



Metode Numerik

Pengantar Bahasa Pemrograman Python

Sigit Adinugroho, S.Kom, M.Sc
Universitas Brawijaya



Outline

- Sejarah
- Python vs Java
- Tipe Data
- Operasi Aritmatika
- Struktur Kendali
- Pembacaan berkas
- Error control
- Fungsi dan Modul
- Modul Numpy
- Plotting
- Jupyter

Sejarah Python

- Python dikembangkan Guido van Rossum tahun 1991
- Awalnya sebagai proyek “hobby” supaya tidak menganggur saat libur Natal
- Nama Python diinspirasi dari grup dan acara komedi “Monty Python”, **bukan nama ular**



Java vs Python

Java	Python
<i>Static type variable</i>	<i>Dynamic type variable</i>
Blok menggunakan { }	Blok menggunakan <i>indentation</i> (tabulasi) – beda tab beda blok
<i>Compiler + Interpreter</i>	<i>Interpreter</i> (meski bisa juga di- <i>compile</i>)
Hanya mendukung object oriented	Mendukung Object Oriented + Prosedural + Fungsional
<i>Faster to run</i>	<i>Faster to code</i>
	Sintaksis (<i>syntax</i>) lebih ringkas

Hello world.....

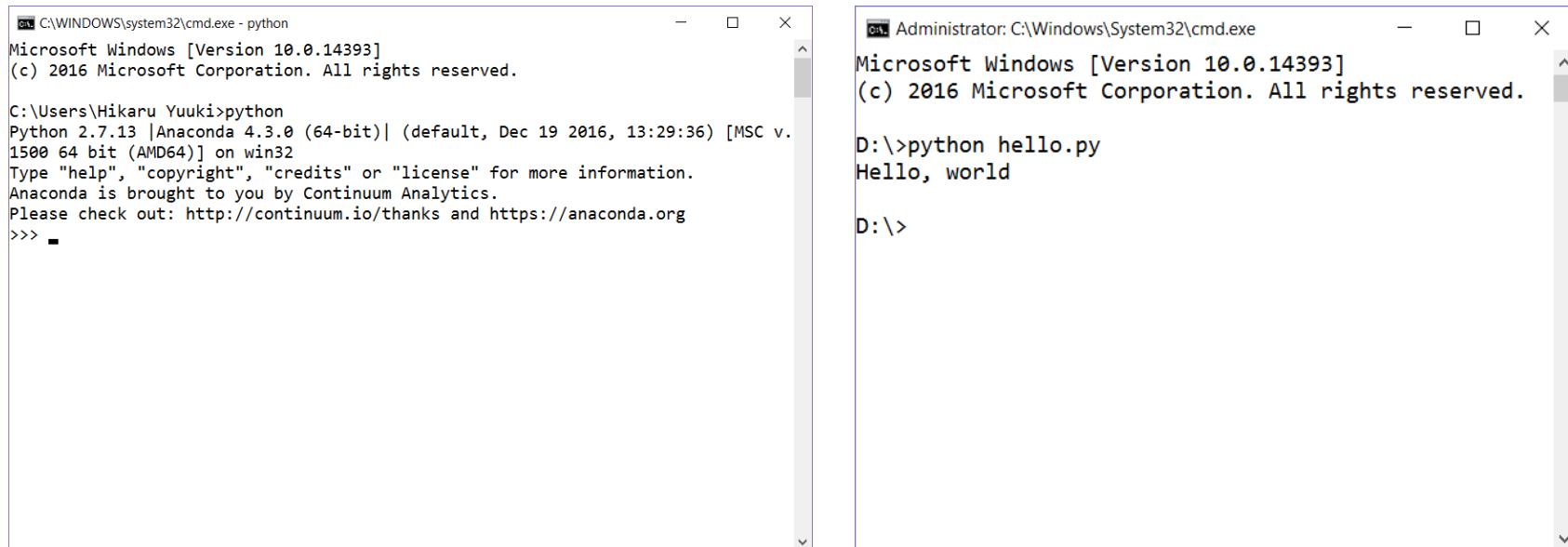
Java	Python
<pre>public class HelloWorld { public static void main (String[] args) { System.out.print("Hello, world!"); } }</pre>	<pre>print "Hello, world!"</pre>

OOP Python vs Java

JAVA	PYTHON
<pre>class Animal{ private String name; public Animal(String name){ this.name = name; } public void saySomething(){ System.out.println("I am" + name); } } class Cat extends Animal{ public Cat(String name) { super(name); } public void saySomething(){ System.out.println("Meow"); } } public class Main{ public static void main(String[] args){ Cat cat = new Cat("Siamese"); cat.saySomething(); } }</pre>	<pre>class Animal(): def __init__(self, name): self.name = name def saySomething(self): print "I am " + self.name class Cat(Animal): def saySomething(self): print "I am " + self.name \ + ", Meow" cat = Cat("Siamese") cat.saySomething()</pre>

Cara menjalankan kode program

- Kode python dapat dijalankan secara langsung pada terminal ataupun disimpan dalam file *.py dan dieksekusi melalui terminal



The image shows two separate instances of the Windows Command Prompt (cmd.exe) running on a Windows 10 system. Both windows have a blue title bar and a white body.

Left Window:

```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Hikaru.Yuuki>python
Python 2.7.13 |Anaconda 4.3.0 (64-bit)| (default, Dec 19 2016, 13:29:36) [MSC v.
1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>> -
```

Right Window:

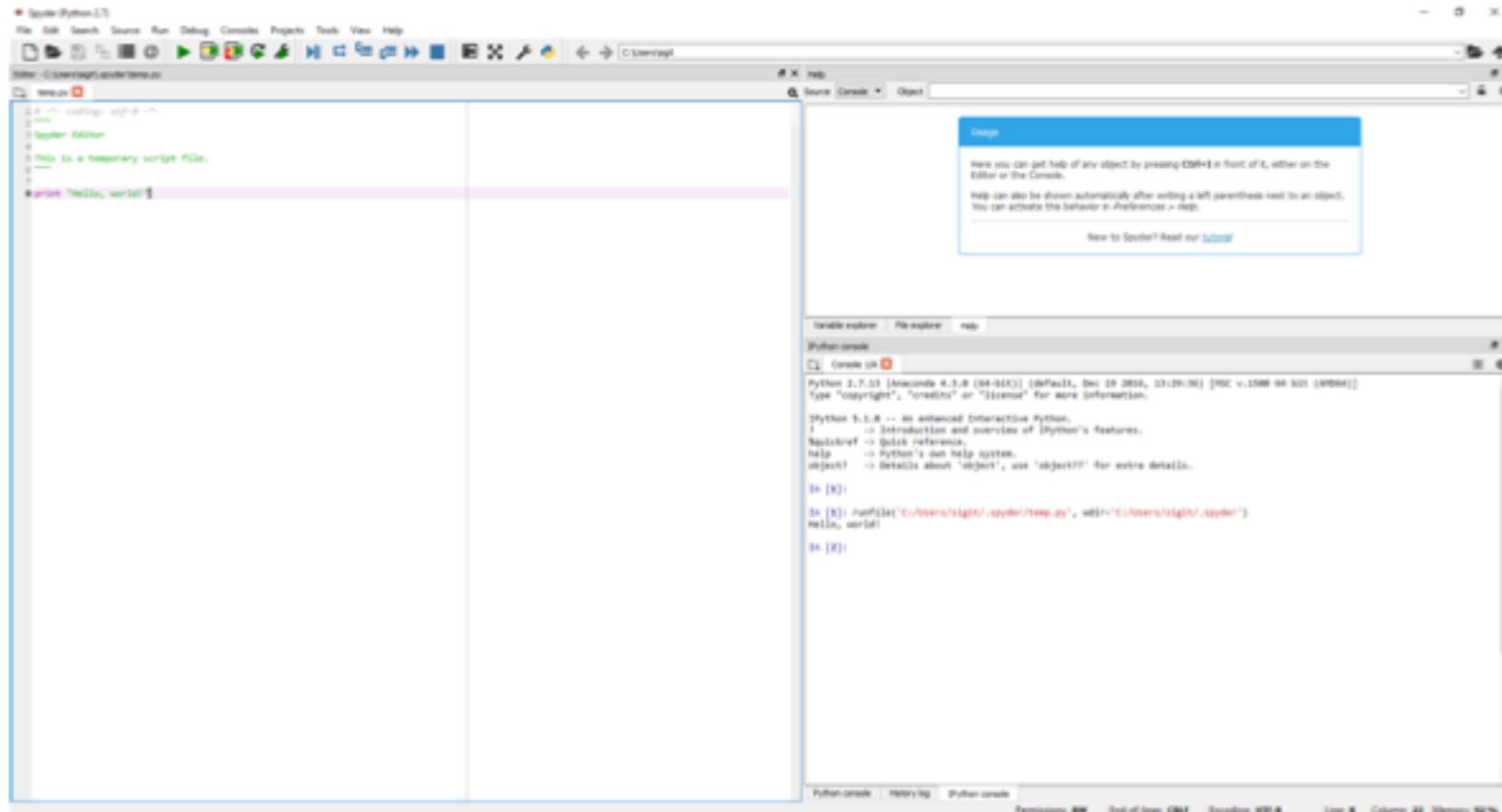
```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

D:\>python hello.py
Hello, world

D:\>
```

Cara menjalankan kode program

- Atau menggunakan IDE seperti Spyder, PyCharm, dll



Variabel & Tipe Data

Variabel

- Variabel merupakan “tempat” menyimpan data
- Variabel pada Python bersifat dinamis :
 - Tidak perlu dideklarasikan tipe datanya
 - Tipe data pada sebuah variabel dapat berubah-ubah

Tipe data angka

- Integer (int) : 1,2,34,5
- Floating point (float) : 3.145, 2.2478
- Bilangan kompleks : 1.1+3.5j, 9.322e-36j
- Boolean : True, False

NULL

Python memiliki **None** yang identik dengan **null** di Java

Tipe data angka

```
D:\>python
Python 2.7.13 |Anaconda 4.3.0 (64-bit)| (default, Dec 19 2016, 13:29:36)
[MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>> a = 5
>>> a = 2.0
>>> a = 1+2j
>>> b = 7.5
>>> a + b
(8.5+2j)
>>> b1 = True
>>> b2 = False
>>> b1 and (b2 or b1)
True
>>> b1 + b2
1
>>> b1 + True
2
>>>
```

Boolean dianggap sebagai integer

Tipe data sekuensial

- String : 'Filkom UB', "Informatika"
- List : [1,2,3,4], [25,'FILKOM', 9.322e-36j]
- Tuple : (1, 2, 3)
- Dictionary (dict) : {'nama':'andi', 'fakultas':'fikom'}
- Set : {1,2,2,3,3,3} → {1,2,3}



String

- String dianggap sebagai kumpulan karakter
- String bersifat *immutable* (nilai elemennya tidak bisa diganti setelah deklarasi)
 - Variable yang *mutable*, isinya bisa diubah
 - Cek *mutability* bisa dengan `id(variable)`, jika id sebelum dan sesudah sama maka bersifat *mutable*
- Penggabungan string menggunakan **+**

String

```
[>>> s1 = 'Filkom UB'
[>>> print(s1)
Filkom UB
[>>> len(s1)
9
[>>> s1[0]
'F'
[>>> s1[7]           Mengakses karakter ke n-1 pada string
'U'
[>>> s1[8]
'B'
[>>> s1[8]='M'      String bersifat immutable
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
[>>> s1 + 'Juara'   Penggabungan 2 string menggunakan +
'Filkom UBJuara'
[>>> a = 100
[>>> s1 + a          String tidak bisa digabungkan dengan int
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str, not int
[>>> s1 + str(a)    Int harus diubah menjadi string terlebih dahulu
'Filkom UB100'
>>> ]
```

List

Python vs Java

List (Python) mirip dengan **array, ArrayList, Vector** di Java

- Python tidak memiliki tipe *array* statis seperti Java.
- Elemen-elemen pada *list* **tidak harus** memiliki tipe data yang sama
- *List* dapat bersifat multidimensi
- *List* bersifat ***mutable***
- Indeks *list* dimulai dari 0

