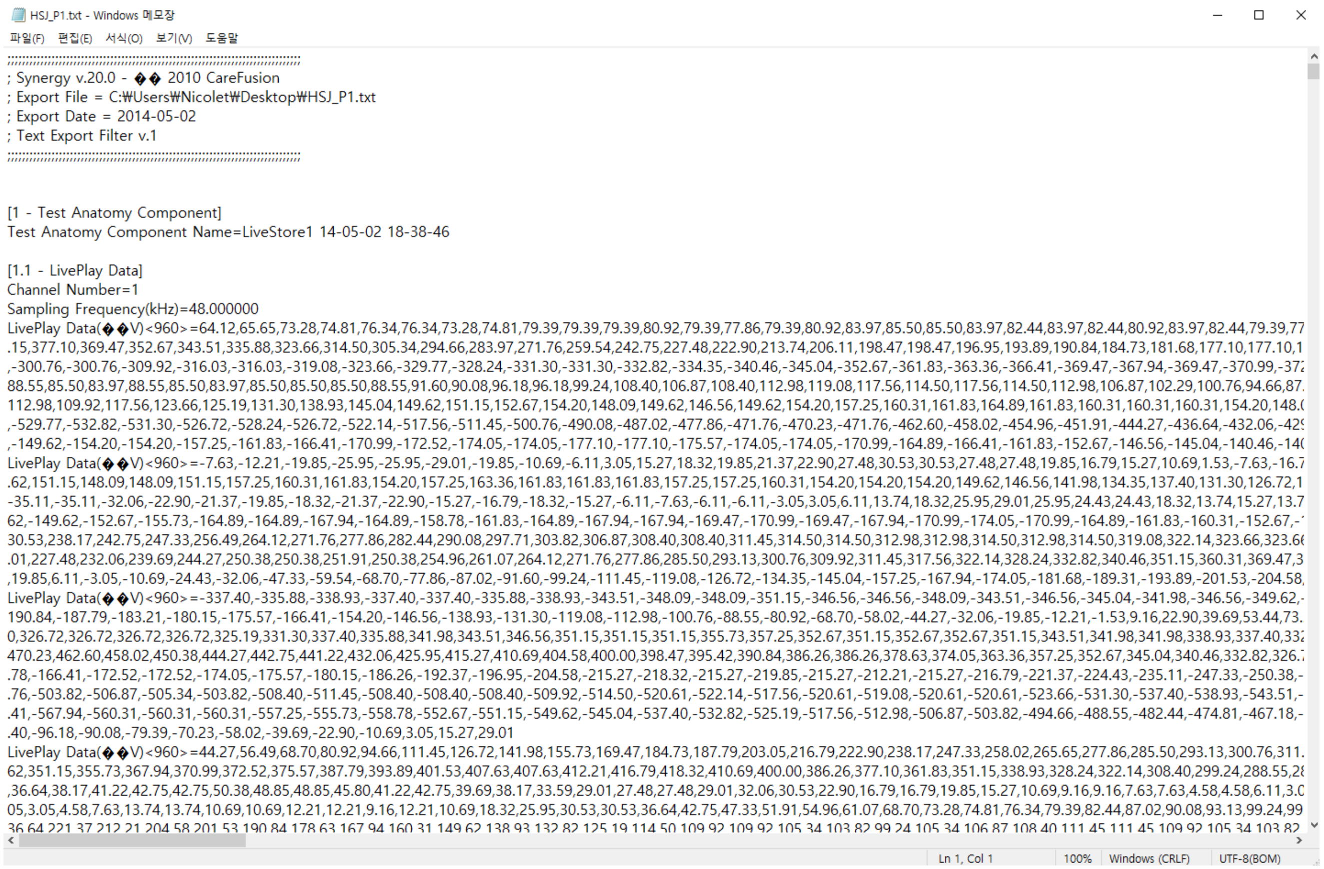
```
import csv

filename = '/content/MyDrive/My Drive/Colab Notebooks/data/data_csv.csv'
csv_data_list = []
f = open(filename,'r')
reader = csv.reader(f, delimiter=',')
# next(reader)    # 헤더라인 skip… 필요한 경우 사용
for line in reader:
    print(line, type(line))
    csv_data_list.append(line)
f.close()

print(csv_data_list)
```

```
['cnu'] <class 'list'>
['college of medicine'] <class 'list'>
['2020'] <class 'list'>
['sophomore'] <class 'list'>
['basic computer programming'] <class 'list'>
['python'] <class 'list'>
['## End of CSV file ##'] <class 'list'>
[['cnu'], ['college of medicine'], ['2020'], ['sophomore'], ['basic computer programming'], ['python'], ['## End of CSV file ##']]
```

# Data Example

## VEMP (VESTIBULAR EVOKED MYOGENIC POTENTIAL )

VEMP test is a neurophysiological examination technique used to diagnose the function of the otolithic organs( Inner Ear). There are two different types of VEMPs called the oVEMP and cVEMP. This test has been conducted in patients who are finding trouble with balance, visual fixation, posture, and lower muscular control. This test has played a key role in Ménière's disease, vestibular neuritis, otosclerosis and central disorders such as Multiple Sclerosis. The symptoms for testing are: Dizziness caused by loud sounds, Sensitivity to changes in pressure of the surrounding air, Hearing your own inner body sounds, such as your heartbeat or movements of your eyes or joints, Ringing in the affected ear.


HSJ_P1.txt - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8(BOM)

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Synergy v.20.0 - �� 2010 CareFusion
; Export File = C:₩Users₩Nicolet₩Desktop₩HSJ_P1.txt
; Export Date = 2014-05-02
; Text Export Filter v.1
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


[1 - Test Anatomy Component]
Test Anatomy Component Name=LiveStore1 14-05-02 18-38-46

