

1. PENGENALAN ALGORITMA PEMROGRAMAN & PYTHON

Apa itu Python

Python adalah bahasa pemrograman yang populer saat ini. Bahasa Python dibuat oleh Guido van Rossum, dan dirilis pada tahun 1991. Python dapat digunakan untuk Pengembangan web (sisi server), Pengembangan perangkat lunak atau membuat aplikasi (software), menyelesaikan persamaan matematika, pembuatan skrip sistem dan pemrograman mikrokontroler (Micro- Python) .

Beberapa fungsi Bahasa Python adalah Python dapat digunakan di server untuk membuat aplikasi web, Python dapat digunakan bersama perangkat lunak untuk membuat alur kerja, Python dapat terhubung ke sistem database, Bahasa Python juga dapat membaca dan memodifikasi file, Python dapat digunakan untuk menangani data besar dan melakukan matematika yang kompleks, dan Python dapat digunakan untuk pembuatan prototipe dengan cepat, atau untuk pengembangan perangkat lunak siap produksi.

Alasan untuk menggunakan dan mempelajari Python adalah Python dapat bekerja pada platform yang berbeda (Windows, Mac, Linux, Raspberry Pi, dll), Python memiliki sintaks sederhana yang mirip dengan bahasa Inggris, Python memiliki sintaks yang memungkinkan pengembang untuk menulis program dengan lebih sedikit baris daripada beberapa bahasa pemrograman lainnya, Python berjalan pada sistem interpreter, artinya kode dapat dieksekusi segera setelah ditulis. Ini berarti pembuatan prototipe bisa sangat cepat, Python dapat diperlakukan dengan cara prosedural, cara berorientasi objek atau cara fungsional, dan Python memiliki banyak Pustaka.

Dalam materi ini Python akan ditulis dalam editor teks. Programmer dapat menulis Python dalam Lingkungan Pengembangan Terintegrasi, seperti Visual Studio Code, Jupyter, Thonny, Pycharm, Netbeans Code Visual, atau Eclipse yang sangat berguna saat mengelola koleksi file Python yang lebih besar.

Sintak Python dibandingkan dengan bahasa pemrograman lain memiliki beberapa kelebihan. Python dirancang agar mudah dibaca, dan memiliki beberapa kesamaan dengan Bahasa Inggris dengan pengaruh dari matematika. Python menggunakan baris baru untuk menyelesaikan perintah, berbeda dengan bahasa pemrograman lain yang sering menggunakan titik koma atau tanda kurung. Python mengandalkan indentasi, menggunakan spasi, untuk mendefinisikan ruang lingkup; seperti cakupan loop, fungsi, dan kelas. Bahasa pemrograman lain sering menggunakan tanda kurung kurawal untuk tujuan ini.

Contoh perbandingan Pemrograman C++, C#, Java dan Python untuk menampilkan tulisan "Hello World!". Terlihat bahwa bahasa pemrograman Python memiliki sintak terpendek, lebih sederhana, mudah, tidak perlu menggunakan titik koma sebagai akhiran program dan tidak perlu menggunakan kurung kurawal.

Pemrograman C# = 5 Baris Program

```
using System;
namespace HelloWorld {
    class Program {
        static void Main(string[] args) {
            Console.WriteLine("Hello World!");
        }
    }
}
```

Pemrograman C++ = 4 Baris

```
#include <iostream>
using namespace std;
int main() {
    cout << "Hello World!";
    return 0;
}
```

Pemrograman Java = 3 Baris

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

Pemrograman Python = 1 Baris

```
print("Hello, World!")
```

Instalasi Python

Banyak komputer sudah menginstal python. Untuk memeriksa apakah Anda telah menginstal python pada PC Windows, jalankan perintah berikut di Command Line (cmd.exe):

```
C:\Users\Your Name>python --version
```

Untuk memeriksa apakah Anda telah menginstal python di Linux atau Mac, maka di linux buka baris perintah atau di Mac buka Terminal dan ketik:

```
python --version
```

Jika ternyata Anda tidak menginstal python di komputer Anda, Anda dapat mengunduhnya secara gratis dari situs web berikut: <https://www.python.org/>

Panduan Memulai Python

Python adalah bahasa pemrograman yang diinterpretasikan, artinya pengembang menulis file Python (.py) di editor teks dan kemudian memasukkan file-file itu ke dalam interpreter python untuk dieksekusi. Cara menjalankan file python seperti ini pada baris perintah:

```
C:\Users\Your Name>python helloworld.py
```

Dengan "helloworld.py" adalah nama file python Anda.

Mari tulis file Python pertama kita, bernama helloworld.py, yang bisa dilakukan di editor teks manapun seperti notepad.

```
helloworld.py
```

```
print("Hello, World!")
```

Sederhana seperti itu. Simpan file Anda. Buka baris perintah Anda, arahkan ke direktori tempat Anda menyimpan file Anda, dan jalankan:

```
C:\Users\Your Name>python helloworld.py
```

Outputnya harusnya terbaca:

```
Hello, World!
```

Selamat, Anda telah menulis dan menjalankan program Python pertama Anda.

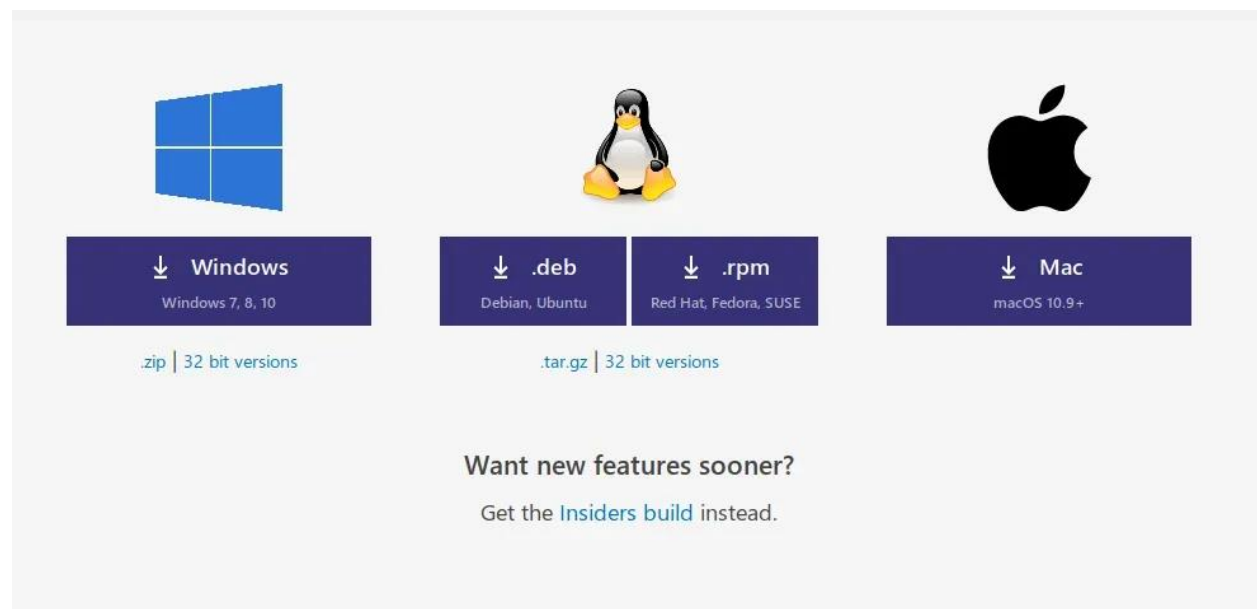
Menggunakan Visual Studio Code Untuk Python

Visual Studio Code merupakan salah satu teks editor terpopuler dalam beberapa tahun terakhir. Teks Editor open-source besutan Microsoft ini berhasil menarik perhatian developer dari berbagai bidang, mulai dari web development, desktop, mobile, dan lain sebagainya.