List

```
>>> l1 = [1,2,3,'Filkom',8.5,3.14567] Deklarasi List
>>> l1[1] Mengakses elemen ke n+1
2
>>> l1[1:4] Mengakses elemen pada rentang tertentu
[2, 3, 'Filkom']
>>> l1[3] = 'Brawijaya' Mengganti nilai pada elemen ke n+1
>>> l1[2:]
[3, 'Brawijaya', 8.5, 3.14567]
>>> l2 = [[3, 9], [8, 5], [11, 1]]
>>> l2[1]
[8, 5]
>>> l2[1][0] List dua dimensi
8
>>> len(l2)
3
>>> l1.append('UB') Menambahkan elemen pada List
>>> l1
[1, 2, 3, 'Brawijaya', 8.5, 3.14567, 'UB']
>>>
```

List

- List dapat digunakan untuk membuat array 2 dimensi

```
>>> a = [[1,2,3],[4,5,6],[6,7,8]]  
>>> print(a)  
[[1, 2, 3], [4, 5, 6], [6, 7, 8]]  
>>> print(a[1])  
[4, 5, 6]  
>>> print(a[1][2])  
6  
>>> 
```

Tuple

Python vs Java

Java tidak memiliki padanan untuk tuple (Python)

- Tuple merupakan list yang bersifat *immutable*

```
>>> t1 = (1, 'Python', 2.7, 'Anaconda')          Deklarasi tuple
>>> t1[1]                                         Mengakses elemen pada tuple
'Python'
>>> t1[1:3]
('Python', 2.7)
>>> t1[2]=3.0                                     Tuple bersifat immutable
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
>>> t1.append(4)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'tuple' object has no attribute 'append'
>>> -
```

Tuple

Penggunaan Tuple

Tuple biasanya digunakan untuk mengembalikan nilai balik suatu fungsi yang berisi lebih dari satu nilai

- Fungsi di Java hanya boleh mengembalikan satu nilai, misalnya
 - ```
int add(int x, int y){
 return x+y
}
```

maka hanya boleh mengembalikan satu nilai bertipe int.
- Python lebih fleksibel, fungsi bisa mengembalikan banyak nilai

```
def operation(x, y):
 return (x, y, x+y)
```

# Dictionary

## Python vs Java

Dictionary (Python) identik dengan **HashMap** di Java

- **Dictionary** merupakan kumpulan pasangan key-value (*key-value pair*)
- *Key* digunakan sebagai indeks untuk mengakses nilai pada *value*
  - Analoginya di array, sama dengan indeks berupa angka
- *Key* bersifat immutable sedangkan *value* bersifat mutable

# Dictionary

```
Type "help", "copyright", "credits" or "license" for more information.
[>>> nilai = {'Andi':8,'Dina':9,'Doni':7,'Dinda':10} Deklarasi dictionary
[>>> nilai
{'Doni': 7, 'Dinda': 10, 'Andi': 8, 'Dina': 9} Mengakses nilai dengan key tertentu
[>>> nilai['Dina']
9
[>>> nilai.keys()
['Doni', 'Dinda', 'Andi', 'Dina'] Memperoleh daftar key
[>>> nilai.values()
[7, 10, 8, 9] Memperoleh daftar value
[>>> nilai['Doni']=9
[>>> nilai
{'Doni': 9, 'Dinda': 10, 'Andi': 8, 'Dina': 9} Memperoleh value dengan key tertentu
[>>> nilai.update({'Anto':7})
[>>> nilai
{'Doni': 9, 'Dinda': 10, 'Anto': 7, 'Andi': 8, 'Dina': 9} Menambahkan pasangan key-value baru
[>>> del nilai['Dinda']
[>>> nilai
{'Doni': 9, 'Anto': 7, 'Andi': 8, 'Dina': 9} Menghapus pasangan key-value tertentu
[>>> nilai['Dimas']
Traceback (most recent call last):
 File "<stdin>", line 1, in <module>
KeyError: 'Dimas' Mengakses value dengan key yang tidak valid
```

# Copy-by-reference

---

```
[>>> a = [1,3,5]
[>>> b = a
[>>> print('a',a)
a [1, 3, 5]
[>>> print('b',b)
b [1, 3, 5]
[>>> b[1]=7
[>>> print('b',b)
b [1, 7, 5]
[>>> print('a',a)
a [1, 7, 5]
>>>]
```

KENAPA????

# Operasi Aritmatika

# Operator Aritmatika

---

|    |                           |
|----|---------------------------|
| +  | : Penjumlahan             |
| -  | : Pengurangan             |
| *  | : Perkalian               |
| /  | : Pembagian               |
| ** | : Pangkat                 |
| %  | : Modulo (sisa pembagian) |

# Operator Perbandingan

---

- > : Lebih besar dari
- < : Lebih kecil dari
- $\geq$  : Lebih besar dari atau sama dengan
- $\leq$  : Lebih kecil dari atau sama dengan
- $=$  : Sama dengan
- $\neq$  : Tidak sama dengan

# Struktur Kendali

# Kondisional *if*

---

- Struktur kondisional *if* pada Python:

```
if kondisi1:
 perintah1
elif kondisi2:
 perintah2
elif kondisi3:
 perintah3
else:
 perintah
```

# Kondisional *if*

---

```
7
8 angka = float(input('Masukkan sebuah angka : '))
9
10 if angka>0:
11 print 'Bilangan positif' Gunakan Tab untuk memulai blok
12 elif angka==0:
13 print 'Nol'
14 else:
15 print 'Bilangan negatif'|
```

# Perulangan *while*

---

- Struktur perulangan *while* pada Python:

```
while kondisi_bersifat_benar:
 perintah
```

# Perulangan *while*

---

```
8 x=1
9 n=input('Masukkan nilai n : ')
10 jumlah=0
11
12 while x<=n:
13 jumlah = jumlah + x
14 x = x + 1
15
16 print "Jumlah "+str(n)+" bilangan pertama adalah "+str(jumlah)
```

# Perulangan *for*

---

- Python mendukung beberapa model perulangan *for*

```
for nilai in list:
 perintah
```

```
for index in range(len(list)):
 perintah
```

```
8 angka = [1,3,5,7,9,11]
9
10 for x in angka:
11 print x
```

```
8 angka = [1,3,5,7,9,11]
9 for index in range(len(angka)):
10 print angka[index]
```