[1.1 - LivePlay Data]
Channel Number=1
Sampling Frequency(kHz)=48.000000
LivePlay Data(��V)<960>=64.12,65.65,73.28,74.81,76.34,76.34,73.28,74.81,79.39,79.39,79.39,80.92,79.39,77.86,79.39,80.92,83.97,85.50,85.50,83.97,82.44,83.97,82.44,80.92,83.97,82.44,79.39,77
.15,377.10,369.47,352.67,343.51,335.88,323.66,314.50,305.34,294.66,283.97,271.76,259.54,242.75,227.48,222.90,213.74,206.11,198.47,198.47,196.95,193.89,190.84,184.73,181.68,177.10,177.10,1
,-300.76,-300.76,-309.92,-316.03,-316.03,-319.08,-323.66,-329.77,-328.24,-331.30,-331.30,-332.82,-334.35,-340.46,-345.04,-352.67,-361.83,-363.36,-366.41,-369.47,-367.94,-369.47,-370.99,-37
88.55,85.50,83.97,88.55,85.50,83.97,85.50,85.50,85.50,88.55,91.60,90.08,96.18,96.18,99.24,108.40,106.87,108.40,112.98,119.08,117.56,114.50,117.56,114.50,112.98,106.87,102.29,100.76,94.66,87.
112.98,109.92,117.56,123.66,125.19,131.30,138.93,145.04,149.62,151.15,152.67,154.20,148.09,149.62,146.56,149.62,154.20,157.25,160.31,161.83,164.89,161.83,160.31,160.31,160.31,154.20,148.
,-529.77,-532.82,-531.30,-526.72,-528.24,-526.72,-522.14,-517.56,-511.45,-500.76,-490.08,-487.02,-477.86,-471.76,-470.23,-471.76,-462.60,-458.02,-454.96,-451.91,-444.27,-436.64,-432.06,-42
,-149.62,-154.20,-154.20,-157.25,-161.83,-166.41,-170.99,-172.52,-174.05,-174.05,-177.10,-177.10,-175.57,-174.05,-174.05,-170.99,-164.89,-166.41,-161.83,-152.67,-146.56,-145.04,-140.46,-14
LivePlay Data(��V)<960>=-7.63,-12.21,-19.85,-25.95,-25.95,-29.01,-19.85,-10.69,-6.11,3.05,15.27,18.32,19.85,21.37,22.90,27.48,30.53,30.53,27.48,27.48,19.85,16.79,15.27,10.69,1.53,-7.63,-16.7
.62,151.15,148.09,148.09,151.15,157.25,160.31,161.83,154.20,157.25,163.36,161.83,161.83,161.83,157.25,157.25,160.31,154.20,154.20,154.20,149.62,146.56,141.98,134.35,137.40,131.30,126.72,1
-35.11,-35.11,-32.06,-22.90,-21.37,-19.85,-18.32,-21.37,-22.90,-15.27,-16.79,-18.32,-15.27,-6.11,-7.63,-6.11,-6.11,-3.05,3.05,6.11,13.74,18.32,25.95,29.01,25.95,24.43,24.43,18.32,13.74,15.27,13.7
62,-149.62,-152.67,-155.73,-164.89,-164.89,-167.94,-164.89,-158.78,-161.83,-164.89,-167.94,-167.94,-169.47,-170.99,-169.47,-167.94,-170.99,-174.05,-170.99,-164.89,-161.83,-160.31,-152.67,-
30.53,238.17,242.75,247.33,256.49,264.12,271.76,277.86,282.44,290.08,297.71,303.82,306.87,308.40,308.40,311.45,314.50,314.50,312.98,312.98,314.50,312.98,314.50,319.08,322.14,323.66,323.6
.01,227.48,232.06,239.69,244.27,250.38,250.38,251.91,250.38,254.96,261.07,264.12,271.76,277.86,285.50,293.13,300.76,309.92,311.45,317.56,322.14,328.24,332.82,340.46,351.15,360.31,369.47,3
,19.85,6.11,-3.05,-10.69,-24.43,-32.06,-47.33,-59.54,-68.70,-77.86,-87.02,-91.60,-99.24,-111.45,-119.08,-126.72,-134.35,-145.04,-157.25,-167.94,-174.05,-181.68,-189.31,-193.89,-201.53,-204.58,
LivePlay Data(��V)<960>=-337.40,-335.88,-338.93,-337.40,-337.40,-335.88,-338.93,-343.51,-348.09,-348.09,-351.15,-346.56,-346.56,-348.09,-343.51,-346.56,-345.04,-341.98,-346.56,-349.62,-
190.84,-187.79,-183.21,-180.15,-175.57,-166.41,-154.20,-146.56,-138.93,-131.30,-119.08,-112.98,-100.76,-88.55,-80.92,-68.70,-58.02,-44.27,-32.06,-19.85,-12.21,-1.53,9.16,22.90,39.69,53.44,73.
0,326.72,326.72,326.72,326.72,325.19,331.30,337.40,335.88,341.98,343.51,346.56,351.15,351.15,351.15,355.73,357.25,352.67,351.15,352.67,352.67,351.15,343.51,341.98,341.98,338.93,337.40,33
470.23,462.60,458.02,450.38,444.27,442.75,441.22,432.06,425.95,415.27,410.69,404.58,400.00,398.47,395.42,390.84,386.26,386.26,378.63,374.05,363.36,357.25,352.67,345.04,340.46,332.82,326.
.78,-166.41,-172.52,-172.52,-174.05,-175.57,-180.15,-186.26,-192.37,-196.95,-204.58,-215.27,-218.32,-215.27,-219.85,-215.27,-212.21,-215.27,-216.79,-221.37,-224.43,-235.11,-247.33,-250.38,-
.76,-503.82,-506.87,-505.34,-503.82,-508.40,-511.45,-508.40,-508.40,-508.40,-509.92,-514.50,-520.61,-522.14,-517.56,-520.61,-519.08,-520.61,-520.61,-523.66,-531.30,-537.40,-538.93,-543.51,-
.41,-567.94,-560.31,-560.31,-560.31,-557.25,-555.73,-558.78,-552.67,-551.15,-549.62,-545.04,-537.40,-532.82,-525.19,-517.56,-512.98,-506.87,-503.82,-494.66,-488.55,-482.44,-474.81,-467.18,-
.40,-96.18,-90.08,-79.39,-70.23,-58.02,-39.69,-22.90,-10.69,3.05,15.27,29.01
LivePlay Data(��V)<960>=44.27,56.49,68.70,80.92,94.66,111.45,126.72,141.98,155.73,169.47,184.73,187.79,203.05,216.79,222.90,238.17,247.33,258.02,265.65,277.86,285.50,293.13,300.76,311.
62,351.15,355.73,367.94,370.99,372.52,375.57,387.79,393.89,401.53,407.63,407.63,412.21,416.79,418.32,410.69,400.00,386.26,377.10,361.83,351.15,338.93,328.24,322.14,308.40,299.24,288.55,2
,36.64,38.17,41.22,42.75,42.75,50.38,48.85,48.85,45.80,41.22,42.75,39.69,38.17,33.59,29.01,27.48,27.48,29.01,32.06,30.53,22.90,16.79,16.79,19.85,15.27,10.69,9.16,9.16,7.63,7.63,4.58,4.58,6.11,3.0
05,3.05,4.58,7.63,13.74,13.74,10.69,10.69,12.21,12.21,9.16,12.21,10.69,18.32,25.95,30.53,30.53,36.64,42.75,47.33,51.91,54.96,61.07,68.70,73.28,74.81,76.34,79.39,82.44,87.02,90.08,93.13,99.24,99
36.64,221.37,212.21,204.58,201.53,190.84,178.63,167.94,160.31,149.62,138.93,132.82,125.19,114.50,109.92,109.92,105.34,103.82,99.24,105.34,106.87,108.40,111.45,111.45,109.92,105.34,103.82
```

http://www.baranagarspeech.co.in/services/vestibular-envoked-myogenic-potencial

## Python Scientific Modules

- Common scientific ecosystem modules
- **NumPy, SciPy, Pandas, Matplotlib**

Python Scientific Modules

## Python Scientific Modules

- **SciPy** is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing.

- **pandas** is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

## Machine Learning Frameworks

- 현업에서 사용되는 인공지능 개발 tool들이 Python Scientific Modules을 기반으로 구성되어 있음.

scikits learn
machine learning in Python

TensorFlow

K Keras

## Machine Learning Frameworks

- **Scikit-learn** is a free software machine learning library for the Python programming language. It is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

- **TensorFlow** is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

- **Keras** is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the **TensorFlow** library.

## Numpy

- A library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

- The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers.

- In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications.

- NumPy is open-source software and has many contributors.

https://en.wikipedia.org/wiki/NumPy

## ndarray

- The core functionality of NumPy is its "ndarray", for *n*-dimensional array, data structure.

- Fixed-size homogeneous multidimensional array
  - Vectorization, broadcasting 지원

- 성능 향상을 위해 같은 데이터 타입만을 요소로 가질 수 있고, 크기 역시 고정
  - 만약 크기를 변경하면 새로 메모리에 할당되고 이전 값은 삭제

- List, tuple 등의 시퀀스 자료형은 서로 다른 data type을 저장할 수 있고(heterogeneous sequence), 크기가 자동으로 커질 수 있음.

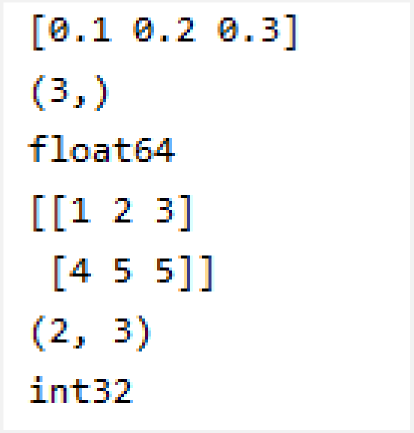
https://wikidocs.net/14569

# ndarray