Hal itu tidak mengherankan mengingat fitur-fitur dari VSCode sangat membantu proses *ngoding*, plus didukung dari marketplace dengan extensions yang sangat banyak siap untuk mempermudah developer.

1. Pastikan VSCode sudah terinstall

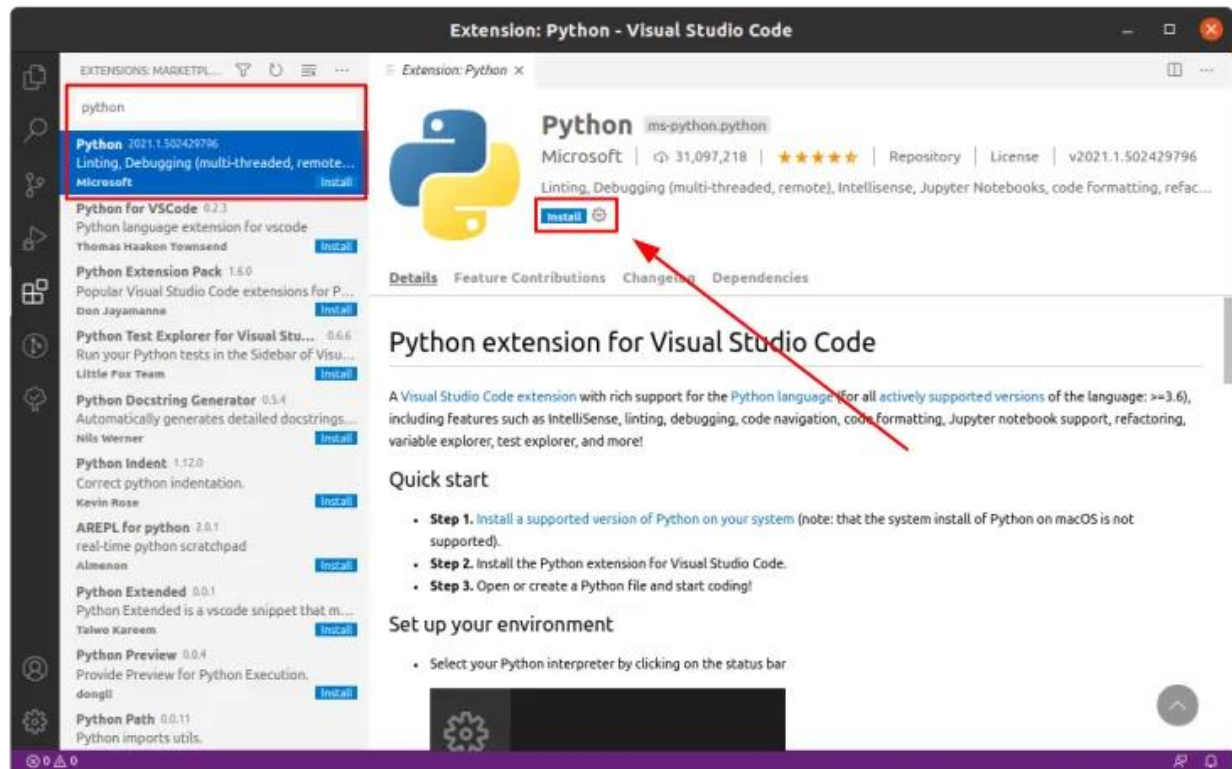
Untuk mengunduh paket instalasi kunjungi situs resmi <https://code.visualstudio.com/>. Pastikan VSCode telah terinstall dengan baik di PC.



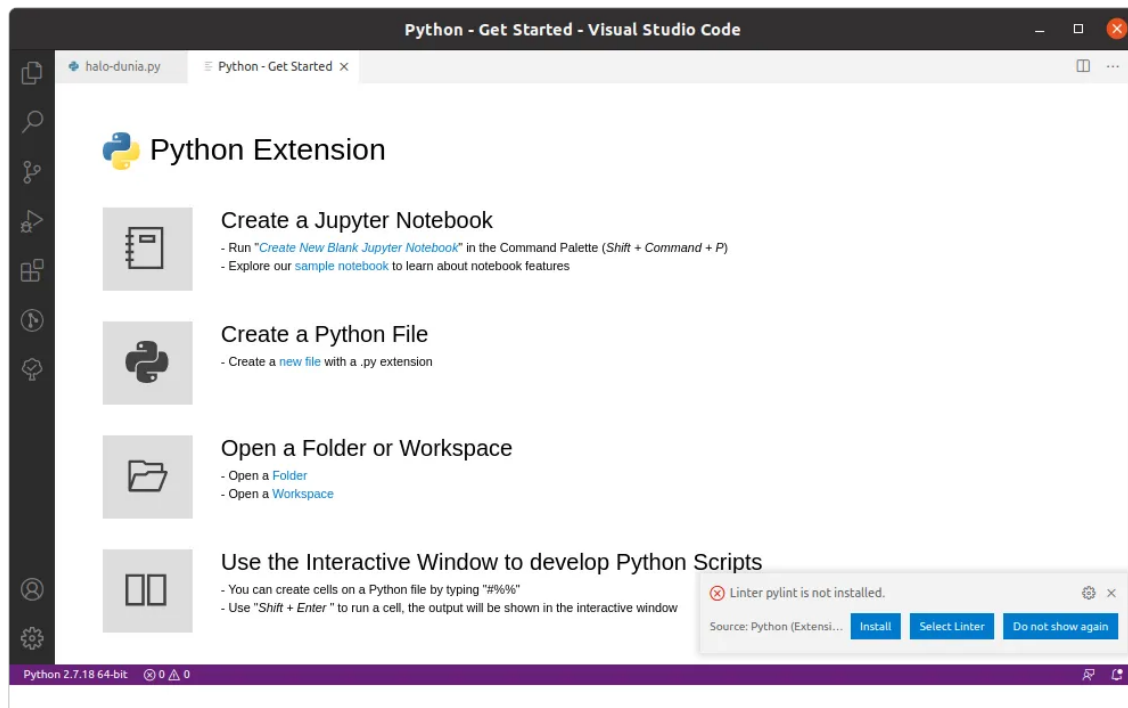
2. Install Python Extension

Buka marketplace extension dari menu yang ada di sebelah kiri, atau dengan menekan *shortcut* **Ctrl+Shift+X** pada kibor.

