

# Basic Python

Pengenalan Bahasa  
Pemrograman Python



# Instalasi Python Notebook

01

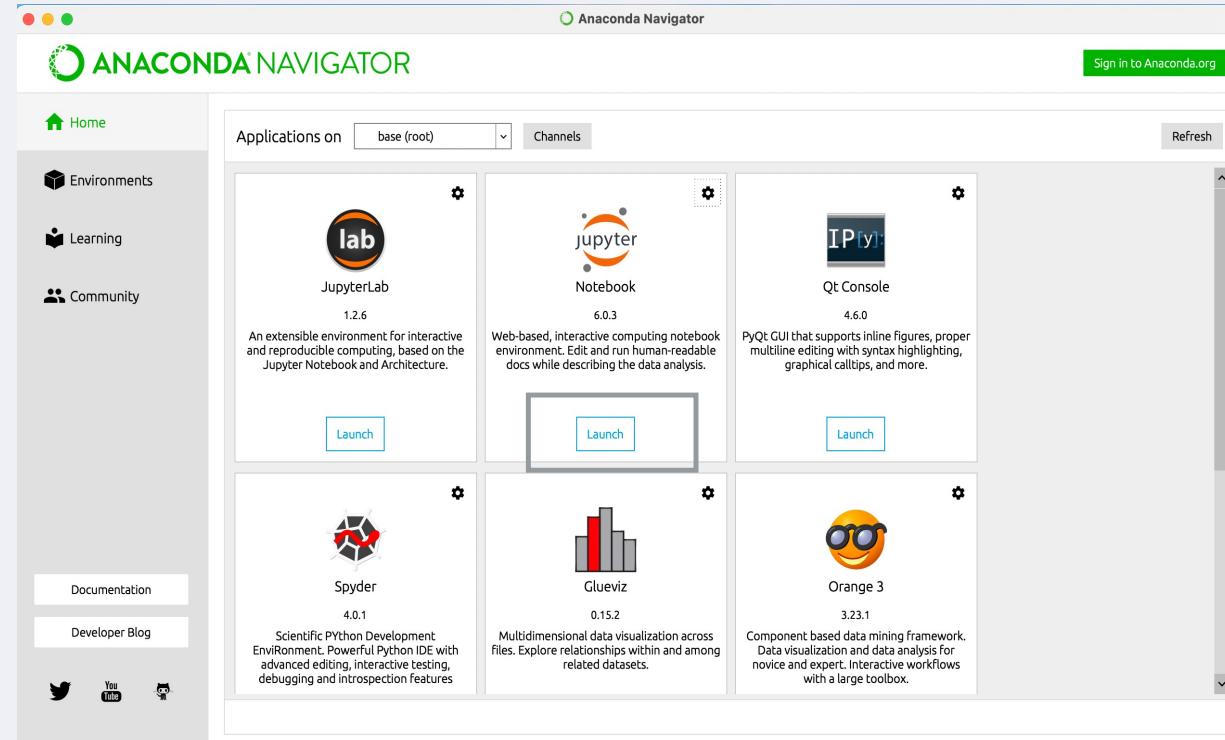
Download Anaconda Navigator sesuai OS yang Anda pakai dari link:  
<https://www.anaconda.com/product/s/individual#Downloads>

