



# DASAR PEMROGRAMAN

## PYTHON

**UNIT PENGEMBANG AKADEMIK**

## **Kata Pengantar**

Puji syukur Alhamdulillah, penulis panjatkan kehadirat Allah ta'ala, yang telah melimpahkan Rahmat dan Karunia-Nya sehingga pada akhirnya penulis dapat menyelesaikan modul ini dengan baik. Dimana modul ini penulis sajikan dalam bentuk modul yang sederhana. Adapun modul ini penulis buat untuk menambah wawasan para pembaca pada umumnya dan untuk menambah bahan materi untuk mata kuliah Dasar Pemrograman bagi mahasiswa prodi Sistem Informasi Universitas Bina Sarana Informatika.

Sebagai bahan penulisan diambil berdasarkan pencarian di beberapa sumber, seperti buku, internet dan masih banyak lagi yang lainnya. Dalam modul ini menjelaskan materi Dasar Pemrograman mulai pertemuan 1 s.d 14. Penulis menyadari bahwa tanpa bimbingan dan dorongan dari semua pihak, maka penulisan dan pembuatan modul ini tidak akan berjalan dengan lancar.

Penulis mengucapkan terima kasih kepada tim sehingga bisa menyelesaikan penyusunan modul ini. Semoga modul ini berguna bagi para pembaca baik mahasiswa ataupun siapapun yang bisa dijadikan bahan referensi untuk pembelajaran.

Karawang, September 2022

Tim Penyusun  
Unit Pengembangan Akademik  
Program Studi Sistem Informasi

## Daftar isi

Kata Pengantar .....	ii
Daftar isi.....	iii
Pertemuan 1 Struktur bahasa pemrograman Python .....	1
1.1 Instalasi Program.....	3
1.2 Pengenalan Bahasa Python.....	12
1.3 Struktur Program Python.....	15
1.3.1 Aturan Penulisan.....	15
1.3.2 Indentasi.....	15
1.3.3 Variabel.....	16
1.3.4 Nama Variabel .....	16
1.3.5 Keyword / Kata Kunci pada Python .....	17
1.4 Membuat File Editor .....	18
1.4.1 Menyimpan File Editor .....	20
1.4.2 Menjalankan Program.....	21
1.4.3 Keluar dari VS Code .....	21
1.5 Program Pertama dengan Python .....	21
Pertemuan 2 Sintaks Dasar Python, Tipe data & Variable.....	23
2.1 Baris dan Indentasi.....	23
2.1.1 Tanda Kutip di Python .....	24
2.1.2 Komentar di Python .....	24
2.2 Variabel dan Tipe Data Python .....	25
2.3 Input dan Output pada Python.....	26
2.3.1 Input pada Python .....	27
2.3.2 Output pada Python.....	27
Pertemuan 3 String, Bilangan & Operator .....	29
3.1 Mengakses Nilai String .....	29
3.2 Mengupdate String .....	29
3.3 Menggabung String .....	30
3.4 Mengetahui Panjang String .....	31
3.4.1 Karakter Escape .....	31
3.4.2 Raw String untuk Mengabaikan Karakter Escape .....	33

3.5 Mengatur Format String .....	33
3.5.1 Metode format() .....	33
3.5.2 Metode / Fungsi Bawaan String.....	35
3.6 Bilangan (Number).....	36
3.6.1 Konversi Jenis Bilangan .....	36
3.6.2 Python Decimal.....	36
3.6.3 Bilangan Pecahan.....	38
3.7 Matematika dengan Python.....	38
3.7.1 Operator Aritmatika .....	40
3.7.2 Operator Perbandingan .....	41
3.7.3 Operator Penugasan .....	42
3.7.4 Operator Logika .....	43
3.7.5 Operator Bitwise .....	44
3.7.6 Operator Identitas.....	44
3.7.7 Operator Keanggotaan .....	45
Pertemuan 4 Percabangan .....	47
4.1 Pernyataan if.....	47
4.2 Pernyataan if...elif...else.....	50
4.3 Studi Kasus : Penjualan Tiket .....	52
Pertemuan 5 Perulangan.....	57
5.1 Perulangan dengan Menggunakan For.....	57
5.1.1 Fungsi range.....	58
5.1.2 Perulangan Menggunakan While .....	59
5.1.3 Infinite Loop .....	61
5.1.4 Kendali Looping .....	61
5.2 While else .....	63
Pertemuan 6 List & Tuple .....	66
6.1 List.....	66
6.1.1 Mengakses anggota List.....	66
6.1.2 List dengan Indeks Negatif .....	67
6.1.3 Memotong (Slicing) List.....	67
6.1.4 Mengubah Anggota List .....	68
6.2 Tuple.....	71

6.2.1	Membuat Tuple.....	71
6.2.2	Mengakses anggota Tuple.....	72
6.2.3	Mengubah Anggota Tuple .....	73
6.2.4	Metode dan Fungsi Bawaan Tuple .....	75
Pertemuan 7	Matrix dan Library Pandas.....	79
7.1	Matrix .....	80
7.1.1	Membuat matrix di python.....	80
7.1.2	Melakukan Penjumlahan Matriks .....	81
7.1.3	Melakukan Pengurangan pada Matriks.....	81
7.1.4	Melakukan Perkalian Matriks .....	82
7.2	Pandas Pada Python .....	83
7.2.1	Instalasi Pandas .....	83
7.2.2	Contoh program dengan pengunaan modul Pandas.....	92
Pertemuan 8	UTS .....	94
Pertemuan 9	Fungsi.....	95
9.1	Mendefinisikan Fungsi.....	95
9.2	Memanggil Fungsi.....	96
Pertemuan 10	Modul dan Eksepsi .....	102
10.1	Menggunakan Perintah Import .....	102
10.2	Menggunakan Perintah Alias pada Modul .....	103
10.3	Mengimpor sebagian fungsi modul pada Python .....	104
10.4	Mengimpor semua fungsi modul pada Python .....	105
10.5	Eksepsi.....	106
Pertemuan 11	Object-Oriented Programming (OOP).....	112
11.1	Konsep OOP .....	112
11.2	Class.....	113
11.3	Function/Method .....	113
11.4	Constructor .....	114
11.5	Object.....	115
11.6	Inheritance .....	115
11.7	Overriding.....	116
11.8	Private Attribute/Function .....	116
11.9	Polymorphism.....	116

# **Pertemuan 1**

## **Struktur bahasa pemrograman Python**

Capaian Pembelajaran Matakuliah

Dasar Pemrograman :

1. Mahasiswa mampu Menganalisis Tools
2. Mahasiswa mampu membuat Dokumen Kode Program
3. Mahasiswa mampu melakukan Debuging
4. Mahasiswa Menulis Kode Dengan Prinsip Sesuai Guidelines dan Best Practice
5. Mahasiswa Menerapkan Pemecahan Permasalahan Menjadi Subrutin

Mahasiswa mampu Menganalisis *Tools*

Unit ini menentukan kompetensi, pengetahuan dan sikap kerja yang diperlukan untuk menganalisis tools yang diperlukan untuk mengembangkan perangkat lunak aplikasi sesuai dengan kebutuhan.

Mahasiswa mampu membuat dokumen kode program

Kompetensi ini berhubungan dengan sikap, pengetahuan, dan keterampilan yang Diperlukan untuk membuat dokumentasi dari kode program yang telah ditulis secara hardcopy termasuk identifikasi penjelasan dari dokumen tersebut.

Mahasiswa mampu melakukan Debugging

Unit kompetensi ini berhubungan dengan sikap, pengetahuan, dan keterampilan yang dibutuhkan dalam memeriksa kode program dari kesalahan (bug).

Mahasiswa menulis kode dengan prinsip sesuai Guidelines dan Best Practice

Menentukan kompetensi, pengetahuan dan Sikap kerja yang diperlukan dalam menerapkan Prinsip penulisan kode yang baik agar kode tersebut dapat dirawat (maintainability).

Mahasiswa menerapkan pemecahan Permasalahan menjadi Subrutin

Kompetensi ini berhubungan dengan sikap, pengetahuan, dan keterampilan yang dibutuhkan dalam memecah permasalahan menjadi permasalahan –permasalahan yang lebih kecil dan

menyelesaikan permasalahan lebih kecil tersebut berupa fungsi, prosedur, library, atau representasi yang lain sesuai paradigma bahasa pemrograman yang digunakan.

### Uji Kompetensi

1. Individu
2. Final Project (Berkelompok)

#### Uji Kompetensi Individu

1. Untuk Matakuliah Dasar Pemrograman Tidak Ada UTS Dan UAS, digantikan Uji kompetensi (Individu dan Final Project)
2. Uji Kompetensi Individu dilaksanakan pada pertemuan 12. Masing-masing mahasiswa diminta mengerjakan soal yang sudah ditentukan. Wajib membawa laptop.

#### Uji Kompetensi (Final Projek)

1. Final Project dilakukan di pertemuan 13-15 dengan ketentuan sebagai berikut:
  - a. Jumlah anggota setiap kelompok adalah 5 anggota (optional) tergantung jumlah mahasiswa pada kelas tersebut.
  - b. Isi dari final project :
    - Nilai Running Program diambil berdasarkan: (Logika Program, Debuging, penulisan Kode Program, Tampilan output Program)
    - c. Masing-masing kelompok membuat paper laporan pembuatan final project
    - d. Program, Paper dan Presentasi di Burning Kedalam CD/ Kirim Link Google Drive kepada Dosen Pengampu Matakuliah
    - e. Template Paper Project Silahkan Check pada Link Berikut:  
<https://drive.google.com/drive/folders/1xQxKn7TltO4KLZdzDuTofnhO28sAaHAq?usp=sharing>
    - f. Masing-masing kelompok mempresentasikan hasil final projectnya.
    - g. Presentasi disajikan dengan media presentasi yang isinya berupa alur logika program dan eksekusi running program.
    - h. Penilaian ditentukan oleh dosen pengajar diruang kelas.
2. Tema Projek di serahkan ke dosen pengajar di Pertemuan ke 2
3. Projek sudah bisa di kerjakan setelah di lakukan penyerahan tema kepada dosen pengajar
4. Penilaian dilakukan oleh dosen pengajar ketika presentasi

#### Tema Projek UAS :

1. Berbasis Bisnis (Optional) Contoh :
  - a. Penjualan dan Pembelian

- b. Pengadaan barang
- 2. Berbasis Science
  - a. Science (Bidang Matematika, Fisika, Kimia atau IPA)
  - b. Animasi Edukasi
  - c. Berbasis Kesehatan (Diagnosa Penyakit)
- 3. Kreatifitas tampilan
- 4. Tema Harus Menarik
- 5. Penilaian ditentukan oleh Dosen Pengajar.

## 1.1 Instalasi Program

Link Download Program

- Link Download Phyton <http://www.python.org/download/>
- Link Download Ms Visual Studio Code <https://code.visualstudio.com/download>

Cara install Phyton 3.8.2 dan Microsoft Visual Studio Code

1. Siapkan Laptopnya, Bisa menggunakan windows 7, 8 atau windows 10, untuk proses instalasi nya tidak berbeda.
2. Master Phyton / Aplikasi mentahan bisa di download melalui halaman : <https://www.python.org/downloads/>
3. Master Phyton / Aplikasi mentahan bisa di download melalui halaman : <https://code.visualstudio.com/download>
4. Jika sudah klik linknya, berikut gambar untuk cara mengunduh aplikasi tersebut.



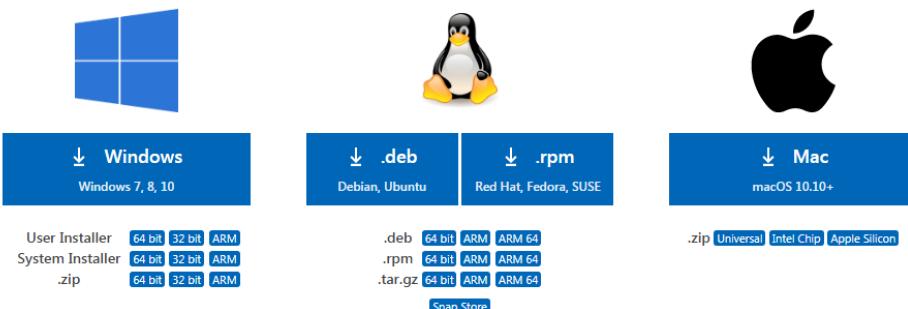
 Visual Studio Code Docs Updates Blog API Extensions FAQ Learn

Version 1.57 is now available! Read about the new features and fixes from May.

Search Docs Download

## Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



5. Selanjutnya klik tombol download seperti pada gambar berikut :

Screenshot of the Python.org download page for Windows.

The page features a navigation bar with tabs: Python, PSF, Docs, PyPI, Jobs, and Community.

The main content area has a blue header with the Python logo and a search bar.

**Download the latest version for Windows**

A button labeled "Download Python 3.8.2" is circled in red.

Text below the button includes:

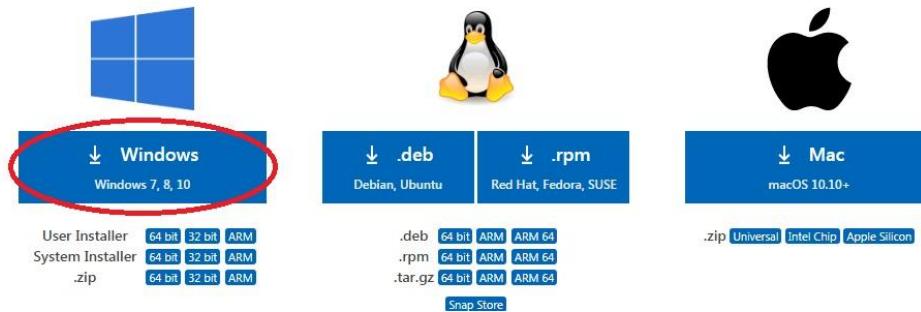
- Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)
- Want to help test development versions of Python? [Prereleases](#), [Docker images](#)
- Looking for Python 2.7? See below for specific releases

To the right of the text is a cartoon illustration of two boxes hanging from parachutes.

The screenshot shows the official Visual Studio Code download page. At the top, there's a navigation bar with links to Docs, Updates, Blog, API, Extensions, FAQ, and Learn. To the right of the navigation bar are search and download buttons. A banner at the bottom of the header area informs users about Version 1.57. Below the header, the main title "Download Visual Studio Code" is centered, followed by the subtitle "Free and built on open source. Integrated Git, debugging and extensions." On the left side, there's a large Windows logo icon. In the center, there are download links for different operating systems: Windows (Windows 7, 8, 10), Linux (.deb for Debian, Ubuntu; .rpm for Red Hat, Fedora, SUSE), and Mac (macOS 10.10+). Each link has a small icon above it. The Windows link is circled in red.

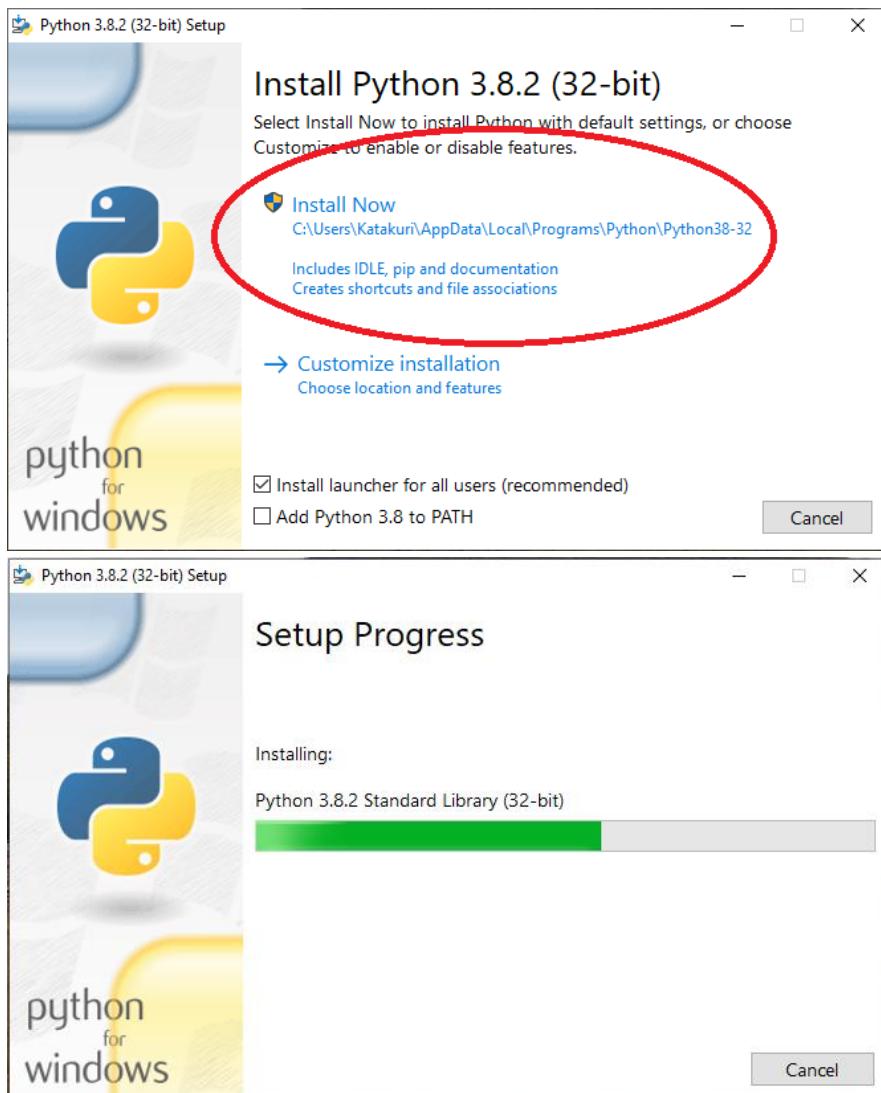
## Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