# *List comprehension*

---

- *List comprehension* memungkinkan transformasi *list* menjadi *list* lainnya tanpa menggunakan perulangan
- Misal kita ingin membuat *list* yang tiap elemennya merupakan kuadrat dari elemen *list* yang lain
- *List comprehension* juga memungkinkan pemilihan elemen (*filtering*) pada *list*

# List Comprehension vs For Loop

for loop



```
8 angka = [1,2,3,4,5,6,7,8,9,10]
9 angkakuadrat = []
10
11 for x in angka:
12 angkakuadrat.append(x*x)
13
14 print angkakuadrat |
```

List  
comprehension



```
8 angka = [1,2,3,4,5,6,7,8,9,10]
9 angkakuadrat = [x*x for x in angka]
10
11 print angkakuadrat
```

# List Comprehension vs For Loop

for loop



```
8 angka = [1,2,3,4,5,6,7,8,9,10]
9 angkakuadrat = []
10
11 for x in angka:
12 if x % 2 == 0:
13 angkakuadrat.append(x*x)
14
15 print angkakuadrat
```

List  
comprehension



```
9 angka = [1,2,3,4,5,6,7,8,9,10]
10 angkakuadrat = [x*x for x in angka if x % 2 == 0]
11
12 print angkakuadrat
```

# Pembacaan Berkas

# Alur pembacaan berkas

---



# Buka berkas

---

- Cara pembukaan berkas

```
file_object = open(namafile,action)
```

| Jenis action | Arti                                                                |
|--------------|---------------------------------------------------------------------|
| 'r'          | Baca berkas yang sudah ada                                          |
| 'w'          | Simpan ke berkas. Jika file tidak ada maka akan dibuat              |
| 'a'          | Tambahkan data ke akhir berkas                                      |
| 'r+'         | Baca dan tulis ke berkas yang sudah ada                             |
| 'w+'         | Seperti 'a+' namun akan buat file terlebih dahulu jika file blm ada |
| 'a+'         | Sama seperti 'w+' dan data akan ditambahkan ke akhir file           |

# Operasi Berkas

---

- Baca n karakter dari berkas
  - `file_object.read(n)`
- Baca semua karakter dari berkas
  - `file_object.read()`
- Baca baris aktif dan ambil n karakter
  - `file_object.readline(n)`
- Baca satu baris aktif dari berkas
  - `file_object.readline()`
- Baca semua isi berkas
  - `file_object.readlines()`
- Menulis sebuah string ke dalam berkas
  - `file_object.write(string)`
- Menulis list of string ke dalam berkas
  - `file_object.writelines(list_of_string)`

# Membaca berkas

Baca berkas



```
8 berkas = open('berkas1.txt','r')
9 isiBerkas = berkas.read()
10 berkas.close()
11 |
12 print isiBerkas
```

Tulis berkas



```
7 #Membuka 2 buah berkas
8 berkas1 = open('berkas1.txt','r')
9 berkas2 = open('berkas2.txt','r')
10 #Membaca 2 berkas dan memasukkannya ke dalam variabel
11 isiBerkas1 = berkas1.read()
12 isiBerkas2 = berkas2.read()
13 #Menutup berkas yang telah selesai dibaca
14 berkas1.close()
15 berkas2.close()
16 #Menggabungkan isi kedua berkas
17 isiGabungan = isiBerkas1 + "\n" + isiBerkas2
18 #Menulis string ke dalam berkas|
19 berkasTulis = open('berkas3.txt','w')
20 berkasTulis.write(isiGabungan)
21 berkasTulis.close()
~~
```

# Membaca Berkas Per Baris

---

```
1 with open('d:/berkas1.txt', 'r') as berkas:
2 for line in berkas:
3 print line.strip()
4
5 berkas.close()
```

## Strip

Fungsi `strip()` digunakan untuk menghilangkan (*trim*) *white space* di awal dan akhir dari suatu text

# Error Control

# Error control

---

- Statement :

```
try:
 operasi
catch error_type:
 operasi alternatif
```

# Error control

| Tanpa error control                                                  | Dengan Error control                                                                                           |
|----------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| code                                                                 | Code                                                                                                           |
| <pre>8 9 hasil = 17.5 /0</pre>                                       | <pre>11 try : 12     hasil = 17.5 /0 13 except ZeroDivisionError: 14     print("Penyebut tidak boleh 0")</pre> |
| Output                                                               | Output                                                                                                         |
| <pre>hasil = 17.5 /0 ZeroDivisionError: float division by zero</pre> | <pre>....: Penyebut tidak boleh 0</pre>                                                                        |

# Fungsi dan Modul

# Fungsi

---

- Struktur fungsi

```
def nama_fungsi (param1,param2,...) :
 statement
 return return_values
```

# Contoh fungsi

```
9 def sum_list(a_list):
10 hasil = 0
11 for x in a_list:
12 hasil +=x
13 return x
14
15 def hitung():
16 input = [2,7,5,9,13,25]
17 hasil = sum_list(input)
18 print(hasil)
19
20 if __name__ == "__main__":
21 hitung()
```

# Modul

---

- Modul merupakan kumpulan fungsi dan variabel.
- Jenis modul :
  - Built-in module : Modul bawaan Python
  - User defined module : Modul yang dibuat sendiri oleh pengguna

# User defined module (UDM)

---

- Pengguna dapat membuat modul sesuai kebutuhan
- Modul disimpan dalam berkas berekstensi .py
- UDM dipanggil menggunakan perintah `import`

# Contoh UDM

```
lingkaran.py
1 def luas(r):
2 luas = 22 / 7 * r * r
3 return luas
4
5 def keliling(r):
6 keliling = 22/7 * r * 2
7 return keliling
```

```
hitung_lingkaran.py
1 import lingkaran
2 print(lingkaran.luas(20))
3 print(lingkaran.keliling(20))
```

# Built-in function

---

- Beberapa fungsi dasar telah disediakan oleh Python
- Contoh built-in modules :
  - math : Memuat fungsi matematika dasar (sin, cos, pi, dll)
  - numpy : Memuat fungsi yang berhubungan dengan numerik, array, dll

# Modul math

---

- Berisi fungsi-fungsi dasar matematika
- Cara penggunaan modul

```
import math
```

```
import math as m
```

```
from math import sqrt
```

```
from math import *
```