02

Buka aplikasi Anaconda Navigator

03

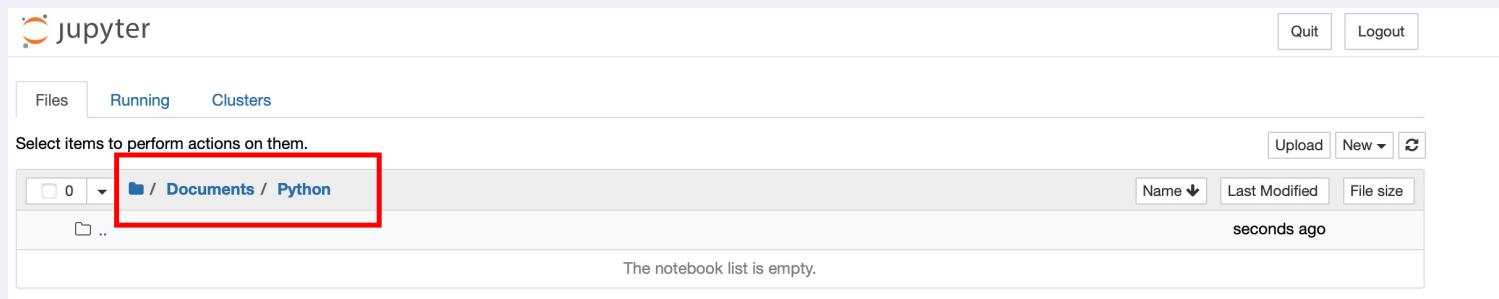
Buka jupyter notebook



# Instalasi Python Notebook (2)

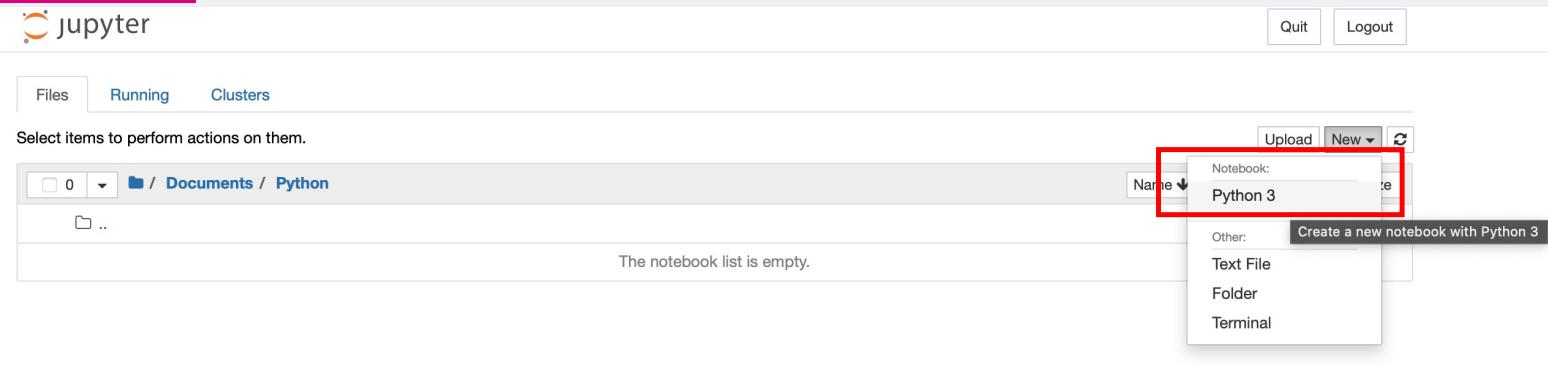
04

Tentukan direktori python notebook anda



05

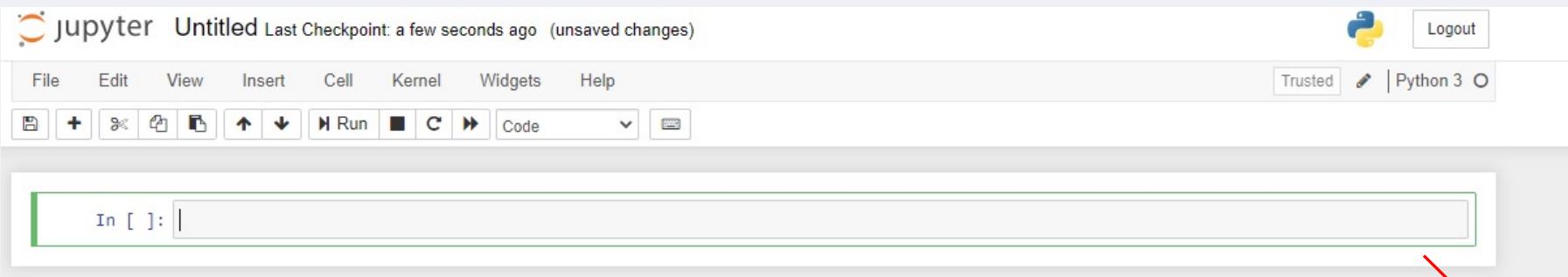
Klik new > python 3



# Instalasi Python Notebook (2)

06

Coding dilakukan pada cell yang ada di dalam notebook



07

Klik “+” untuk menambah cell baru

08

Klik “Run” untuk menjalankan kode yang ada di dalam cell yang di-highlight

09

Klik “Cell” + “Run All” untuk menjalankan kode yang ada di semua cell

10

Klik “Kernel” + “Restart & Clear Output” untuk menghapus semua output dari cell

11

Cell yang telah dijalankan kodanya akan terus berfungsi sampai dilakukan restart

# 1. Fungsi dan Argumen

**FUNGSI** adalah kode yang digunakan untuk menjalankan suatu aksi.

01 Aksi dari fungsi akan dilakukan kepada suatu **argumen**.

02 Contoh fungsi adalah **print()**

✓ **print()** digunakan untuk menampilkan argumen pada output

✓ Argumen yang diterima oleh fungsi **print()**, antara lain:

- Teks / string

```
print("Hello World")
```

Hello World

- Operasi aritmatik

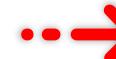
```
print(1+3)
```

4

- Output dari fungsi lain

```
print(type("Hello World"))
```

<class 'str'>



```
print(len("Hello World"))
```

11



Tipe dari argumen "Hello World" adalah string

Argumen "Hello World" mengandung 11 karakter

# Latihan 1: Fungsi dan Argumen

1

Pada cell ketikkan kode:

- `print("NAMA ANDA")`
- `print("TANGGAL HARI INI")`
- `print(100/10)`
- `print(2*5)`
- `print(type(100/10))`
- `print(len("NAMA ANDA"))`

Klik “Run”.

2

## 2. Variabel

**VARIABEL** adalah container yang dipakai untuk menyimpan data.

01

Variabel dipakai agar tidak perlu mengulang menuliskan data setiap akan menjalankan suatu fungsi terhadap data tersebut

02

Variabel dibuat dengan menuliskan nama variable diikuti dengan '='

```
tabungan = 100
```

```
print(type(tabungan))
```

```
<class 'int'>
```

```
nama = "Agung Surya"
```

```
print(nama)
```

```
Agung Surya
```



Tipe dari data yang disimpan di variabel tabungan adalah integer

# 3. Tipe Data

Beberapa tipe data yang dikenal oleh PYTHON:

01 Integer: bilangan bulat

```
a = 56  
print(type(a))  
<class 'int'>
```

```
b = 56.46
```

```
print(type(b))  
<class 'float'>
```

```
c = "lima puluh enam"
```

```
print(type(c))  
<class 'str'>
```

02 Float: bilangan pecahan

```
d = 56 + 1 == 57
```

```
e = 56 + 1 != 57
```

```
print(d)  
print(e)  
print(type(d))  
print(type(e))
```

03 String: teks

```
True  
False  
<class 'bool'>  
<class 'bool'>
```

04 Boolean: true / false

# Latihan 2: Variabel dan Tipe Data

Buat 4 variabel:

01



Buat 4 variable dengan 4 tipe data berbeda: int, float, string, boolean



02

Identifikasi tipe-tipe data pada 4 variabel yang telah dibuat menggunakan fungsi `print()` dan `type()`



**LIST** adalah kumpulan data yang terstruktur dengan urutan tertentu.

01

List dibuat dengan menuliskan "[" dan "]" pada data-data yang dipisahkan dengan ","

02

Dalam list, tiap data memiliki index dengan urutan dari 0, 1, 2, dst

- Index diperlukan untuk **subsetting**: mengidentifikasi/memanggil data dengan urutan tertentu dari list



```
print(c[0])
print(c[1])
print(c[3])
print(c[-1])
```

```
True
35.46
Agung surya
Agung surya
```



Indeks -1 menunjukkan data dengan urutan terakhir

```
a = 100
```

```
b = "Agung surya"
```

```
c = [True, 35.46, a, b]
```

```
print(c)
```

```
[True, 35.46, 100, 'Agung surya']
```

```
print(type(c))
```

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

# 5. Melakukan slicing pada list

**SLICING** adalah proses memotong list dengan range indeks tertentu

01

Range indeks ditulis dengan [“indeks awal” : “indeks akhir”]

02

Indeks awal akan diikutsertakan pada output

03

Indeks akhir tidak diikutsertakan pada output

```
c = [True, 35.46, 100, 'Agung surya']

print(type(c))
<class 'list'>

print(c[0:3])
[True, 35.46, 100]

print(c[:3])
[True, 35.46, 100]

print(c[1:])
[35.46, 100, 'Agung surya']
```

→ Data dengan indeks 0 diikutsertakan, data dengan indeks 3 tidak diikutsertakan

→ Apabila slicing dilakukan dari awal list, indeks awal tidak perlu dituliskan

→ Apabila slicing dilakukan sampai akhir list, indeks akhir tidak perlu dituliskan

# 6. Memodifikasi List

01

Menambah elemen list

```
d = [1, 3, 2, 0]
d = d + [5, 7]
print(d)
```

[1, 3, 2, 0, 5, 7]

03

Menghapus elemen list

```
d = [1, 3, 2, 0]
del(d[0])
print(d)
```

[3, 2, 0]

02

Mengganti elemen list

```
d = [1, 3, 2, 0]
d[1] = 4
print(d)
```

[1, 4, 2, 0]

04

Mengurutkan elemen list

```
d = [1, 3, 2, 0]
d = sorted(d)
print(d)
```

[0, 1, 2, 3]



Fungsi sorted() tidak bisa dipakai untuk list yang elemennya merupakan gabungan tipe string dan bilangan

```
d = [1, 3, 2, 0]
d = sorted(d, reverse = True)
print(d)
```

[3, 2, 1, 0]



Fungsi sorted() merupakan contoh fungsi yang menerima multiple argument

# Latihan 3: List

Buat 4 variabel:

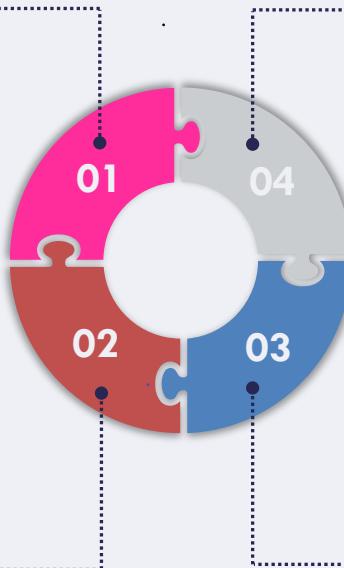
nama = "NAMA ANDA"

jarak\_km = 5

jarak\_mil = 5 \* 0.62

jarak\_beda = jarak\_km == jarak\_mil

↓  
Jadikan list!