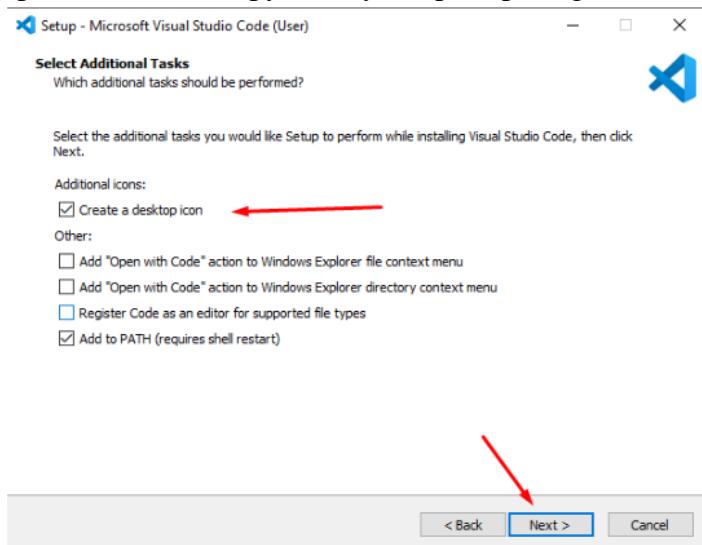


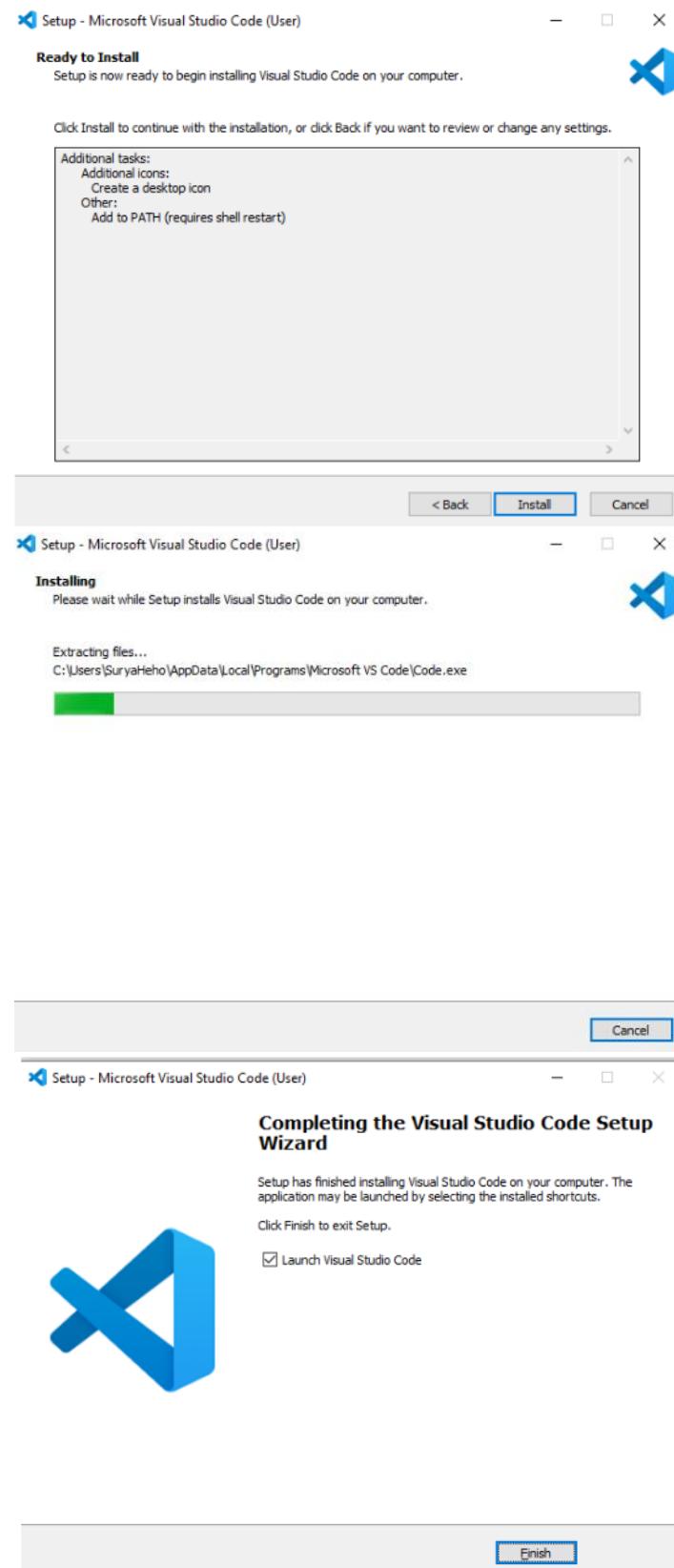
6. Setelah selesai mengunduh kedua aplikasi tersebut, kita double click aplikasi Phyton yang sudah kita unduh terlebih dahulu untuk proses instalasi, seperti pada gambar berikut :





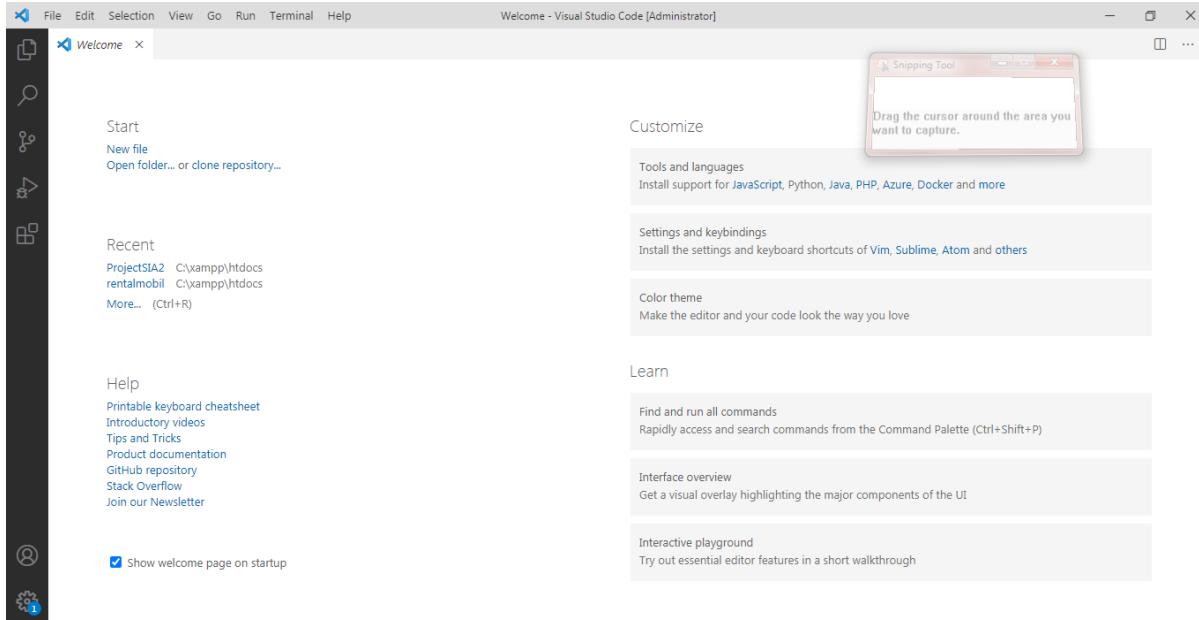
7. Setelah selesai proses instalasi Python, berikutnya kita install Microsoft Visual Studio Code untuk aplikasi text editor python nya, seperti pada gambar berikut:



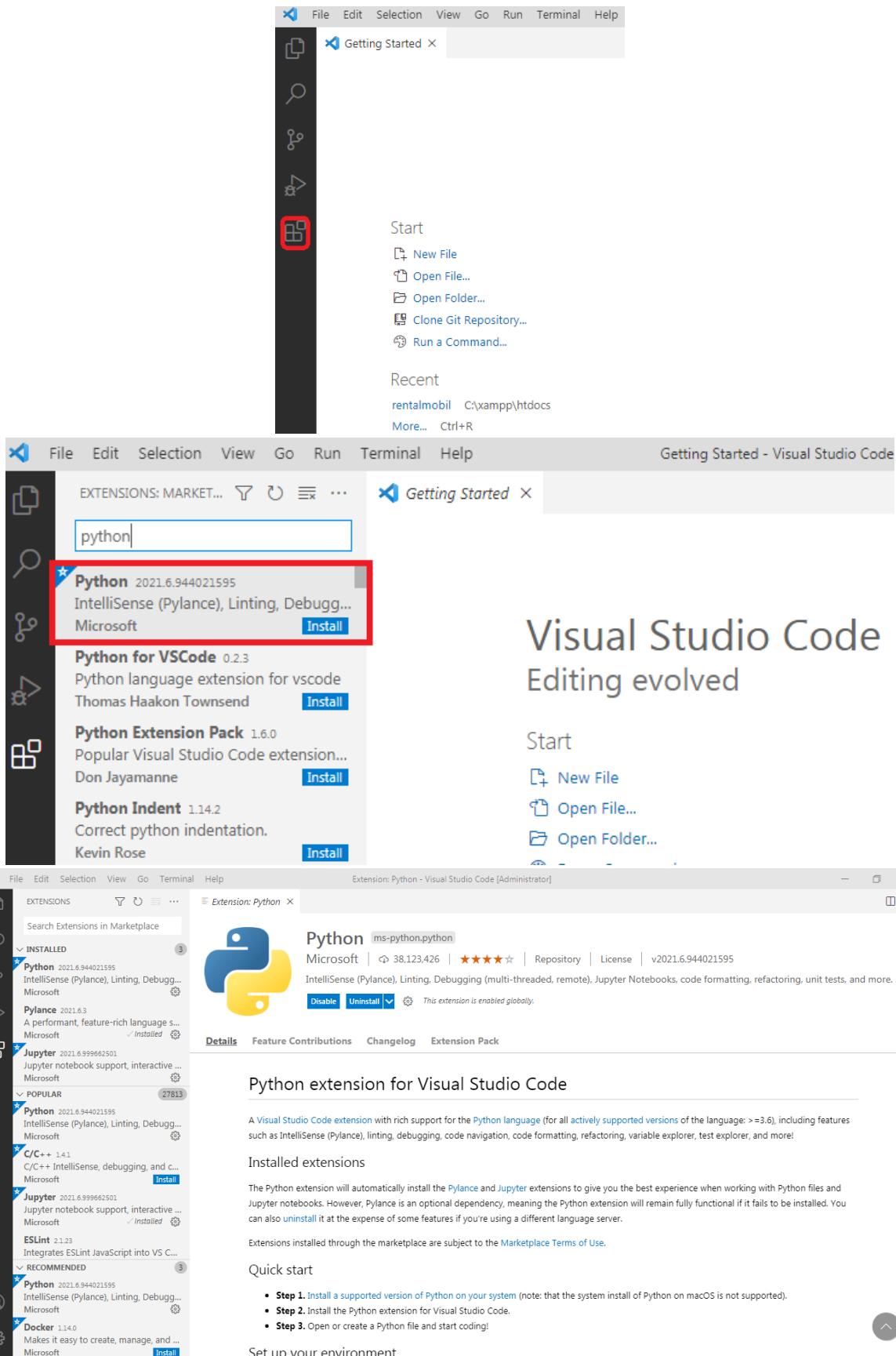


8. Setelah instalasi berhasil, jika ingin mengoperasikan Visual Studio Code, kita pergi ke folder dimana Visual Studio Code terinstall atau bisa dengan mengakses icon vs code pada Desktop.

bin	23/04/2021 19:43	File folder
locales	23/04/2021 19:43	File folder
resources	23/04/2021 19:43	File folder
swiftshader	23/04/2021 19:44	File folder
tools	23/04/2021 19:44	File folder
chrome_100_percent.pak	13/04/2021 9:37	PAK File 122 KB
chrome_200_percent.pak	13/04/2021 9:37	PAK File 182 KB
<b>Code</b>	<b>13/04/2021 9:54</b>	<b>Application 101.061 KB</b>
Code.VisualElementsManifest	13/04/2021 9:37	XML Document 1 KB
d3dcompiler_47	13/04/2021 9:53	Adobe Acrobat D... 3.637 KB
ffmpeg	13/04/2021 9:53	Adobe Acrobat D... 2.303 KB
icudtl.dat	13/04/2021 9:37	NCH.VideoPad.dat 10.282 KB
libEGL	13/04/2021 9:53	Adobe Acrobat D... 368 KB
libGLESv2	13/04/2021 9:53	Adobe Acrobat D... 6.486 KB
resources.pak	13/04/2021 9:37	PAK File 4.715 KB
snapshot_blob.bin	13/04/2021 9:37	BIN File 51 KB
unins000.dat	23/04/2021 19:44	NCH.VideoPad.dat 2.089 KB
unins000	23/04/2021 19:43	Application 2.567 KB
unins000	23/04/2021 19:44	Outlook Item 23 KB
v8_context_snapshot.bin	13/04/2021 9:37	BIN File 168 KB
vk_swiftshader	13/04/2021 9:53	Adobe Acrobat D... 3.810 KB
vk_swiftshader_icd.json	13/04/2021 9:37	JSON File 1 KB
vulkan-1	13/04/2021 9:53	Adobe Acrobat D... 634 KB



9. Sebelum aplikasi terbuka kita juga harus menginstall Ekstensi Python pada Visual Studio Code agar bahasa pemrograman python dapat dikenali. Tuliskan “Python” pada kolom pencarian, untuk mencari ekstensi yang akan di install pada VS Code seperti pada gambar berikut :



## Visual Studio Code Editing evolved

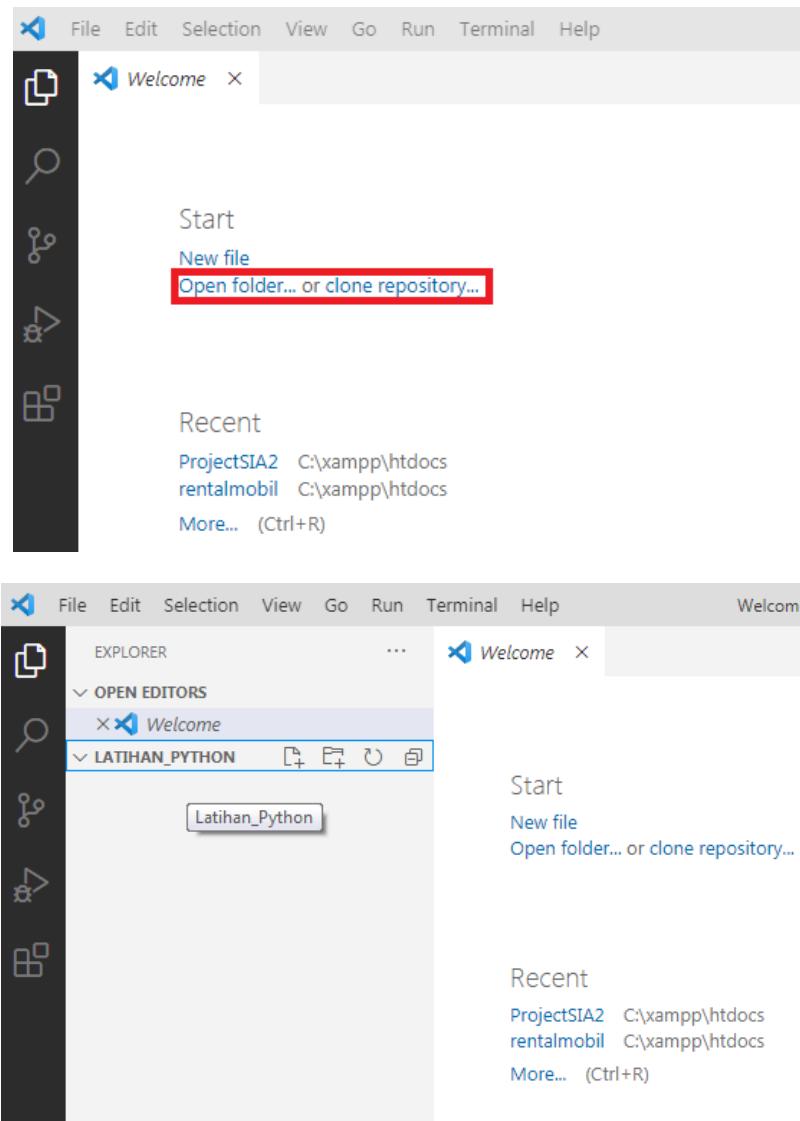
### Start

- New File
- Open File...
- Open Folder...

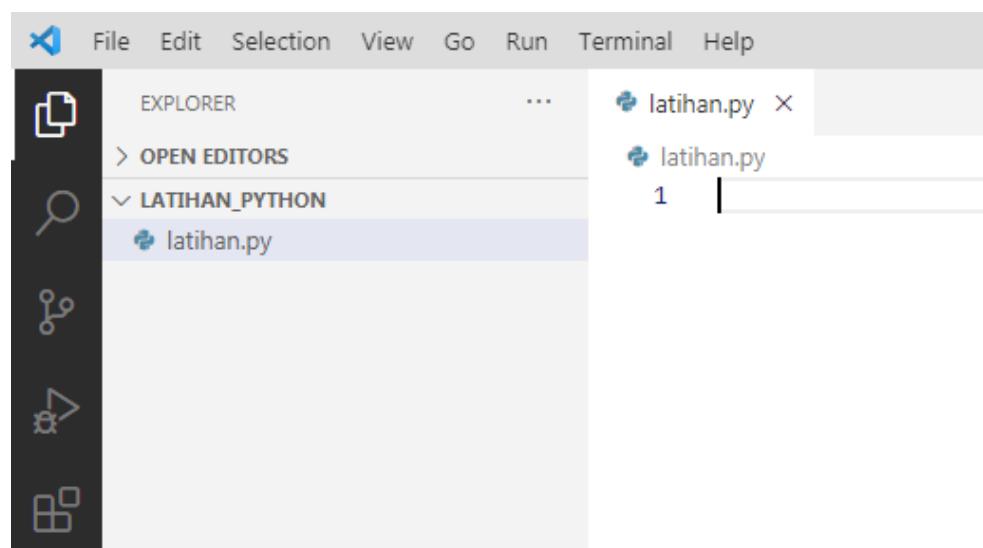
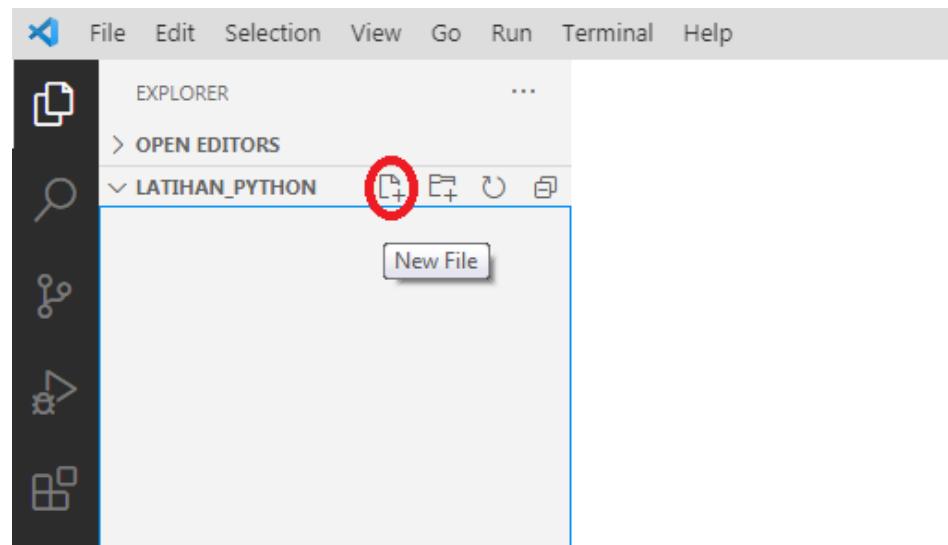
10. Ketika proses instalasi extension python selesai dan Untuk memudahkan pembuatan file baru dan manajemen file pemrograman python, buat sebuah folder terlebih dahulu

dengan nama “Latihan\_Python”, untuk penempatan folder silahkan disesuaikan masing-masing.