# Modul math

---

Daftar fungsi pada modul math

|         |            |         |             |            |          |           |         |
|---------|------------|---------|-------------|------------|----------|-----------|---------|
| 'acos'  | 'atanh'    | 'e'     | 'factorial' | 'gcd'      | 'isnan'  | 'log2'    | 'sin'   |
| 'acosh' | 'ceil'     | 'erf'   | 'floor'     | 'hypot'    | 'ldexp'  | 'modf'    | 'sinh'  |
| 'asin'  | 'copysign' | 'erfc'  | 'fmod'      | 'inf'      | 'lgamma' | 'nan'     | 'sqrt'  |
| 'asinh' | 'cos'      | 'exp'   | 'frexp'     | 'isclose'  | 'log'    | 'pi'      | 'tan'   |
| 'atan'  | 'cosh'     | 'expm1' | 'fsum'      | 'isfinite' | 'log10'  | 'pow'     | 'tanh'  |
| 'atan2' | 'degrees'  | 'fabs'  | 'gamma'     | 'isinf'    | 'log1p'  | 'radians' | 'tau'   |
|         |            |         |             |            |          |           | 'trunc' |

<https://docs.python.org/3/library/math.html>

# Penggunaan modul math

---

```
[>>> import math
[>>> print(math.ceil(0.2))
1
```

```
[>>> import math as m
[>>> print(m.factorial(5))
120
```

```
[>>> from math import sqrt
[>>> print(sqrt(289))
17.0
```

```
[>>> from math import *
[>>> log10(1000)
3.0
```

# Modul numpy

---

- NumPy = Numerical Python
- Library yang berisi objek array multidimensi dan sekumpulan fungsi untuk memproses array tersebut
- Memudahkan operasi matematika

# Numpy array

---

- Merupakan array berdimensi n
- Biasa disebut ndarray
- Pembuatan array dilakukan dengan perintah

```
numpy.array(object, dtype = None,
copy = True, order = None, subok =
False, ndmin = 0)
```

# Array 1 dimensi dan 2 dimensi

---

```
import numpy as np
#membuat array 1 dimensi
a = np.array([1,2,3])
print(a)
print()

#membuat array 2 dimensi
b = np.array([[1,2,3], [4,5,6]])
print(b)
```

[1 2 3]

[[1 2 3]  
 [4 5 6]]

# Atribut array

---

```
import numpy as np
b = np.array([[1,2,3], [4,5,6]])
print(b.shape)
print()
print(b)
```

(2, 3)

[[1 2 3]  
 [4 5 6]]

# Atribut array

```
import numpy as np
b = np.array([[1,2,3], [4,5,6]])
print(b.shape)
print()
print(b)
b.shape = (3,2)
print()
print(b)
print()
print(b.shape)
```

## Output

```
(2, 3)
[[1 2 3]
 [4 5 6]]

[[1 2]
 [3 4]
 [5 6]]

(3, 2)
```

# Resize array

```
import numpy as np
b = np.array([[1,2,3], [4,5,6]])
print(b)
print()

b.shape = (3,2)
print(b)

c = b.reshape(2,3)
print()
print(c)
```

Output:

```
[[1 2 3]
 [4 5 6]]
```

```
[[1 2]
 [3 4]
 [5 6]]
```

```
[[1 2 3]
 [4 5 6]]
```

# Resize Array

---

```
import numpy as np
a = np.arange(24)
print(a)
print()
print(a.ndim)

b = a.reshape(3,8)
print()
print(b)
print(b.ndim)
```

# Resize array

---

Output:

```
[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 1
5 16 17 18 19 20 21 22 23]
```

```
1
```

```
[[0 1 2 3 4 5 6 7]
 [8 9 10 11 12 13 14 15]
 [16 17 18 19 20 21 22 23]]
```

```
2
```

# Resize Array

---

```
import numpy as np
a = np.arange(10,22,2)
print(a)
a = a.reshape(2,3)
print()
print(a)
```

```
[10 12 14 16 18 20]
```

```
[[10 12 14]
 [16 18 20]]
```

# Akses elemen array

```
import numpy as np
a = np.arange(10,22,2)
a = a.reshape(2,3)
print(a), print()
print(a[1][2]), print()
print(a[1,2]), print()
print(a[0:1,0:2]), print()
print(a[0:2,1:3])
```

Output:

```
[[10 12 14]
 [16 18 20]]
```

```
20
```

```
20
```

```
[[10 12]]
```

```
[[12 14]
 [18 20]]
```

# Operasi array

---

```
import numpy as np
a = np.array([[1,4],[5,2]])
b = np.array([[3,1],[4,3]])
print(a)
print(b)
print()
print(a+b), print()
print(a-b), print()
```

Output:

```
[[1 4]
 [5 2]]
 [[3 1]
 [4 3]]
```

```
[[4 5]
 [9 5]]
```

```
[[-2 3]
 [1 -1]]
```

# Operasi array

```
import numpy as np
a = np.array([[1,4],[5,2]])
b = np.array([[3,1],[4,3]])
print(a)
print(b)
print()
print(a*b), print()
print(np.dot(a,b))
```

Output:

```
[[1 4]
 [5 2]]
[[3 1]
 [4 3]]

[[3 4]
 [20 6]]

[[19 13]
 [23 11]]
```

# Operasi array

---

```
import numpy as np
a = np.array([[1,4],[5,2]])
print(a), print()
print(a**2)
```

```
[[1 4]
 [5 2]]
```

```
[[1 16]
 [25 4]]
```

# Modul aljabar linear

---

```
import numpy.linalg as lg
a = np.array([[1,4],[3,1]])
b = np.array([6,7])
print(a), print()
print(b), print()
print(lg.inv(a)), print()
print(lg.solve(a,b))|
```

# Modul aljabar linear

---

Output:

```
[[1 4]
 [3 1]]

[6 7]

[[-0.09090909 0.36363636]
 [0.27272727 -0.09090909]]

[2. 1.]
```