~~Identifikasi tipe-tipe data pada 1 variabel~~

~~yang telah dibuat menggunakan fungsi~~

~~print() dan type()~~

• Slice 2 data terakhir dari list yang dibuat

↓  
di print

• Slice 2 data pertama dari list yang dibuat

↓  
di print

1. Pakai list yang sudah dibuat di latihan sebelumnya.
2. Tambahkan 2 elemen di akhir list
3. modifikasi elemen index 2 dengan elemen lain
4. delete elemen index 3
5. print hasil modifikasi list

**METODE** adalah fungsi yang dimiliki oleh objek

01

Contoh metode yang dimiliki oleh objek string:

- `upper()` digunakan untuk mengubah semua karakter pada objek menjadi huruf kapital.

```
a = "Agung Surya"  
b = a.upper()  
print(b)
```

AGUNG SURYA

- `count()` digunakan untuk menghitung karakter tertentu pada objek

```
a = "Agung Surya"  
b = a.count("u")  
print(b)
```

2

02

Contoh metode yang dimiliki oleh objek list:

- `index()` digunakan untuk mengidentifikasi indeks dari elemen tertentu

```
a = [1, 3, 2, 0]  
a.index(3)
```

1

- `count()` digunakan untuk menghitung elemen tertentu pada list

```
a = ["b", "b", "c", "d"]  
a.count("b")
```

2

1. buat 1 variabel berisi data string, gunakan metode .upper() dan .count() pada variabel ini, print outputnya
2. buat 1 variabel berupa list, gunakan metode .index() dan .count() pada variabel ini, print outputnya

# 8. Mengimpor Modul

**MODUL** adalah file berisikan kumpulan definisi dan pernyataan python.

01

Modul perlu diimpor untuk mengakses definisi yang terdapat didalamnya

02

Contohnya adalah modul **math** yang didalamnya terdapat definisi konstanta **pi**.

```
import math  
print(math.pi)
```

3.141592653589793

```
#Berapa luas lingkaran dengan jari-jari 2 cm?  
L = 2**2 * math.pi  
print(str(L)+" cm^2")
```

12.566370614359172 cm<sup>2</sup>

03

Modul juga bisa diimpor secara selektif, di mana hanya definisi atau pernyataan tertentu yang diimpor

```
from math import pi  
print(pi)
```

3.141592653589793

```
#Berapa luas lingkaran dengan jari-jari 2 cm?  
L = 2**2 * pi  
print(str(L)+" cm^2")
```

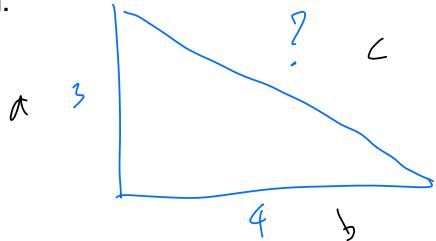
12.566370614359172 cm<sup>2</sup>

①

Dengan menggunakan variabel pi dari modul math. Hitung volume bola dengan jari-jari 12 cm.

$$\text{Volume bola} = \frac{4}{3} * \pi * r^3$$

② Dengan menggunakan fungsi sqrt dari modul math. Hitung sisi miring dari suatu segita tiga siku-siku dengan:



$$c = \sqrt{a^2 + b^2}$$

# 9. Numpy Array

NUMPY adalah modul pada python yang dapat membuat dan mengolah struktur data array

01

Elemen array bersifat homogen (satu tipe)

02

Struktur data array optimal untuk analisis numerik

03

Data array dapat dibuat dari list

```
#Mengimpor modul numpy
import numpy as np

#Mengkonversi list menjadi array
a = [0, 1, 2, 4]
b = np.array(a)

#Memeriksa dan menampilkan array yang dibuat
print(type(b))
print(b)

<class 'numpy.ndarray'>
[0 1 2 4]
```

04

Operasi aritmatik dapat dilakukan pada data array

```
c = b + 1
d = b ** 2
print(b)
print(c)
print(d)
print(b * c)
```

```
[0 1 2 4]
[1 2 3 5]
[ 0  1  4 16]
[ 0  2  6 20]
```

# 10. 2D Numpy Array

01

Array dapat berbentuk 2 dimensi

02

Array 2D dibuat dengan mengkonversi list of list

atribut → variabel  
yang  
dimiliki  
objek

```
# membuat list of list dengan variabel 'c' dari 2 list 'a' dan 'b'

a = [0, 1, 2, 4]
b = [3, 5, 7, 9]
c = [a, b] •.....•

print(c)
print(type(c))

[[0, 1, 2, 4], [3, 5, 7, 9]]
<class 'list'>
```

c adalah list yang elemen-elemennya  
berupa list

```
# membuat array 2D dengan variabel 'd' dari list 'c'

import numpy as np
d = np.array(c)

print(d)
print(type(d))
print(d.shape) •.....•

[[0 1 2 4]
 [3 5 7 9]]
<class 'numpy.ndarray'>
(2, 4)
```

Shape merupakan atribut atau variable yang  
dimiliki oleh objek array.

(2, 4) menunjukkan bahwa array mengandung 2  
row dan 4 column



# 11. Subsetting 2D Array

01

Subset pada 2D Array dilakukan dengan menuliskan row dan column (dihitung dari 0, 1, 2 dst.) dengan format  
nama\_array[nomor\_row,nomor\_column]

02

Subset juga bisa dilakukan dengan menggunakan range row atau column (slicing)

```
a = [0, 1, 2, 4]
b = [3, 5, 7, 9]
c = [a, b]

import numpy as np
d = np.array(c)

print(d)
print(d[0,2])
print(d[:,1:3])
```

[[0 1 2 4]  
 [3 5 7 9]]  
2  
[[1 2]  
 [5 7]]

Mengidentifikasi elemen pada row 1 dan column 3  
(yaitu 2)

Mengidentifikasi elemen-elemen pada semua row dan  
kolom 2-3

# 12. Analisis Statistika Dasar pada Array

Modul numpy memiliki fungsi untuk menjalankan analisis statistic dasar seperti:

-  min()
-  max()
-  sum()
-  mean()
-  median()

```
a = [0, 1, 2, 3, 5, 9]
b = np.array(a)

print(np.min(b))
print(np.max(b))
print(np.sum(b))
print(np.mean(b))
print(np.median(b))
```

```
0
9
20
3.333333333333335
2.5
```

# 13. Array generator

Modul numpy memiliki beberapa fungsi untuk men-generate array

- ✓ np.random.rand() → men-generate array dengan angka float antara 0-1
- ✓ np.random.randint() → men-generate array dengan angka integer
- ✓ np.arange → men-generate array dengan menentukan jarak antar angkanya
- ✓ np.linspace → men-gerate array dengan menentukan jumlah angka yang diinginkan dalam array dengan jarak yang konstan

```
import numpy as np

np.random.rand(10)

array([0.05211176, 0.30335649, 0.23113106, 0.91397175, 0.25675568,
       0.60495387, 0.54836746, 0.94938337, 0.47720038, 0.58999834])

np.random.randint(1,10,5)

array([9, 9, 3, 1, 9])

np.arange(0,100,20)

array([ 0, 20, 40, 60, 80])

np.linspace(0,100,21)

array([ 0., 5., 10., 15., 20., 25., 30., 35., 40., 45., 50.,
       55., 60., 65., 70., 75., 80., 85., 90., 95., 100.])
```

$$\begin{array}{rcl} 0 \rightarrow 10 & a + b = 10 \\ 1 \rightarrow 20 & 1 \times a + b = 20 \\ & \hline & -a = -10 \\ & a = 10 & \\ & b = 10 & \end{array}$$

# Latihan 4: Array

01

Import modul numpy

02

Buat 2D Array “c” dari 2  
list:  
`a = [0,1,2,3]`  
`b = [2,0,9,2]`

03

Subset data pada  
posisi row 1 dan  
kolom 3 dari Array  
yang dibuat

04

Tentukan nilai mean dan  
median dari nilai yang  
ada di Array  
menggunakan fungsi  
yang ada di dalam  
modul numpy.