11. Kemudian pilih “Open Folder” dan pilih folder “Latihan\_Python” untuk memulai menggunakan aplikasi VS Code, seperti gambar berikut :



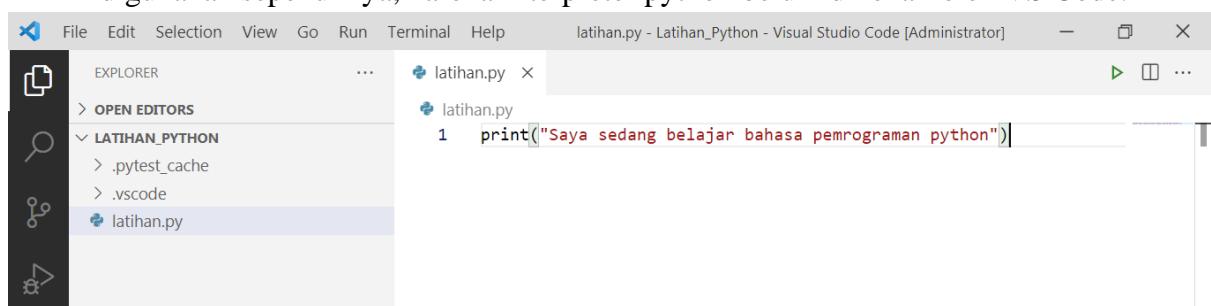
12. Setelah folder “Latihan\_Python” terbuka dalam aplikasi VS Code selanjutnya kita buat file Python nya dengan mengklik icon buat file baru dengan format .py. contohnya “latihan.py” seperti digambar berikut :



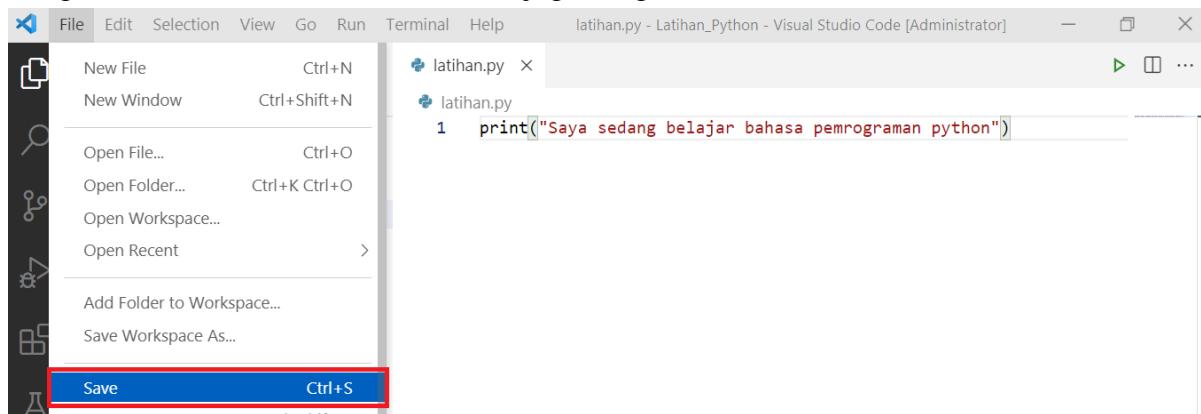
13. Setelah file dibuat, maka file python akan otomatis ada dalam folder Latihan\_Python, seperti gambar berikut :



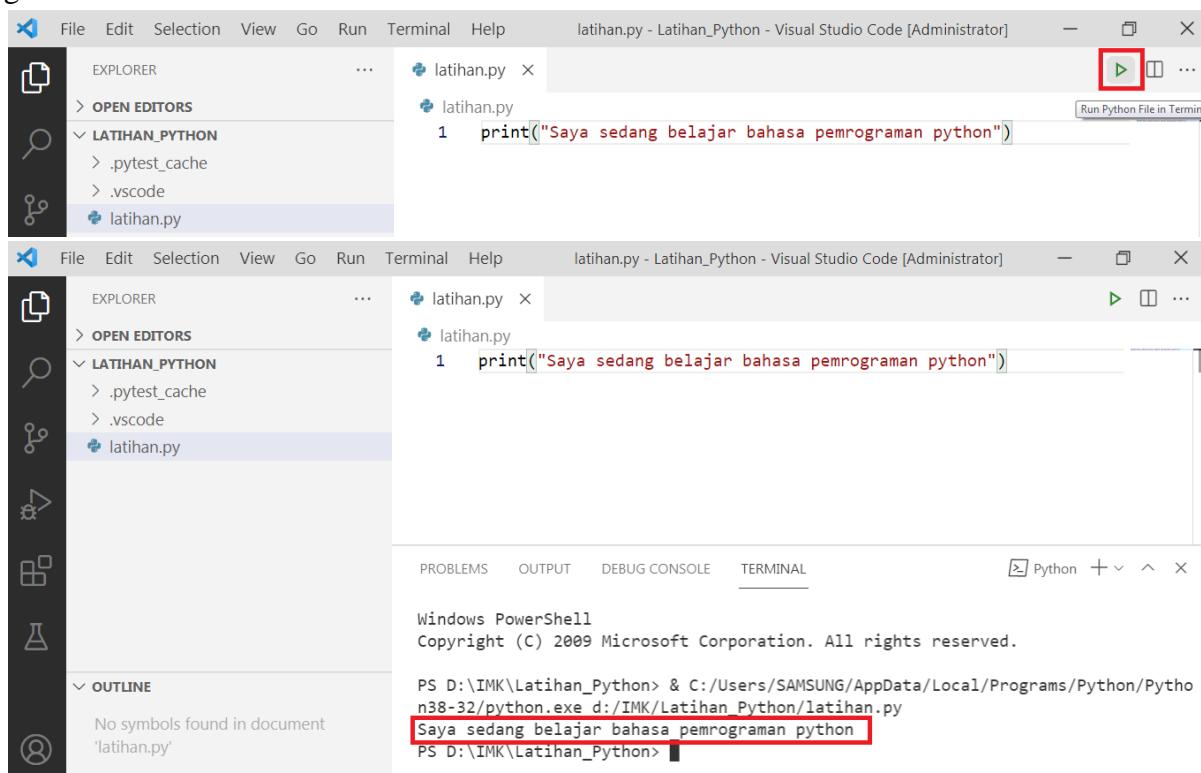
14. Buatlah sebuah baris kode pemrograman python dibawah ini agar VS Code dapat digunakan sepenuhnya, karena Interpreter python belum dikenali oleh VS Code.



Biasakan setiap selesai mengetikan sebuah baris kode, simpan file terlebih dahulu, dengan mengakses menu File → Save, atau bisa juga dengan menekan tombol Ctrl + s.



Untuk mengeksekusi sebuah kode program kita bisa dengan mengklik tombol run seperti pada gambar berikut :



## 1.2 Pengenalan Bahasa Python

### 1.1. Sejarah singkat perkembangan bahasa pemrograman Python

#### 1.1.1. Sekilas Perkembangan Bahasa Python

Python diciptakan oleh Guido van Rossum pertama kali di Centrum Wiskunde & Informatica (CWI) di Belanda pada awal tahun 1990-an. Bahasa python terinspirasi dari bahasa pemrograman ABC. Sampai sekarang, Guido masih menjadi penulis utama untuk python, meskipun bersifat open source sehingga ribuan orang juga berkontribusi dalam mengembangkannya.

Di tahun 1995, Guido melanjutkan pembuatan python di Corporation for National Research Initiative (CNRI) di Virginia Amerika, di mana dia merilis beberapa versi dari python.

Pada Mei 2000, Guido dan tim Python pindah ke BeOpen.com dan membentuk tim BeOpen PythonLabs. Di bulan Oktober pada tahun yang sama, tim python pindah ke Digital Creation (sekarang menjadi Perusahaan Zope). Pada tahun 2001, dibentuklah Organisasi Python yaitu Python Software Foundation (PSF). PSF merupakan organisasi nirlaba yang dibuat khusus untuk semua hal yang berkaitan dengan hak intelektual Python. Perusahaan Zope menjadi anggota sponsor dari PSF.

Semua versi python yang dirilis bersifat open source. Dalam sejarahnya, hampir semua rilis python menggunakan lisensi GFL-compatible. Berikut adalah versi mayor dan minor python berikut tanggal rilisnya.

- Python 1.0 – Januari 1994
- Python 1.2 – 10 April 1995
- Python 1.3 – 12 Oktober 1995
- Python 1.4 – 25 Oktober 1996
- Python 1.5 – 31 Desember 1997
- Python 1.6 – 5 September 2000
- Python 2.0 – 16 Oktober 2000
- Python 2.1 – 17 April 2001
- Python 2.2 – 21 Desember 2001
- Python 2.3 – 29 Juli 2003
- Python 2.4 – 30 Nopember 2004
- Python 2.5 – 19 September 2006
- Python 2.6 – 1 Oktober 2008
- Python 2.7 – 3 Juli 2010
- Python 3.0 – 3 Desember 2008
- Python 3.1 – 27 Juni 2009
- Python 3.2 – 20 Februari 2011
- Python 3.3 – 29 September 2012
- Python 3.4 – 16 Maret 2014
- Python 3.5 – 13 September 2015
- Python 3.6 – 23 Desember 2016
- Python 3.7 – 27 Juni 2018

Nama python sendiri tidak berasal dari nama ular yang kita kenal. Guido adalah penggemar grup komedi Inggris bernama Monty Python. Ia kemudian menamai bahasa ciptaannya dengan nama Python.

### 1.1.2. Mengapa Harus Python

Mengapa harus Python? Bukankah masih banyak bahasa pemrograman lain di luar sana? Apa kelebihan Python?

Pertanyaan – pertanyaan tersebut sering menjadi pertanyaan yang muncul sebelum seseorang mempelajari Python. Berikut adalah beberapa di antara kelebihan Python:

1. Python adalah bahasa pemrograman yang populer. Per September 2018, Python berada di urutan ke 3 bahasa program yang paling populer di dunia.
2. Python relatif lebih mudah dipelajari dan digunakan dibandingkan bahasa pemrograman lain. Sintaksnya sederhana, mudah dibaca dan diingat karena filosofi python sendiri menekankan pada aspek kemudahan dibaca (*readibility*). Kode python mudah ditulis dan mudah dibaca, sehingga lebih mudah diperbaiki kalau ada kesalahan, dan juga mudah untuk dipelihara.
3. Selain lebih mudah dibaca, python juga lebih efisien dibandingkan bahasa lain seperti C, C++, maupun Java. Untuk melakukan sesuatu dengan 5 baris kode pada bahasa lain, bisa jadi di python hanya diperlukan 1 baris kode. Hal ini menyebabkan pembuatan program dalam Python menjadi lebih ringkas dan lebih cepat dibandingkan bahasa lain.
4. Python merupakan bahasa multifungsi. Dengan python Anda bisa melakukan berbagai hal mulai dari memproses teks, membuat website, membuat program jaringan, robotika, data mining, sampai dengan kecerdasan buatan. Dengan python Anda bisa membuat aplikasi berbasis desktop maupun berbasis smartphone.
5. Python kaya akan dukungan library (pustaka) standar. Tersedia banyak sekali modul-modul dan ekstensi program yang sudah siap Anda pakai untuk membuat program sesuai kebutuhan Anda. Komunitas python adalah komunitas yang sangat aktif mengembangkan python sehingga menjadi bahasa yang sangat handal.
6. Python bisa berinteraksi dengan bahasa lain. Kode python bisa memanggil oleh bahasa C, C++, dan sebaliknya juga bisa dipanggil dari bahasa lain.

Tapi, itu hanya kelebihannya. Terus, apa kekurangannya? Python adalah bahasa interpreter. Kekurangan python dibanding bahasa lain yang menggunakan kompiler adalah ‘sedikit’ lebih lambat pada saat dijalankan bila dibandingkan bahasa C maupun C++. Tapi hal inipun sangat bersifat relatif. Tergantung dari besar ukuran program yang dibuat.

Untuk program besar yang membutuhkan kecepatan pemrosesan tinggi mungkin Python kalah cepat dari bahasa C, tapi untuk hal selain itu Python lebih mudah dan lebih baik dari bahasa lain. Selain itu, kode sumber sekarang sudah dioptimasi menggunakan bahasa C, sehingga kecepatannya juga sudah sangat mendekati kecepatan bahasa C. Spesifikasi komputer juga sekarang ini sudah semakin tinggi sehingga bisa memproses program dengan cepat, sehingga sering kali ini tidak menjadi hal penting dan bisa diabaikan.

#### 1.1.3. Sekilas Tentang Bahasa Python

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai pemrograman yang menggabungkan kapabilitas bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif.

Python mendukung multi paradigma pemrograman, utamanya namun tidak dibatasi pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang tersedia pada python adalah sebagai

bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada pemrograman dinamis lainnya. Python umumnya digunakan sebagai bahasa skrip meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan untuk berbagai pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi.

### 1.3 Struktur Program Python

#### 1.3.1 Aturan Penulisan

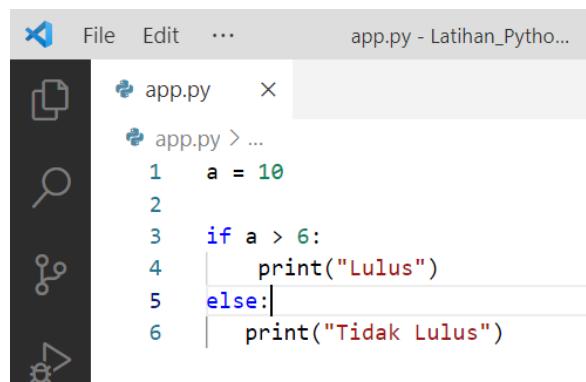
Program-program yang ditulis dalam Python secara khas jauh lebih pendek dibandingkan dengan program-program C atau C++, karena beberapa pertimbangan:

- Tipe data tingkat tinggi digunakan untuk menyatakan operasi kompleks dalam suatu statemen tunggal.
- Pengelompokan statemen telah selesai dengan indentasi sebagai pengganti dari pengurungan mulai dan akhiran.
- Tidak ada deklarasi-deklarasi argumentasi atau variabel yang diperlukan.

#### 1.3.2 Indentasi

Bahasa pemograman Python adalah bahasa pemograman yang mudah dibaca dan terstruktur, hal ini karena digunakannya sistem indentasi. Yaitu memisahkan blok - blok program dengan susunan indentasi. Jadi untuk memasukan sub - sub program dalam suatu blok, sub - sub program tersebut diletakkan satu atau lebih spasi dari kolom suatu blok program.

Python memiliki sedikit perbedaan pada cara penulisan program dengan bahasa pemrograman yang lain seperti C/Java. Kalau pada C/Java menggunakan tanda kurung sebagai pemisah blok program, di Python kita hanya menggunakan spasi sebagai pemisah blok program yang biasa disebut sebagai Indentasi. Karena Python menjalankan perintah secara berurutan, maka kita harus pintar menyusun perintah agar mendapatkan hasil seperti yang diinginkan.



```
File Edit ... app.py - Latihan_Pytho...
app.py < ...
1 a = 10
2
3 if a > 6:
4     print("Lulus")
5 else:
6     print("Tidak Lulus")
```

Contoh Penggunaan Indentasi yang kurang tepat

Pada contoh diatas kita dapat melihat jika suatu kondisi nilai  $a > 6$  dipenuhi maka program akan menjalankan baris perintah yang ada di dalam suatu blok kondisi tersebut, yang ditandai dengan penggunaan satu spasi atau lebih dari blok kondisi

sebelumnya, dalam contoh diatas perintah yang akan dilaksanakan jika suatu kondisi diatasnya terpenuhi menggunakan dua (2) spasi, sedangkan pada pernyataan else, menggunakan satu spasi. Perbedaan penggunaan spasi ini tidak dianjurkan meskipun dalam program Python dibenarkan, karena struktur program akan lebih sulit dibaca. Seharusnya blok-blok program diatas adalah sebagai berikut :

```
File Edit ... app.py - Latihan_Pytho...
app.py < ...
1 a = 10
2
3 if a > 6:
4     print("Lulus")
5 else:
6     print("Tidak Lulus")
```

Penggunaan Indentasi yang tepat

### 1.3.3 Variabel

Sebuah variabel adalah sebuah nama yang mempunyai sebuah nilai. Pendeklarasian kalimat membuat sebuah variabel - variabel baru dan memberinya nilai.

```
File Edit ... app.py - Latihan_Pytho...
app.py < ...
1 a = "Belajar Python"
2 b = 851
3 phi = 3.14
```

Pada contoh di atas, pendeklarasian tersebut menciptakan 3 variabel baru. Pendeklarasian pertama, menunjukkan string "Belajar Python" ke sebuah variabel yang bernama **a**. Kedua, variabel **b** diberi nilai 851 sebagai integer. Dan yang terakhir variabel phi diberi nilai 3.14 sebagai nilai pecahan/float.