```python
import numpy as np

x = np.array((0.1, 0.2, 0.3))

print(x)
print(x.shape)
print(x.dtype)

y = np.array(((1, 2, 3), (4, 5, 5)))

print(y)
print(y.shape)
print(y.dtype)
```

# ndarray

```python
import numpy as np

x = np.array((0.1, 0.2, 0.3))

print(x)
print(x.shape)
print(x.dtype)

y = np.array(((1, 2, 3), (4, 5, 5)))

print(y)
print(y.shape)
print(y.dtype)
```



```
[0.1 0.2 0.3]
(3,)
float64
[[1 2 3]
 [4 5 5]]
(2, 3)
int32
```

- dtype: data type 확인
- shape: data dimension 확인
    - 1차원 배열: (m,), 2차원 배열: (m,n), 3차원 배열: (p,q,r)

- x: float64 type, 크기 3의 1차원 배열
- y: int32 type, 크기 (2,3)의 2차원 배열

# ndarray 주요 속성

- shape : 배열의 형태
- dtype : 요소의 데이터 타입
  - int32, float32, etc.
- ndim : 차원수
  - x.ndim = 1, y.ndim=2 등이며 len(x.shape) 와 동일
- size : 요소의 개수
  - shape의 모든 값의 곱. x.size = 3, y.size=6 등
- itemsize : 요소 데이터 타입의 크기(byte 단위)
  - x.itemsize=8 등

## ndarray 주요 기능 리스트

- Indexing 및 array 처리
  - Concatenate, hstack, vstack
- Operations (연산) & Universal functions
  - np.linspace(-np.pi, np.pi, 100) , np,sin, np.cos, dot(), matmul(), @ 연산자, np.amin, np.amax, np.argmin, np.argmax, np.sort()
- Broadcasting
- Copy

## Data Read & Write

- np.loadtxt(), np.savetxt(): Text file Read & Write
    - Built-in package인 csv처럼 헤더 read는 불가능
    - Comma, Space로 구분된 Data 사용 시 편리
    - Default delimiter (구분자): Space
    - Comma 사용 시 delimiter=','로 지정
    - Skiprows: 초기 무시한 행 개수 지정
    - 필요 시 comment 문자 지정

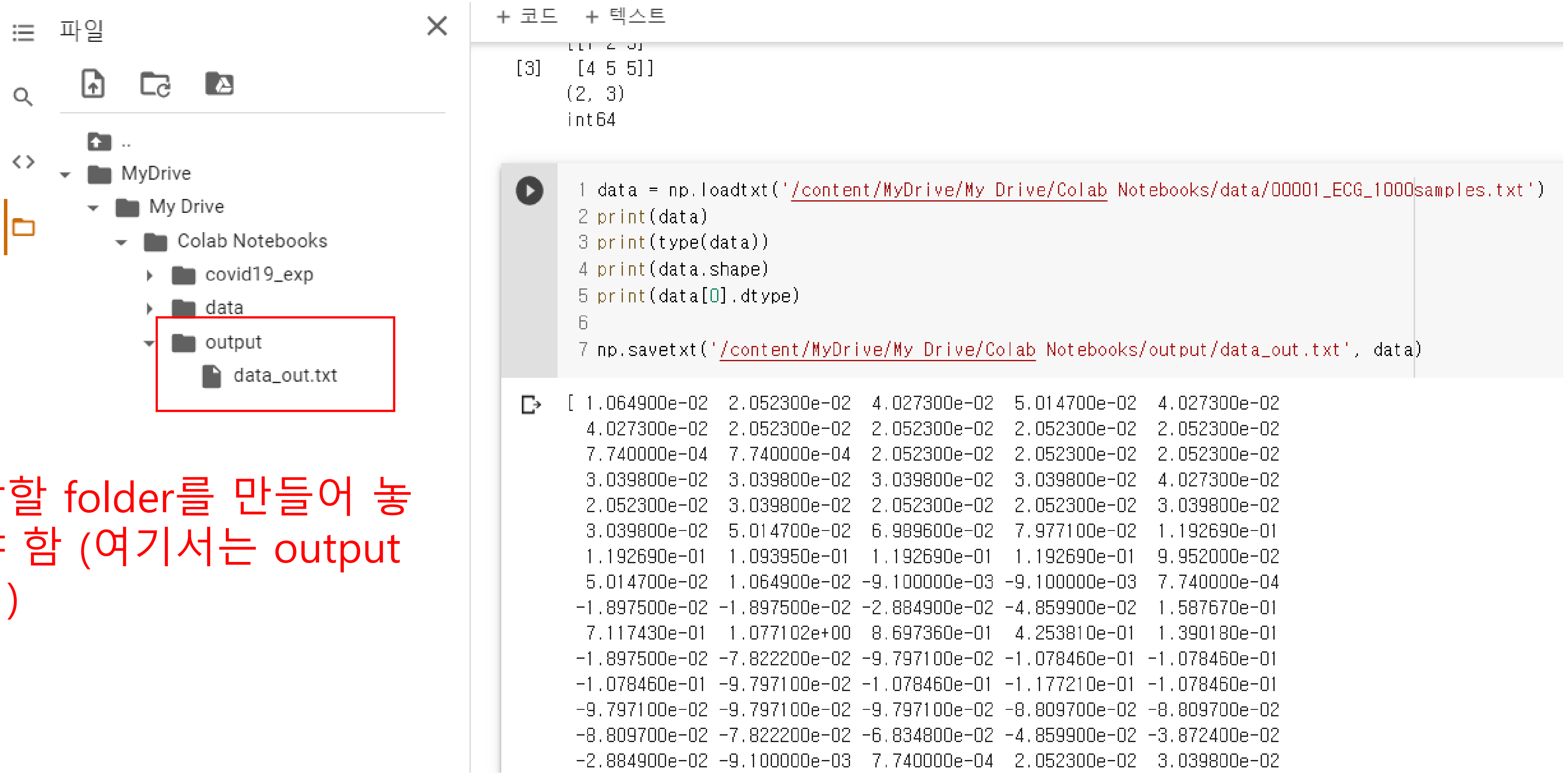
```
np.savetxt(fname, X, delimiter=' ', ...)
X= np.loadtxt(fname, delimiter=' ', skiprows=0,
comments='#', ...)
```

## Data Read & Write