# Latihan 4: Array

05

Generate array berisi 10  
float dengan range ~~0-100~~

random ✓  
25 - 100

06

Generate array berisi 10  
integer dengan range 0-  
10

random

07

Generate array berisi  
angka dengan range  
0-10 dengan jarak  
antar angkanya 2

↓  
gunakan  
np.arange

08

Generate array berisi 5  
angka dengan range 0-  
10 dengan jarak antar  
angka yang tetap

↓  
gunakan  
np.linspace

$$0 \rightarrow 25$$

$$1 \rightarrow 50$$

$$0a + b = 25$$

$$1a + b = 50$$

—

$$-a = -25$$

$$a = 25$$

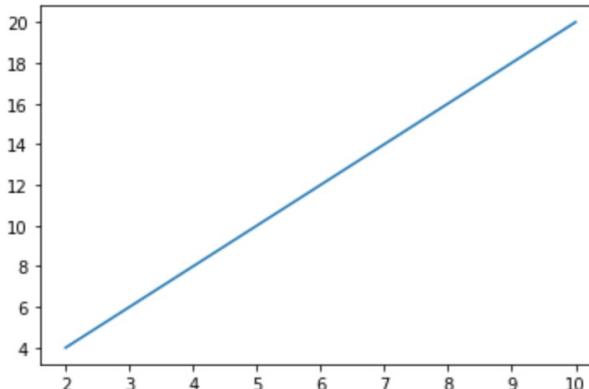
$$b = 50$$

# 14. Membuat Line Plot

## MODUL MATPLOTLIB

adalah salah satu modul yang memberikan cara untuk membuat plot

```
#menentukan posisi titik pada plot dengan membuat list posisi axis-x dan posisi axis-y  
  
#list posisi axis x diberi variabel 'x'  
x = [2, 4, 6, 8, 10]  
  
#list posisi axis y diberi variabel 'y'  
y = [4, 8, 12, 16, 20]  
  
#mengimpor kumpulan fungsi pyplot dari modul matplotlib  
import matplotlib.pyplot as plt  
  
#melakukan plot dengan fungsi .plot()  
plt.plot(x, y)  
  
#menampilkan plot dengan fungsi .show()  
plt.show()
```



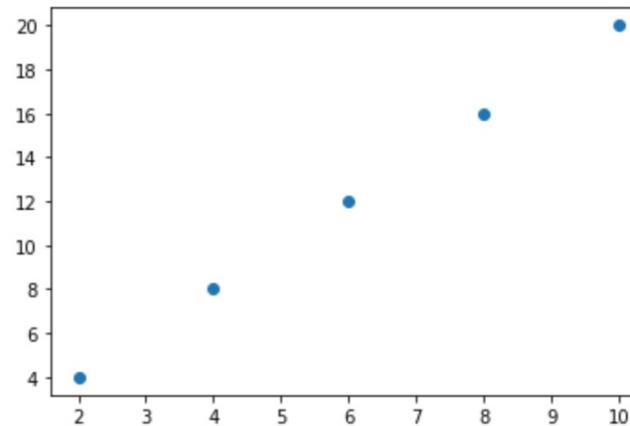
# 15. Membuat Scatterplot

Scatterplot dapat dibuat dengan menjalankan fungsi scatter() dari modul matplotlib

```
x = [2, 4, 6, 8, 10]
y = [4, 8, 12, 16, 20]

import matplotlib.pyplot as plt

#membuat scatter plot dengan fungsi scatter()
plt.scatter(x, y)
plt.show()
```



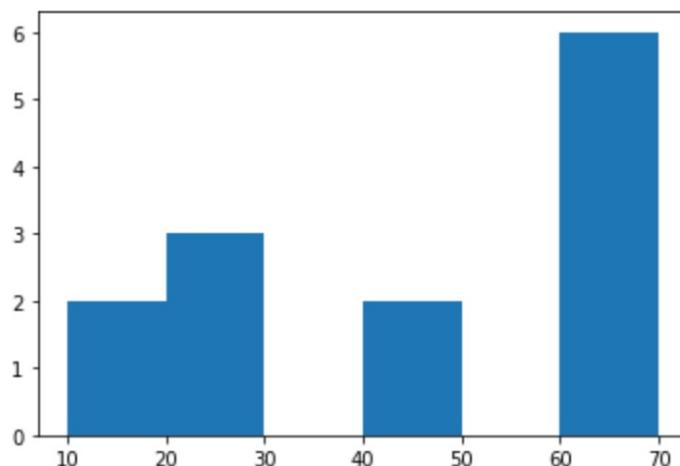
# 16. Membuat histogram

**Fungsi** adalah kode yang digunakan untuk menjalankan suatu aksi.

```
x = [10, 10, 20, 20, 20, 40, 40, 70, 70, 70, 70, 70, 70, 70]

import matplotlib.pyplot as plt

#membuat histogram dengan fungsi hist()
plt.hist(x, bins = 6)
plt.show()
```



► Fungsi **hist()** menerima argumen **bins** untuk menentukan jumlah **batang** yang dibuat

# 17. Mengedit Plot

Beberapa fungsi untuk mengedit plot:

```
x = [2, 4, 6, 8, 10]
y = [4, 8, 12, 16, 20]

import matplotlib.pyplot as plt

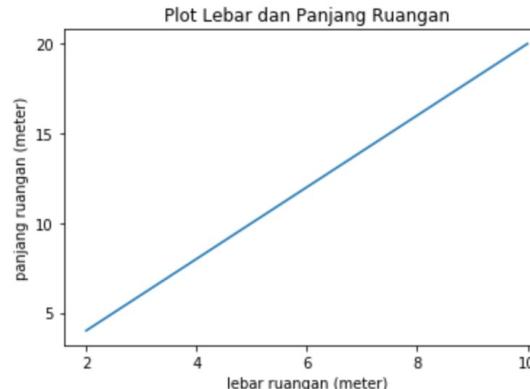
plt.plot(x, y)

#mengedit judul plot
plt.title('Plot Lebar dan Panjang Ruangan')

#mengedit label pada x-axis dan y-axis
plt.xlabel('lebar ruangan (meter)')
plt.ylabel('panjang ruangan (meter)')

#mengedit nilai yang ditampilkan pada x-axis dan y-axis
plt.xticks([2, 4, 6, 8, 10])
plt.yticks([5, 10, 15, 20])

plt.show()
```