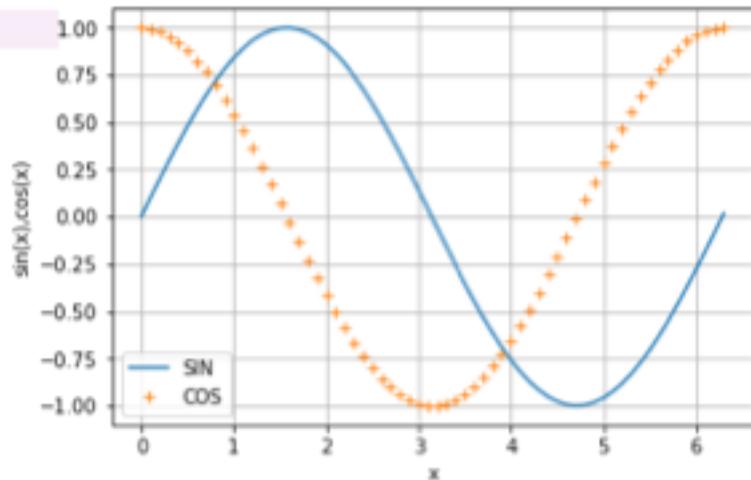
# Plotting

# Plotting

---

- Plotting dilakukan dengan library matplotlib

```
9 import matplotlib.pyplot as plt
10 import numpy as np
11
12 x = np.arange(0.0,2*np.pi+0.1,0.1)
13 plt.plot(x,np.sin(x),'-',x,np.cos(x),'+'.)
14 plt.xlabel('x')
15 plt.ylabel('sin(x),cos(x)')
16 plt.legend(('SIN','COS'),loc=0)
17 plt.grid(True)
18 plt.show()
```



# Marker Style

---

|         |                 |
|---------|-----------------|
| ' - '   | Solid line      |
| ' -- '  | Dashed line     |
| ' - . ' | Dash-dot line   |
| ' : '   | Dotted line     |
| ' o '   | Circle marker   |
| ' ^ '   | Triangle marker |
| ' s '   | Square marker   |
| ' h '   | Hexagon marker  |
| ' x '   | x marker        |

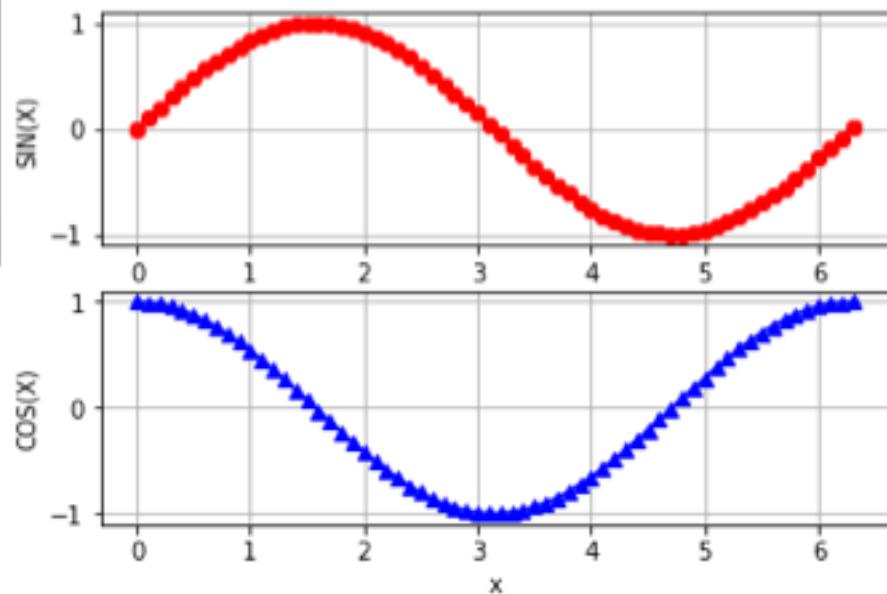
# Loc style

---

|   |                 |
|---|-----------------|
| 0 | "Best" location |
| 1 | Upper right     |
| 2 | Upper left      |
| 3 | Lower left      |
| 4 | Lower right     |

# Subplot

```
9 import matplotlib.pyplot as plt
10 import numpy as np
11
12 x = np.arange(0.0,2*np.pi+0.1,0.1)
13 plt.subplot(2,1,1)
14 plt.plot(x,np.sin(x), 'ro-')
15 plt.xlabel('x')
16 plt.ylabel('SIN(X)')
17 plt.grid(True)
18 plt.subplot(2,1,2)
19 plt.plot(x,np.cos(x), 'b^-')
20 plt.xlabel('x')
21 plt.ylabel('COS(X)')
22 plt.grid(True)
23
24 plt.show()
```



