

Dasar Pemrograman

Pendahuluan dan Tipe Data di Python



01

Pengenalan Python

Pengenalan Python

- Python merupakan bahasa pemrograman yang dibuat oleh Guido van Rossum pada tahun 1991
- Saat ini, Python dikelola oleh lembaga non-komersial Python Software Foundation (PSF)
- Python memiliki kemampuan dalam penanganan masalah dan mengutamakan sintaks yang mudah dimengerti dan dibaca (*readability*)
- Python mengadopsi *dynamic typing* secara opsional, artinya dalam membuat variable tidak perlu deklarasi tipe variabelnya walaupun bisa dilakukan.
- Sejak versi ke 3.6, Python sudah tersedia pilihan untuk *static typing*
- Zen of Python merupakan salah satu patokan dalam pengembangan Python.

Zen of Python

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!

Kelebihan Python

- Python merupakan bahasa pemrograman yang memiliki banyak pengguna.
- Memiliki sintaks sederhana dengan banyak kata kunci dalam Bahasa Inggris sehingga lebih *user friendly*.
- Mudah diaplikasikan dalam pengembangan produk.
- Memiliki dukungan komunikasi dan memiliki repository *library* dan modul yang *open source*.
- Menyediakan *library* yang dapat digunakan seperti *unit testing*, *regular expression*, *databases*, dll.
- Python Package Index (PIP) memiliki lebih dari 209.000 modul yang dapat diinstal dan digunakan di Python.

Kelebihan Python

- Python merupakan salah satu Bahasa pilihan untuk masuk ke *data science*
- Beberapa pemanfaatan Bahasa pemrograman Python:
 1. Pengembangan software (*software development*)
 2. Aplikasi website (*server-side*)
 3. Matematika dan ilmu data (*data science*)
 4. Machine Learning (ML)
 5. Internet of Things (IoT)



02

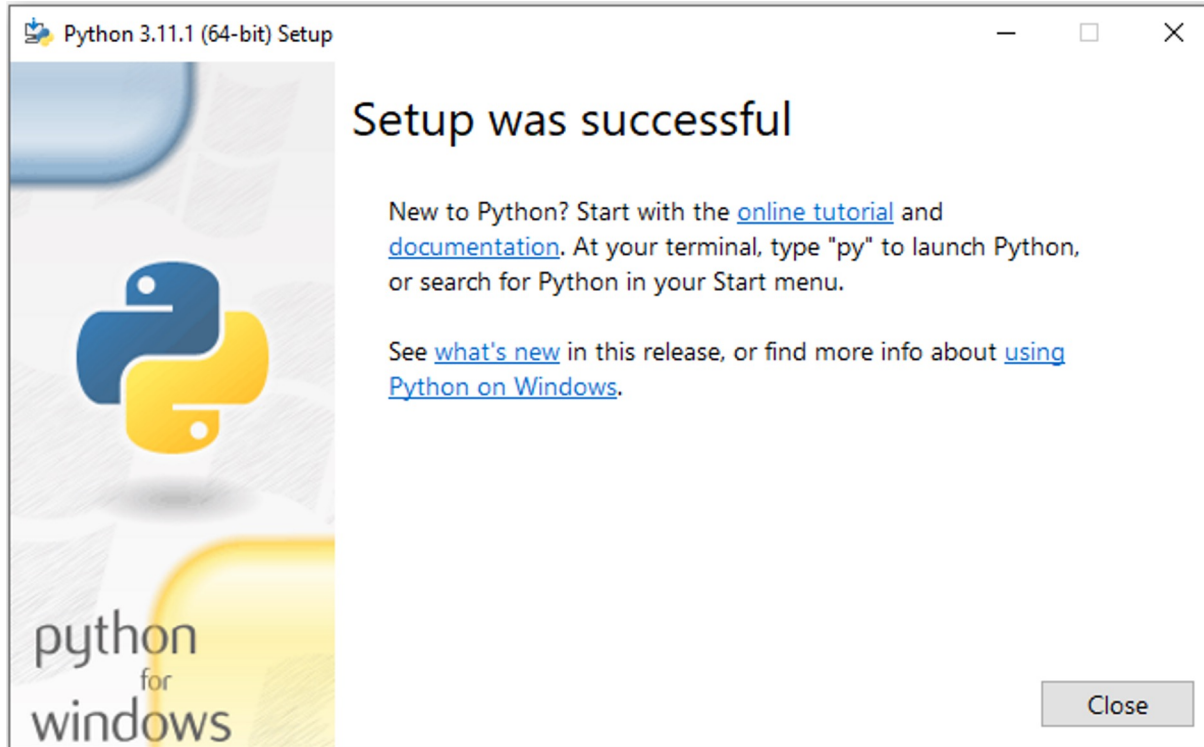
Instalasi Python

Instalasi Python: Windows

- Unduh pada laman <https://www.python.org/downloads/windows/> lalu pilih versi Windows 64 Bit atau Windows 32 Bit.
- Lakukan instalasi seperti pada gambar ini