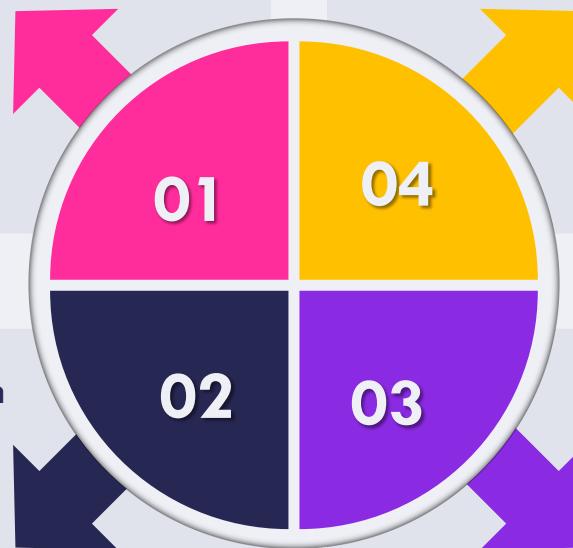
# Latihan 5: Plotting

Import matplotlib.pyplot

Buat line plot dan scatter plot dengan nilai axis berikut:

$x = [10, 20, 30, 40, 50]$

$y = [50, 40, 30, 20, 10]$



Buat histogram dengan 5 bins dari data-data berikut:

$z = [0, 0, 0, 100, 100, 200, 200, 200, 400, 500, 500, 500, 500]$

Tambahkan label axis x, label axis y, dan judul plot pada plot yang dibuat (bebas)

# 18. Dictionary

01

**Dictionary** adalah struktur data yang elemennya berupa pasangan key dan value

02

**Dictionary** dapat dibuat dengan menuliskan {key1: value1, key2: value2}

```
# Membuat dictionary dengan variabel 'ibukota'
ibukota = {'Indonesia': 'Jakarta', 'USA': 'Washington DC', 'UK': 'London', 'Thailand': 'Bangkok'}

print(ibukota)
print(type(ibukota))

# Mengidentifikasi value suatu key dari dictionary
print(ibukota['Indonesia'])

# Menambah elemen dictionary
ibukota['France'] = 'Rome'
print(ibukota)

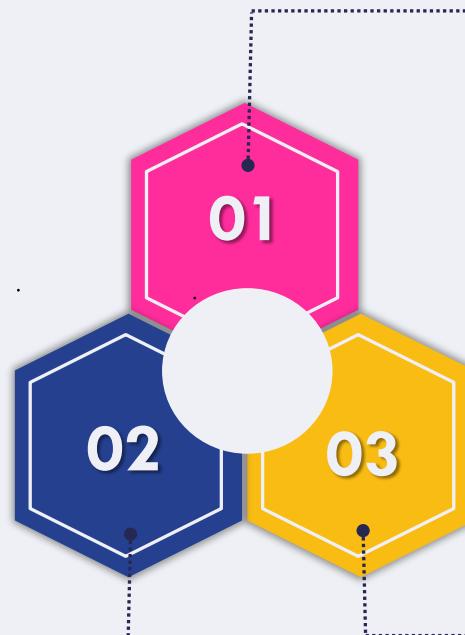
# Mengedit elemen dictionary
ibukota['France'] = 'Paris'
print(ibukota)

# Menghapus elemen dictionary
del ibukota['Indonesia']
print(ibukota)

{'Indonesia': 'Jakarta', 'USA': 'Washington DC', 'UK': 'London', 'Thailand': 'Bangkok'}
<class 'dict'>
Jakarta
{'Indonesia': 'Jakarta', 'USA': 'Washington DC', 'UK': 'London', 'Thailand': 'Bangkok', 'France': 'Rome'}
{'Indonesia': 'Jakarta', 'USA': 'Washington DC', 'UK': 'London', 'Thailand': 'Bangkok', 'France': 'Paris'}
{'USA': 'Washington DC', 'UK': 'London', 'Thailand': 'Bangkok', 'France': 'Paris'}
```

# Latihan 6: Dictionary

Tambahkan 1 pasang key dan value  
baru sehingga ada 6 key di dictionary



- Buat dictionary “mata\_uang” dengan key berupa 5 negara dan value-nya berupa mata uang negara tersebut (bebas)

- Hapus key ke-3 dari dictionary

# 19. Operator Pembanding

Operator pembanding dalam python antara lain:

- == (sama dengan)
- != (tidak sama dengan)
- < (lebih kecil dari)
- <= (lebih kecil atau sama dengan)
- > (lebih besar dari)
- >= (lebih besar atau sama dengan)

```
print(1 == 2)
print(1 != 2)
print(1 < 2)
print(1 <= 2)
print(1 > 2)
print(1 >= 2)
```

```
False
True
True
True
False
False
```

# 20. Conditional Statement

Conditional statement pada python dapat dibuat dengan menggunakan fungsi if(), elif(), dan else

Fungsi elif dapat ditulis lebih dari satu kali untuk membuat lebih dari satu kondisi tambahan

```
angka = 16  
  
if(angka < 5) :  
    print("kecil")  
elif(angka < 10) :  
    print("sedang")  
elif(angka < 15) :  
    print("besar")  
else :  
    print("sangat besar")  
  
sangat besar
```

Tugas conditional statement

jarak = 60 km

tuliskan 4 conditional statement:

- 1: jarak > 100 km print 'jauh'
- 2: jarak > 75 km pring 'lumayan jauh'
- 3: jarak > 50 km print 'sedang'
- 4: jarak <= 50 km print 'dekat'

# 21. While Loop

While digunakan untuk membuat loop berdasarkan suatu kondisi yang ditentukan

Fungsi while juga dapat dikombinasikan dengan conditional statement

```
x = 1
while x < 5 :
    print(x)
    x = x + 1
```

1  
2  
3  
4

```
x = 5
while x != 0 :
    print(x)
    if x > 0 :
        x = x - 1
    else :
        x = x + 1
```

5  
4  
3  
2  
1

# Latihan 7: While Loop

1

**BUAT VARIABEL**

2

Buat while loop dengan kondisi:



Installation\_progress = 0



apabila installation\_progress < 100, print "installing...",



kemudian tambah nilai installation\_progress dengan 10



apabila installation\_progress >=100, print "complete"

# 22. For Loop

For digunakan untuk membuat loop berdasarkan elemen-elemen yang ada di suatu list

For juga bisa digunakan untuk list of list

```
negara = ['Indonesia', 'USA', 'UK', 'France', 'Thailand']

for x in negara :
    print(x)
```

Indonesia  
USA  
UK  
France  
Thailand

```
negara_dan_ibukota = [['Indonesia', 'Jakarta'], ['USA', 'Washington DC'], ['UK', 'London']]
for x in negara_dan_ibukota :
    print('Ibukota Negara ' + x[0] + ' adalah ' + x[1] + '.')
```

Ibukota Negara Indonesia adalah Jakarta.  
Ibukota Negara USA adalah Washington DC.  
Ibukota Negara UK adalah London.

# Latihan 8: For Loop

1

**BUAT LIST**

2

Buat for loop untuk list nilai dan kombinasikan dengan conditional statement dengan kondisi:



nilai = [6, 9, 5, 7, 3, 5, 9]



apabila `data < 6`, print "tidak lulus"



apabila `data >= 6`, print "lulus"

nilai

nilai

$\geq 6$

1. Dengan menggunakan for loop 2 variabel (GUNAKAN DICTIONARY), print 3 kalimat:

"Monumen ini ... berada di kota ..."