```python
import numpy as np
```

```python
data = np.loadtxt('00001_ECG_1000samples.txt')
print(data)
print(type(data))
print(data.shape)
print(data[0].dtype)



np.savetxt('data_out.txt', data)
```

# Data Read & Write



```
[3]  [4 5 5]]
    (2, 3)
    int64
```

```
data = np.loadtxt('/content/MyDrive/My Drive/Colab Notebooks/data/00001_ECG_1000samples.txt')
print(data)
print(type(data))
print(data.shape)
print(data[0].dtype)

np.savetxt('/content/MyDrive/My Drive/Colab Notebooks/output/data_out.txt', data)
```

```
[ 1.064900e-02  2.052300e-02  4.027300e-02  5.014700e-02  4.027300e-02
  4.027300e-02  2.052300e-02  2.052300e-02  2.052300e-02  2.052300e-02
  7.740000e-04  7.740000e-04  2.052300e-02  2.052300e-02  2.052300e-02
  3.039800e-02  3.039800e-02  3.039800e-02  3.039800e-02  4.027300e-02
  2.052300e-02  3.039800e-02  2.052300e-02  2.052300e-02  3.039800e-02
  3.039800e-02  5.014700e-02  6.989600e-02  7.977100e-02  1.192690e-01
  1.192690e-01  1.093950e-01  1.192690e-01  1.192690e-01  9.952000e-02
  5.014700e-02  1.064900e-02 -9.100000e-03 -9.100000e-03  7.740000e-04
 -1.897500e-02 -1.897500e-02 -2.884900e-02 -4.859900e-02  1.587670e-01
  7.117430e-01  1.077102e+00  8.697360e-01  4.253810e-01  1.390180e-01
 -1.897500e-02 -7.822200e-02 -9.797100e-02 -1.078460e-01 -1.078460e-01
 -1.078460e-01 -9.797100e-02 -1.078460e-01 -1.177210e-01 -1.078460e-01
 -9.797100e-02 -9.797100e-02 -9.797100e-02 -8.809700e-02 -8.809700e-02
 -8.809700e-02 -7.822200e-02 -6.834800e-02 -4.859900e-02 -3.872400e-02
 -2.884900e-02 -9.100000e-03  7.740000e-04  2.052300e-02  3.039800e-02
```

저장할 folder를 만들어 놓아야 함 (여기서는 output 폴더)

# Data Read & Write

```
import numpy as np
```

```
data = np.loadtxt('00001_ECG_1000samples.txt')
print(data)
print(type(data))
print(data.shape)
print(data[0].dtype)