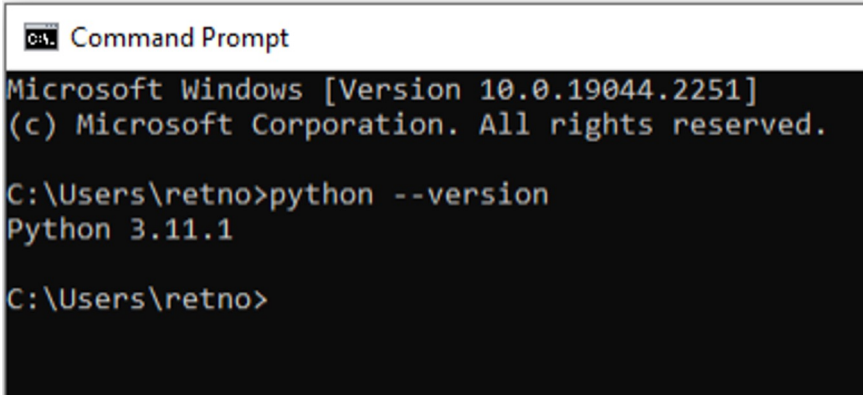


Instalasi Python: windows



Instalasi Python: Windows

Apabila proses instalasi Python sudah selesai, buka Command Prompt lalu ketik `python --version` apabila hasil yang ditampilkan seperti ini maka selamat! Python sudah terpasang di komputer Anda.



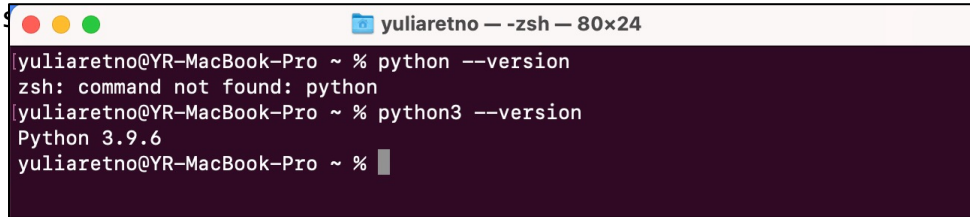
```
Command Prompt
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\retno>python --version
Python 3.11.1

C:\Users\retno>
```

Instalasi Python: Linux dan MacOS

- Umumnya Python sudah otomatis terpasang di Linux dan MacOS
- Untuk mengecek Python yang terpasang pada Linux atau MacOS buka terminal lalu ketik `python --version`

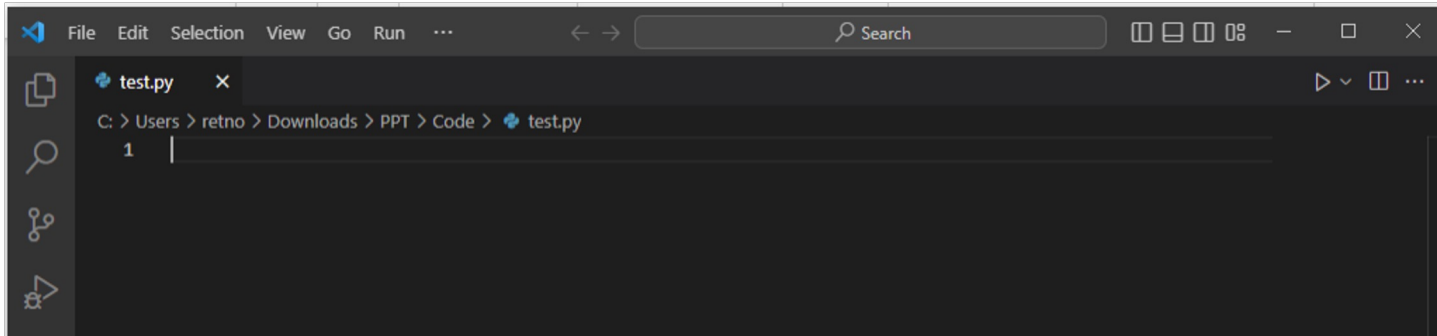


```
yuliaretno@YR-MacBook-Pro ~ % python --version
zsh: command not found: python
yuliaretno@YR-MacBook-Pro ~ % python3 --version
Python 3.9.6
yuliaretno@YR-MacBook-Pro ~ %
```

- Namun apabila Python belum terinstall bisa melakukan instalasi Python terlebih dahulu yang dapat diakses pada link <https://www.python.org/downloads/>

Setup Python: visual Studio Code

- *Download dan install software* Visual Studio Code dengan mengklik pada tautan <https://code.visualstudio.com/download> sesuaikan dengan Operating System anda
- Apabila sudah selesai, jalankan program Visual Studio Code
- Anda dapat membuat file Python dengan cara klik file -> ketik .py -> pilih folder dimana file tersebut akan disimpan -> tulis nama file. Contoh: test.py -> klik OK