2. Dengan menggunakan for loop 3 variabel (GUNAKAN ZIP), print 3 kalimat:

"Makanan ... berasal dari kota ..., negara ...."

# 23. Custom Function

Custom function dapat dibuat dengan menggunakan def atau lambda

```
def sqrt(x):  
    return x**(1/2)
```

```
sqrt(4)
```

```
2.0
```

```
def pythagoras(a,b):  
    return (a**2+b**2)**(1/2)
```

```
pythagoras(3,4)
```

```
5.0
```

```
sqrt = lambda x: x**(1/2)
```

```
sqrt(4)
```

```
2.0
```

```
pythagoras = lambda a, b: (a**2+b**2)**(1/2)
```

```
pythagoras(3,4)
```

```
5.0
```

# Latihan 8: Custom function

1

BUAT custom function untuk mengubah dolar US menjadi rupiah, 1 dolar = 14383 rupiah, buat contoh penggunaan fungsinya

2

Buat custom function untuk mencari luas segitiga apabila diketahui alas dan tingginya, buat contoh penggunaan fungsinya

def

lambda

3. buat fungsi merubah yen jadi rupiah (1 yen =109.35 rupiah). buat contoh penggunaannya

4. buat fungsi volume kerucut bila diketahui jari-jari alas dan tingginya

rumus volume kerucut:  $1/3 * \text{alas} * \text{tinggi}$

# 24. DataFrame

Modul pandas memberikan struktur data berupa DataFrame

DataFrame adalah struktur data yang tersusun atas row dan column, di mana column pertama adalah index

Struktur data yang ada dalam satu kolom DataFrame juga dikenal sebagai **Series**.

DataFrame dapat dibuat dari dictionary:

 Key akan menjadi label dari kolom

 Value akan menjadi elemen yang ada di kolom

```
# Membuat list elemen untuk tiap kolom pada DataFrame
negara = ['Indonesia', 'USA', 'UK', 'Thailand', 'France']
ibukota = ['Jakarta', 'Washington DC', 'London', 'Bangkok', 'Paris']
mata_uang = ['Rupiah', 'US Dollar', 'Poundsterling', 'Baht', 'Euro']

# Membuat dictionary dengan format {label_kolom_1: elemen_kolom_1, label_kolom_2: elemen_kolom_2}
df_dict = {'negara': negara, 'ibukota': ibukota, 'mata_uang': mata_uang}

# Mengimpor modul pandas
import pandas as pd

# Membuat DataFrame dengan fungsi DataFrame()
df = pd.DataFrame(df_dict)

print(df)
print(type(df))

      negara     ibukota    mata_uang
0  Indonesia   Jakarta      Rupiah
1      USA  Washington DC  US Dollar
2       UK        London  Poundsterling
3  Thailand      Bangkok      Baht
4  France         Paris      Euro
<class 'pandas.core.frame.DataFrame'>
```

1. Buatlah dataframe 2 kolom 5 row dengan cara konversi dari dictionary, kolom pertama: nama universitas, kolom kedua: negara.

# 25. Mengkonversi .csv ke DataFrame

File .csv dapat dikonversi menjadi DataFrame dengan menggunakan fungsi `read_csv()` pada modul pandas

Tempatkan file .csv pada direktori yang sama dengan file notebook sebelum mengkonversi

```
# Mengkonversi file .csv menjadi DataFrame dengan fungsi read_csv()
import pandas as pd
df = pd.read_csv('negara.csv')

print(df)
print(type(df))
```

	negara	ibukota	mata_uang
0	indonesia	Jakarta	Rupiah
1	USA	Washington DC	US Dollar
2	UK	London	Pondsterling
3	France	Paris	Euro
4	Thailand	Bangkok	Baht
5	Italy	Rome	Euro

```
<class 'pandas.core.frame.DataFrame'>
```

# 26. Subsetting dataframe menggunakan ~~atribut~~ metode .loc

.loc dipakai dengan menggunakan row  
label dan column label

✓ DataFrame.loc[[row\_label\_1, row\_label\_2],  
[column\_label\_1, column\_label\_2]]

```
import pandas as pd
df = pd.read_csv('negara.csv')

print(df)

      negara        ibukota   mata_uang  total_kasus_covid
0  indonesia       Jakarta    Rupiah          1280000
1        USA  Washington DC  US Dollar          28200000
2        UK         London  Pondsterling          4120000
3     France        Paris        Euro          3610000
4   Thailand       Bangkok       Baht            25504
5      Italy         Rome        Euro          2810000
```

```
df_loc = df.loc[:,['negara','ibukota']]
print(df_loc)
```

```
      negara        ibukota
0  indonesia       Jakarta
1        USA  Washington DC
2        UK         London
3     France        Paris
4   Thailand       Bangkok
5      Italy         Rome
```

# 27. Subsetting dataframe menggunakan atribut .iloc

.iloc dipakai dengan menggunakan row number dan column number (dihitung mulai dari 0)



DataFrame.iloc[[row\_number\_1, row\_number\_2], [column\_number\_1, column\_number\_2]]

```
import pandas as pd  
df = pd.read_csv('negara.csv')
```

```
print(df)
```

	negara	ibukota	mata_uang	total_kasus_covid
0	indonesia	Jakarta	Rupiah	1280000
1	USA	Washington DC	US Dollar	28200000
2	UK	London	Pondsterling	4120000
3	France	Paris	Euro	3610000
4	Thailand	Bangkok	Baht	25504
5	Italy	Rome	Euro	2810000

```
df_iloc = df.iloc[:,0:2]  
print(df_iloc)
```

	negara	ibukota
0	indonesia	Jakarta
1	USA	Washington DC
2	UK	London
3	France	Paris
4	Thailand	Bangkok
5	Italy	Rome

# Latihan 9: Membuat dataframe dan melakukan subsetting

1

Buat dataframe dengan mengkonversi  
titanic.csv

→ pakai .loc

2

Subset dataframe dengan hanya meng-  
include kolom 1-5

print output dari  
metode .head()

3. Subset dataframe dengan hanya  
include kolom 6-10

→ pakai .iloc

# 28. Filtering DataFrame

01

DataFrame dapat difilter dengan menetapkan kondisi tertentu

02

Gunakan '&' (and) dan '|' (or) untuk menetapkan beberapa kondisi sekaligus

```
import pandas as pd
df = pd.read_csv('negara.csv')

print(df)

      negara      ibukota     mata_uang  total_kasus_covid
0  indonesia    Jakarta      Rupiah        1280000
1       USA  Washington DC   US Dollar      28200000
2       UK        London  Pondsterling      4120000
3    France        Paris        Euro        3610000
4   Thailand      Bangkok       Baht         25504
5     Italy         Rome        Euro        2810000

print(df[df['total_kasus_covid'] < 1000000])

      negara      ibukota     mata_uang  total_kasus_covid
4  Thailand      Bangkok       Baht         25504

print(df[(df['total_kasus_covid'] > 1000000) & (df['total_kasus_covid'] < 3000000)])

      negara      ibukota     mata_uang  total_kasus_covid
0  indonesia    Jakarta      Rupiah        1280000
5     Italy         Rome        Euro        2810000
```