1. Lalu ketik **"python"** pada kolom pencarian.
2. Pilih hasil yang paling atas
3. Klik tombol **install**

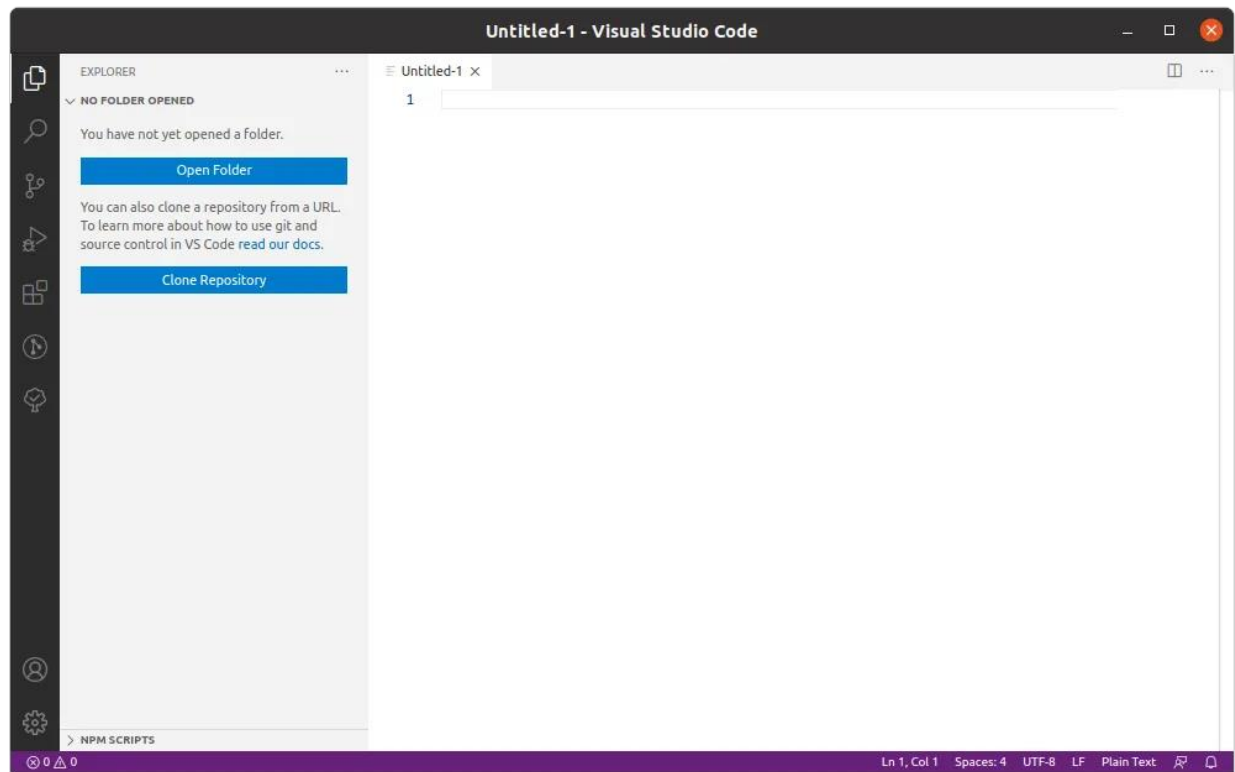


Setelah proses instalasi selesai, kita biasanya akan mendapatkan overview yang menjelaskan beberapa fitur kunci dari plugin tersebut.



3. Buat File Baru

Selanjutnya kita bisa langsung membuat file baru dengan klik menu **File** pada toolbar, lalu memilih tombol **New File**.

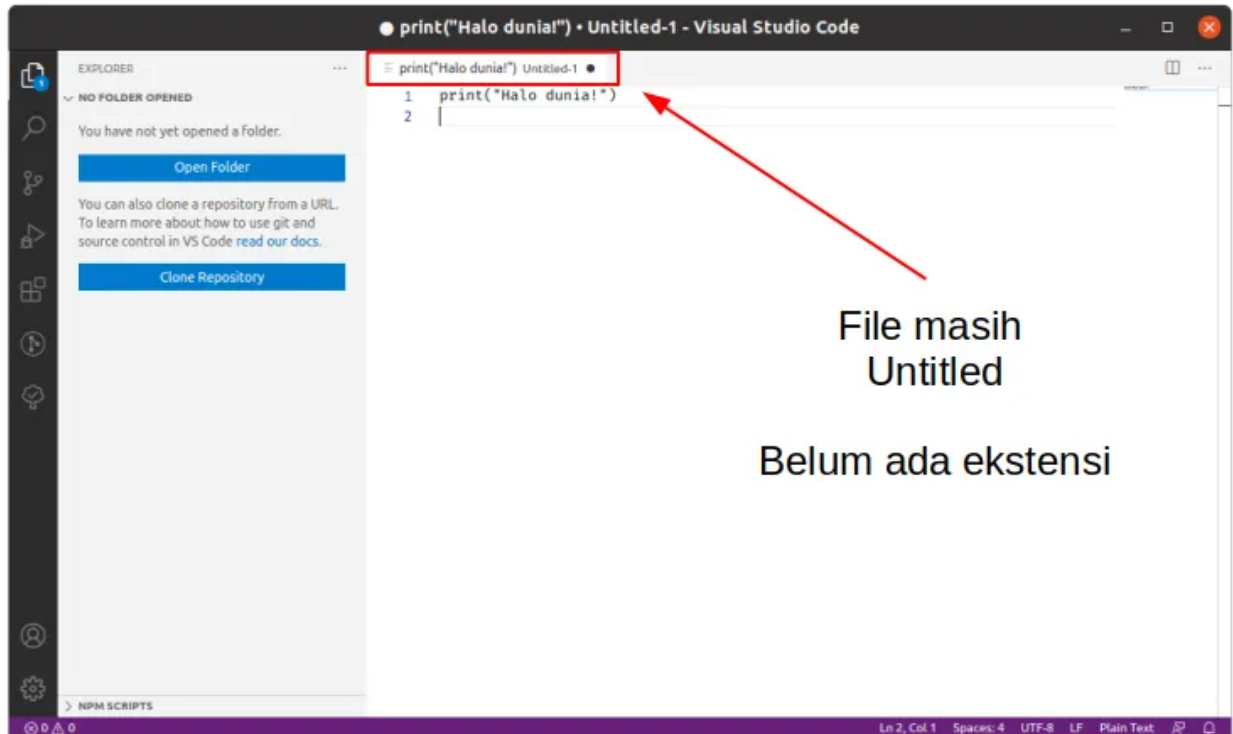


Setelah itu langsung saja tuliskan kode program berikut:

```
print("Halo dunia!")
```

Loh, kok?

Tulisannya tidak berwarna?



Iya, untuk sekarang, file kita masih belum berekstensi apa pun. Oleh karena itu, VSCode tidak tahu ini file dari bahasa pemrograman apa, sehingga fitur *syntax highlighting*-nya pun belum berfungsi.

4. Save dengan ekstensi .py

Agar berfungsi dengan baik, mari kita save file yang sudah kita buat dengan ekstensi `.py`.

Untuk menyimpan file baru, kita bisa menekan tombol `Ctrl+S` dari kibor.

Sekarang, kode python yang kita tulis sudah berwarna, dan fitur-fitur yang ditawarkan oleh Python Extension juga telah berfungsi.

Tips

Untuk kebutuhan project, anda bisa membuat folder tersendiri yang berisi file-file python. Kemudian buka folder tersebut secara keseluruhan menggunakan visual studio code.

5. Python di VSCode *in action*

Sebelum kita run project pertama kita di VSCode.

Kita bisa mencoba kode program berikut:

```
import time

for i in range(5):
    print('Halo dunia!')
    time.sleep(1)
```

6. Run Python di Visual Studio Code

Sekarang, kita akan menjalankan kode python yang telah sudah ditulis.