np.savetxt('data_out.txt', data)
```

```
  3.039800e-02  1.064900e-02 -9.100000e-03  7.740000e-04 -1.897500e-02
 -1.897500e-02 -2.884900e-02 -9.100000e-03  7.740000e-04  1.064900e-02
  3.039800e-02  6.002200e-02  6.989600e-02  6.989600e-02  6.989600e-02
  8.964500e-02  9.952000e-02  6.002200e-02  2.052300e-02 -9.100000e-03
 -1.897500e-02 -1.897500e-02 -1.897500e-02 -1.897500e-02 -1.897500e-02
 -4.859900e-02 -1.897500e-02  2.278890e-01  8.894860e-01  1.116601e+00
  7.907400e-01  3.858830e-01  1.390180e-01 -9.100000e-03 -4.859900e-02]
<class 'numpy.ndarray'>
(1000,)
float64
```

## Matplotlib

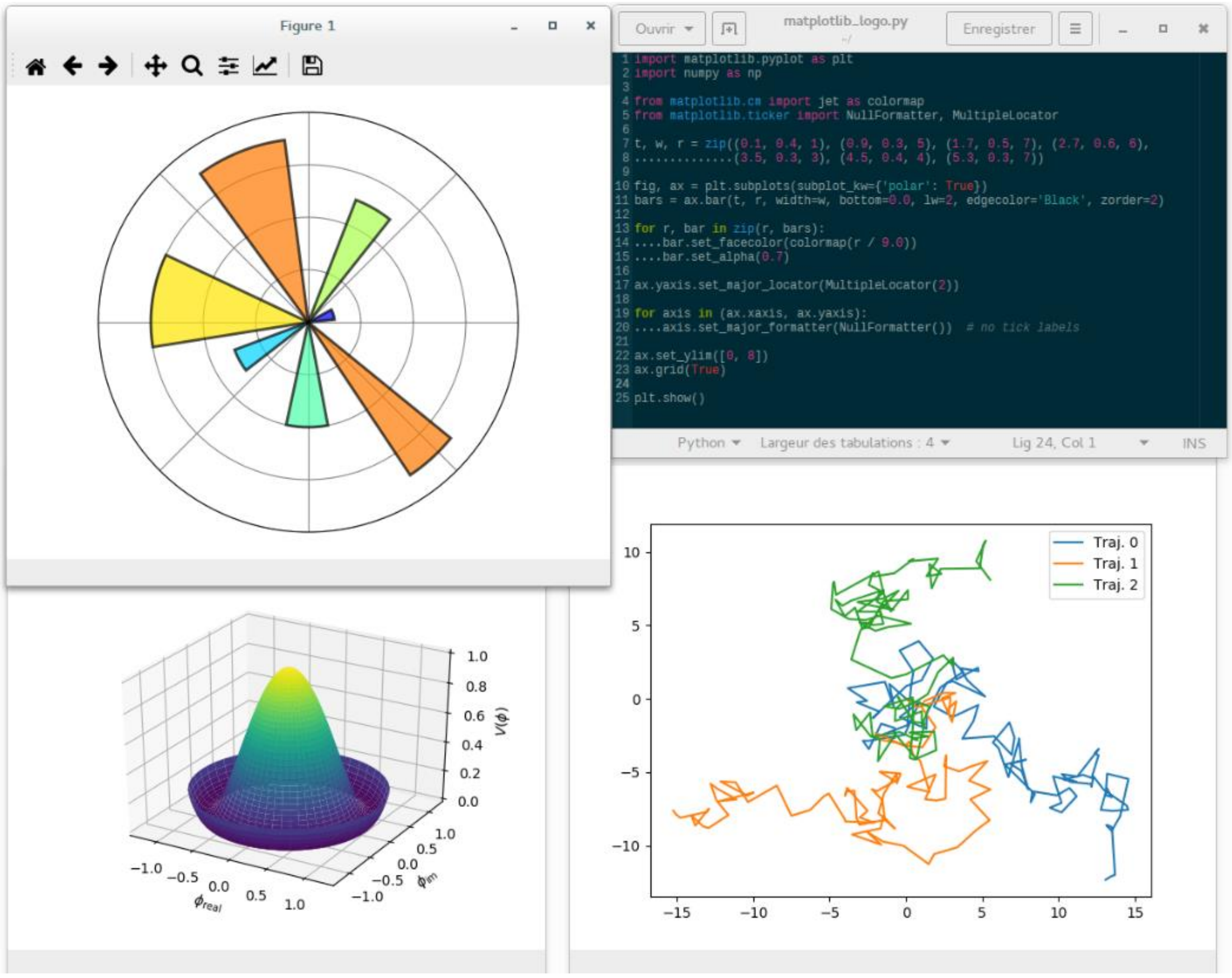
- Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

- It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.

- SciPy makes use of Matplotlib.

https://en.wikipedia.org/wiki/Matplotlib

# Matplotlib



```python
import matplotlib.pyplot as plt
import numpy as np

from matplotlib.cm import jet as colormap
from matplotlib.ticker import NullFormatter, MultipleLocator

t, w, r = zip((0.1, 0.4, 1), (0.9, 0.3, 5), (1.7, 0.5, 7), (2.7, 0.6, 6),
..............(3.5, 0.3, 3), (4.5, 0.4, 4), (5.3, 0.3, 7))

fig, ax = plt.subplots(subplot_kw={'polar': True})
bars = ax.bar(t, r, width=w, bottom=0.0, lw=2, edgecolor='Black', zorder=2)

for r, bar in zip(r, bars):
....bar.set_facecolor(colormap(r / 9.0))
....bar.set_alpha(0.7)

ax.yaxis.set_major_locator(MultipleLocator(2))

for axis in (ax.xaxis, ax.yaxis):
....axis.set_major_formatter(NullFormatter())  # no tick labels

ax.set_ylim([0, 8])
ax.grid(True)