# Latihan 10: Filtering dataframe

1

Filter df hanya meng-include data  
penumpang dengan umur > 30 dan jenis  
kelamin wanita

2 Filter df hanya menginclude data  
penumpang dengan umur > 30  
ATAU jenis kelamin wanita

3. Filter df hanya menginclude data  
penumpang male DAN umur < 30  
DAN survived == 1

masing2 tampilan  
.head() lah .shape

# 29. Melting

01

Fungsi `.melt()` digunakan untuk mengubah bentuk DataFrame dari melebar menjadi memanjang

02

Fungsi `.melt()` menerima argumen:



`id_vars`: kolom yang akan dijadikan acuan perubahan bentuk DataFrame



`value_vars`: kolom yang elemennya akan diikutsertakan setelah perubahan struktur

```
print(negara_populasi)
```

```
negara      ibukota     mata_uang total_kasus_covid populasi_juta \
0   USA    Washington DC      US Dollar        28200000          328
1   UK       London DC  Pondsterling        41200000           66
2 France        Paris       Euro            36100000           66
3 Thailand      Bangkok      Baht            25504             69
4 Italy         Rome        Euro            28100000           60
```

```
region
```

```
0 Amerika
1 Eropa
2 Eropa
3 Asia
4 Eropa
```

```
negara_melted = pd.melt(negara_populasi,
                         id_vars = 'negara',
                         value_vars = ['region', 'populasi_juta'])

print(negara_melted)
```

```
negara      variable  value
0   USA        region Amerika
1   UK        region Eropa
2 France      region Eropa
3 Thailand     region Asia
4 Italy        region Eropa
5   USA  populasi_juta    328
6   UK  populasi_juta     66
7 France  populasi_juta     66
8 Thailand  populasi_juta     69
9 Italy  populasi_juta      60
```

# Latihan 10: Melting dataframe

1

Gunakan fungsi melt pada df, gunakan kolom 'Name' sebagai id\_vars dan kolom 'Age', 'Sex', 'Fare' sebagai value\_vars

'Pclass', 'Embarked', 'Survived') → value\_vars

2 Urutkan dataframe hasil melt berdasarkan kolom 'Name' secara ascending



. sort\_values ('Name')

Grouping menggunakan metode .groupby()

Mengelompokkan dataframe berdasarkan 1 atau lebih kolom.

Karena ada agregasi/penggabungan data, maka perlu ditambahkan metode agregasi: .sum(), .mean(), .count(), .max(), .min()

Latihan grouping:

1. Gunakan dataset titanic
2. Lakukan grouping data menggunakan kolom 'Pclass' sebagai acuan, gunakan .mean() sebagai metode agregasi
3. Lakukan grouping data menggunakan kolom ['Sex', 'Pclass'], gunakan .max() sebagai metode agregasi

# 30. Pivot Table (1)

01

Metode `.pivot_table()` berfungsi untuk mengubah bentuk DataFrame dengan menentukan index dan kolom baru

02

Metode ini menerima argumen:



Index = kolom yang akan dijadikan index



Columns = kolom yang elemennya akan dijadikan label kolom



Values = kolom yang elemennya akan dijadikan elemen pada DataFrame baru



Aggfunc = metode yang dilakukan apabila dibutuhkan agregasi value (optional)

# 30. Pivot Table (2)

```
covid = pd.read_csv('covid.csv', sep = ';')
print(covid.head())
print(covid.tail())
```

	tanggal	negara	kasus	meninggal	sembuh
0	2021-01-01	Indonesia	751270	22329	617936
1	2021-01-02	Indonesia	758473	22555	625518
2	2021-01-03	Indonesia	765350	22734	631937
3	2021-01-04	Indonesia	772103	22911	639103
4	2021-01-05	Indonesia	779548	23109	645746
	tanggal	negara	kasus	meninggal	sembuh
52	2021-01-15	Thailand	11680	70	8906
53	2021-01-16	Thailand	12054	70	9015
54	2021-01-17	Thailand	12423	70	9206
55	2021-01-18	Thailand	12594	70	9356
56	2021-01-19	Thailand	12653	71	9621



```
covid_pivoted = covid.pivot_table(index = 'tanggal',
                                    columns = 'negara',
                                    values = 'kasus')
print(covid_pivoted)
```

	negara	Indonesia	Singapore	Thailand
tanggal				
2021-01-01	751270	58629	7379	
2021-01-02	758473	58662	7694	
2021-01-03	765350	58697	8439	
2021-01-04	772103	58721	8966	
2021-01-05	779548	58749	9331	
2021-01-06	788402	58780	9636	
2021-01-07	797723	58813	9841	
2021-01-08	808340	58836	10053	
2021-01-09	818386	58865	10298	
2021-01-10	828026	58907	10547	
2021-01-11	836718	58929	10834	
2021-01-12	846765	58946	10991	
2021-01-13	858043	58984	10991	
2021-01-14	869600	59029	11450	
2021-01-15	882418	59059	11680	
2021-01-16	896642	59083	12054	
2021-01-17	907929	59113	12423	
2021-01-18	917015	59127	12594	
2021-01-19	927380	59157	12653	

File 'covid.csv' adalah tabel bersikan data kasus covid negara Indonesia, Singapore, dan Thailand

Metode .pivot\_table() di samping mengubah bentuk DataFrame di mana data jumlah kasus masing-masing negara dipisahkan di kolom yang berbeda

# Latihan 10: Pivoting dataframe

1

**Gunakan fungsi pivot\_table pada df,  
gunakan kolom ‘Sex’ sebagai index, kolom  
‘Pclass’ sebagai columns, kolom ‘Survived’  
sebagai values, dan ‘sum’ sebagai aggfunc**

Latihan pd.crosstab()

Buat tabulasi kolom 'Survived' dan 'Pclass'

```
print tabulasi tanpa normalisasi  
print tabulasi normalisasi keseluruhan  
print tabulasi normalisasi 'index'  
print tabulasi normalisasi 'columns'
```

# 31. Plotting DataFrame (1)

DataFrame memiliki metode `.plot()` yang berfungsi secara otomatis membuat plot dari data yang ada di DataFrame

Index dari DataFrame akan menjadi nilai x-axis

Elemen di tiap kolom akan menjadi nilai y-axis

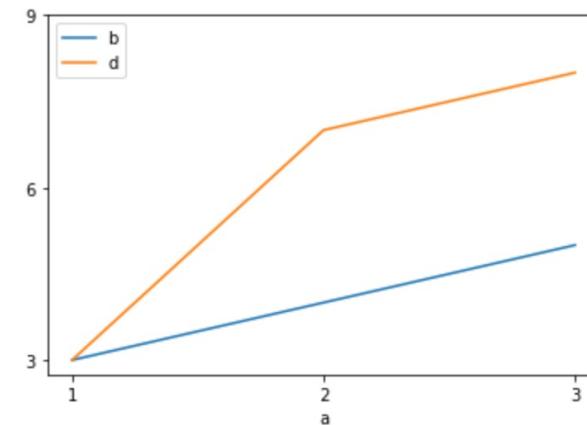
Elemen tiap kolom akan diplot secara terpisah

```
df = pd.DataFrame({'a': [1,2,3],  
                   'b': [3,4,5],  
                   'd': [3,7,8]}).set_index('a')
```

```
print(df)
```

```
   b   d  
a  
1  3  3  
2  4  7  
3  5  8
```

```
a.plot()  
plt.xticks([1, 2, 3])  
plt.yticks([3, 6, 9])  
plt.show()
```