Pastikan Pilih Interpreter Yang Benar

Sejauh ini python memiliki 2 versi besar:

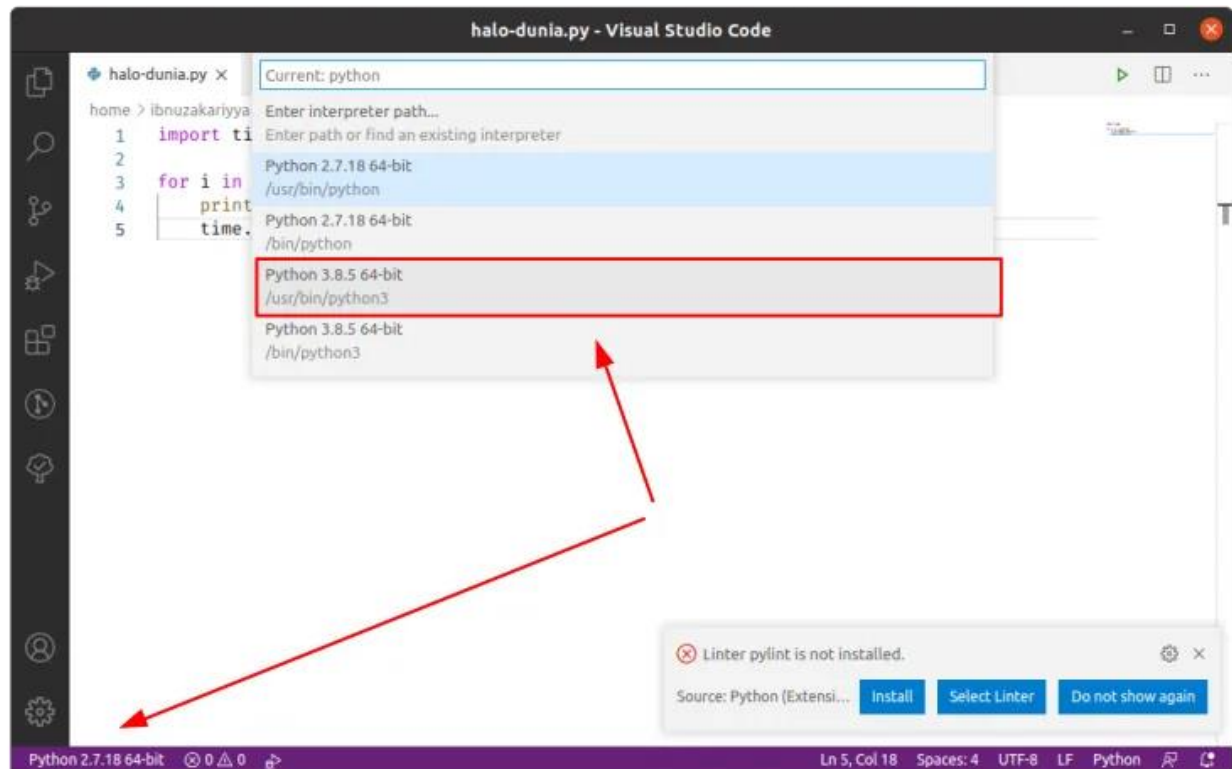
- Versi 3
- dan Versi 2

Python versi 2 memang sudah *discontinue*, akan tetapi, masih banyak aplikasi yang bergantung pada python versi 2, sehingga di berbagai distro linux pun python 2 masih terpasang secara default.

Kalau memang di OS kita terpasang lebih dari satu versi python, pastikan kita memilih interpreter yang benar.

Kalau masih akan memulai belajar python, nggak perlu ragu lagi untuk memilih python 3 versi yang terbaru.

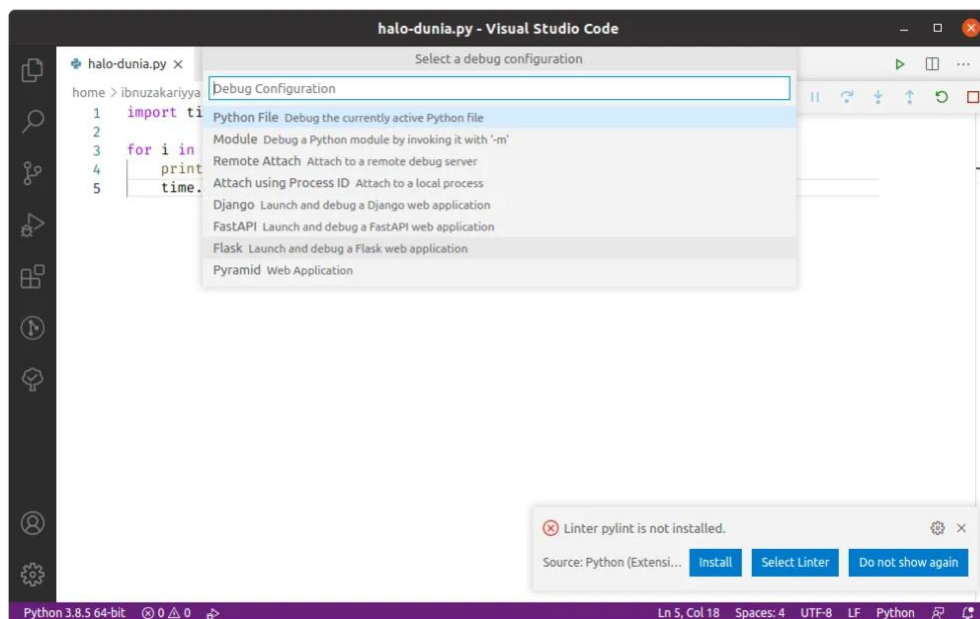
Untuk memilih interpreter python pada VSCode, anda bisa klik tombol yang ada di pojok kiri bawah:



Pilih Debug Configuration dan... Run!

Untuk menjalankan kode python, kita bisa langsung menekan tombol **f5**.

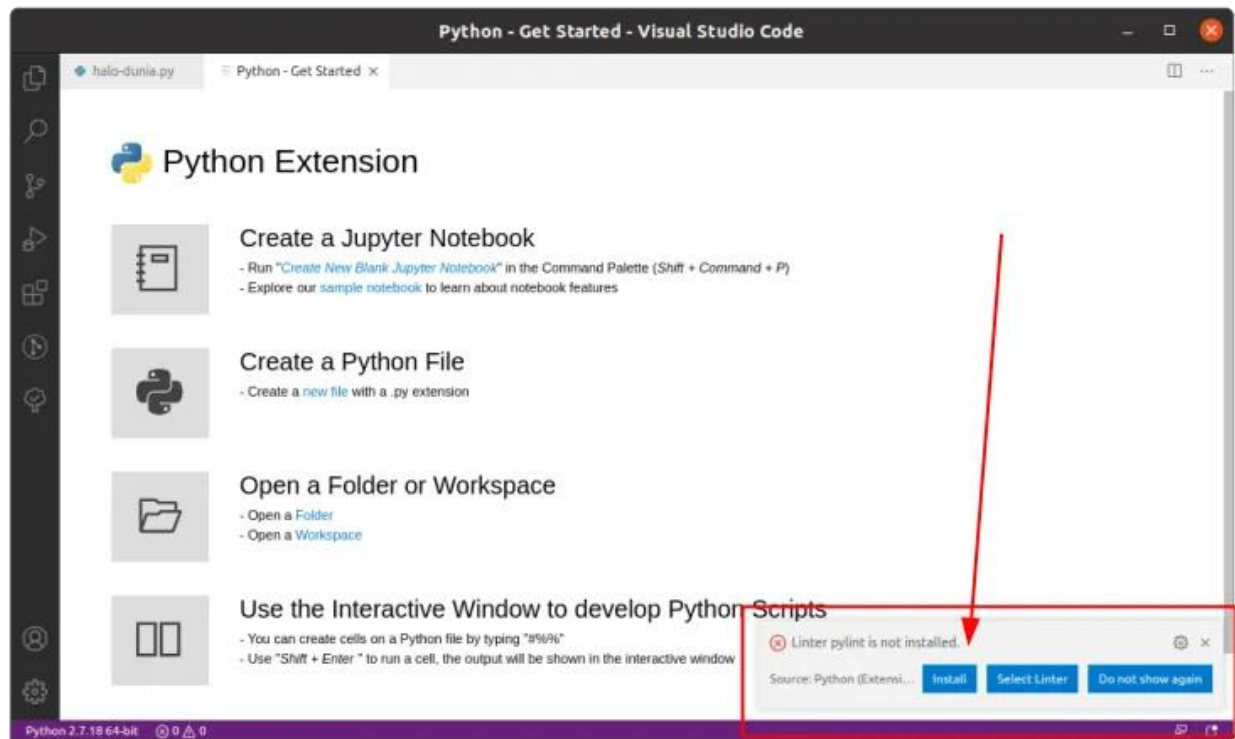
Jika muncul pilihan debug configuration, langsung saja pilih **Python File**.