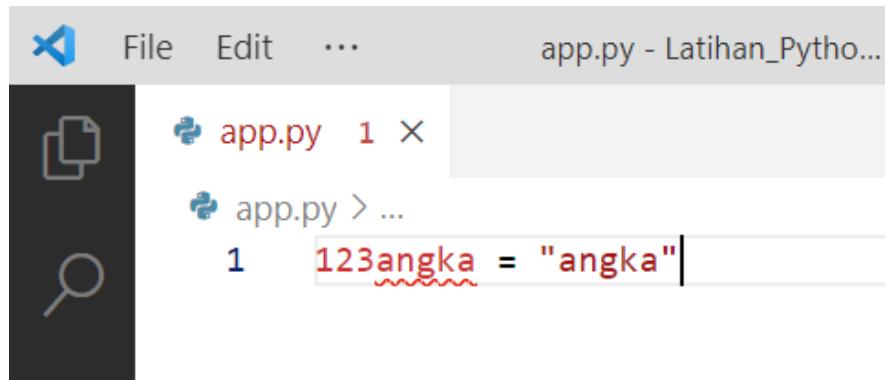
### 1.3.4 Nama Variabel

Pada umumnya, programmer memakai nama variabel sesuai dengan keterangan isi dari variabel tersebut dan variabel juga merupakan simbol yang mewakili nilai tertentu. Pembuatan variabel dalam python sangat sederhana. Berikut adalah ketentuan mengenai variabel dalam pyton,

- Variabel tidak perlu dideklarasikan mempunyai tipe data tertentu
- Jenis data dalam variabel dapat berubah-ubah
- Penulisan variabel harus diawali dengan huruf, dan untuk karakter selanjutnya bisa berupa huruf atau angka

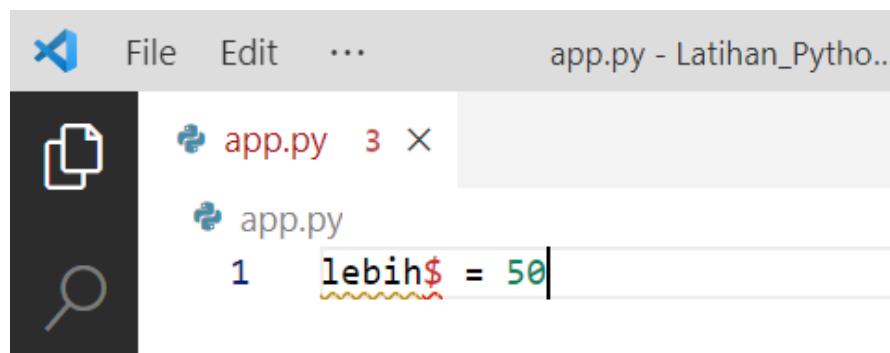
- Dapat berupa huruf Kapital, tetapi bersifat case-sensitive, nama Kapital dengan kapital adalah variabel yang berlainan.
- Penulisan variabel tidak boleh dipisah oleh <spasi>
- Untuk variabel yang terdiri dari 2 suku kata, dapat dipisah dengan simbol underscore ( \_ ) seperti nama\_saya.

Statemen yang tidak boleh dijadikan nama variabel adalah keywords pada Python. Contoh :



The screenshot shows a code editor window titled "app.py - Latihan\_Pytho...". On the left is a sidebar with icons for file operations. The main area shows two files: "app.py" (version 1) and "app.py" (version 2). The second file contains the line "1 123angka = "angka"" with the number "1" and the variable name "123angka" underlined in red, indicating a syntax error.

Variabel 123satu adalah penamaan variabel tidak benar karena diawali dengan sebuah angka.



The screenshot shows a code editor window titled "app.py - Latihan\_Pytho...". On the left is a sidebar with icons for file operations. The main area shows two files: "app.py" (version 3) and "app.py". The second file contains the line "1 lebih\$ = 50" with the dollar sign (\$) underlined in red, indicating a syntax error.

lebih\$ juga tidak benar karena terdapat karakter yang tidak semestinya ada dalam penamaan variabel.

### 1.3.5 Keyword / Kata Kunci pada Python

Kata kunci mendefinisikan aturan - aturan dan struktur bahasa, dan mereka tidak dapat digunakan sebagai nama variabel.

Python mempunyai 28 kata kunci :

and	continue	else	for	import	not	raise
assert	def	except	from	in	or	return
break	del	exec	global	is	pass	try
class	elif	finally	if	lambda	print	while

Anda mungkin ingin menyimpan daftar - daftar ini, pada saat interpreter mengeluarkan kesalahan sintaks dari salah satu nama variabel Anda dan Anda tidak mengetahui penyebabnya, lihat mereka pada daftar ini.

#### C++ "Hello World"

```
#include <iostream.h>
main()
{
    cout << "Hello World! ";
}
return 0
```

#### Java "Hello World"

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

#### Python

```
print "Hello world"
```

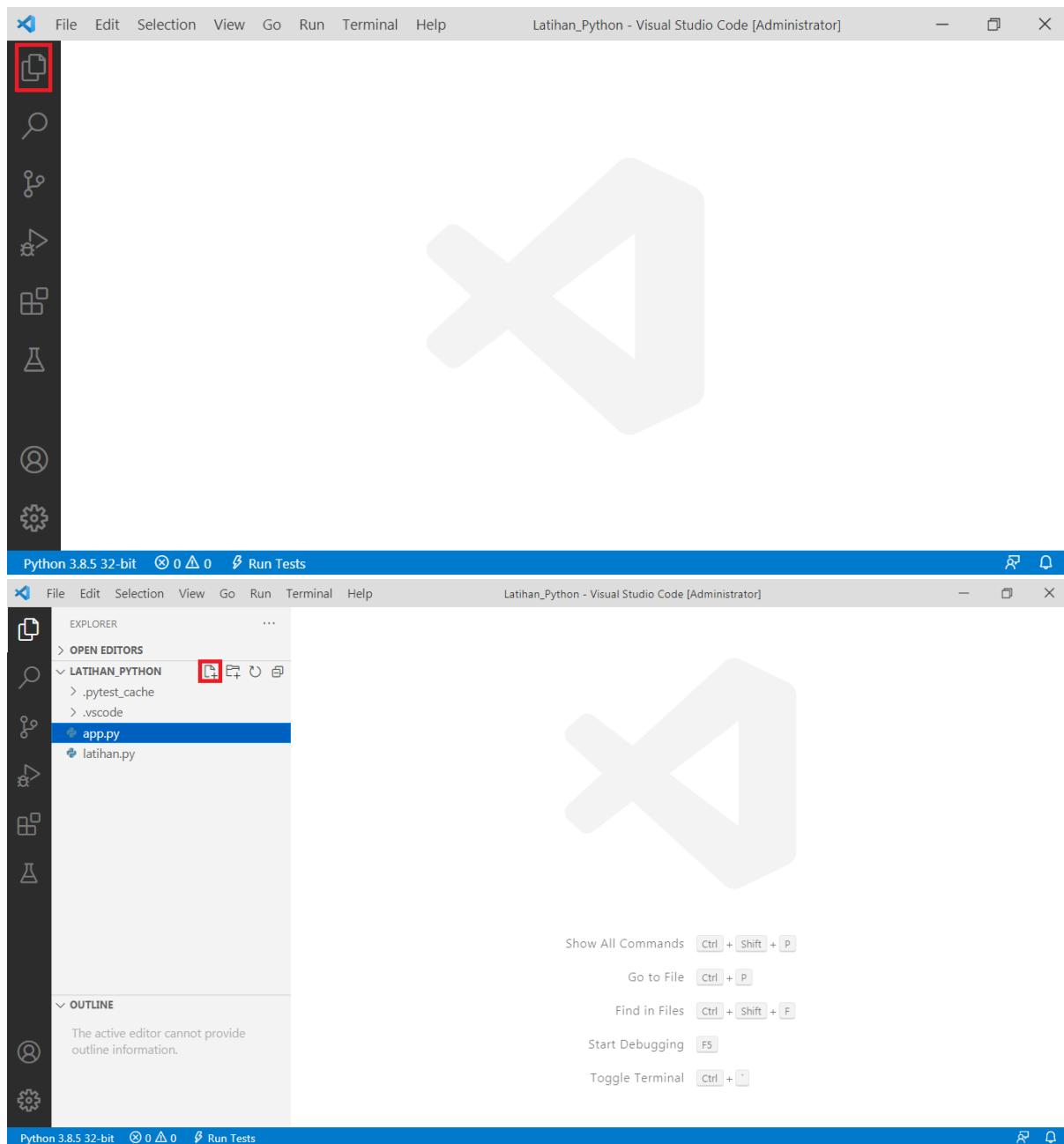


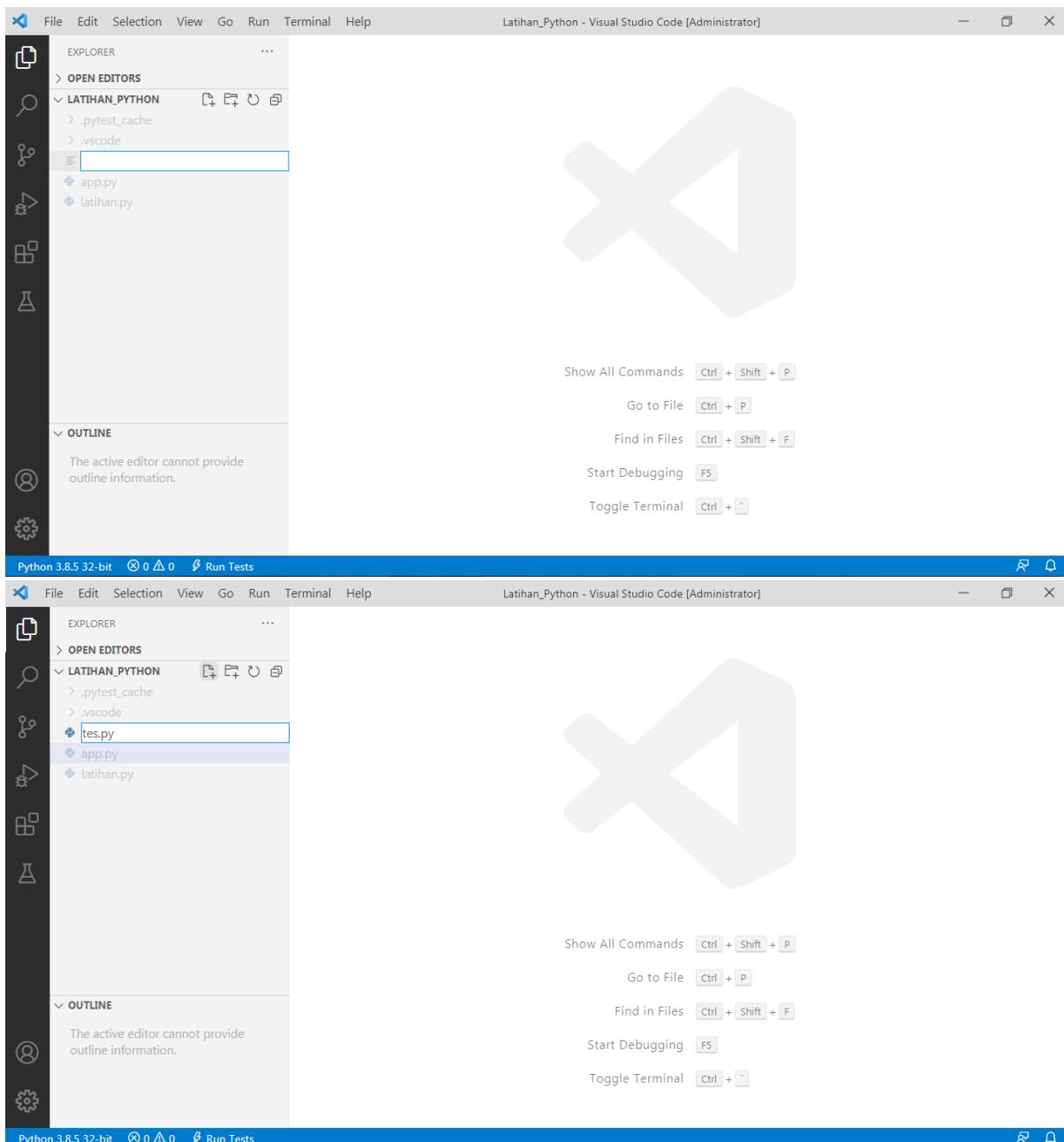
Gambar 1.12 Perbandingan Python dengan bahasa pemrograman lainnya.

## 1.4 Membuat File Editor

File Editor merupakan file kode program yang dapat dikompilasi, kemudian dijalankan untuk menampilkan hasilnya yang mempunyai ekstensi .PY.

Cara megkatifkan nya klik menu File Explorer → Klik New File → Beri Nama File → Tekan tombol Enter





Gambar 1.13 Membuat File Baru

#### 1.4.1 Menyimpan File Editor

Setelah selesai mengetikan naskah program yang baru pada jendela Tex Editor, maka selanjutnya disimpan dengan cara :

- a. Klik Menu File → Save.

Pada PyCharm VS Code terdapat 2 cara menyimpan file editor, diantaranya yaitu :

**Save** digunakan untuk menyimpan semua file Python pada file projek yang sedang aktif.

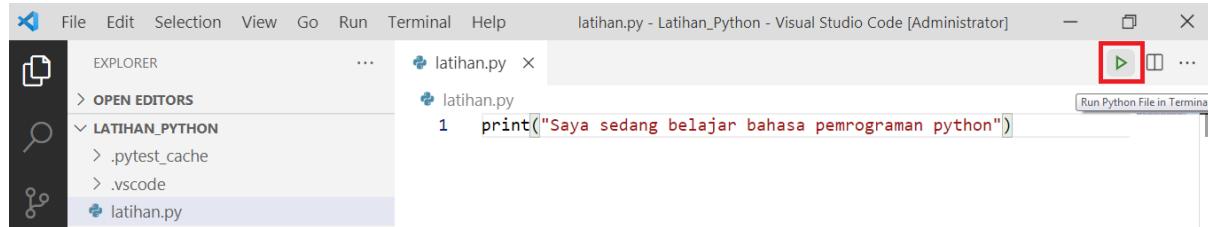
**Save As** digunakan untuk menyimpan file Pyhton yang sedang aktif dengan nama file Python yang berbeda

#### 1.4.2 Menjalankan Program

Proses Run merupakan suatu proses menerjemahkan program, melakukan proses linking, dan sekaligus menjalankan program, yaitu dengan cara :

- Klik button Run pada pojok kiri atas.
- Menekan hotkey Shift + F10

Setelah proses menterjemahkan program, proses linking, selanjutnya tampil hasil seperti gambar 1.14 dibawah ini:



Gambar 1.14 Contoh Hasil Keluaran Program

#### 1.4.3 Keluar dari VS Code

Keluar dari aplikasi VS Code, dengan cara klik Menu File → Exit

Contoh struktur sederhana dalam PyCharm 2019.3



Gambar 1.19 Input dan Output Program

### 1.5 Program Pertama dengan Python

Seringkali, program “Hello World!” digunakan untuk mengenalkan suatu bahasa pemrograman ke pemula. Program “Hello World!” adalah sebuah program sederhana yang menampilkan kalimat “Hello World!” di monitor. Akan tetapi, Python adalah bahasa yang paling mudah dipelajari, dan membuat program “Hello World!” hanya sesederhana menuliskan perintah `print("Hello World!")`. Oleh karena itu, kita akan lebih memilih untuk membuat program penjumlahan dua bilangan.

## Program Penjumlahan Dua Bilangan

```
File Edit Selection View Go Run Terminal Help
contoh01.py - Latihan_Python - Visual Studio Code [Administrator]
contoh01.py > ...
1 bil1 = 45
2 bil2 = 55
3 jumlah = bil1 + bil2
4
5 print(jumlah)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe d:/IMK/latihan_python/contoh01.py
100
PS D:\IMK\Latihan_Python>
```

### Penjelasan Program

Python adalah bahasa pemrograman yang menggunakan interpreter. Pada interpreter program akan dieksekusi baris perbaris. Bila ada error maka program akan terhenti, kecuali dengan menggunakan metode penanganan eksepsi. Pada program di atas, pada baris pertama kita menciptakan sebuah objek bilangan yaitu 45. Kita membuat variabel bil1 menunjuk ke objek 45. Dengan kata lain, objek 45 ditugaskan ke variabel bil1. Penjelasan untuk baris 2 sama dengan penjelasan untuk baris ke 1.

Selanjutnya pada baris ke 3, objek yang ditunjuk oleh bil1 yaitu 15 dan yang ditunjuk oleh bil2 yaitu 55 dijumlahkan. Hasilnya ditugaskan ke variabel jumlah. Di baris terakhir, kita menggunakan fungsi bawaan (builtin) python, yaitu print() untuk menampilkan variabel jumlah ke monitor. Demikian program sederhana menggunakan python.

## Pertemuan 2

### Sintaks Dasar Python, Tipe data & Variable

Python merupakan bahasa pemrograman yang memiliki sintaks yang sederhana dan mudah dimengerti. Python memiliki filosofi bahwa kode program harus mudah dibaca.

**Statement (Pernyataan) di Python**, Semua perintah yang bisa dieksekusi oleh Python disebut statement. Misalnya, `a = 1` adalah sebuah statement penugasan. Selain statement penugasan ada statement lain seperti statement if, statement for, dan lain sebagainya.