# 31. Plotting DataFrame (2)

01

Fungsi `.pivot_table()` dapat dipakai untuk mengubah bentuk DataFrame sehingga dapat dibuat plot sesuai kebutuhan

```
covid = pd.read_csv('covid.csv', sep = ',')
covid['tanggal'] = pd.to_datetime(covid['tanggal'])
print(covid.head())
```

	tanggal	negara	kasus	meninggal	sembuh
0	2021-01-01	Indonesia	751270	22329	617936
1	2021-01-02	Indonesia	758473	22555	625518
2	2021-01-03	Indonesia	765350	22734	631937
3	2021-01-04	Indonesia	772103	22911	639103
4	2021-01-05	Indonesia	779548	23109	645746

02

(\*) dipakai agar panda mengenali elemen yang ada di kolom 'tanggal' sebagai data waktu

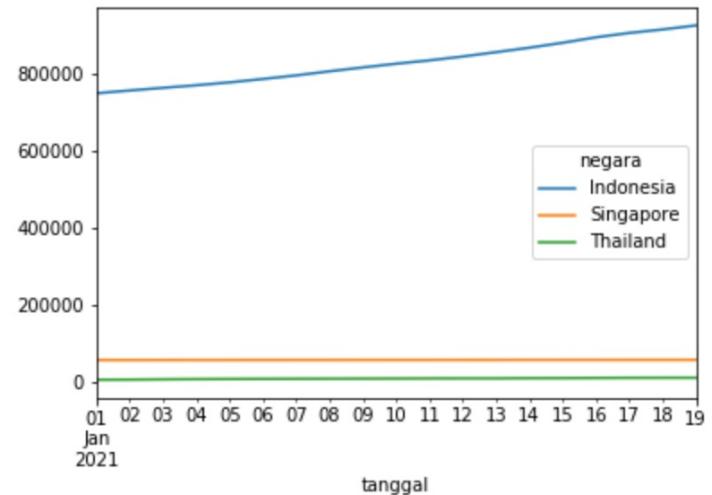


Untuk memudahkan pengolahan data sebaiknya format tanggal diubah terlebih dahulu menjadi 'YYYY'-'MM'-'DD'

```
covid_pivoted = covid.pivot_table(index = 'tanggal',
                                    columns = 'negara',
                                    values = 'kasus')

print(covid_pivoted.head())
covid_pivoted.plot()
plt.show()
```

negara	Indonesia	Singapore	Thailand
tanggal			
2021-01-01	751270	58629	7379
2021-01-02	758473	58662	7694
2021-01-03	765350	58697	8439
2021-01-04	772103	58721	8966
2021-01-05	779548	58749	9331



# 31. Plotting DataFrame (3)

01

Fungsi plot dari modul matplotlib dapat digunakan untuk melakukan plot dataframe.

```
df = pd.read_csv('titanic.csv')
```

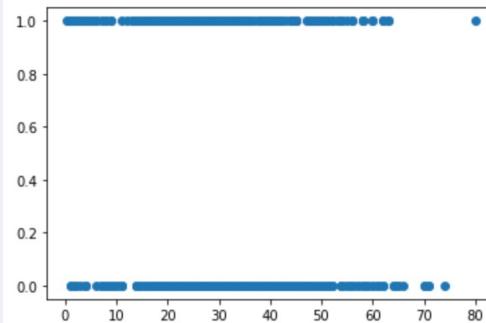
```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th... Heikkinen, Miss. Laina	female	38.0	1	0	PC 17599 STON/O2. 3101282	71.2833 7.9250	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0			NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

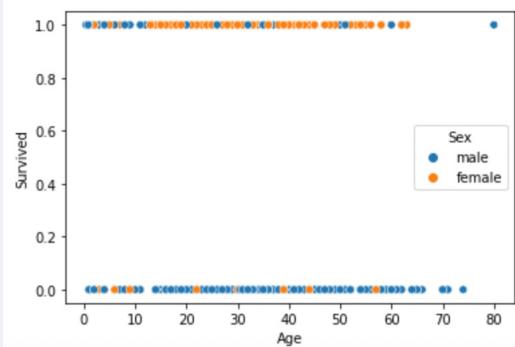
02

Fungsi plot dari modul seaborn dapat digunakan untuk melakukan plot dataframe

```
import matplotlib.pyplot as plt  
plt.scatter(data = df, x = 'Age', y= 'Survived')  
plt.show()
```



```
import seaborn as sns  
sns.scatterplot(data = df, x='Age', y='Survived', hue='Sex')  
plt.show()
```



# Latihan 11: Plotting dataframe

1

Buat scatter plot untuk df menggunakan fungsi `.scatterplot()` dari library seaborn. Gunakan kolom 'Age' sebagai nilai untuk axis-x. Gunakan kolom 'Survived' sebagai nilai untuk axis-y. Gunakan kolom 'Pclass' sebagai hue.

titanic

2.

a. lakukan grouping menggunakan kolom 'Country/Region' dan 'Date' sebagai acuan dan gunakan metode agregasi `.sum()`

b. lakukan reset index menggunakan metode `.reset_index()`

c.

Buat line plot dari data set covid yang menunjukkan perkembangan kasus aktif negara us, australia, spain, dengan axis-x = tanggal.

DATA SCIENCE BOOTCAMP SHARING VISION™

merging / menyatukan 2 data frame

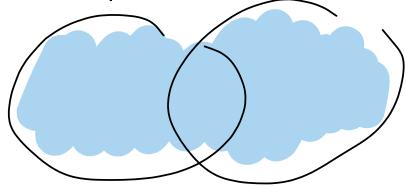
- `df1.join(df2, ...)`
- `pd.merge(df1, df2, ...)`
- `pd.concat([df1, df2], ..., axis=0/1)`

parameter "how"

"outer"

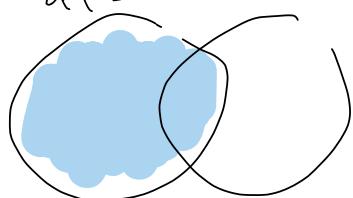
$df1 \cup df2$

$df1$        $df2$



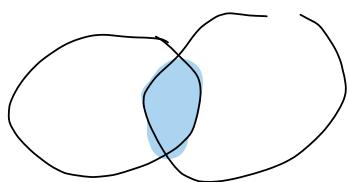
"left"

$df1$        $df2$



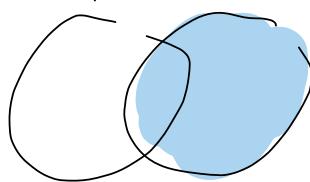
"inner"

$df1 \cap df2$



"right"

$df1$        $df2$



buat 2 dataframe:

titanic\_c: row 11-15, kolom PassengerId + 2-6

titanic\_d: row 13-17, kolom PassengerId + 7-12

lakukan merging menggunakan metode .join() dengan parameter how='outer'

lakukan merging menggunakan fungsi merge() dengan parameter how ='inner'

Terimakasih  
Danke Thank you  
Gracias Grazie Merci  
Arigatou Obrigado  
Syukron Gamsa Hamnida  
Xie-Xie Kheilil Mamnun