Setelah itu, jika tidak ada error, maka akan melakukan sebuah perulangan (for) untuk menampilkan tulisan "Halo dunia" sebanyak 5 kali, dengan jeda 1 detik setiap perulangan.

7. Python Linting (PyLint)

Mungkin kalian penasaran dengan popup yang dari tadi muncul di pojok kanan bawah VSCode:



Popup tersebut menawarkan kita untuk memasang python linting ke sistem sehingga VSCode bisa memanfaatkannya.

Apa gunanya linting?

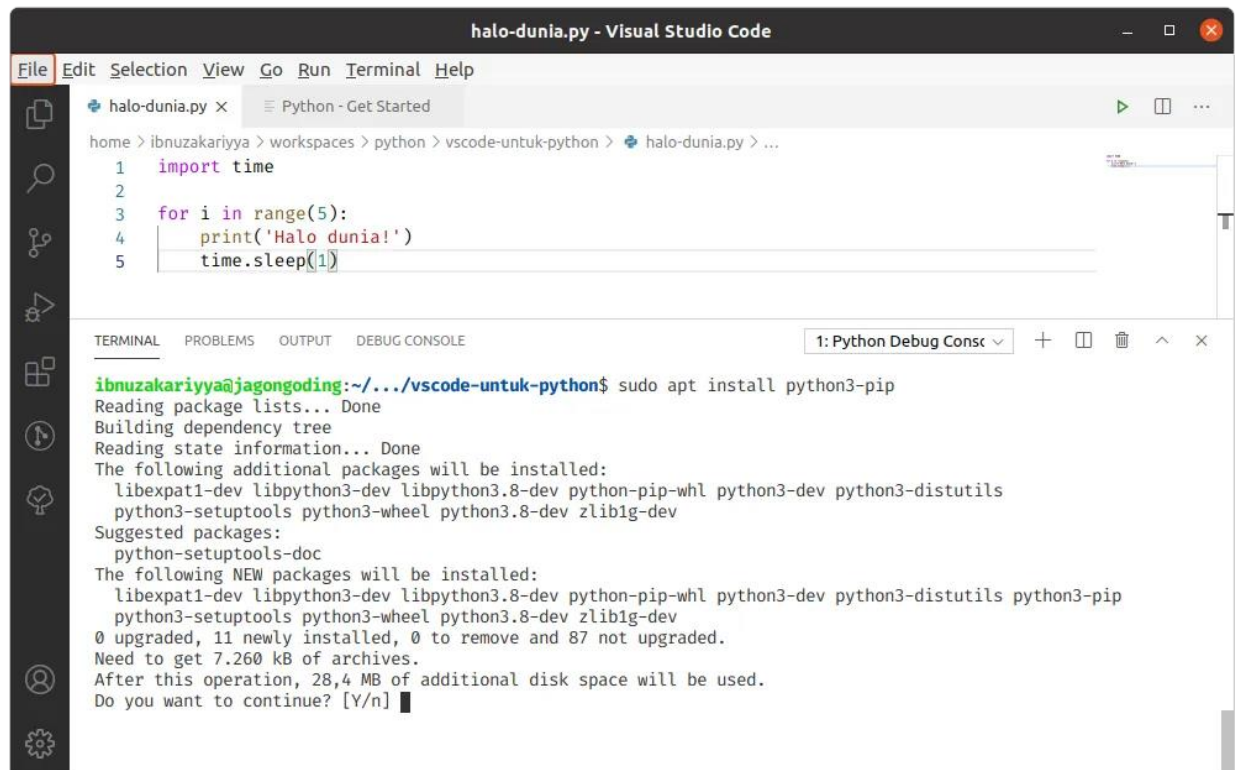
Linting berguna untuk menampilkan mana baris yang error pada kode program kita secara langsung.

Install python linting

Untuk menginstall-nya, kita hanya perlu untuk menekan tombol install pada popup yang dari tadi muncul.

Akan tetapi, pastikan terlebih dahulu bahwa kita telah memasang pip pada OS kita.

VSCode memiliki fitur terminal internal, maka langsung saja install pip python 3 dari vscode:



The screenshot shows the Visual Studio Code interface with a file named `halo-dunia.py` open. The code in the editor is as follows:

```
1 import time
2
3 for i in range(5):
4     print('Halo dunia!')
5     time.sleep(1)
```

Below the editor, the integrated terminal is active, showing the command `sudo apt install python3-pip` and its output:

```
ibnuzakariyya@jagongoding:~/.../vscode-untuk-python$ sudo apt install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libexpat1-dev libpython3-dev libpython3.8-dev python-pip-whl python3-dev python3-distutils
  python3-setuptools python3-wheel python3.8-dev zlib1g-dev
Suggested packages:
  python-setuptools-doc
The following NEW packages will be installed:
  libexpat1-dev libpython3-dev libpython3.8-dev python-pip-whl python3-dev python3-distutils python3-pip
  python3-setuptools python3-wheel python3.8-dev zlib1g-dev
0 upgraded, 11 newly installed, 0 to remove and 87 not upgraded.
Need to get 7.260 kB of archives.
After this operation, 28,4 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Tes linting

Kita bisa mencoba linting yang telah terpasang dengan menyengaja melakukan sebuah error.

Pada contoh di bawah, saya meng-*comment* kode program untuk mengimpor modul `time`.

Walhasil, pada baris ke-5, tool linting yang sudah terpasang langsung memberikan informasi bahwa pada baris tersebut telah terjadi error.