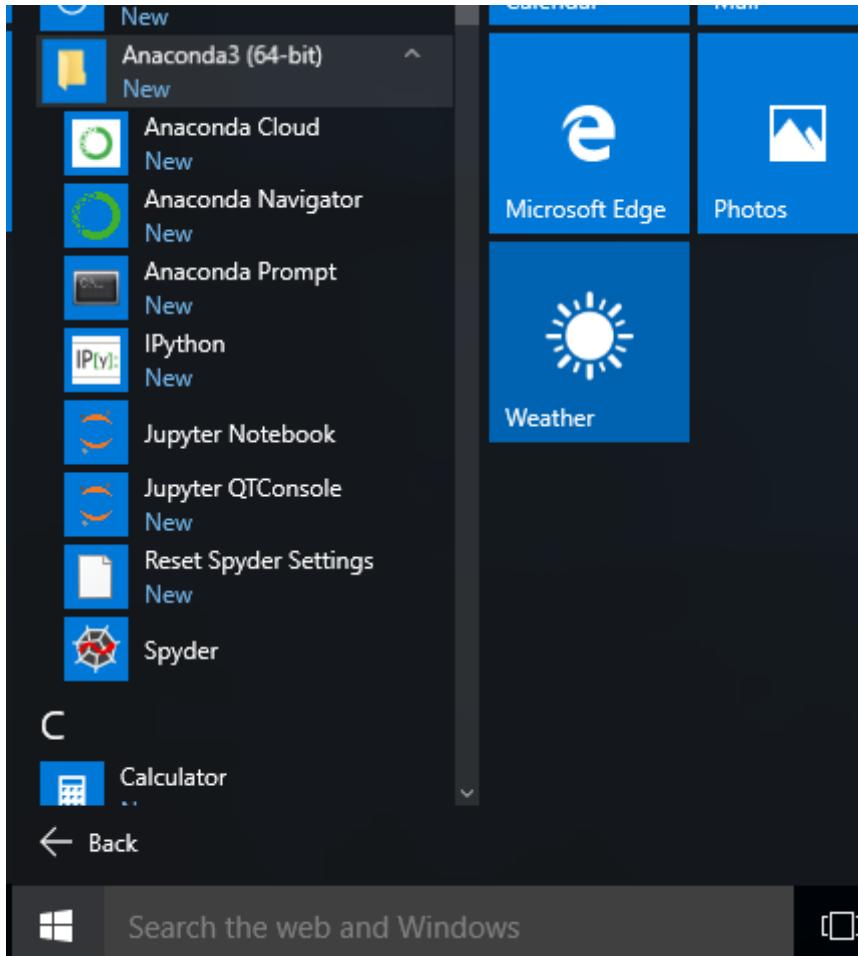
# Jupyter Notebook

# Definisi

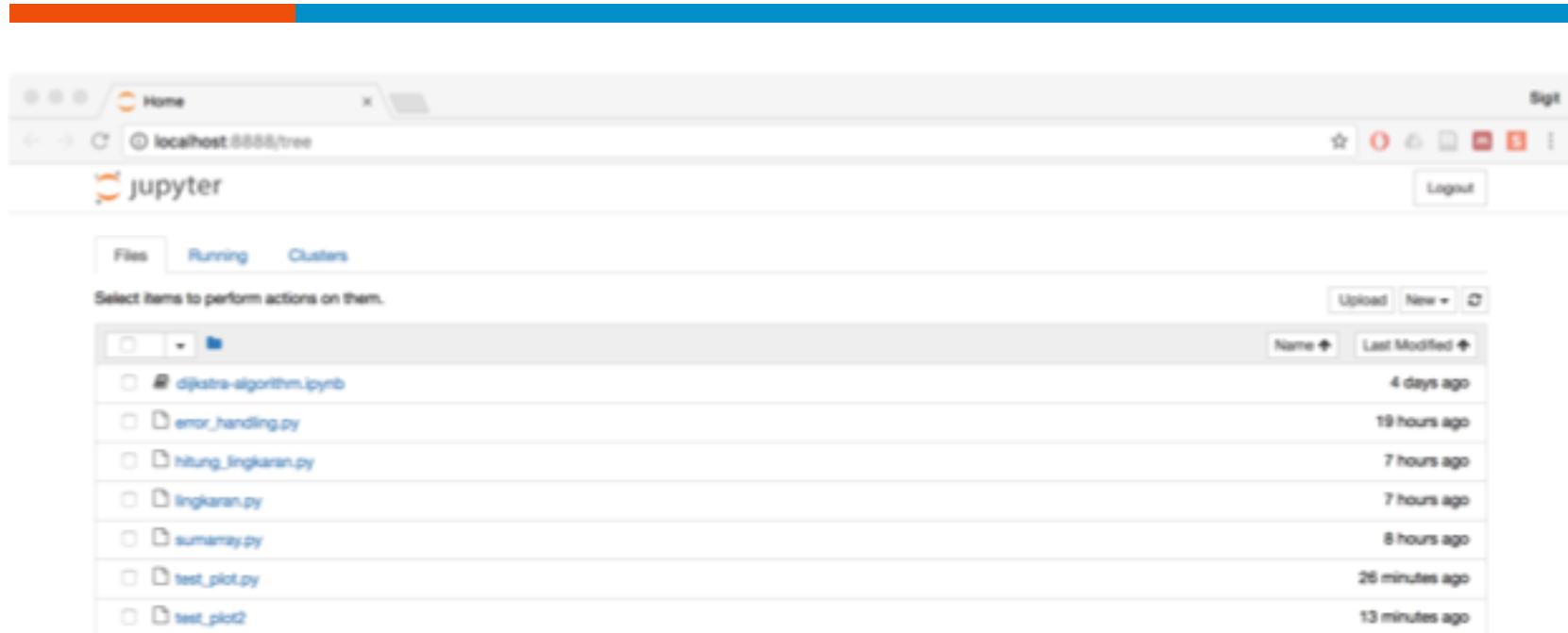
---

- Notebook merupakan dokumen yang berisi *source code* dan *rich text elements* (dapat berupa teks, persamaan, gambar, link, dll)
- Lebih mudah dibaca daripada komentar pada *source code*
- Jupyter merupakan aplikasi untuk mengedit dan menjalankan *notebook*