03

Mode di Python

Mode di Python

01

Script

Compiler/interpreter
mengeksekusi file
dengan ekstensi .py

02

Interactive

Diakses melalui cmd
atau terminal

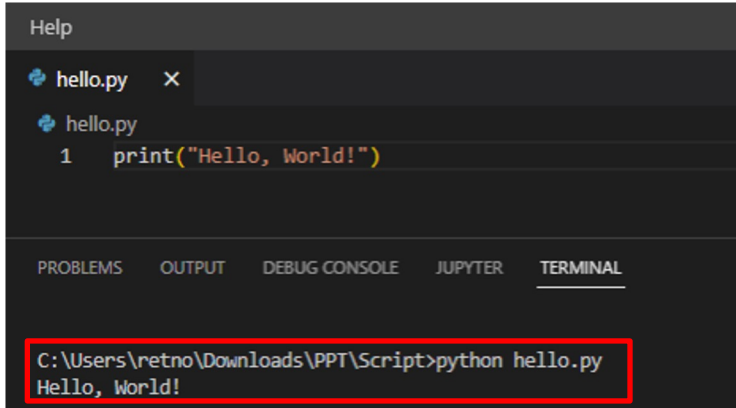
03

Notebook

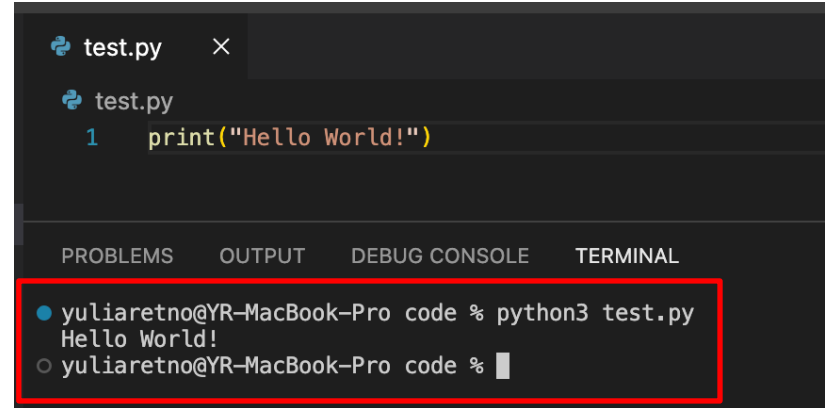
Melalui aplikasi web
seperti Jupyter
Notebook atau Google
Colaboratory

Mode Script

- File dengan ekstensi .py dieksekusi melalui terminal dengan menuliskan `python <nama_file>.py` atau `python3 <nama_file>.py` seperti pada gambar ini. Kemudian *output* akan ditampilkan di terminal



The screenshot shows a code editor with a file named `hello.py` containing the line `print("Hello, World!")`. The `TERMINAL` tab is active, displaying the command `C:\Users\retno\Downloads\PPT\Script>python hello.py` and its output `Hello, World!`. The terminal output is highlighted with a red box.



The screenshot shows a code editor with a file named `test.py` containing the line `print("Hello World!")`. The `TERMINAL` tab is active, displaying the command `yuliaretno@YR-MacBook-Pro code % python3 test.py` and its output `Hello World!`. The terminal output is highlighted with a red box.



05

Syntax Dasar Python

Sintax Dasar: Print

- Dalam pemrograman Python tidak diperlukan titik koma (;) di akhir kode
- Perintah `print()` *strings* dalam Python dilakukan dengan menggunakan kutip satu atau dua.

```
1 print("Hello, World!")
2 print('Hello, World!')
3 print("Good Morning, 'Ms. Bella'!")
4 print('Good Morning, "Ms. Bella"!')
```

```
Hello, World!
Hello, World!
Good Morning, 'Ms. Bella'!
Good Morning, "Ms. Bella"!
```

- Operator aritmatika dapat dilakukan di perintah `print()` untuk *strings*.

```
1 print("Hello" * 3)
```

```
HelloHelloHello
```

Sintax Dasar: Print

- Perintah `print()` juga dapat melakukan aritmatika sederhana untuk nilai *numeric*

```
1 print(1+1)
2 print(10-3)
3 print(2*6)
4 print(10/5)
```

```
2
7
12
2.0
```

Sintax Dasar: Comment

- *Comment* dapat digunakan untuk menjelaskan kode Python, membuat kode lebih mudah dibaca dan untuk mencegah kode dieksekusi saat sedang menguji kode.

```
1 # ini adalah komentar
2 print("Hi, everyone!")
3
4 # print("Kode ini tidak akan dieksekusi")
5 print("Kode ini akan dieksekusi")
```

Hi, everyone!