plt.show()
```

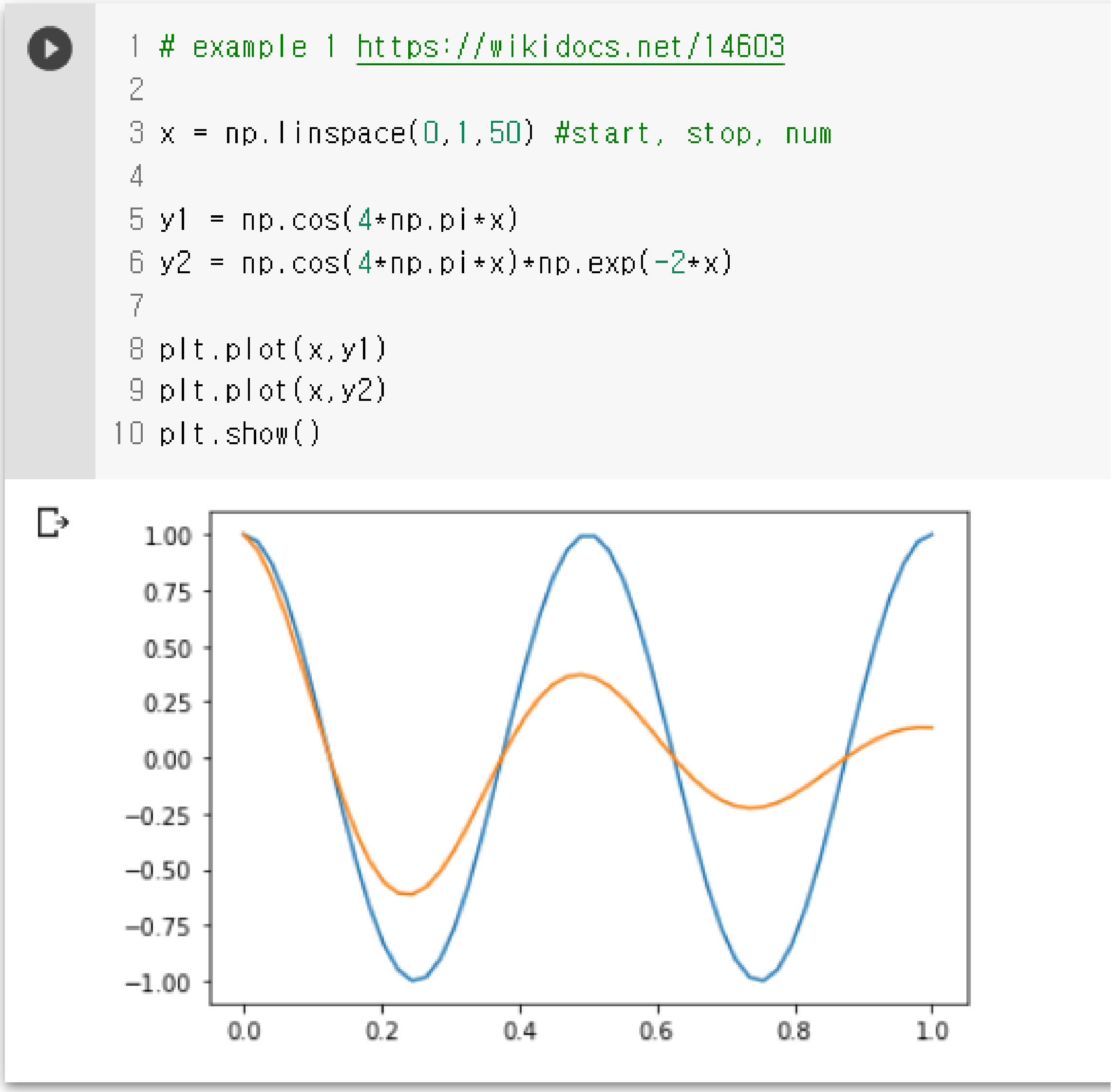
Python ▾ Largeur des tabulations : 4 ▾ Lig 24, Col 1 ▾ INS

## Plot

```
import numpy as np
```

```
# example 1 https://wikidocs.net/14603
import matplotlib.pyplot as plt

x = np.linspace(0,1,50) #start, stop, num

y1 = np.cos(4*np.pi*x)
y2 = np.cos(4*np.pi*x)*np.exp(-2*x)

plt.plot(x,y1)
plt.plot(x,y2)
plt.show()
```

# Plot

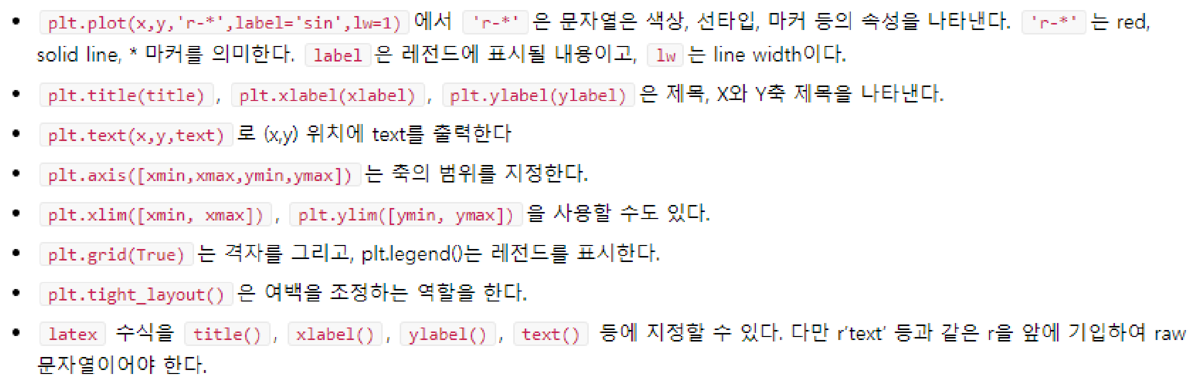


```
# example 1 https://wikidocs.net/14603

x = np.linspace(0,1,50) #start, stop, num

y1 = np.cos(4*np.pi*x)
y2 = np.cos(4*np.pi*x)*np.exp(-2*x)

plt.plot(x,y1)
plt.plot(x,y2)
plt.show()
```

## Plot – 꾸미기

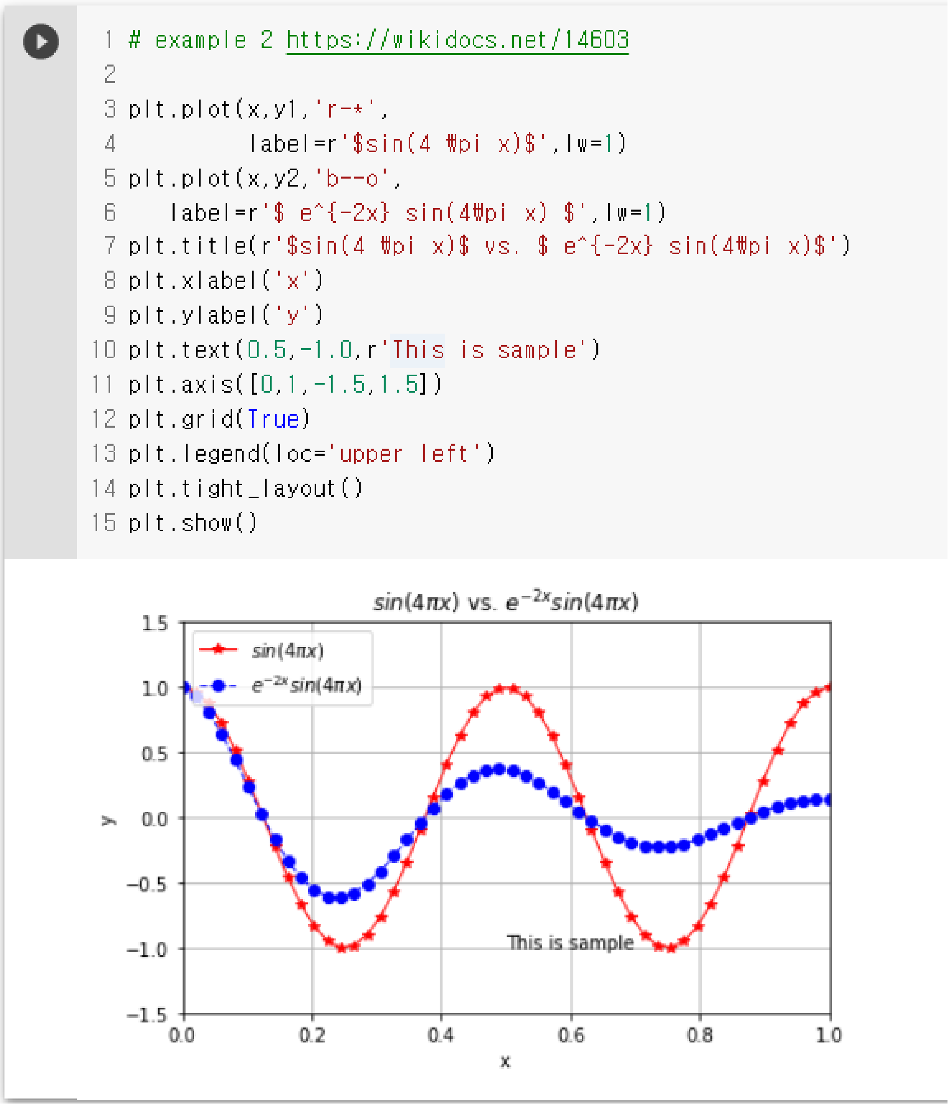
```python
import numpy as np
import matplotlib.pyplot as plt
```