```
halo-dunia.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
halo-dunia.py x
home > ibnuzakariyya > workspaces > python > vscode-untuk-python > halo-dunia.py > ...
1 # import time
2
3 for i in range(5):
4     print('Halo dunia!')
5     time.sleep(1)
```

Python 3.8.5 64-bit 1 0 Ln 1, Col 3 Spaces: 4 UTF-8 LF Python

Aturan Sintaks Python

Bahasa pemrograman Python adalah bahasa yang dieksekusi oleh sebuah interpreter. Interpreter tersebut bertugas untuk memarsing sintaks python, dan kemudian mengkonversinya menjadi sebuah instruksi mesin satu persatu (maksudnya adalah satu baris-satu baris).

Terdapat beberapa aturan penulisan sintaks pada bahasa pemrograman Python. Aturan-aturan ini harus ditaati oleh programmer agar interpreter bisa memarsing dan menjalankan aplikasi dengan baik. Jika tidak, maka aplikasi akan memproduksi sebuah error.

Apa saja aturan-aturan penulisan sintaks python?

Aturan penulisan python terbagi menjadi beberapa sub bahasan; mulai dari struktur baris kode, statemen, komentar, penugasan, indentasi, dan lain sebagainya.

Secara umum, sintaks penulisan python bersifat:

- **Case Sensitive**

- **Tidak Menggunakan Titik Koma**
- **Indentasi Sebagai Pembentuk Struktur**
- **Tidak ketat terhadap tipe data**
- **Human friendly**

Kita akan bahas satu persatu.

NB: Beberapa contoh dari tutorial ini diambil dari materi-materi yang akan datang. Sehingga sangat wajar jika kita masih belum memahami beberapa istilah atau beberapa kode program yang dijadikan contoh.

Case Sensitive

Bahasa pemrograman python bersifat *case sensitive*. Ia akan membedakan antara huruf kecil dan huruf besar walaupun sebuah kata itu terlihat sama.

Contoh:

```
ibu_kota = 'Jakarta'
print(iBu_kota)
```

Jika dijalankan, kita akan mendapatkan error:

```
Exception has occurred: NameError
name 'iBu_kota' is not defined
File "case-sensitive.py", line 3, in <module>
    print(iBu_kota)
```

Kenapa?

Karena variabel yang kita definisikan adalah `ibu_kota`, dengan huruf kecil semuanya. Sedangkan variabel yang berusaha kita panggil adalah `iBu_kota` yang mana huruf `B` nya adalah huruf kapital, dan interpreter python menganggap keduanya berbeda.

Tidak Menggunakan Titik Koma

Sebelum membahas lebih lanjut tentang titik koma, ada baiknya kita mengetahui terlebih dahulu pengertian statement pada bahasa pemrograman (lebih khusus pada python).

Apa itu statement?

Statemen adalah sebuah pernyataan atau instruksi yang akan dieksekusi oleh mesin. Interpreter python akan bertugas untuk menginterpretasikan setiap statemen menjadi perintah mesin yang sesuai.

Penulisan Statement

Di dalam python, penulisan antar statemen **tidak diakhiri dengan titik koma**—berbeda dengan bahasa pemrograman lain pada umumnya di mana setiap statement akan dibedakan berdasarkan adanya karakter titik koma (;).

Lalu, bagaimana cara interpreter python membedakan antar satu statemen dengan statemen lainnya?

Dengan karakter ganti baris (`\n`).

Setiap pergantian baris, interpreter akan menganggap bahwa **sebuah statemen telah sempurna**.

Perhatikan contoh berikut:

```
a = 5
b = 8
c = a + b

print(c)
```

Pada program di atas, terdapat 4 buah statemen. Dan setiap statemen dipisahkan oleh sebuah karakter `\n` atau karakter yang menandakan bahwa suatu baris telah selesai.

Kita tetap bisa menggunakan titik koma

Pada kasus-kasus tertentu, kita tetap bisa menggunakan titik koma.

Kapan?

Ketika terdapat **lebih dari 1 statemen** dalam satu baris.

Perhatikan contoh berikut:

```
a = 5; b = 8; c = a + b;

print(c)
```

Pada contoh di atas, terdapat 5 buah statement:

- 4 di antaranya ditulis dalam satu baris sekaligus.
- dan sisanya ditulis pada baris yang berbeda.

Satu Statemen Multi Baris

Kebalikannya...

Kita bisa memecah satu statemen menjadi lebih dari satu baris.

Dalam kasus-kasus tertentu, satu statemen bisa sangat panjang. Sehingga kode programnya melebihi ukuran layar.

Contoh:

```
kondisi = 10 < 5 and 10 > 9 or 11 == 6 + 5 and 0 == 100 * 5 /  
(25 - 15)
```

Hal ini tentu saja membuat kode program menjadi lebih susah dibaca, didebug, dan dipahami.

Kita inginnya kode program terlihat lebih ringkas dan mudah dibaca.

Sedangkan jika kita buat baris baru, statemen di atas akan terputus, yang kemudian akan dianggap error oleh interpreter python.

Solusinya?

Kita bisa memecah satu statemen panjang menjadi multiple baris dengan tanda *backslash* (\).

Seperti contoh berikut:

```
kondisi = 10 < 5 \  
and 10 > 9 \  
or 11 == 6 + 5 \  
and 0 == 100 * 5 / (25 - 15)
```

Meskipun kode program di atas memiliki total 4 baris, tapi interpreter tetap menganggapnya sebagai satu statemen utuh.

Indentasi Sebagai Pembentuk Struktur

Pada bahasa pemrograman lain, umumnya indentasi adalah sesuatu yang tidak penting. Bahkan cenderung diabaikan oleh mesin.

Indentasi hanya digunakan untuk mempermudah manusia dalam membaca kode program.

Tapi tidak dengan python.

Dalam python, **indentasi adalah hal yang super-super penting**, karena ia bertugas untuk mendefinisikan struktur blok kode program.

Sehingga, **melakukan kesalahan indentasi** juga bisa berujung pada sebuah error (yang mungkin akan sulit dipecahkan bagi yang belum terbiasa). Atau dalam istilah lain: kita menggunakan indentasi untuk mengelompokkan blok kode program di dalam python.

Apa itu Indentasi?

Indentasi adalah penulisan paragraf yang agak menjorok masuk ke dalam. Biasanya jika kita membaca majalah atau koran, kita akan dapati indentasi pada kalimat awal setiap paragrafnya.

Kesalahan-Kesalahan Penulisan Indentasi pada Python

Untuk memahami betapa pentingnya indentasi pada python, mari kita perhatikan contoh-contoh berikut tentang kesalahan indentasi yang umum terjadi pada bahasa pemrograman python.

Perhatikan contoh berikut:

```
print('Selamat')
    print('Pagi')
print('Dunia')
```

Kode program di atas, secara sekilas tidak ada masalah. Tapi, ketika kita eksekusi, ternyata interpreter python memberitahukan kita bahwa terdapat error yang disebabkan oleh indentasi yang tidak pada tempatnya:

```
IndentationError: unexpected indent
```

Yang benar harusnya tiga perintah `print()` berada pada satu indentasi, karena memang ketiganya berada dalam satu blok yang sama.

```
print('Selamat')
```



```
print('Pagi')
print('Dunia')
```

Contoh kode program yang benar :

```
a = 10

if a > 5:
    print('nilai a lebih dari 5')
```

Contoh yang salah :

```
a = 10

if a > 5:
print('nilai a lebih dari 5')
```

Pesan error:

```
IndentationError: expected an indented block
```

Contoh yang salah tapi samar (**kalau di-copy paste kemungkinan akan tetap working**) :

```
a = 10

if a > 5:
    print('nilai a lebih dari 5')
print('nilai a lebih dari 5')
```

Pesan error:

```
TabError: inconsistent use of tabs and spaces in indentation
```

Kenapa yang terakhir error?

Karena ke-tidak-konsisten-an penggunaan tab. Yang pertama menggunakan **4 spasi untuk tab**, sedangkan yang kedua menggunakan **1 tab asli (bukan spasi)**.

Meski secara visual terlihat sama, tapi tidak bagi interpreter python.

Jika kita menggunakan teks editor atau IDE yang canggih, **error seperti ini tidak akan terjadi** karena IDE atau Teks Editor yang kita gunakan akan otomatis mengkonversi

semua tab menjadi seragam, entah menggunakan spasi atau menggunakan `tab` itu sendiri.

Error seperti ini biasanya akan terjadi jika:

- Kita melakukan *copy-paste* kode program dari internet
- Dan kita tidak menggunakan teks editor atau IDE yang kekinian

Tidak Ketat Terhadap Tipe Data

Sifat yang berikutnya adalah: insensitifitas terhadap tipe data.

Artinya:

Kita bisa memberi dan mengubah nilai apa pun dari tipe data apa pun ke dalam sebuah variabel.

Bagi yang baru mempelajari bahasa pemrograman, ini mungkin biasa saja. Tapi bagi *old-school* yang pernah mempelajari Java dan semisalnya, akan merasa aneh. Karena pada bahasa-bahasa tersebut, satu variabel hanya diperuntukkan untuk satu tipe data saja.

Contoh ketidak-ketat-an python terhadap tipe data:

```
# nilai awal berupa integer
a = 5
# kita ubah menjadi string dan tidak error
a = 'Nurul Huda'
```

Tanda Petik dan Tanda Petik Dua

Dalam bahasa pemrograman python, kita bisa mendefinisikan string dengan tanda petik satu `'` maupun tanda petik dua `"`.

Contoh yang benar :

```
nama = 'Nurul Huda'
asal = "Indonesia"
```

Contoh yang salah :

```
nama = 'Nurul Huda"
asal = "Indonesia'
```

Penulisan Komentar

Komentar adalah sebuah baris kode atau statemen yang diabaikan oleh interpreter python. Ia hanya ditulis dengan tujuan **agar dibaca oleh manusia**, bukan mesin.

Komentar juga sangat penting sebagai penjelasan alur dari kode program yang kita tulis. Jika tidak, kita sendiri (si penulis kode) bisa lupa dan kebingungan jika harus menjelaskan kode program lama yang pernah kita tulis pada masa lalu.

Penulisan komentar pada python **terdiri dari 2 jenis**:

- satu baris
- dan multi baris

Komentar **satu baris** ditulis dengan tanda **#**. Sedangkan komentar **lebih dari satu baris** ditulis dengan *triple doublequote* (tanda petik dua sebanyak 3x).

Contoh:

```
# variabel a merepresentasikan panjang
a = 5
b = 10 # variabel b merepresentasikan tinggi

"""
Dan variabel c merepresentasikan luas
persegi dari hasil perkalian
variabel a dan variabel b
"""
c = a * b
```

Tipe Data dan Variabel

Python adalah bahasa pemrograman yang berorientasi objek secara menyeluruh. Artinya semua variabel di dalam python adalah sebuah objek. Meskipun begitu, tentu saja terdapat jenis-jenis tipe data pada Python yang harus kita pahami.

Apa itu variabel?



Anggap saja variabel adalah sebuah keranjang, tempat di mana kita bisa memasukkan sesuatu di dalamnya, yaitu data.

Di python, kita bisa memasukkan tipe data apa saja ke dalam keranjang (yakni variabel) tanpa harus mendefinisikan tipe datanya terlebih dahulu (hal ini berbeda dengan beberapa bahasa pemrograman lain yang mengharuskan kita mendefinisikan tipe data terlebih dahulu).

Apa itu tipe data?

Tipe data –*sesuai namanya*– ia adalah **jenis** dari suatu data. Setiap data memiliki nilai, dan setiap nilai memiliki jenis. Ada data-data yang bertipe angka, ada pula yang bertipe huruf/karakter, ada juga yang bertipe benar/salah dan sebagainya.

Sebagai ibarat, kalau variabel adalah keranjang, maka tipe data adalah jenis barang atau jenis benda yang akan kita masukkan ke dalam keranjang tersebut.

Gambar di bawah ini saya kira bisa memberikan ilustrasi dasar bagaimana hubungan sebuah variabel dan tipe data.



Kita bisa lihat bahwa di dalam gambar di atas, terdapat banyak kotak dan banyak buah. Setiap **kotak** tertentu digunakan untuk menyimpan **jenis** buah tertentu.

Sehingga bisa kita tarik kesimpulan bahwak:

- Kotak keranjang merepresentasikan **variabel**.
- Buah merepresentasikan **data**.
- Dan jenis-jenis buah tersebut merepresentasikan **tipe data**.

Cara Membuat Variabel

Secara singkat, membuat variabel di Python sangat mudah sekali. Kita hanya perlu menuliskan **nama variabel** lalu diikuti oleh **nilai** yang kita inginkan.

Perhatikan contoh skrip berikut ini:

```
nama = 'Nurul Huda'  
usia = 24
```

```
sudah_menikah = True

print('nama:', nama)
print('usia:', usia)
print('sudah menikah:', sudah_menikah)
```

Kode program 01

Jika kita eksekusi, program di atas akan menghasilkan output seperti berikut:

```
nama: Nurul Huda
usia: 24
sudah menikah: True
```

Penjelasan

Pada skrip di atas, kita membuat 3 buah variabel:

- nama
- usia
- sudah_menikah

Masing-masing variabel kita berikan sebuah nilai.

- Variabel `nama` memiliki nilai `"Nurul Huda"`
- Variabel `usia` memiliki nilai `24`
- Variabel `sudah_menikah` memiliki nilai `True`

Lalu di baris selanjutnya, kita menampilkan isi dari masing-masing variabel menggunakan perintah `print()`.

Aturan Penamaan Variabel

Secara umum, kita bisa membuat nama variabel apa saja yang kita mau di dalam python. Akan tetapi, **terdapat beberapa aturan dan pengecualian**.

Berikut ini aturan-aturannya secara sederhana:

1. Nama variabel hanya boleh diawali oleh huruf atau underscore.
2. Nama variabel tidak boleh diawali oleh angka.
3. Nama variabel hanya bisa terdiri dari karakter *alpha-numeric* dan underscore (A-z, 0-9, and `_`)

4. Nama variabel bersifat case sensitive. Artinya variabel `nama` berbeda dengan `Nama` atau `naMA`

Sebagai contoh, berikut ini adalah variabel-variabel yang benar xdan variabel-variabel yang salah:

1. `_nama` ✓
2. `1nama` ✗
3. `nama depan` ✗
4. `namaDepan` ✓
5. `nama_belakang` ✓
6. `nama%lengkap` ✗

Aturan Assignment

Aturan *assignment* atau aturan pemberian nilai terhadap variabel di dalam Python bisa selesai secara *multiple* mau pun secara *single*.

Cara *single* adalah dengan memberikan satu nilai terhadap satu variabel dalam satu baris, sedangkan cara *multiple* adalah dengan memberikan *multiple* nilai terhadap *multiple* variables dalam satu baris.

Perhatikan contoh berikut:

```
a, b, c = 1, 2, "Mantap"

print('a:', a)
print('b:', b)
print('c:', c)

# kita juga bisa memberikan satu nilai yang sama untuk
# beberapa variabel

d = e = f = 10

print('d:', d)
print('e:', e)
print('f:', f)
```

Jika dijalankan, program di atas akan menghasilkan output:

```
a: 1
b: 2
c: Mantap
```

```
d: 10
e: 10
f: 10
```

Memeriksa Tipe Data Pada Python

Sebelum kita memasuki contoh-contoh tipe data dasar pada python, kita akan mempelajari cara untuk memeriksa atau mengetahui tipe data dari suatu variabel.

Untuk melakukannya, kita bisa menggunakan fungsi `type()` bawaan python.

Perhatikan kode program berikut:

```
a = 'Madura'
b = 50

print(type(a))
print(type(b))
```

Output:

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

Jenis-Jenis Tipe Data Python

Jika kita lihat kembali **kode program 01** di atas, maka kita akan mendapati bahwa data dari masing-masing 3 variabel memiliki tipe data yang berbeda-beda.