Kode ini akan dieksekusi

Sintax Dasar: Multi Line Comment

- Untuk menulis komentar lebih dari satu baris, dapat menggunakan tiga buah kutip dua (") seperti gambar di bawah ini.

```
1  # komentar ini
2  # ditulis lebih dari
3  # satu baris
4  print("Hello, World!")
5
6  """komentar ini
7  ditulis lebih dari
8  satu baris"""
9  print("Hello, World!")
```

```
Hello, World!
Hello, World!
```

Quiz 1

1. Bahasa pemograman Python mewajibkan titik koma (;) di akhir kode
 - a. Benar
 - b. Salah

2. Apa *output* dari kode `print("Hello, World!" * 2)`
 - a. Error
 - b. Hello, World
 - c. Hello, World!Hello, World!
 - d. Hello, World! Hello, World!



06

Variabel

Variabel

- Variabel adalah tempat penyimpanan data yang nilainya dapat berubah (*mutable*)
- Tidak perlu mendeklarasikan tipe datanya (walaupun bisa dilakukan)

nama_variable = <nilai>

```
1 a = "Hello"
2 b = 20
3
4 print(a)
5 print(b)
```

Hello
20

```
1 a = "Hello" #ini adalah string
2 print(a)
3
4 a = 20 #ini adalah integer
5 print(a)
```

Hello
20

Variabel

Ada beberapa aturan memberi nama variable di Python, yaitu antara lain:

- Nama variable harus dimulai dengan huruf atau *underscore*, **tidak** bisa dimulai dengan angka
- Nama variable hanya dapat mengandung huruf, angka, dan *underscore* (A-z, 0-9, dan _)
- Nama variable bersifat *case-sensitive* (contoh: nama, Nama, dan NAMA masing- masing merupakan variable yang berbeda)
- Tidak boleh ada spasi dalam nama variable

```
1 #contoh penamaan variabel yang dapat dilakukan
2 namavariabel = "Hello"
3 namaVariabel = "Hello"
4 NamaVariabel = "Hello"
5 _namavariabel = "Hello"
6 nama_variabel = "Hello"
7 namavariabel2 = "Hello"
```


Variabel: Casting Variabel

- Jika anda ingin membuat variable dengan menentukan tipe datanya, dapat dilakukan dengan menggunakan *casting*

nama_variable = tipe_data(nilai)

```
1 a = int(3) #ini adalah integer
2 b = float(3) #ini adalah float
3 c = str(3) #ini adalah string
4
5 print(a)
6 print(b)
7 print(c)
```

```
3
3.0
3
```

Variabel: Multi Words Variabel

Bila memberi nama variable lebih dari satu kata (multi-words) akan menyulitkan dalam membacanya, maka dapat dilakukan dengan beberapa teknik, yaitu:

- Snake case: setiap kata dipisah dengan menggunakan *underscore*
- Camel case: menuliskan huruf kapital di kata yang baru kecuali kata pertama
- Pascal case: setiap kata pertama dimulai dengan huruf kapital

```
1 nama_variabel_pertama = "Hello" #Snake Case
2 namaVariabelPertama = "Hello" #Camel Case
3 NamaVariabelPertama = "Hello" #Pascal Case
```

Variabel: Assign Multi Variabel

- Menyimpan banyak nilai ke dalam beberapa variabel

```
1 a, b, c = "apel", "belimbing", "ceri"  
2 print(a)  
3 print(b)  
4 print(c)
```

```
apel  
belimbing  
ceri
```

- Menyimpan satu nilai ke dalam beberapa variabel

```
1 a = b = c = "Buah"  
2 print(a)  
3 print(b)  
4 print(c)
```

```
Buah  
Buah  
Buah
```

Variabel: Output Variabel

- Perintah `print()` untuk beberapa variable sekaligus .

```
1 a = "Halo"  
2 b = "Selamat"  
3 c = "Pagi"  
4 print(a,b,c)
```

Halo Selamat Pagi

- Menggunakan operator `+` untuk menambahkan beberapa variable

```
1 a = "Halo"  
2 b = "Selamat"  
3 c = "Pagi"  
4 print(a + b + c)  
5  
6 x = 2  
7 y = 5  
8 print(x + y)
```

HaloSelamatPagi

7

Quiz 2

1. Apa output dari kode di bawah ini?

```
1 x = 2
2 y = 6
3 print(str(a + b) + "5" * 4)
```

- a. 2620
- b. 820
- c. 265555
- d. 85555



07

Input dan Output

Output

- Fungsi `print()` merupakan cara *output* ke *console* atau layar.
- Untuk memasukkan nilai *variable* pada *print string* dapat dilakukan dengan beberapa cara, berikut adalah contohnya

```
1 a = 20
2 print("Nilai a adalah", a)
```

Nilai a adalah 20

```
1 a = 20
2 b = 10
3 print("Nilai a adalah {0} dan b adalah {1}".format(a, b))
```