```python
# example 2 https://wikidocs.net/14603
plt.plot(x,y1,'r-*',
         label=r'$sin(4 \pi x)$',lw=1)
plt.plot(x,y2,'b--o',
   label=r'$ e^{-2x} sin(4\pi x) $',lw=1)
plt.title(r'$sin(4 \pi x)$ vs. $ e^{-2x} sin(4\pi x)$')
plt.xlabel('x')
plt.ylabel('y')
plt.text(0.5,-1.0,r'This is sample')
plt.axis([0,1,-1.5,1.5])
plt.grid(True)
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```



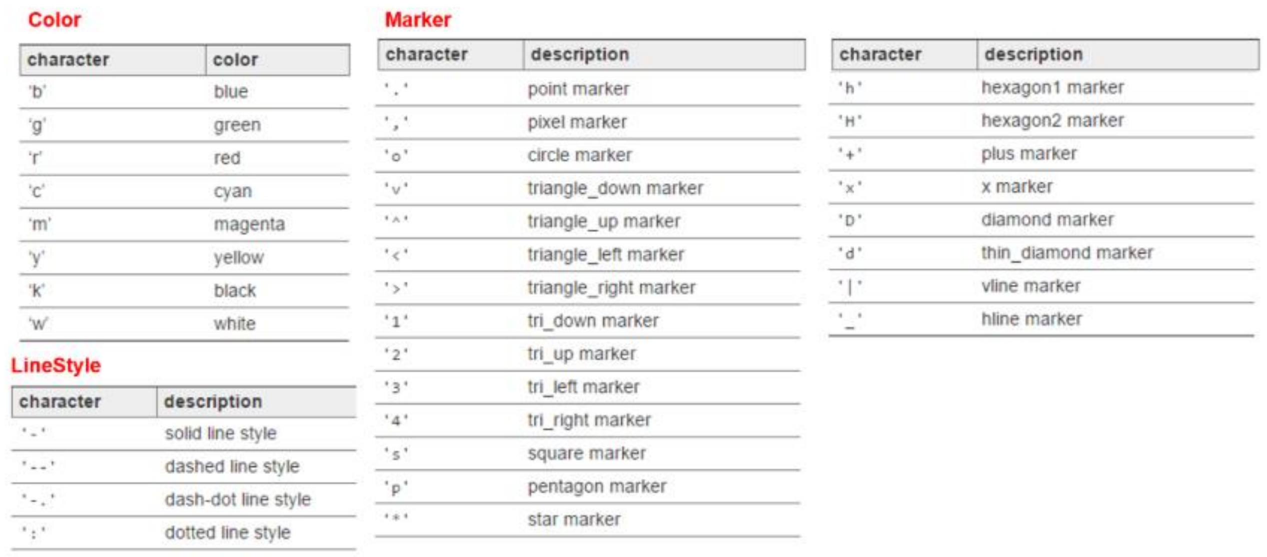
- `plt.plot(x,y,'r-*',label='sin',lw=1)` 에서 `'r-*'` 은 문자열은 색상, 선타입, 마커 등의 속성을 나타낸다. `'r-*'` 는 red, solid line, * 마커를 의미한다. `label` 은 레전드에 표시될 내용이고, `lw` 는 line width이다.
- `plt.title(title)`, `plt.xlabel(xlabel)`, `plt.ylabel(ylabel)` 은 제목, X와 Y축 제목을 나타낸다.
- `plt.text(x,y,text)` 로 (x,y) 위치에 text를 출력한다
- `plt.axis([xmin,xmax,ymin,ymax])` 는 축의 범위를 지정한다.
- `plt.xlim([xmin, xmax])`, `plt.ylim([ymin, ymax])` 을 사용할 수도 있다.
- `plt.grid(True)` 는 격자를 그리고, plt.legend()는 레전드를 표시한다.
- `plt.tight_layout()` 은 여백을 조정하는 역할을 한다.
- `latex` 수식을 `title()`, `xlabel()`, `ylabel()`, `text()` 등에 지정할 수 있다. 다만 r'text' 등과 같은 r을 앞에 기입하여 raw 문자열이어야 한다.