```
nama = 'Nurul Huda'
usia = 24
sudah_menikah = True
```

- Variabel `nama` memiliki tipe data `string` (teks)
- Variabel `usia` memiliki tipe data `number` (numerik)
- Dan variabel `sudah_menikah` memiliki tipe data `boolean` (benar/salah)

Sekarang, kita akan mencoba dan memahami lebih jauh tentang masing-masing dari tipe data di atas.

Tipe Data Numbers (Numerik)

Yang pertama adalah tipe data numerik. Tipe data numerik adalah semua jenis tipe yang bersifat angka, bisa ditambah, bisa dikurangi, bisa dikali, bisa dibagi, dan lain sebagainya.

Ada beberapa tipe data numerik pada python; seperti **integer**, **float**, dan **kompleks**.

Integer

Tipe data **integer** adalah tipe data bilangan bulat. Sehingga setiap variabel yang memiliki nilai bilangan bulat, maka ia akan dikategorikan sebagai integer.

Float

Hampir sama dengan tipe data **integer**, hanya saja tipe data **float** dipergunakan untuk variabel-variabel yang memiliki nilai pecahan / desimal.

Complex

Sedangkan tipe data numerik yang lainnya adalah tipe data **complex**, sesuai namanya, ini adalah tipe data yang kompleks. Ia merepresentasikan nilai imajiner.

Mencoba tipe data numerik

Setelah pengenalan singkat dengan 3 tipe data numerik pada python, sekarang waktunya untuk kita mencoba masing-masing dari jenis tipe data tersebut.

```
panjang = 5
lebar = 10.5
luas = panjang * lebar

print(panjang, '*', lebar, '=', luas)
print("Tipe dari variabel panjang:", type(panjang))
print("Tipe dari variabel lebar:", type(lebar))
print("Tipe dari variabel luas:", type(luas))
```

Dengan memanggil fungsi **type(nama_variabel)**, kita akan bisa mengetahui tipe data dari sebuah variabel.

Jika dijalankan, kode program di atas akan menghasilkan output seperti berikut:

```
5 * 10.5 = 52.5
Tipe dari variabel panjang: <class 'int'>
Tipe dari variabel lebar: <class 'float'>
Tipe dari variabel luas: <class 'float'>
```

Dan untuk tipe data complex, silakan coba tulis dan jalankan kode program berikut

```
a = 5j
```

```

b = 10j
c = a + b

print(a, '+', b, '=', c)
print('Tipe dari a:', type(a))
print('Tipe dari b:', type(b))
print('Tipe dari c:', type(c))

```

Output dari kode program di atas adalah:

```

5j + 10j = 15j
Tipe dari a: <class 'complex'>
Tipe dari b: <class 'complex'>
Tipe dari c: <class 'complex'>

```

Tipe Data String (Teks)

Selanjutnya yang akan kita bahas adalah tipe data string. Ia adalah tipe data yang digunakan untuk menyimpan sebuah teks.

Data yang bertipe string harus diapit oleh tanda petik, baik tanda petik satu (') mau pun tanda petik dua (").

Silakan perhatikan contoh kode program berikut:

```

nama_depan = "Wahit"
nama_belakang = 'Abdulloh'
nama_lengkap = nama_depan + ' ' + nama_belakang
usia = '12'
alamat = 'Bangkalan'
kata_mutiara = "Don't judge a book by it's cover"

print(nama_lengkap, '(' + usia + ')', ', ', 'dari', alamat, ', ',
kata_mutiara:', kata_mutiara)

print('\nTipe dari nama_lengkap:', type(nama_lengkap))
print('Tipe dari usia:', type(usia))
print('Tipe dari alamat:', type(alamat))
print('Tipe dari kata_mutiara:', type(kata_mutiara))

```

Jika kita menjalankan kode program di atas, hasil yang akan kita dapat adalah seperti berikut:

```

Wahit Abdulloh (12) , dari Bangkalan , kata mutiara: Don't
judge a book by it's cover

```

```
Tipe dari nama_lengkap: <class 'str'>
Tipe dari usia: <class 'str'>
Tipe dari alamat: <class 'str'>
Tipe dari kata_mutiara: <class 'str'>
```

Catatan

Coba perhatikan variabel `usia`, meskipun isinya adalah sebuah angka numerik, tetap saja di situ dia bertipe data `string`.

Kenapa? karena ia diapit oleh tanda petik.

Lalu, apa perbedaan antara tipe data numerik dan tipe data teks (string)?

Perbedaannya terletak pada fungsi dan cara mengoperasikannya.

Misalkan kita ingin menambahkan dua buah variabel bertipe data numerik, yang kita dapatkan adalah **hasil penjumlahannya**.

Berbeda jika kita menambahkan dua buah variabel bertipe data string (teks), yang kita dapatkan adalah **hasil penggabungan keduanya**.

Perhatikan contoh berikut:

```
# penjumlahan dua data numerik
print(5 + 5) # output 10

# penjumlahan dua data string
print('5' + '5') # output 55
```

Output dari kode program di atas:

```
10
55
```

Oleh karena itu: pemilihan tipe data yang tepat sangatlah penting agar tidak terjadi pada kesalahan operasi.

Tipe Data Boolean (Benar/Salah)

Selanjutnya adalah tipe data boolean.

Tipe data boolean adalah tipe data yang paling simpel dan mudah. Akan tetapi dia sangat penting sekali bahkan untuk membangun program/aplikasi skala besar sekalipun.

Tipe data boolean **hanya memiliki dua buah nilai**, yaitu; `True` dan `False`.

Nilai `True` untuk pernyataan bernilai benar, dan `False` untuk merepresentasikan pernyataan yang bernilai salah.

Sederhananya, kita bisa mempraktikkan kode program berikut:

```
saya_orang_indonesia = True
saya_adalah_robot = False

print('Apakah saya orang Indonesia?', saya_orang_indonesia)
print('Apakah saya adalah robot?', saya_adalah_robot)
print('Tipe dari saya_orang_indonesia',
      type(saya_orang_indonesia))
print('Tipe dari saya_adalah_robot', type(saya_adalah_robot))
```

Jika dijalankan, maka output yang akan kita dapatkan adalah:

```
Apakah saya orang Indonesia? True
Apakah saya adalah robot? False
Tipe dari saya_orang_indonesia <class 'bool'>
Tipe dari saya_adalah_robot <class 'bool'>
```

Tipe data Boolean adalah tipe data yang sangat penting. Ia bisa berfungsi untuk mengontrol laju dan alur dari program yang kita bangun.

Tipe Data Kompleks

Masih ada beberapa tipe data lagi di dalam bahasa pemrograman Python. Tipe data tersebut sebenarnya tidak terlalu canggih seperti yang dibayangkan, hanya saja saya katakan itu sebagai tipe data canggih karena cara kerjanya tidak seperti tipe data yang sudah dibahas. Materi ini akan dibahas pada materi yang akan datang.

Di antara tipe data tersebut adalah tipe data:

- List
- Tuple
- Set
- Dictionary

Tugas

1. Enroll pada LMS class.ipb.ac.id
2. Instalasi dan integrasikan Python dan Visual Studio Code
3. Upload screenshot bukti poin 2 pada LMS poin 1

Referensi

- [1] <https://www.bbc.co.uk/bitesize/guides/zgmpr82/revision/2> - diakses 14 Februari 2021
- [2] <https://www.greenteapress.com/thinkpython/thinkCSpy/html/chap02.html> - diakses 14 Februari 2021