Nilai a adalah 20 dan b adalah 10

```
1 a = 20
2 b = 10
3 print(f"Nilai a adalah {a} dan b adalah {b}")
```

Nilai a adalah 20 dan b adalah 10

Input

- Untuk membuat *user* memberikan input pada program, bisa menggunakan fungsi `input()`

```
1 nama = input("Masukan nama Anda: ")
2
3 print(f"Selamat datang, {nama}")
```

Masukan nama Anda:

```
1 nama = input("Masukan nama Anda: ")
2
3 print(f"Selamat datang, {nama}")
```

Masukan nama Anda: Yulia

Selamat datang, Yulia

Input

- Ketika menggunakan fungsi `input()`, program baru akan berhenti sampai *user* memberikan input
- Apapun yang dimasukan sebagai input, fungsi `input()` akan mengubah tipe datanya menjadi strings, oleh sebab itu perlu menggunakan *typecasting* untuk merubahnya menjadi tipe data yang diinginkan. (Note: materi tipe data ada pada slide selanjutnya).

```
1 umur = input("Masukan umur Anda")
2
3 print(f"Umur anda: {umur}")
4 print(type(umur))
```

```
Masukan umur Anda20
Umur anda: 20
<class 'str'>
```

```
1 umur = int(input("Masukan umur Anda"))
2
3 print(f"Umur anda: {umur}")
4 print(type(umur))
```

```
Masukan umur Anda20
Umur anda: 20
<class 'int'>
```



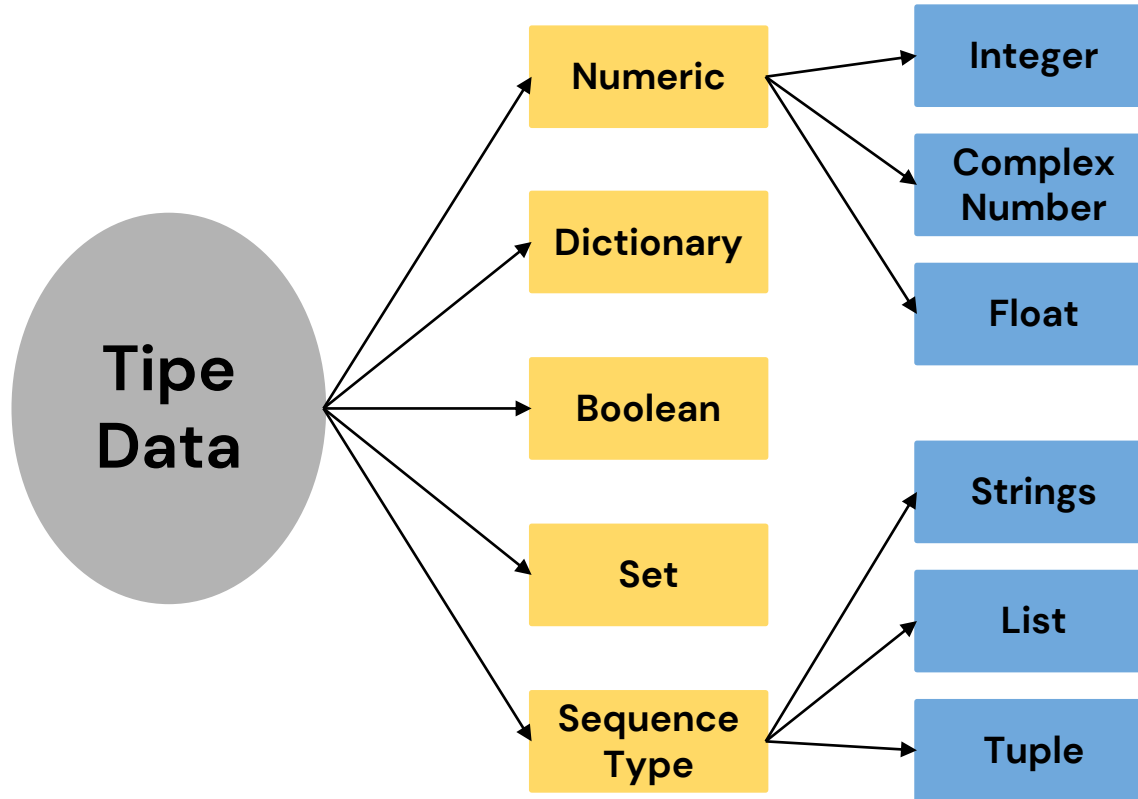
08

Tipe Data

Tipe Data

- Tipe data merupakan cara untuk mengklasifikasi data berdasarkan jenis data tersebut. Hal ini dilakukan agar kompiler dapat mengetahui bagaimana sebuah data akan dilakukan.
- Dalam pemrograman, tipe data merupakan salah satu konsep yang penting
- Sebuah variabel dapat menyimpan data dengan tipe data yang berbeda dan setiap tipe data tersebut dapat melakukan berbagai hal yang berbeda.