# Plot – 꾸미기



```
# example 2 https://wikidocs.net/14603

plt.plot(x,y1,'r-*',
         label=r'$sin(4 \pi x)$',lw=1)
plt.plot(x,y2,'b--o',
   label=r'$ e^{-2x} sin(4\pi x) $',lw=1)
plt.title(r'$sin(4 \pi x)$ vs. $ e^{-2x} sin(4\pi x)$')
plt.xlabel('x')
plt.ylabel('y')
plt.text(0.5,-1.0,r'This is sample')
plt.axis([0,1,-1.5,1.5])
plt.grid(True)
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```

# Plot Style Options 및 주요 함수



**Color**

| character | color |
|---|---|
| 'b' | blue |
| 'g' | green |
| 'r' | red |
| 'c' | cyan |
| 'm' | magenta |
| 'y' | yellow |
| 'k' | black |
| 'w' | white |

**LineStyle**

| character | description |
|---|---|
| '-' | solid line style |
| '--' | dashed line style |
| '-.' | dash-dot line style |
| ':' | dotted line style |

**Marker**

| character | description |
|---|---|
| '.' | point marker |
| ',' | pixel marker |
| 'o' | circle marker |
| 'v' | triangle_down marker |
| '^' | triangle_up marker |
| '<' | triangle_left marker |
| '>' | triangle_right marker |
| '1' | tri_down marker |
| '2' | tri_up marker |
| '3' | tri_left marker |
| '4' | tri_right marker |
| 's' | square marker |
| 'p' | pentagon marker |
| '*' | star marker |

| character | description |
|---|---|
| 'h' | hexagon1 marker |
| 'H' | hexagon2 marker |
| '+' | plus marker |
| 'x' | x marker |
| 'D' | diamond marker |
| 'd' | thin_diamond marker |
| '\|' | vline marker |
| '_' | hline marker |

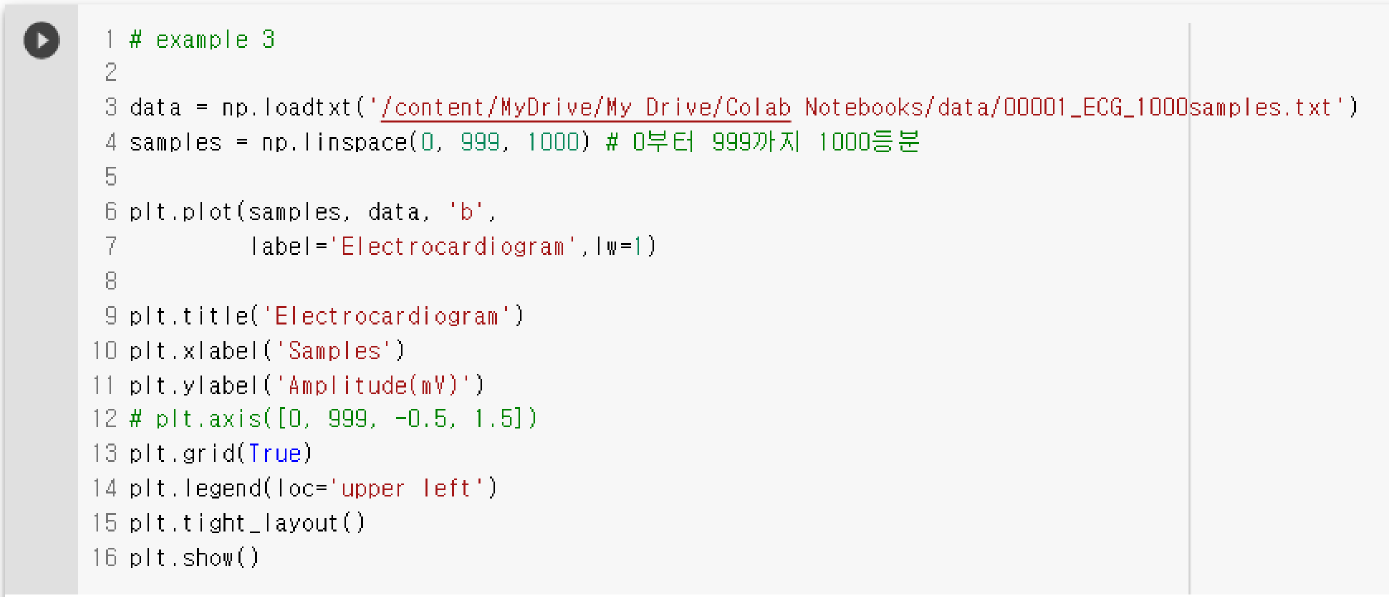
- `plot()`
- `subplot()`
- `title()`
- `xlabel()`
- `ylabel()`
- `axis()`
- `xlim()`
- `ylim()`
- `tight_layout()`
- `grid()`
- `legend()`
- `show()`
- `figure()`
- `text()`
- `subplots()`

# 입력 Data Plot

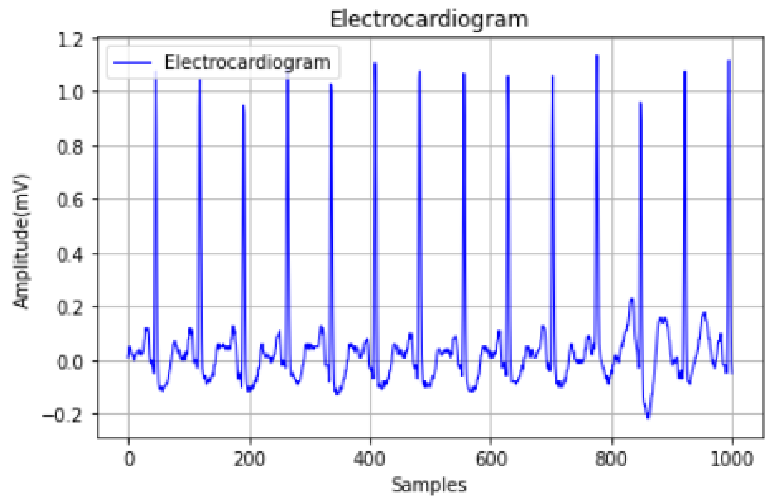
```python
import numpy as np
import matplotlib.pyplot as plt
```

```python
data = np.loadtxt('00001_ECG_1000samples.txt')
samples = np.linspace(0, 999, 1000) # 0부터 999까지 1000등분

plt.plot(samples, data, 'b',
         label='Electrocardiogram',lw=1)

plt.title('Electrocardiogram')
plt.xlabel('Samples')
plt.ylabel('Amplitude(mV)')
plt.grid(True)
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```

## 입력 Data Plot



```
# example 3

data = np.loadtxt('/content/MyDrive/My Drive/Colab Notebooks/data/00001_ECG_1000samples.txt')
samples = np.linspace(0, 999, 1000) # 0부터 999까지 1000등분

plt.plot(samples, data, 'b',
         label='Electrocardiogram',lw=1)

plt.title('Electrocardiogram')
plt.xlabel('Samples')
plt.ylabel('Amplitude(mV)')
# plt.axis([0, 999, -0.5, 1.5])
plt.grid(True)
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```



## [Wrap-up] Python 기본

- 자료형
- 연산자
- 제어문
- 함수
- 클래스/모듈/패키지
- 파일 입출력
- Numpy
- Matplotlib