**Statement Multibaris**, Pada Python, akhir dari sebuah statement adalah karakter baris baru (newline). Kita dapat membuat sebuah statement terdiri dari beberapa baris dengan menggunakan tanda backslash (\). Misalnya:

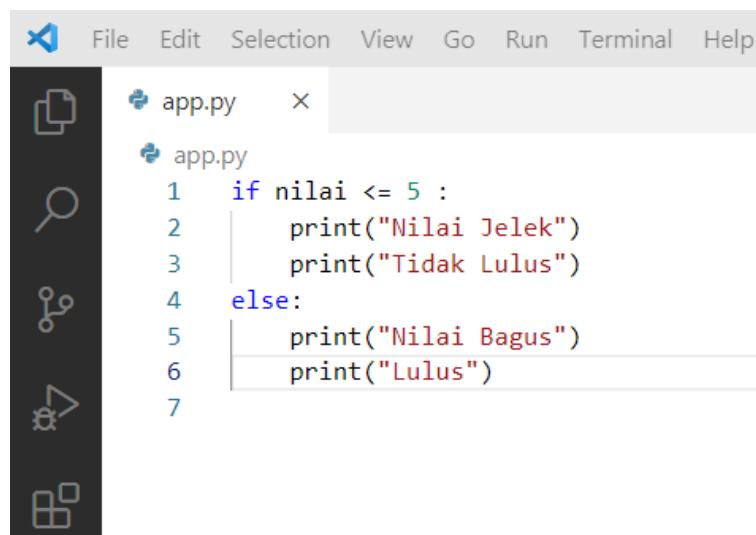
```
a = panjang1 + panjang2 + \
panjang3 + \
panjang4
```

Statement yang ada di dalam tanda kurung [ ], { }, dan ( ) tidak memerlukan tanda \. Contohnya:

```
nama_bulan = ['Januari', 'Februari', 'Maret', 'April', 'Mei',
'Juni']
```

#### 2.1 Baris dan Indentasi

Python tidak menggunakan tanda { } untuk menandai blok / grup kode. Blok kode di python menggunakan tanda indentasi (spasi). Jumlah spasi untuk setiap baris yang ada dalam satu blok kode harus sama. Contoh yang benar adalah sebagai berikut:



The screenshot shows a code editor window with a dark theme. The menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. On the left is a sidebar with icons for file operations like new, open, save, and run. The main area shows a file named 'app.py' with the following code:

```
File Edit Selection View Go Run Terminal Help
app.py ×
app.py
1 if nilai <= 5 :
2     print("Nilai Jelek")
3     print("Tidak Lulus")
4 else:
5     print("Nilai Bagus")
6     print("Lulus")
7
```

Bila indentasi dalam satu grup kode tidak sama, python akan menampilkan sintaks error.

```
File Edit Selection View Go Run Terminal Help
contoh01.py - Latihan_Python - Visual Studio Code [Administrator]
contoh01.py 6 ×
contoh01.py > ...
1 nilai = 8
2 if nilai <= 5 :
3     print("Nilai Jelek")
4     print("Tidak Lulus")
5 else:
6     print("Nilai Bagus")
7     print("Lulus")

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL
Python +
```

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe d:/IMK/Latihan_Python/contoh01.py
File "d:/IMK/Latihan_Python/contoh01.py", line 4
    print("Tidak Lulus")
               ^
IndentationError: unindent does not match any outer indentation level
PS D:\IMK\Latihan_Python>
```

### 2.1.1 Tanda Kutip di Python

Python menggunakan tanda kutip tunggal ('), ganda ("'), maupun triple (''' atau ''") untuk menandai string, sepanjang stringnya diawali oleh tanda kutip yang sama di awal dan akhir string. Tanda kutip tiga digunakan untuk string multibaris. Ketiga contoh berikut, semuanya adalah benar.

```
kata = 'kata'
kalimat = "Ini adalah kalimat"
paragraf = """Ini adalah Paragraf terdiri dari beberapa
baris"""

```

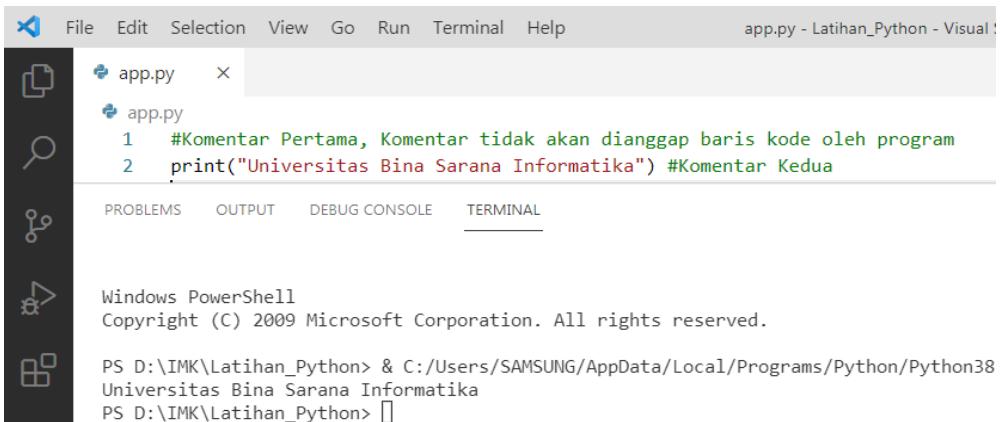
```
File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python - Visual Studio Code [Admin]
app.py 6 ×
app.py > ...
1 kata = 'Kuliah'
2 kalimat = "BSI Aja!"
3 paragraf = """Kuliah..?
4 BSI Aja!"""
5
6 print(kata)
7 print(kalimat)
8 print(paragraf)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe
Kuliah
BSI Aja!
Kuliah..?
BSI Aja!
PS D:\IMK\Latihan_Python>
```

### 2.1.2 Komentar di Python

Tanda pagar (#) digunakan untuk menandai komentar di python. Komentar tidak akan diproses oleh interpreter Python. Komentar hanya berguna untuk programmer untuk memudahkan memahami maksud dari kode.



```

File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python - Visual Studio Code

app.py
1 #Komentar Pertama, Komentar tidak akan dianggap baris kode oleh program
2 print("Universitas Bina Sarana Informatika") #Komentar Kedua

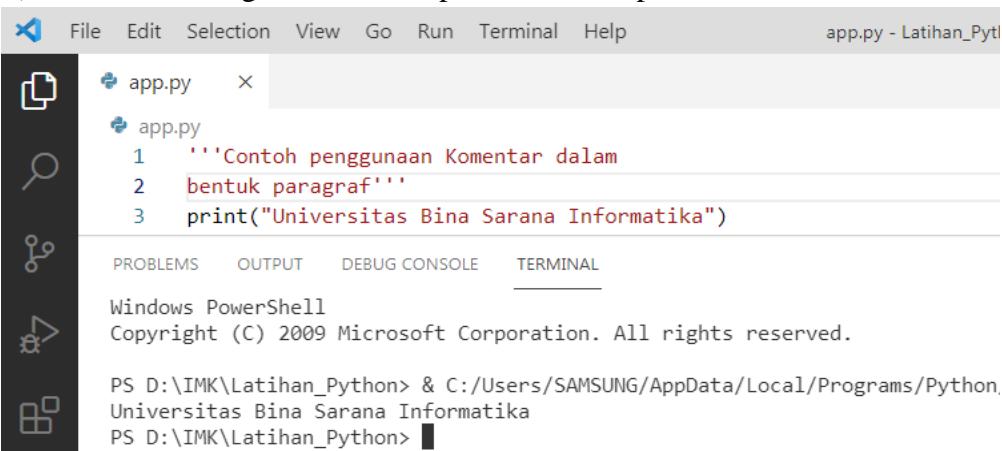
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38
Universitas Bina Sarana Informatika
PS D:\IMK\Latihan_Python>

```

Python juga memiliki fitur komentar multibaris dengan perintah triple tanda kutip (""). Kita harus mengomentari satu persatu baris seperti berikut:



```

File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python - Visual Studio Code

app.py
1 '''Contoh penggunaan Komentar dalam
2 bentuk paragraf'''
3 print("Universitas Bina Sarana Informatika")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

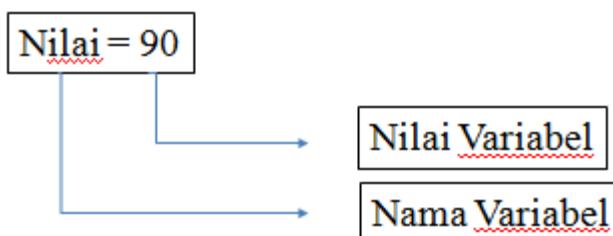
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38
Universitas Bina Sarana Informatika
PS D:\IMK\Latihan_Python>

```

## 2.2 Variabel dan Tipe Data Python

Variabel adalah lokasi di memori yang digunakan untuk menyimpan nilai. Memberi Nilai Variabel, Di python, variabel tidak perlu dideklarasikan secara eksplisit. Deklarasi atau pembuatan variabel terjadi secara otomatis pada saat kita memberi (menugaskan) suatu nilai ke variabel. Tanda sama dengan (=) digunakan untuk memberikan nilai ke variabel. Operand di sebelah kiri tanda = adalah nama variabel dan di sebelah kanan tanda = adalah nilai yang disimpan di dalam variabel.



Data yang disimpan di memori memiliki tipe yang berbeda – beda. Misalnya untuk panjang, akan disimpan dengan tipe bilangan. Nama orang akan disimpan dalam tipe string/karakter. Suhu akan disimpan dalam bentuk bilangan berkoma. Dan lain sebagainya. Tipe data adalah suatu media atau memori pada komputer yang digunakan untuk menampung informasi. Python sendiri mempunyai tipe data yang cukup unik

bila kita bandingkan dengan bahasa pemrograman yang lain. Berikut adalah tipe data dari bahasa pemrograman Python :

Tabel 2.1 Tipe data Python

Tipe Data	Contoh	Penjelasan
Boolean	True atau False	Menyatakan benar True yang bernilai 1, atau salah False yang bernilai 0
String	"Ayo belajar Python"	Menyatakan karakter/kalimat bisa berupa huruf angka, dll (diapit tanda " atau ')
Integer	25 atau 1209	Menyatakan bilangan bulat
Float	3.14 atau 0.99	Menyatakan bilangan yang mempunyai koma
Hexadecimal	9a atau 1d3	Menyatakan bilangan dalam format heksa (bilangan berbasis 16)
Complex	1 + 5j	Menyatakan pasangan angka real dan imajiner
List	['xyz', 786, 2.23]	Data untaian yang menyimpan berbagai tipe data dan isinya bisa diubah-ubah
Tuple	('xyz', 768, 2.23)	Data untaian yang menyimpan berbagai tipe data tapi isinya tidak bisa diubah
Dictionary	{'nama': 'adi','id':2}	Data untaian yang menyimpan berbagai tipe data berupa pasangan penunjuk dan nilai

### 2.3 Input dan Output pada Python

Python menyediakan banyak fungsi *built-in* yang bisa kita pergunakan. Salah satunya adalah yang berkaitan dengan fungsi i/o atau input output. Fungsi bawaan untuk melakukan operasi output adalah **print()**, dan fungsi untuk melakukan operasi input adalah fungsi **input()**.

### 2.3.1 Input pada Python

Python 2 memiliki dua fungsi built-in untuk membaca data dari input standar, yang secara default berasal dari keyboard. Fungsi ini adalah `input()` dan `raw_input()`. Dengan Python 3, fungsi `raw_input()` tidak digunakan lagi. Selain itu, `input()` berfungsi membaca data dari keyboard sebagai string, terlepas dari apakah itu tertutup dengan tanda kutip (" atau "") atau tidak. Perhatikan contoh dibawah ini :

#### 1. Jika variable dengan tipe data string

```
nim=input("masukkan NIM anda") #Tekan tombol Enter setelah selesai input.
```

#### 2. Jika variable dengan tipde data Int

```
angka=int(input("Masukkan angka : "))
```

### 2.3.2 Output pada Python

Cara termudah untuk menghasilkan output adalah dengan menggunakan pernyataan cetak di mana Anda bisa melewati nol atau lebih banyak ekspresi yang dipisahkan dengan koma. Fungsi ini mengubah ekspresi yang Anda berikan ke string dan menulis hasilnya ke output standar sebagai berikut :

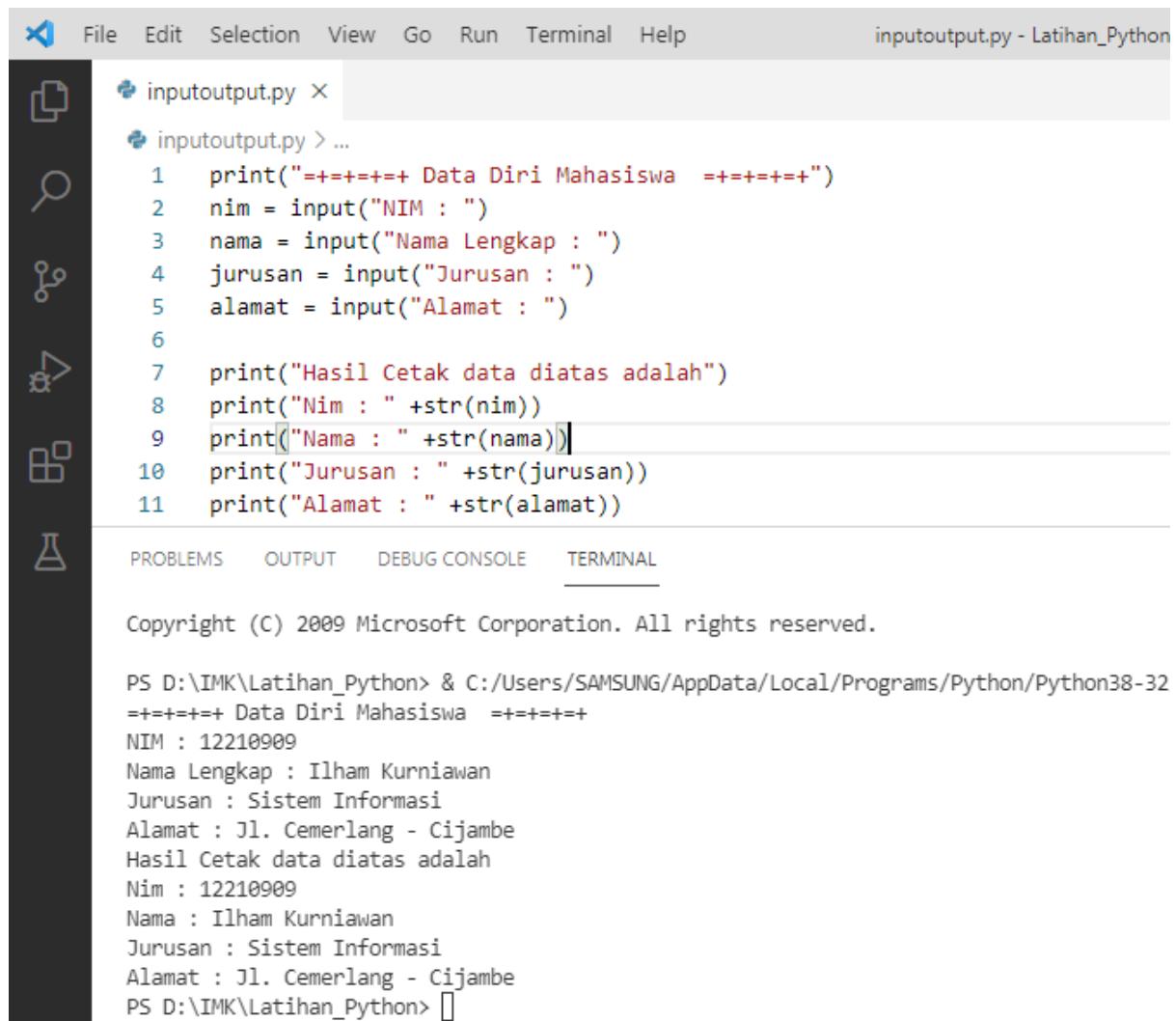
```
print ("Python adalah bahasa pemrograman yang hebat")
```

#### Mencetak Nilai Variabel pada Python (Print)

### Latihan Input & Output

#### Data Diri Mahasiswa

```
Masukkan NIM anda : <input>
Masukkan Nama anda : <input>
Jurusan : <input>
Alamat : <input>
```



File Edit Selection View Go Run Terminal Help

inputoutput.py - Latihan\_Python

inputoutput.py > ...

```
1 print("=+=+=+= Data Diri Mahasiswa =+=+=+=")
2 nim = input("NIM : ")
3 nama = input("Nama Lengkap : ")
4 jurusan = input("Jurusan : ")
5 alamat = input("Alamat : ")
6
7 print("Hasil Cetak data diatas adalah")
8 print("Nim : " +str(nim))
9 print("Nama : " +str(nama))
10 print("Jurusan : " +str(jurusan))
11 print("Alamat : " +str(alamat))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan\_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32

=+=+=+= Data Diri Mahasiswa =+=+=+=

NIM : 12210909

Nama Lengkap : Ilham Kurniawan

Jurusan : Sistem Informasi

Alamat : Jl. Cemerlang - Cijambe

Hasil Cetak data diatas adalah

Nim : 12210909

Nama : Ilham Kurniawan

Jurusan : Sistem Informasi

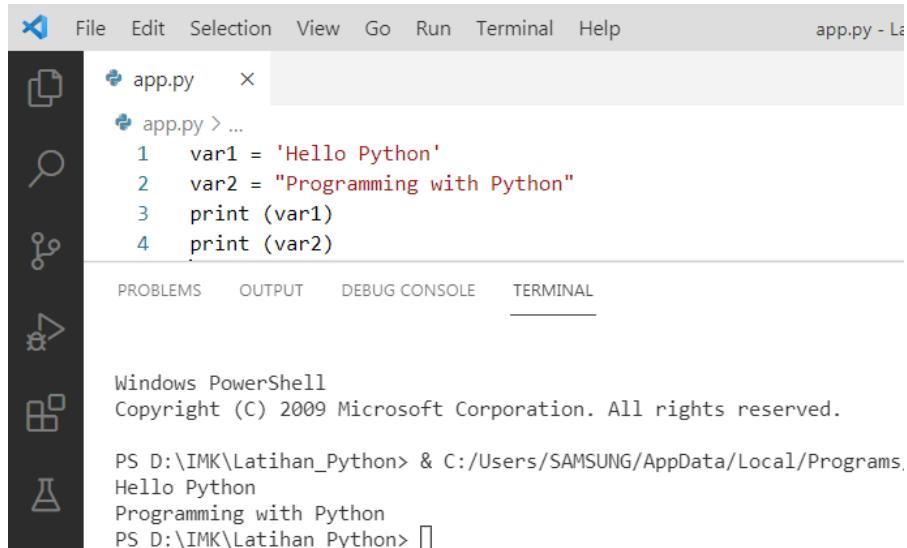
Alamat : Jl. Cemerlang - Cijambe

PS D:\IMK\Latihan\_Python> []