Tipe Data di Python



Tipe Data di Python

- Fungsi `type()` digunakan untuk mengecek tipe data dari suatu variable atau nilai.

```
1 a = "Hello"  
2 b = 20  
3 c = True  
4 d = 10.8  
5  
6 print(type(a))  
7 print(type(b))  
8 print(type(c))  
9 print(type(d))
```

```
<class 'str'>  
<class 'int'>  
<class 'bool'>  
<class 'float'>
```

- Untuk mengubah tipe data tertentu bisa menggunakan *casting*

```
1 a = 5  
2 print(type(a))  
3  
4 x = float(a)  
5 print(type(x))
```

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

Tipe Data di Python

Berikut adalah table contoh *value* dari suatu variable berserta tipe datanya

Contoh Variabel	Tipe Data	Keterangan
a = "halo"	str	Strings
a = 2	int	Integer
a = 2.5	float	Float
a = 2j	complex	Complex
a = ["apel", "belimbing", "cherry"]	list	List
a = ("apel", "belimbing", "cherry")	tuple	Tuple
a = {"buah" : "apel", "jumlah" : 2}	dict	Dictionary
a = {"apel", "belimbing", "cherry"}	set	Sets
a = True	boolean	Boolean

Quiz 3

1. Manakah diantara pilihan berikut yang **bukan** merupakan tipe data di Python?
 - a. Dictionary
 - b. Complex
 - c. Boolean
 - d. Object
 - e. Tuple
2. Manakah variable yang memiliki nilai dengan tipe data float?
 - a. `a = 4j`
 - b. `b = ["apel", "belimbing", "cherry"]`
 - c. `c = 2.0`
 - d. `d = 6`
 - e. `e = {"buah" : "apel", "jumlah" : 2}`



09

Python Numeric

Integer

- Tipe data *numeric* adalah jenis tipe data yang bersifat angka yang dapat ditambah, dikurang, dibagi, dikali, dll.
- Tipe data integer (int) merupakan nilai bilangan bulat positif atau negatif, tanpa decimal dan memiliki panjang yang tak terbatas (*unlimited*)
- Setiap variabel yang memiliki nilai bulat, pasti dikategorikan sebagai integer.

```
1 a = 0
2 b = 298490327849029045723907
3 c = -629
4
5 print(type(a))
6 print(type(b))
7 print(type(c))
```

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

Float

- Variabel dengan tipe data *float* digunakan untuk menyimpan nilai pecahan atau decimal.
- Batas panjang tipe data *float* adalah 17 angka dibelakang koma.
- *Float* juga dapat berupa bilangan ilmiah dengan huruf “e” yang merepresentasikan pangkat 10

```
1 a = 5.0
2 b = -8.3
3 c = 35e3
4
5 print(type(a))
6 print(type(b))
7 print(type(c))
```

```
<class 'float'>
<class 'float'>
<class 'float'>
```

```
1 x = 0.012345678901234567890123456789
2 print(x)
```

```
0.012345678901234568
```

Complex

- Tipe data *complex* merupakan data yang merepresentasikan nilai imajiner
- Nilai imajiner dalam Python ditulis dengan karakter huruf “j”

```
1 a = 2j
2 b = 5+2j
3 c = a + b
4
5 print(type(a))
6 print(type(b))
7 print(c)
```

```
<class 'complex'>
<class 'complex'>
(5+4j)
```

Fungsi Numeric

- Berikut adalah beberapa fungsi untuk tipe data numeric di Python

Fungsi	Nama	Keterangan	Contoh
<code>abs(x)</code>	Absolute	Mengembalikan nilai absolute dari suatu nilai x	<code>abs(-10) = 10</code>
<code>max(x1, x2, ... xn)</code>	Max	Mengembalikan nilai paling besar dari seluruh nilai	<code>max(1, 3, 5) = 5</code>
<code>min(x1, x2, ... xn)</code>	Min	Mengembalikan nilai paling kecil dari seluruh nilai	<code>min(1, 3, 5) = 1</code>
<code>pow(x, y)</code>	Pow	Mengembalikan nilai $x^{**}y$	<code>pow(2, 3) = 8</code>
<code>round(x, a)</code>	Round	Membulatkan nilai x sejumlah nilai a dibelakang koma	<code>round(10.7374, 2) = 10.74</code>

Quiz 4

1. Berapakah panjang digit decimal maksimal untuk tipe data float?
 - a. 15
 - b. 19
 - c. 10
 - d. 17
 - e. 12
2. Apa *output* dari kode `print(pow(3, 3))`
 - a. 9
 - b. 27
 - c. 6
 - d. 3.000
 - e. 3



10 Python Strings

Strings

- Tipe data strings biasa disebut juga sebagai tipe data teks karena digunakan untuk menyimpan sebuah teks.
- Strings harus diapit oleh tanda kutip satu (") atau kutip dua (" ").

```
1 a = "Hello"  
2 b = 'Halo'  
3  
4 print(type(a))  
5 print(type(b))
```

```
<class 'str'>  
<class 'str'>
```