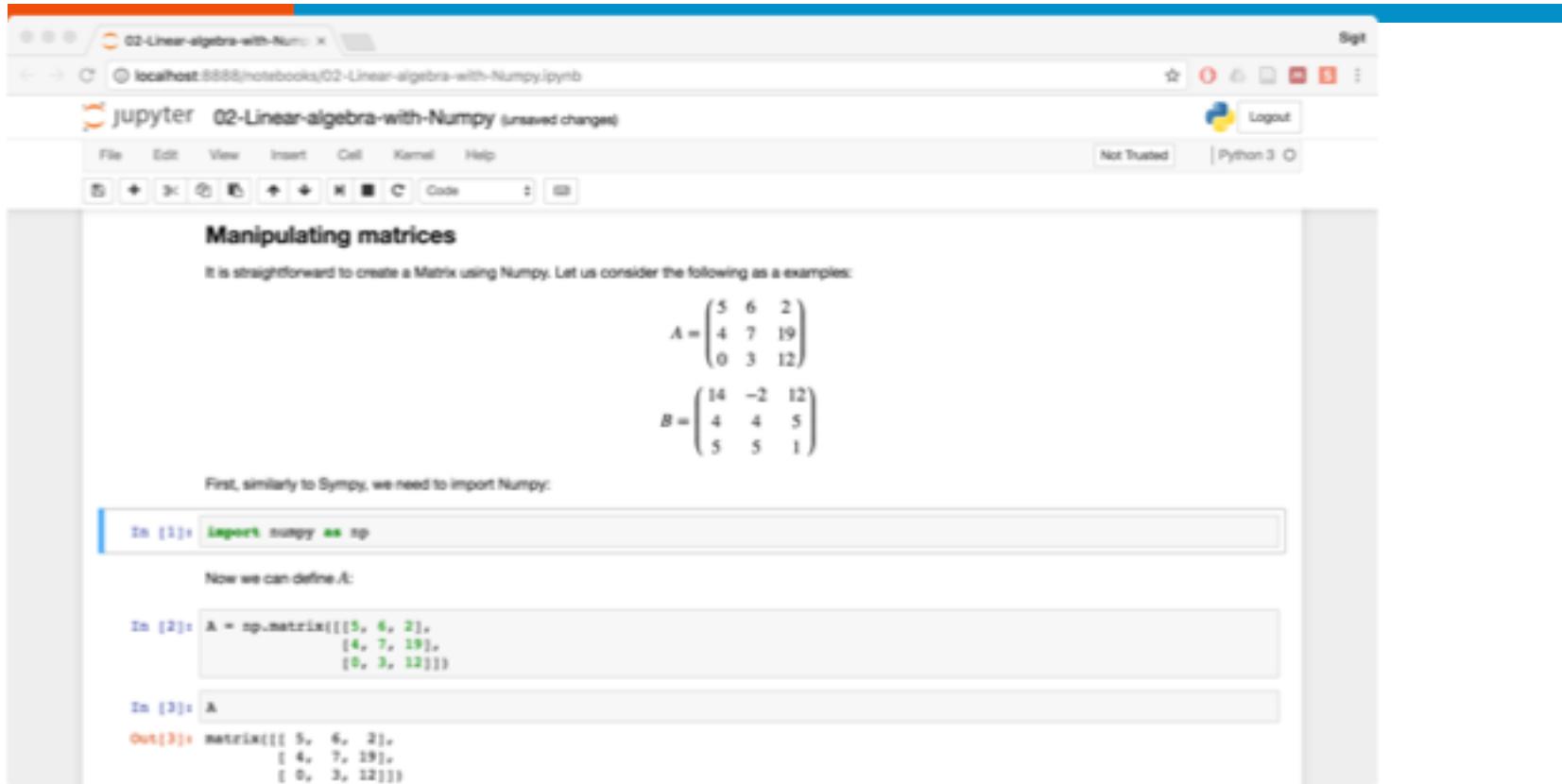
# Menjalankan Jupyter



# Antarmuka Jupyter



# Contoh notebook



The screenshot shows a Jupyter Notebook interface with the title "02-Linear-algebra-with-Numpy.ipynb". The notebook has a header bar with tabs, a toolbar below it, and a main content area. The content area contains a section titled "Manipulating matrices" with text and two equations for matrices A and B. Below this, there's a code cell with imports and matrix definitions, followed by another cell showing the result of printing matrix A.

```
In [1]: import numpy as np

Now we can define A:

In [2]: A = np.matrix([[5, 6, 2],
 ...: [4, 7, 19],
 ...: [0, 3, 12]])

In [3]: A
Out[3]: matrix([[5, 6, 2],
 ...: [4, 7, 19],
 ...: [0, 3, 12]])
```

<https://github.com/drvinceknight/Python-Mathematics-Handbook>

# Komponen notebook

---

Setiap notebook terdiri dari sekumpulan *cell*. Setiap *cell* dapat berbentuk

- Code :
  - Merupakan source code yang ditulis dalam Python
- Markdown :
  - Markdown merupakan teks dalam format *rich text* yang berfungsi menjelaskan source code
  - Markdown dapat mengandung :
    - Header
    - Italic dan bold
    - Link
    - Gambar
    - Tabel
    - Rumus
    - dll

# Membuat notebook

- Pilih menu New -> Python 3



# Menambahkan Markdown

---

- Klik pada *cell* dan pilih opsi *Markdown*



# Menulis Markdown

jupyter Lingkaran Last Checkpoint: a few seconds ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3

In [ ]:

## # Perhitungan Luas dan keliling lingkaran

Luas dan keliling lingkaran dipengaruhi oleh nilai jari-jari( $r$ ) dari lingkaran tersebut. Persamaan untuk menghitung luas dan keliling lingkaran adalah sebagai berikut

### ## - Luas Lingkaran

Luas lingkaran dapat dihitung menggunakan persamaan  $L = \pi r^2$

! [Gambar Luas Lingkaran] (images/Circle\_Area.png)

### ## - Keliling Lingkaran

Keliling lingkaran dapat dihitung menggunakan persamaan  $L = 2 \pi r$

! [Gambar Keliling Lingkaran] (images/Circle-withsegments.png)

### ### Referensi

1. [\[Luas lingkaran\]](https://en.wikipedia.org/wiki/Circle) (<https://en.wikipedia.org/wiki/Circle>) <br/>
2. [\[Keliling lingkaran\]](https://en.wikipedia.org/wiki/Circumference) (<https://en.wikipedia.org/wiki/Circumference>)

# Menampilkan Markdown

- Tekan “Shift” + “Enter” untuk menampilkan hasil markdown

The screenshot shows a Jupyter Notebook interface with a single Markdown cell. The title of the cell is "Perhitungan Luas dan keliling lingkaran". The content discusses the area and circumference of a circle, providing the formulas  $A = \pi r^2$  and  $C = 2\pi r$ , along with diagrams illustrating the radius and diameter.

**Perhitungan Luas dan keliling lingkaran**

Luas dan keliling lingkaran dipengaruhi oleh nilai jari-jari( $r$ ) dari lingkaran tersebut. Persamaan untuk menghitung luas dan keliling lingkaran adalah sebagai berikut:

- **Luas Lingkaran**

Luas lingkaran dapat dihitung menggunakan persamaan  $A = \pi r^2$

Area =  $r^2$

Circle Area =  $\pi \times r^2$

- **Keliling Lingkaran**

Keliling lingkaran dapat dihitung menggunakan persamaan  $C = 2\pi r$

# Referensi markdown

---

- <https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet>
- <http://jupyter-notebook.readthedocs.io/en/latest/examples/Notebook/Working%20With%20Markdown%20Cells.html>
- <https://www.codecogs.com/latex/eqneditor.php>  
(membuat persamaan dalam bentuk LaTex)

# Menambahkan code

---

- Klik pada *cell* dan pilih opsi *Code*
- Tulis source code pada *cell* tersebut
- Tekan “Shift” + “Enter” untuk menampilkan hasil

```
In [3]: import math
r = 25
L = math.pi*math.pow(r,2)
K = 2 * math.pi * r
print("Luas : ",L)
print("Keliling : ", K)

Luas : 1963.4954084936207
Keliling : 157.07963267948966
```