## Pertemuan 3

### String, Bilangan & Operator

String adalah tipe data yang paling sering digunakan di Python. Kita bisa membuat string dengan meletakkan karakter di dalam tanda kutip. Tanda kutipnya bisa kutip tunggal maupun kutip ganda. Contohnya adalah sebagai berikut:



```
File Edit Selection View Go Run Terminal Help
app.py - L
app.py > ...
1 var1 = 'Hello Python'
2 var2 = "Programming with Python"
3 print (var1)
4 print (var2)

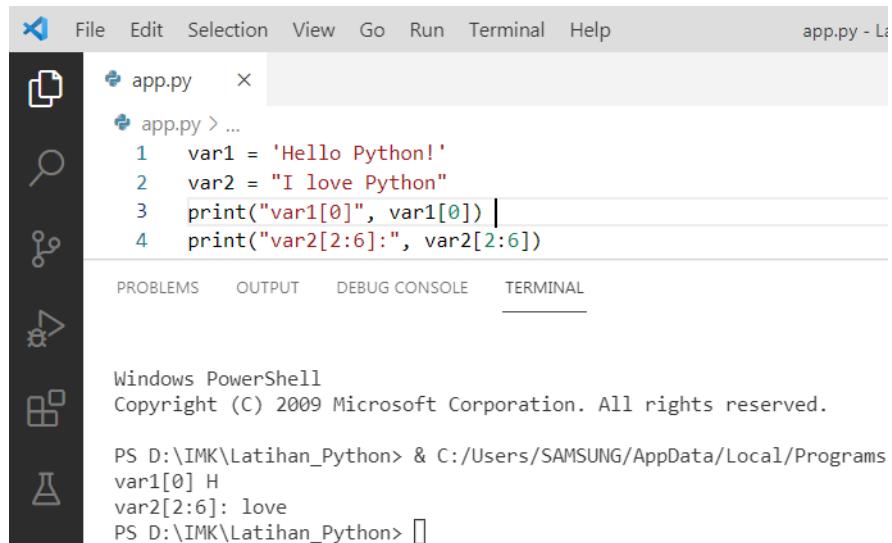
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Hello Python
Programming with Python
PS D:\IMK\Latihan_Python>
```

### 3.1 Mengakses Nilai String

Untuk mengakses nilai atau substring dari string, kita menggunakan indeks dalam tanda [ ].



```
File Edit Selection View Go Run Terminal Help
app.py - L
app.py > ...
1 var1 = 'Hello Python!'
2 var2 = "I love Python"
3 print("var1[0]", var1[0])
4 print("var2[2:6]", var2[2:6])

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
var1[0] H
var2[2:6]: love
PS D:\IMK\Latihan_Python>
```

### 3.2 Mengupdate String

String adalah tipe data immutable, artinya tidak bisa diubah. Untuk mengupdate string, kita perlu memberikan nilai variabel string lama ke string yang baru. Nilai yang baru adalah nilai string lama yang sudah diupdate.

```
File Edit Selection View Go Run Terminal Help
app.py

app.py > ...
1 var1 = 'Hello Python!'
2 var2 = var1[:6]
3 print(var1)
4 print("String Update: - ", var1[:6] + 'World')

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python37-32/python app.py
Hello Python!
String Update: - Hello World
PS D:\IMK\Latihan_Python>
```

### 3.3 Menggabung String

Kita bisa menggabungkan dua atau lebih string menjadi satu dengan menggunakan operator `+`. Selain itu kita juga bisa melipatgandakan string menggunakan operator `*`.

```
File Edit Selection View Go Run Terminal Help
app.py - L

app.py > ...
1 str1 = 'Hello'
2 str2 = 'Python'
3
4 # menggunakan +
5 print('str1 + str2 =', str1 + str2)
6 |
7 # menggunakan *
8 print('str1 * 3 =', str1 * 3)

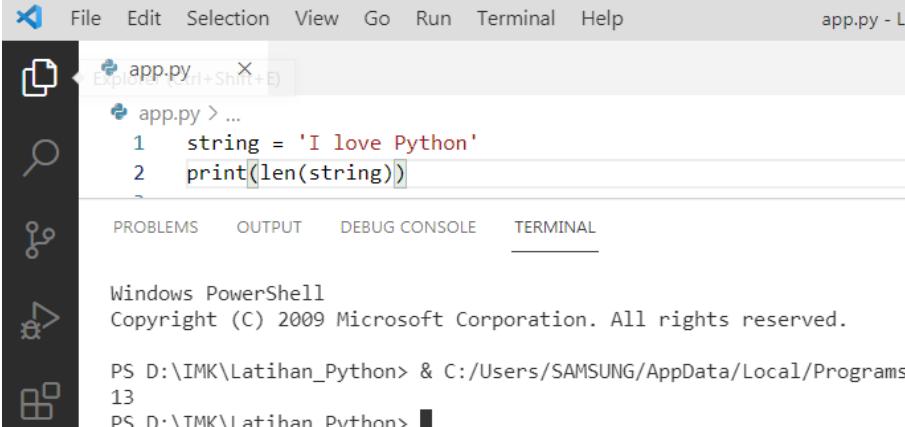
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python37-32/python app.py
HelloPython
HelloHelloHello
PS D:\IMK\Latihan_Python>
```

### 3.4 Mengetahui Panjang String

Untuk mengetahui panjang dari string, kita bisa menggunakan fungsi len().



```
File Edit Selection View Go Run Terminal Help
app.py - La
app.py > ...
1 string = 'I love Python'
2 print(len(string))

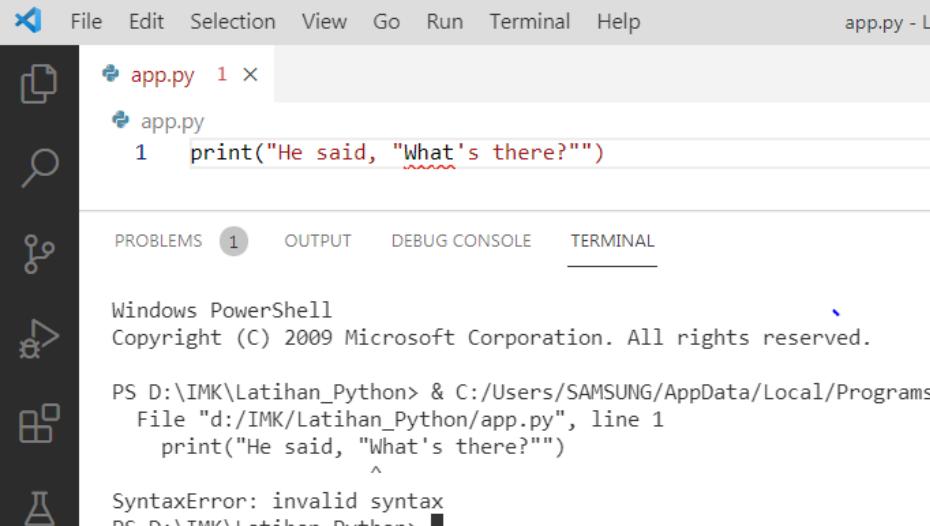
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
13
PS D:\IMK\Latihan_Python>
```

#### 3.4.1 Karakter Escape

Kalau kita hendak mencetak string: He said, "What's there?" kita tidak bisa menggunakan tanda kutip tunggal maupun ganda. Bila kita melakukannya, akan muncul pesan error SyntaxError karena teks berisi kutip tunggal dan ganda.



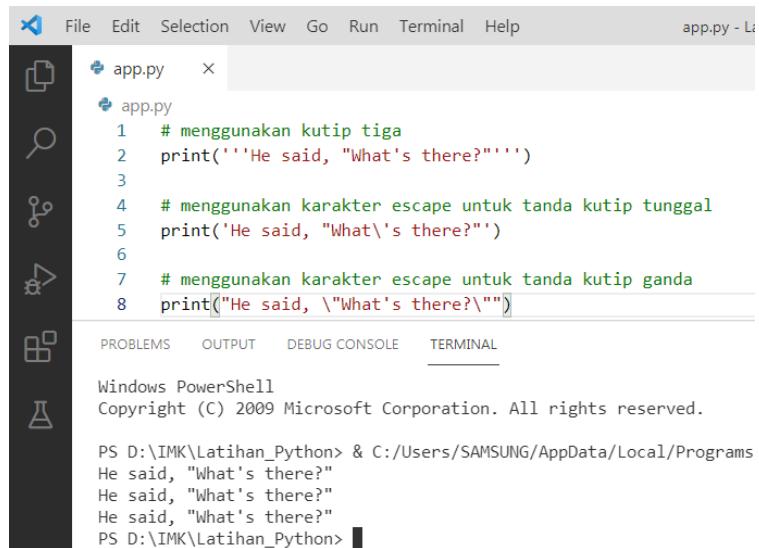
```
File Edit Selection View Go Run Terminal Help
app.py 1 ×
app.py
1 print("He said, "What's there?"")

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
File "d:/IMK/Latihan_Python/app.py", line 1
    print("He said, "What's there?""")
          ^
SyntaxError: invalid syntax
PS D:\IMK\Latihan_Python>
```

Untuk hal seperti ini kita bisa menggunakan tanda kutip tiga atau menggunakan karakter escape. Karakter escape dimulai dengan tanda backslash \. Interpreter akan menerjemahkannya dengan cara berbeda dengan string biasa. Solusi untuk error di atas adalah sebagai berikut:



The screenshot shows a Windows PowerShell window titled 'app.py - La'. The code in 'app.py' is as follows:

```

1 # menggunakan kutip tiga
2 print('''He said, "What's there?''')
3
4 # menggunakan karakter escape untuk tanda kutip tunggal
5 print('He said, "What\'s there?"')
6
7 # menggunakan karakter escape untuk tanda kutip ganda
8 print("He said, \"What's there?\"")

```

The output in the terminal shows the strings being printed:

```

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
He said, "What's there?"
He said, "What's there?"
He said, "What's there?"
PS D:\IMK\Latihan_Python>

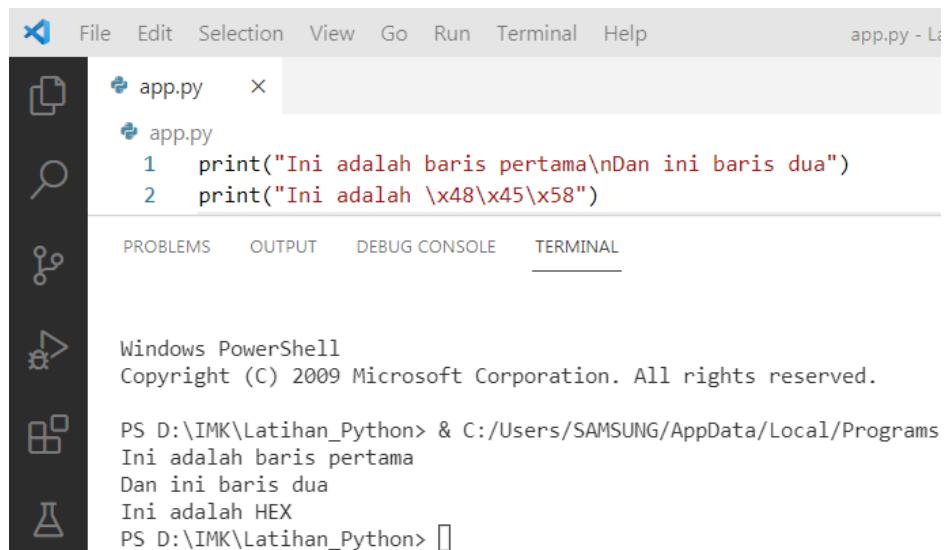
```

Berikut adalah daftar karakter escape yang didukung oleh Python.

Tabel 9.1 Karakter Escape yang didukung Pyhton

Karakter Escape	Deskripsi
\newline	Backslash dan newline diabaikan
\\\	Backslash
\'	Kutip tunggal
\\"	Kutip ganda
\a	ASCII bel
\b	ASCII backspace
\f	ASCII formfeed
\n	ASCII linefeed
\r	ASCII carriage return
\t	ASCII tab horizontal
\v	ASCII tab horizontal
\ooo	karakter dengan nilai oktal oo
\xHH	karakter dengan nilai heksadesimal HH

Berikut ini adalah beberapa contohnya:



```
File Edit Selection View Go Run Terminal Help app.py - La
app.py x
app.py
1 print("Ini adalah baris pertama\nDan ini baris dua")
2 print("Ini adalah \x48\x45\x58")

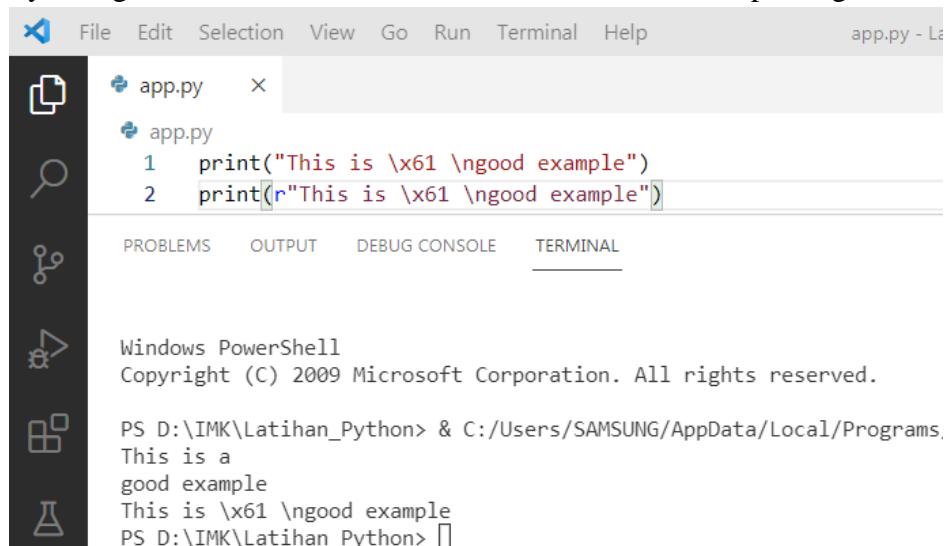
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Ini adalah baris pertama
Dan ini baris dua
Ini adalah HEX
PS D:\IMK\Latihan_Python>
```

### 3.4.2 Raw String untuk Mengabaikan Karakter Escape

Kadang kala kita perlu untuk mengabaikan karakter escape yang ada dalam string. Kita bisa melakukannya dengan meletakkan huruf r atau R sebelum tanda kutip string.



```
File Edit Selection View Go Run Terminal Help app.py - La
app.py x
app.py
1 print("This is \x61 \ngood example")
2 print(r"This is \x61 \ngood example")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

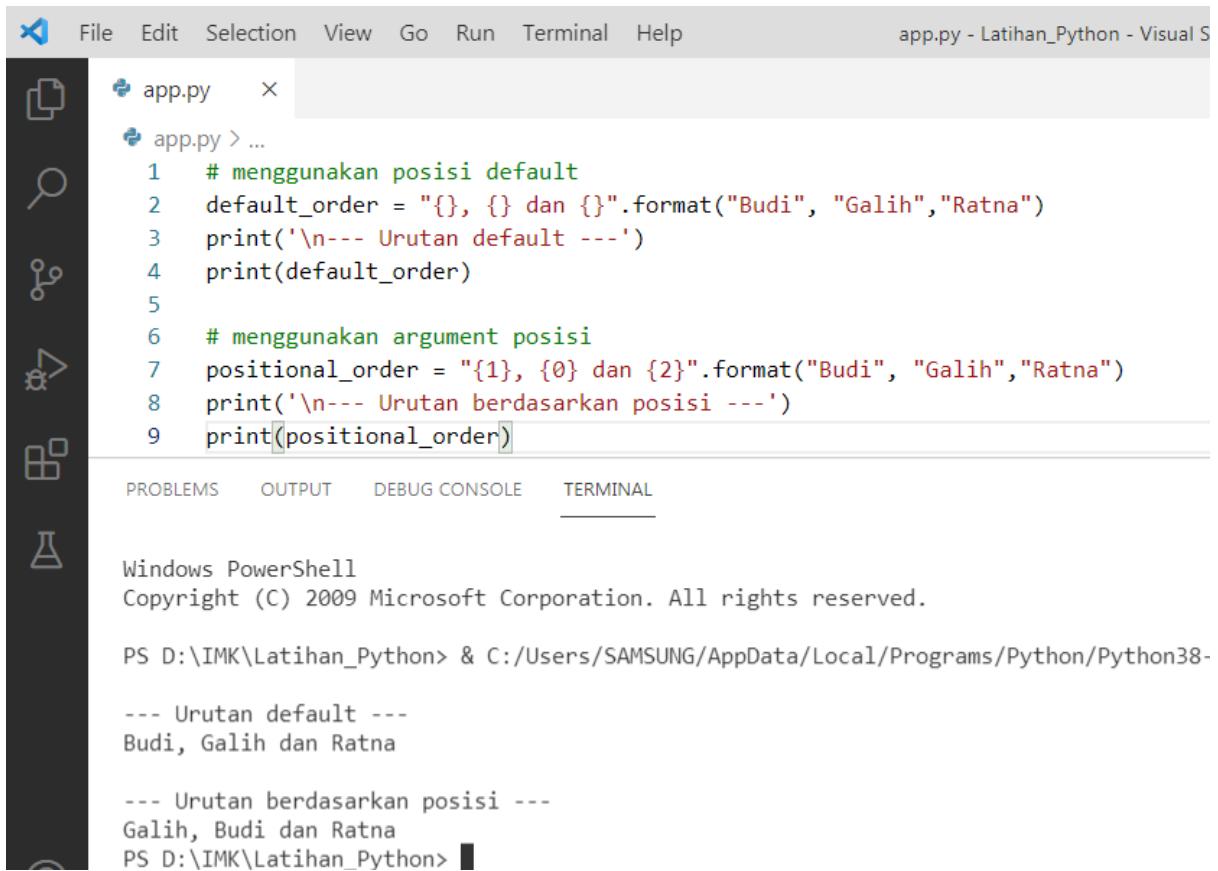
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
This is a
good example
This is \x61 \ngood example
PS D:\IMK\Latihan_Python>
```

## 3.5 Mengatur Format String

Ada dua cara melakukan format pada string. Pertama dengan menggunakan fungsi format(), dan kedua dengan menggunakan cara lama (menggunakan %).

### 3.5.1 Metode format()

Memformat string dengan fungsi format() dibuat dengan menggunakan tanda {} sebagai placeholder atau posisi substring yang akan digantikan. Kita bisa menggunakan argumen posisi atau kata kunci untuk menunjukkan urutan dari substring.