- Untuk mengecek panjang dari sebuah strings dapat menggunakan fungsi `len()`

```
1 a = "Hello, World!"  
2  
3 print(len(a))
```

13

Multiline Strings

- Untuk membuat multiline string bisa diapit menggunakan tiga tanda-kutip-dua atau tiga tanda-kutip-satu

```
1 a = """Ini adalah contoh
2 dari multiline
3 string"""
4
5 print(a)
```

Ini adalah contoh
dari multiline
string

```
1 b = '''Ini adalah contoh
2 dari multiline
3 string'''
4
5 print(b)
```

Ini adalah contoh
dari multiline
string

Slicing Strings

- Slicing strings merupakan mengambil karakter dari strings dengan rentang tertentu

`a = "Halo, selamat pagi"`

H	a	l	o	,		S	e	l	a	m	a	t		P	a	g	i
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

- Slicing dengan range tertentu → `a[index_start:index_finish]`

```
1 a = "Halo, selamat pagi"
2
3 print(a[1:4])
```

alo

Slicing Strings

- Slicing string dari karakter pertama → `a[:index_finish]`

```
1 a = "Halo, selamat pagi"
2
3 print(a[:5])
```

Halo,

- Slicing hingga karakter terakhir → `a[index_start:]`

```
1 a = "Halo, selamat pagi"
2
3 print(a[10:])
```

mat pagi

Modifikasi Strings

- Fungsi `upper()` → mengubah string menjadi huruf kapital

```
1 a = "Hello, World!"  
2  
3 print(a.upper())
```

HELLO, WORLD!

- Fungsi `lower()` → mengubah string menjadi huruf kecil

```
1 a = "Hello, World!"  
2  
3 print(a.lower())
```

hello, world!

Modifikasi Strings

- Fungsi `strip()` → menghapus spasi sebelum/setelah teks sesungguhnya (whitespace)

```
1 a = " Hello, World!"
2 b = "Hello, World! "
3
4 print(a.strip())
5 print(b.strip())
```

```
Hello, World!
Hello, World!
```

- Fungsi `replace()` → mengganti suatu karakter dengan karakter lain dalam string
- Fungsi `split()` → mengembalikan string menjadi list di antara karakter pemisah tertentu

```
1 a = "Hello, World!"
2
3 print(a.replace("H", "J"))
4 print(a.split(","))
```

```
Jello, World!
['Hello', ' World!']
```

Strings Concatenation

- Strings concatenation adalah menggabungkan dua buah strings atau lebih dengan menggunakan operator tambah “+”

```
1 a = "Bahasa"  
2 b = "Pemograman"  
3 c = "Python"  
4  
5 print(a + b + c)  
6 print(a + " " + b + " " + c)
```

BahasaPemogramanPython
Bahasa Pemograman Python

Strings Formatting

- Fungsi `format()` digunakan untuk menggabungkan string dan number di Python.
- Fungsi ini mengambil argument yang diteruskan, memformatnya, dan menempatkannya ke dalam string yang tempat `{}` berbeda

```
1 umur = 20
2 text = "Nama saya Bela umur saya {}"
3
4 print(text.format(umur))
```

Nama saya Bela umur saya 20

```
1 jumlah = 4
2 harga = 5000
3 uang = 20000
4 order = "Saya beli {} buah apel yang harganya Rp {} dengan uang Rp {}"
5
6 print(order.format(jumlah, harga, uang))
```

Saya beli 4 buah apel yang harganya Rp 5000 dengan uang Rp 20000

Strings Formatting

```
1 jumlah = 4
2 harga = 5000
3 uang = 20000
4 order = "Saya beli {0} buah apel yang harganya Rp {2} dengan uang Rp {1}"
5
6 print(order.format(jumlah, harga, uang))
```

Saya beli 4 buah apel yang harganya Rp 20000 dengan uang Rp 5000

```
1 jumlah = 4
2 harga = 5000
3 uang = 20000
4 print(f'Saya beli {jumlah} buah apel yang harganya Rp {harga} dengan uang Rp {uang}')
```

Saya beli 4 buah apel yang harganya Rp 5000 dengan uang Rp 20000

Escape Characters

- Escape character digunakan untuk menyisipkan karakter dalam sebuah strings

```
1 text = "Halo, Selamat Datang \"Bella\" dari Bandung."  
2 print(text)
```

Halo, Selamat Datang "Bella" dari Bandung.

```
1 text = "Halo, Selamat Datang Bella\n dari Bandung."  
2 print(text)
```

Halo, Selamat Datang Bella
dari Bandung.