The screenshot shows the Visual Studio Code interface with the following details:

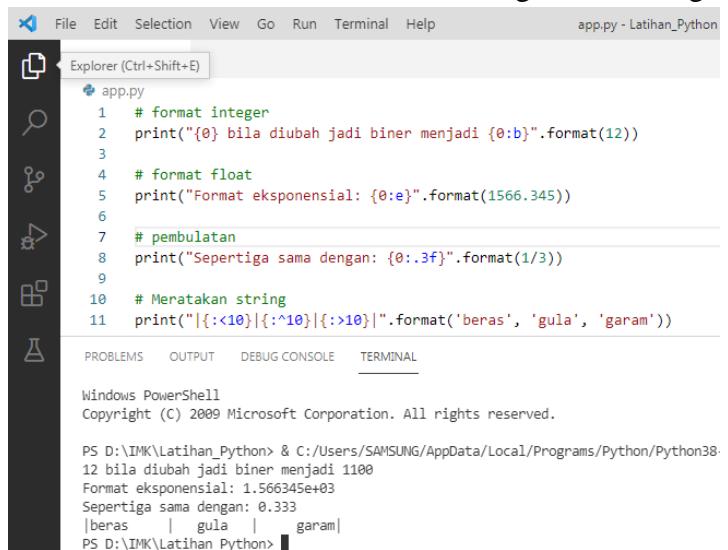
- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** app.py - Latihan\_Python - Visual S
- Left Sidebar:** Explorer (Ctrl+Shift+E), Search, Find, Open, Recent, and a pinned icon.
- Code Editor:** Contains Python code demonstrating positional and keyword arguments with `format`.

```
1 # menggunakan posisi default
2 default_order = "{}, {} dan {}".format("Budi", "Galih", "Ratna")
3 print('\n--- Urutan default ---')
4 print(default_order)
5
6 # menggunakan argument posisi
7 positional_order = "{1}, {0} dan {2}".format("Budi", "Galih", "Ratna")
8 print('\n--- Urutan berdasarkan posisi ---')
9 print(positional_order)
```
- Bottom Navigation:** PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL.
- Terminal Output:** Windows PowerShell, showing the execution of the script and its output:

```
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-
--- Urutan default ---
Budi, Galih dan Ratna

--- Urutan berdasarkan posisi ---
Galih, Budi dan Ratna
PS D:\IMK\Latihan_Python>
```

Metode `format()` dapat memiliki spesifikasi format opsional. Misalnya, kita bisa menggunakan tanda `<` untuk rata kiri, `>` untuk rata kanan, `^` untuk rata tengah, dan sebagainya.



The screenshot shows the Visual Studio Code interface with the following details:

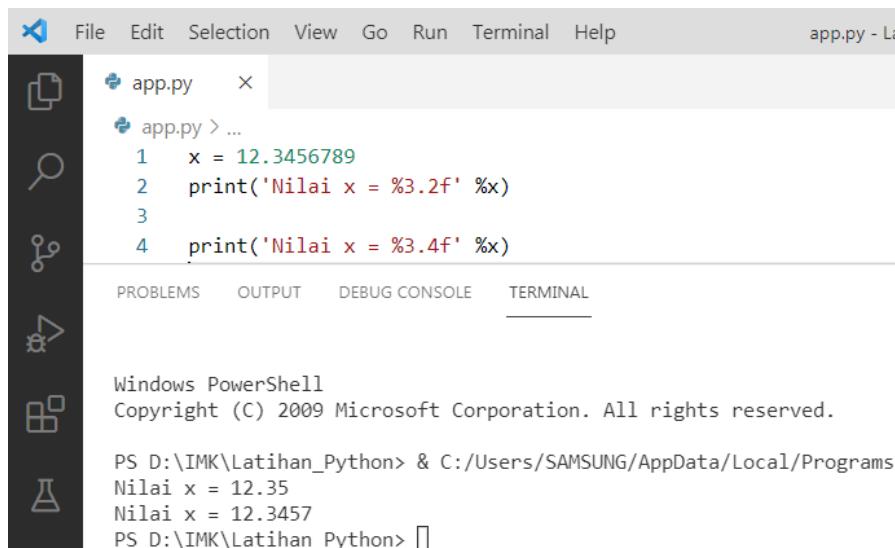
- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** app.py - Latihan\_Python
- Left Sidebar:** Explorer (Ctrl+Shift+E), Search, Find, Open, Recent, and a pinned icon.
- Code Editor:** Contains Python code demonstrating various optional format specifiers.

```
1 # format integer
2 print("{} bila diubah jadi biner menjadi {}".format(12))
3
4 # format float
5 print("Format eksponensial: {}".format(1566.345))
6
7 # pembulatan
8 print("Sepertiga sama dengan: {:.3f}".format(1/3))
9
10 # Meratakan string
11 print("|{:<10}|{:>10}|{:^10}|".format('beras', 'gula', 'garam'))
```
- Bottom Navigation:** PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL.
- Terminal Output:** Windows PowerShell, showing the execution of the script and its output:

```
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-
12 bila diubah jadi biner menjadi 1100
Format eksponensial: 1.566345e+03
Sepertiga sama dengan: 0.333
|beras    |    gula   |      garam|
PS D:\IMK\Latihan_Python>
```

Format cara lama dengan %

Kita bisa menggunakan operator %.



```
File Edit Selection View Go Run Terminal Help app.py - La
app.py > ...
1 x = 12.3456789
2 print('Nilai x = %3.2f' %x)
3
4 print('Nilai x = %3.4f' %x)

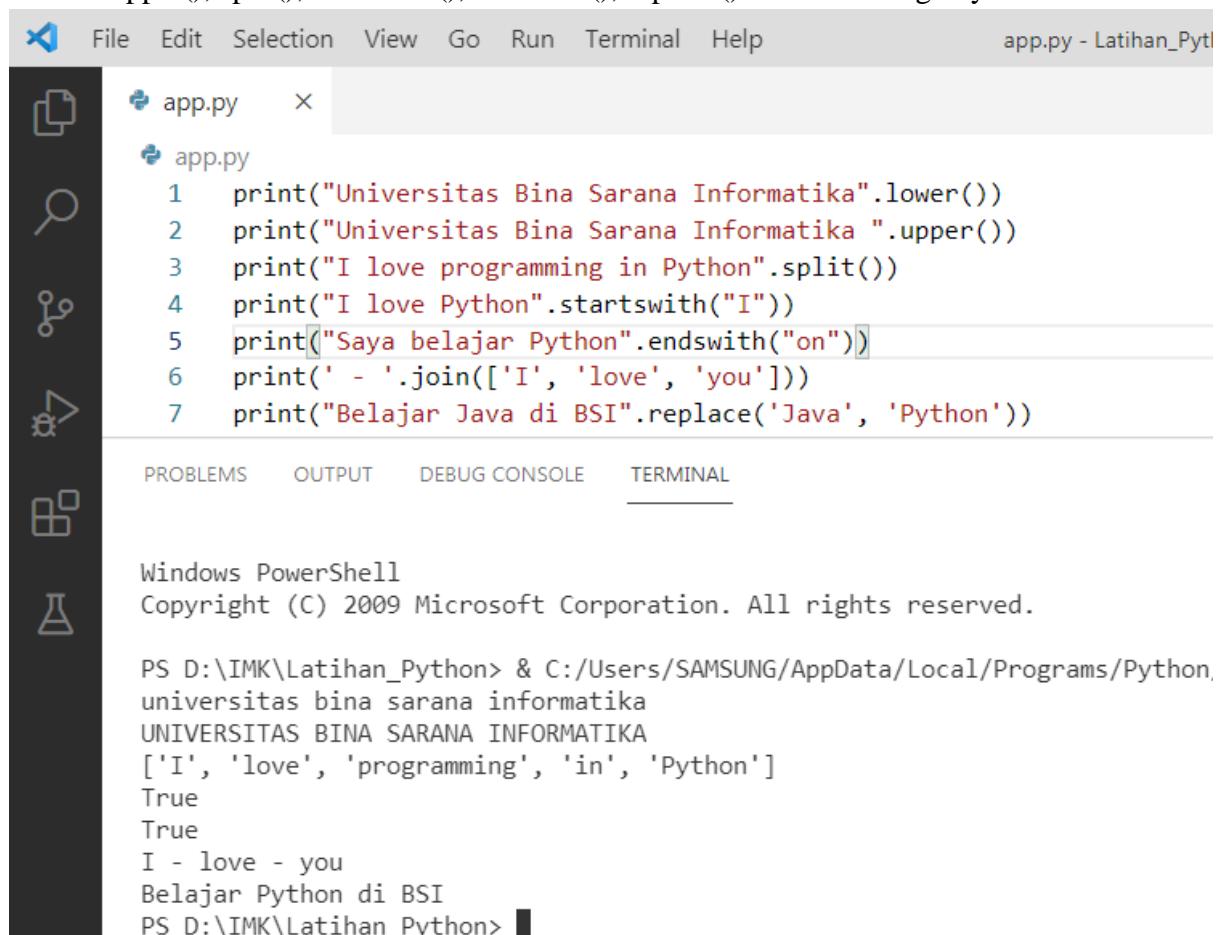
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
Nilai x = 12.35
Nilai x = 12.3457
PS D:\IMK\Latihan_Python>
```

### 3.5.2 Metode / Fungsi Bawaan String

String memiliki banyak fungsi bawaan. `format()` yang kita bahas di atas hanya salah satu darinya. Fungsi atau metode lainnya yang sering dipakai adalah `join()`, `lower()`, `upper()`, `split()`, `startswith()`, `endswith()`, `replace()` dan lain sebagainya.



```
File Edit Selection View Go Run Terminal Help app.py - Latihan_Pyt
app.py > ...
1 print("Universitas Bina Sarana Informatika".lower())
2 print("Universitas Bina Sarana Informatika ".upper())
3 print("I love programming in Python".split())
4 print("I love Python".startswith("I"))
5 print("Saya belajar Python".endswith("on"))
6 print(' - '.join(['I', 'love', 'you']))
7 print("Belajar Java di BSI".replace('Java', 'Python'))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python,
universitas bina sarana informatika
UNIVERSITAS BINA SARANA INFORMATIKA
['I', 'love', 'programming', 'in', 'Python']
True
True
I - love - you
Belajar Python di BSI
PS D:\IMK\Latihan_Python>
```

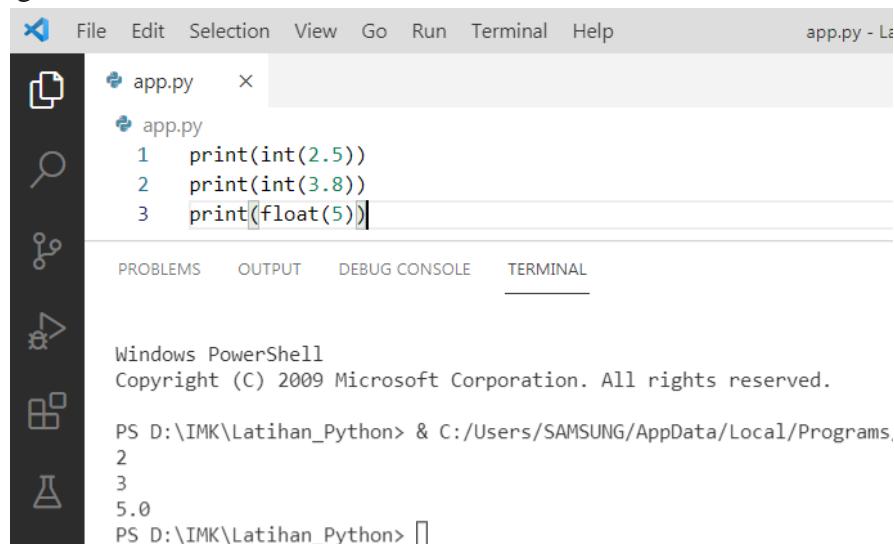
### 3.6 Bilangan (Number)

Bilangan (number) adalah salah satu tipe data dasar di Python. Python mendukung bilangan bulat (integer), bilangan pecahan (float), dan bilangan kompleks (complex). Masing – masing diwakili oleh kelas int, float, dan complex. Integer adalah bilangan bulat, yaitu bilangan yang tidak mempunyai koma. Contohnya 1, 2, 100, -30, -5, 99999, dan lain sebagainya. Panjang integer di python tidak dibatasi jumlah digitnya. Selama memori masih cukup, maka sepanjang itulah jumlah digit yang akan ditampilkan.

Float adalah bilangan pecahan atau bilangan berkoma. Contohnya adalah 5.5, 3.9, 72.8, -1.5, -0.7878999, dan lain sebagainya. Panjang angka di belakang koma untuk float ini adalah 15 digit. Bilangan kompleks (complex) adalah bilangan yang terdiri dari dua bagian, yaitu bagian yang real dan bagian yang imajiner. Contohnya adalah  $3 + 2j$ ,  $9 - 5j$ , dan lain sebagainya.

#### 3.6.1 Konversi Jenis Bilangan

Kita bisa mengubah jenis bilangan dari int ke float, atau sebaliknya. Mengubah bilangan integer ke float bisa menggunakan fungsi `int(num)` dimana num adalah bilangan float.



```
File Edit Selection View Go Run Terminal Help app.py - La
app.py x
app.py
1 print(int(2.5))
2 print(int(3.8))
3 print(float(5))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

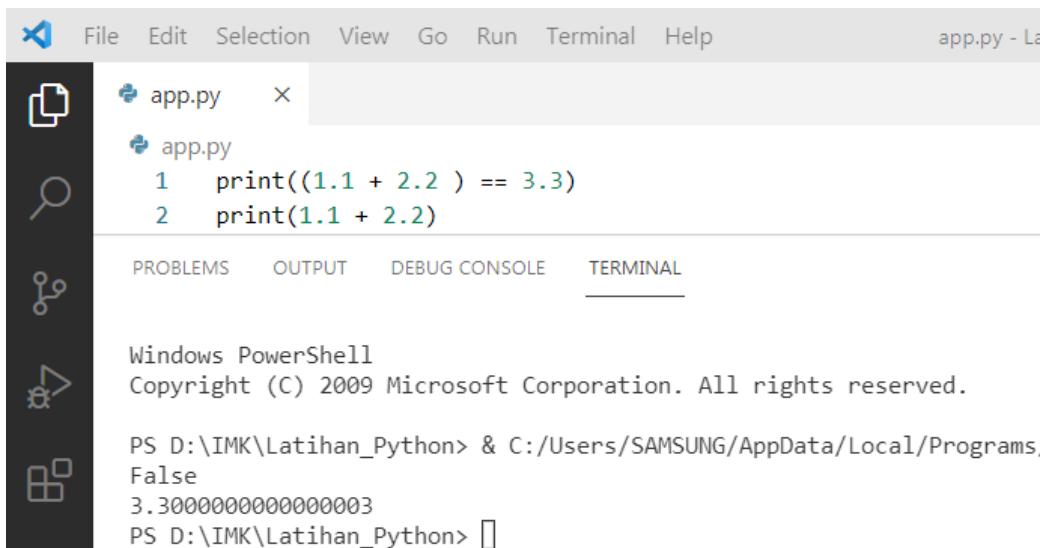
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/
2
3
5.0
PS D:\IMK\Latihan_Python>
```

Pada saat kita mengubah float ke integer, bilangan dibulatkan ke bawah. Sebaliknya saat kita mengubah integer ke float, maka bilangan bulat akan menjadi bilangan berkoma.

#### 3.6.2 Python Decimal

Ada kalanya perhitungan menggunakan float di Python membuat kita terkejut. Kita tahu bahwa  $1.1 + 2.2$  hasilnya adalah  $3.3$ . Tapi pada saat kita lakukan dengan Python, maka hasilnya berbeda.



```

File Edit Selection View Go Run Terminal Help
app.py - La
app.py
1 print((1.1 + 2.2 ) == 3.3)
2 print(1.1 + 2.2)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
False
3.3000000000000003
PS D:\IMK\Latihan_Python>

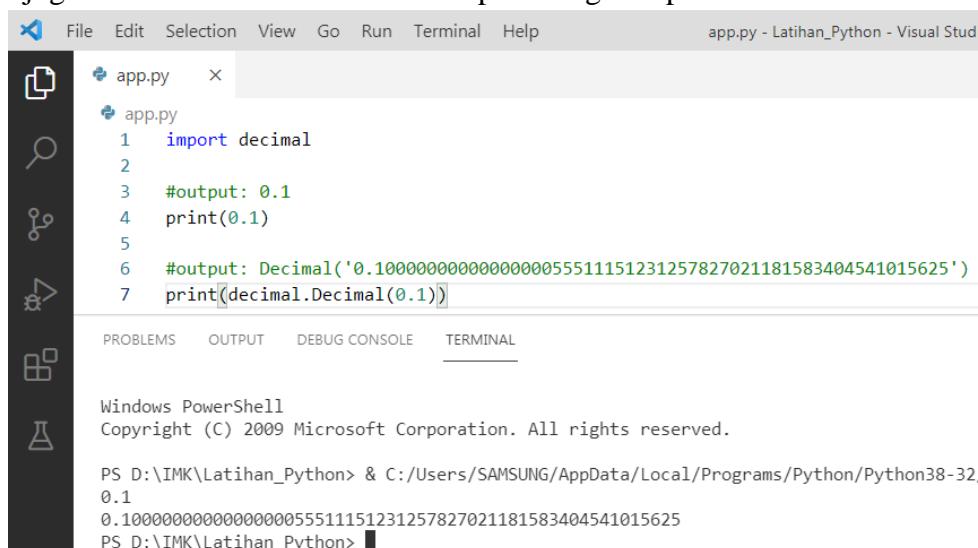
```

Mengapa terjadi demikian?

Hal ini terjadi karena bilangan dalam komputer disimpan dalam bentuk digit 0 atau 1. Bila padanan digitnya tidak sesuai, maka bilangan float seperti 0.1 dalam bilangan biner akan menjadi pecahan yang sangat panjang yaitu 0.00011001100110011001... dan komputer kita hanya akan menyimpan panjang yang terbatas. Hal inilah yang menyebabkan terjadinya masalah seperti pada contoh di atas.

Untuk menangani hal seperti itu, kita bisa menggunakan modul bawaan Python yaitu modul decimal. Float hanya memiliki presisi sampai 15 digit di belakang koma, sementara dengan modul decimal kita bisa mengatur presisi jumlah digit di belakang koma.