Escape Characters

Beberapa escape karakter yang dapat digunakan di Python

Karakter	Keterangan
\'	Single Quote
\\	Backslash
\n	Baris Baru
\r	Carriage Return
\t	Tab
\b	Backspace
\\ooo	Octal Value
\xhh	Hex Value



Latihan

Latihan

- Buatlah program dari Flowchart pada pertemuan minggu lalu dengan menggunakan bahasa pemrograman Python.
- Buatlah sebuah program Python dimana *user* harus memberikan input nama dan umur, lalu *output* program yaitu:

Selamat datang, <nama_user>! Umur Anda sekarang adalah <umur_user>

- Buatlah program kalkulator sederhana untuk pertambahan tiga buah bilangan yang di-*input* oleh *user*. Contoh:

Input a = 1

Input b = 2

Input c = 3

Maka *output*-nya: "Hasil dari penjumlahan $1 + 2 + 3 = 6$ "



Materi Tambahan

Identasi

- Aturan penulisan ini mengacu pada PEP-008
- Python tidak menggunakan perbedaan atau kurung kurawal, tetapi menggunakan identasi untuk menulis kode bertingkat dengan 4 spasi pada setiap tingkatan identasi atau tab dalam notebook.

```
1 Perintah tingkat 1:  
2     Perintah tingkat 2 ke-1  
3     Perintah tingkat 2 ke-2  
4         Perintah tingkat 3 ke-1  
5         Perintah tingkat 3 ke-2
```

Penggantian Baris

- Dalam menulis kode, terkadang kita memerlukan pergantian baris karena kode tidak cukup (terlalu panjang)

```
1  # Rata kiri dengan kurung atau pemisah
2  contoh = nama_function(variabel_satu, variabel_dua,
3                          variabel_tiga, variabel_empat)
4
5  # Hanging indents dengan penambahan level indentasi
6  contoh = nama_function(
7      variabel_satu, variabel_dua,
8      variabel_tiga, variabel_empat)
9
10 # Contoh kesalahan (Tidak rata kiri dengan bagian yang relevan)
11 contoh = nama_function(variabel_satu, variabel_dua,
12                          variabel_tiga, variabel_empat)
```

Penggantian Baris

```
1  # Tambahkan identasi ekstra untuk memisahkan parameter/argumen
2  def nama_function(
3      variabel_satu, variabel_dua, variabel_tiga,
4      variabel_empat):
5      print("Hasil")
6
7  # Contoh kesalahan (sulit dibedakan antara fungsi baru atau baris lanjutan)
8  def nama_function(
9      variabel_satu, variabel_dua, variabel_tiga,
10     variabel_empat):
11     print("Hasil")
```

Sebelum Operator Binary

```
1  # Disarankan
2  rumus_matematika = (variable_satu
3                      + variable_dua
4                      - (variabel_satu - variabel_dua)
5                      * variabel_tiga)
6
7
8  # Tidak disarankan
9  rumus_matematika = (variable_satu +
10                     variable_dua -
11                     (variabel_satu - variabel_dua) *
12                     variabel_tiga)
```


Whitespace

- Hindari penambahan whitespace yang tidak perlu

```
1 # Contoh Benar
2 a = 1
3 b = 2
4 ini_adalah_variabel = 3
5
6 # Contoh Salah
7 a           = 1
8 b           = 2
9 ini_adalah_variabel = 3
10
```

```
1 # Contoh Benar
2 if a > b:
3     print(a, b)
4
5 # Contoh Salah
6 if a > b :
7     print(a , b)
```

Whitespace

- Hindari penambahan whitespace dibelakang statement
- Tambahkan satu spasi di kanan dan kiri operator

```
1  # Contoh Benar
2  x = a + b
3  x += 1
4  x = a*a + b*b
5  x = (a+b) * (a-b)
6
7  # Contoh Salah
8  x=a+b
9  x +=1
10 x = a * b + b * b
11 x = (a + b) * (a - b)
12
```

Dokumentasi

- Buat dokumentasi untuk semua modul, fungsi, kelas, dan method yang bersifat *public*
- Untuk docstring satu baris disarankan untuk menggunakan 3-tanda-kutip-dua `"""` walaupun sebenarnya bisa dilakukan dengan 3-tanda-kutip-satu `"`

```
1 # Contoh
2
3 """Ini adalah dokumentasi
4 yang sifatnya adalah opsional
5 """
```

Quiz

1. Manakah diantara pilihan berikut yang penulisannya sesuai dengan aturan yang dianjurkan?

a.

```
1 def rumus luasPersegi(sisi):  
2     hasil = sisi * sisi  
3     print(hasil)
```

b.

```
1 if a==b:  
2     print("a sama dengan b" )
```

c.

```
1 if a > 0:  
2     a += 2  
3     print(a)
```

Terimakasih

Do you have any questions?

addyouremail@freepik.com

+91 620 421 838

yourcompany.com

CREDITS: This presentation template was created by
Slidesgo, including icons by **Flaticon**, and
infographics & images by **Freepik**

Please keep this slide for attribution