Modul ini juga membuat kita bisa melakukan perhitungan seperti di sekolah.



```

File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python - Visual Stud
app.py
1 import decimal
2
3 #output: 0.1
4 print(0.1)
5
6 #output: Decimal('0.100000000000000055511151231257827021181583404541015625')
7 print(decimal.Decimal(0.1))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32,
0.1
0.100000000000000055511151231257827021181583404541015625
PS D:\IMK\Latihan_Python>

```

Kapan Saatnya Menggunakan Decimal Dibanding Float?

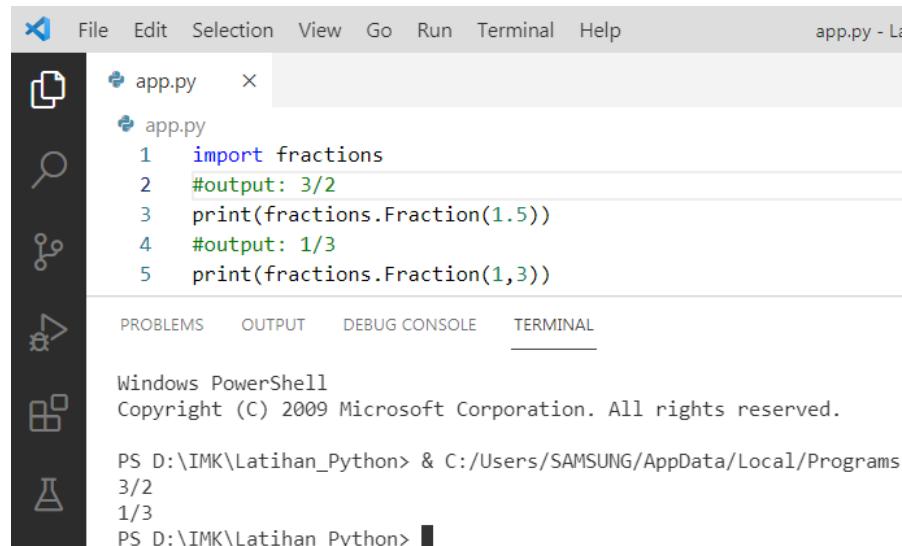
Kita lebih baik menggunakan Decimal dalam kasus:

- Saat kita ingin membuat aplikasi keuangan yang membutuhkan presisi desimal yang pasti
- Saat kita ingin mengontrol tingkat presisi yang diperlukan
- Saat kita ingin menerapkan perkiraan berapa digit decimal yang signifikan

Saat kita ingin melakukan operasi perhitungan sama persis dengan yang kita lakukan di sekolah

### 3.6.3 Bilangan Pecahan

Python menyediakan modul fractions untuk mengoperasikan bilangan pecahan. Pecahan adalah bilangan yang memiliki pembilang dan penyebut, misalnya  $3/2$ . Perhatikan contoh berikut:



The screenshot shows a code editor interface with a dark theme. On the left is a sidebar with icons for file operations, search, and navigation. The main area displays a Python script named 'app.py' with the following code:

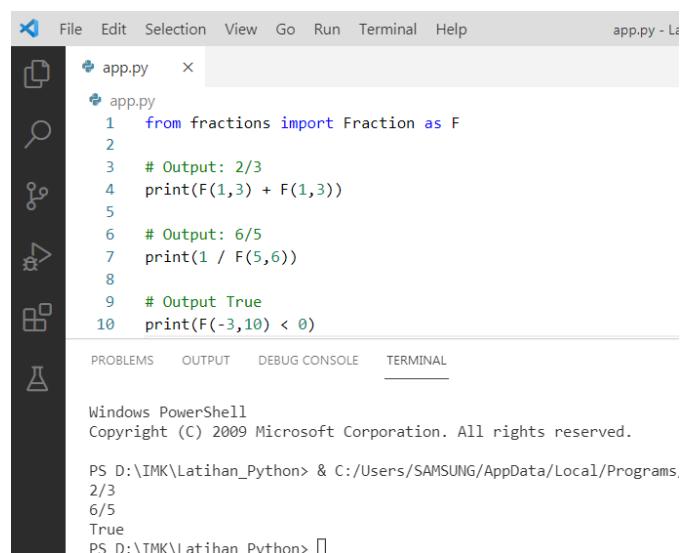
```
File Edit Selection View Go Run Terminal Help app.py - La
app.py x
app.py
1 import fractions
2 #output: 3/2
3 print(fractions.Fraction(1.5))
4 #output: 1/3
5 print(fractions.Fraction(1,3))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
3/2
1/3
PS D:\IMK\Latihan_Python>
```

Operasi dasar seperti penjumlahan atau pembagian pecahan juga bisa dilakukan dengan modul fractions ini



The screenshot shows a code editor interface with a dark theme. On the left is a sidebar with icons for file operations, search, and navigation. The main area displays a Python script named 'app.py' with the following code:

```
File Edit Selection View Go Run Terminal Help app.py - La
app.py x
app.py
1 from fractions import Fraction as F
2
3 # Output: 2/3
4 print(F(1,3) + F(1,3))
5
6 # Output: 6/5
7 print(1 / F(5,6))
8
9 # Output True
10 print(F(-3,10) < 0)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
2/3
6/5
True
PS D:\IMK\Latihan_Python>
```

## 3.7 Matematika dengan Python

Python menyediakan modul math melakukan hal yang berbau matematis seperti trigonometri, logaritma, probabilitas, statistik, dan lain – lain.

The screenshot shows the Visual Studio Code interface. On the left is a dark sidebar with icons for file operations, search, connections, and other tools. The main area has a light background. At the top, a menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. A tab bar on the right shows "app.py - La". The central workspace contains a code editor with the following Python script:

```
app.py
import math
# Output: 3.141592653589793
print(math.pi)
# Output: -1.0
print(math.cos(math.pi))
# Output: 148.4131591025766
print(math.exp(5))
# Output: 2.0
print(math.log10(100))
# Output: 120
print(math.factorial(5))
```

Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, showing the output of the script's execution in a Windows PowerShell window:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
3.141592653589793
-1.0
148.4131591025766
2.0
120
PS D:\IMK\Latihan_Python>
```

Operator adalah simbol tertentu yang digunakan untuk melakukan operasi aritmatika maupun logika. Nilai yang padanya dilakukan operasi disebut operand. Misalnya adalah  $2 + 3$ . Di sini tanda  $+$  adalah operator penjumlahan.  $2$  dan  $3$  adalah operand.

Python memiliki sejumlah operator, yaitu:

- Operator Aritmatika
- Operator Perbandingan
- Operator Penugasan
- Operator Logika
- Operator Bitwise
- Operator Identitas
- Operator Keanggotaan

### 3.7.1 Operator Aritmatika

Operator aritmatika adalah operator yang digunakan untuk melakukan operasi matematika, seperti penjumlahan, pengurangan, perkalian, pembagian, dan sebagainya. Tabel berikut menunjukkan jenis operator aritmatika.

Tabel 3.1 Operator Aritmatika

Operator	Nama dan Fungsi	Contoh
+	Penjumlahan, menjumlahkan 2 buah operand	$a + b$
-	Pengurangan, mengurangkan 2 buah operand	$a - b$
*	Perkalian, mengalikan 2 buah operand	$a * b$
/	Pembagian, membagi 2 buah operand	$a / b$
**	Pemangkatan, memangkatkan bilangan	$a ** b$
//	Pembagian bulat, menghasilkan hasil bagi tanpa koma	$a // b$

```
File Edit Selection View Go Run Terminal Help
app.py

app.py > ...
1 nilai1 = 10
2 nilai2 = 8
3 penjumlahan = nilai1 + nilai2
4 print(nilai1, "+", nilai2, "=", penjumlahan)
5 pengurangan = nilai1 - nilai2
6 print(nilai1, "-", nilai2, "=", pengurangan)
7 perkalian = nilai1 * nilai2
8 print(nilai1, "x", nilai2, "=", perkalian)
9 pembagian = nilai1 / nilai2
10 print(nilai1, "/", nilai2, "=", pembagian)
11 pemangkatan = nilai1 ** nilai2
12 print(nilai1, "**", nilai2, "=", pemangkatan)
13 pembagian_bulat = nilai1 // nilai2
14 print(nilai1, "//", nilai2, "=", pembagian_bulat)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
10 + 8 = 18
10 - 8 = 2
10 x 8 = 80
10 / 8 = 1.25
10 ** 8 = 100000000
10 // 8 = 1
PS D:\IMK\Latihan_Python>
```

Kita juga bisa menentukan suatu variabel sesuai dengan keinginan kita dengan menggunakan fungsi **input()** seperti contoh berikut :

```

File Edit Selection View Go Run Terminal Help
app.py - Lati
app.py > ...
1 nilai1 = int(input("Masukan Nilai Pertama : "))
2 nilai2 = int(input("Masukan Nilai Kedua : "))
3 penjumlahan = nilai1 + nilai2
4 print(nilai1, "+", nilai2, "=", penjumlahan)
5 pengurangan = nilai1 - nilai2
6 print(nilai1, "-", nilai2, "=", pengurangan)
7 perkalian = nilai1 * nilai2
8 print(nilai1, "x", nilai2, "=", perkalian)
9 pembagian = nilai1 / nilai2
10 print(nilai1, "/", nilai2, "=", pembagian)
11 pemangkatan = nilai1 ** nilai2
12 print(nilai1, "**", nilai2, "=", pemangkatan)
13 pembagian_bulat = nilai1 // nilai2
14 print(nilai1, "//", nilai2, "=", pembagian_bulat)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python,
Masukan Nilai Pertama : 8
Masukan Nilai Kedua : 7
8 + 7 = 15
8 - 7 = 1
8 x 7 = 56
8 / 7 = 1.1428571428571428
8 ** 7 = 2097152
8 // 7 = 1
PS D:\IMK\Latihan_Python>

```

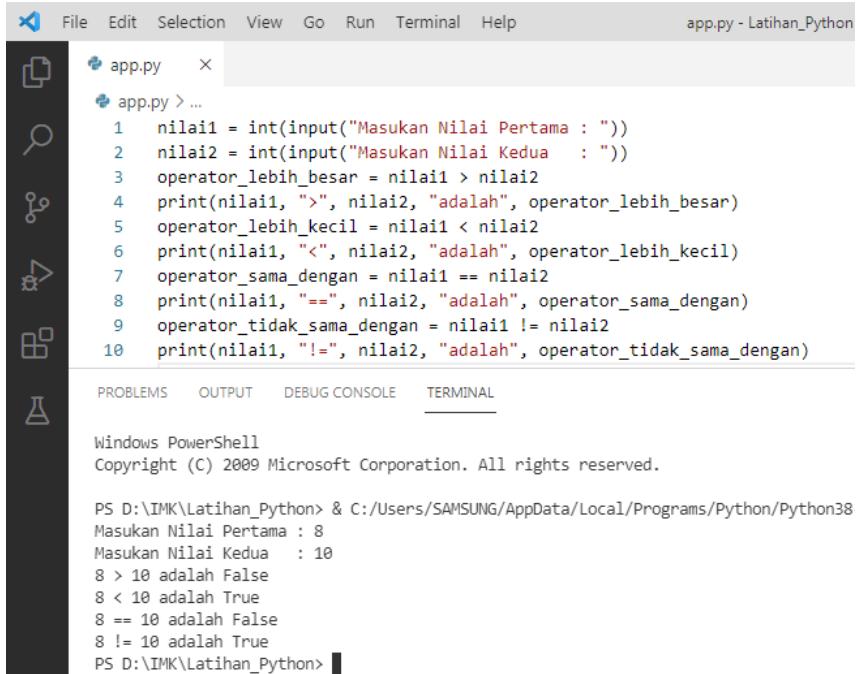
### 3.7.2 Operator Perbandingan

Operator perbandingan adalah operator yang digunakan untuk membandingkan 2 buah nilai. Hasil perbandingannya adalah True atau False tergantung kondisi.

Tabel 3.2 Operator Perbandingan

Operator	Nama dan Fungsi	Contoh
>	Lebih besar dari – Hasilnya True jika nilai sebelah kiri lebih besar dari nilai sebelah kanan	a > b
<	Lebih kecil dari – Hasilnya True jika nilai sebelah kiri lebih kecil dari nilai sebelah kanan	a < b
==	Sama dengan – Hasilnya True jika nilai sebelah kiri sama dengan nilai sebelah kanan	a == b
!=	Tidak sama dengan – Hasilnya True jika nilai sebelah kiri tidak sama dengan nilai sebelah kanan	a != b

$\geq$	Lebih besar atau sama dengan – Hasilnya True jika nilai sebelah kiri lebih besar atau sama dengan nilai sebelah kanan	$a \geq b$
$\leq$	Lebih kecil atau sama dengan – Hasilnya True jika nilai sebelah kiri lebih kecil atau sama dengan nilai sebelah kanan	$a \leq b$



```

File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python
app.py > ...
1 nilai1 = int(input("Masukan Nilai Pertama : "))
2 nilai2 = int(input("Masukan Nilai Kedua : "))
3 operator_lebih_besar = nilai1 > nilai2
4 print(nilai1, ">", nilai2, "adalah", operator_lebih_besar)
5 operator_lebih_kecil = nilai1 < nilai2
6 print(nilai1, "<", nilai2, "adalah", operator_lebih_kecil)
7 operator_sama_dengan = nilai1 == nilai2
8 print(nilai1, "==", nilai2, "adalah", operator_sama_dengan)
9 operator_tidak_sama_dengan = nilai1 != nilai2
10 print(nilai1, "!=" , nilai2, "adalah", operator_tidak_sama_dengan)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38
Masukan Nilai Pertama : 8
Masukan Nilai Kedua : 10
8 > 10 adalah False
8 < 10 adalah True
8 == 10 adalah False
8 != 10 adalah True
PS D:\IMK\Latihan_Python>

```

### 3.7.3 Operator Penugasan

Operator penugasan adalah operator yang digunakan untuk memberi nilai ke variabel.  $a = 7$  adalah contoh operator penugasan yang memberi nilai 7 di kanan ke variabel  $a$  yang ada di kiri.

Tabel 3.3 Operator Penugasan

Operator	Penjelasan	Contoh
$=$	Menugaskan nilai yang ada di kanan ke operand yang ada di sebelah kiri	$c = a + b$ menugaskan $a + b$ ke $c$
$+=$	Menambahkan operand yang di kanan dengan operand yang ada di kiri dan hasilnya ditugaskan ke operand yang di kiri	$c += a$ sama dengan $c = c + a$

<code>-=</code>	Mengurangi operand yang di kanan dengan operand yang ada di kiri dan hasilnya di tugaskan ke operand yang di kiri	$c -= a$ sama dengan $c = c + a$
<code>*=</code>	Mengalikan operand yang di kanan dengan operand yang ada di kiri dan hasilnya di tugaskan ke operand yang di kiri	$c *= a$ sama dengan $c = c * a$
<code>/=</code>	Membagi operand yang di kanan dengan operand yang ada di kiri dan hasilnya di tugaskan ke operand yang di kiri	$c /= a$ sama dengan $c = c * a$
<code>**=</code>	Memangkatkan operand yang di kanan dengan operand yang ada di kiri dan hasilnya ditugaskan ke operand yang di kiri	$c **= a$ sama dengan $c = c ** a$
<code>//=</code>	Melakukan pembagian bulat operand di kanan terhadap operand di kiri dan hasilnya disimpan di operand yang di kiri	$c // a$ sama dengan $c = c // a$
<code>%=</code>	Melakukan operasi sisa bagi operand di kanan dengan operand di kiri dan hasilnya di simpan di operand yang di kiri	$c \% a$ sama dengan $c = c \% a$

### 3.7.4 Operator Logika

Operator logika adalah operator yang digunakan untuk melakukan operasi logika

Tabel 3.4 Operator Logika

Operator	Penjelasan	Contoh
<code>and</code>	Hasilnya adalah True jika kedua operandnya bernilai benar	<code>a and b</code>
<code>or</code>	Hasilnya adalah True jika salah satu atau kedua operandnya bernilai benar	<code>a or b</code>

not	Hasilnya adalah True jika operandnya bernilai salah (kebalikan nilai)	not a
-----	-----------------------------------------------------------------------	-------

### 3.7.5 Operator Bitwise

Operator bitwise adalah operator yang melakukan operasi bit terhadap operand. Operator ini beroperasi bit per bit sesuai dengan namanya. Sebagai misal, angka 2 dalam bit ditulis 10 dalam notasi biner dan angka 7 ditulis 111. Pada tabel di bawah ini, misalkan  $a = 10$  (0000 1010) dalam biner dan  $b = 4$  (0000 0100) dalam biner.

Tabel 3.5 Operator Bitwise

Operator	Nama	Contoh
&	Bitwise AND	$a \& b = 0$ (0000 0000)
	Bitwise OR	$a   b = 14$ (0000 1110)
~	Bitwise NOT	$\sim a = -11$ (1111 0101)
^	Bitwise XOR	$a ^ b = 14$ (0000 1110)
>>	Bitwise right shift	$a >> 2 = 2$ (0000 0010)
<<	Bitwise left shift	$b << 2 = 40$ (0010 1000)

### 3.7.6 Operator Identitas

Operator identitas adalah operator yang memeriksa apakah dua buah nilai ( atau variabel ) berada pada lokasi memori yang sama.

Tabel 3.6 Operator Identitas

Operator	Penjelasan	Contoh

is	True jika kedua operand identik (menunjuk ke objek yang sama)	a is True
is not	True jika kedua operand tidak identik (tidak merujuk ke objek yang sama)	a is not True

### 3.7.7 Operator Keanggotaan

Operator keanggotaan adalah operator yang digunakan untuk memeriksa apakah suatu nilai atau variabel merupakan anggota atau ditemukan di dalam suatu data (string, list, tuple, set, dan dictionary)

Tabel 3.7 Operator Keanggotaan

Operator	Penjelasan	Contoh
in	True jika nilai/variabel ditemukan di dalam data	5 in a
not in	True jika nilai/variabel tidak ada di dalam data	5 not in a

#### 3.1. Latihan 1

- Buatlah 1 Contoh Operator Penugasan
- Buatlah 1 Contoh Operator Logika
- Buatlah 1 Contoh Operator Bitwise
- Buatlah 1 Contoh Operator Identitas
- Buatlah 1 Contoh Operator Keanggotaan

#### 3.2. Latihan 2

- Buatlah program seperti gambar dibawah ini

```
Run: Tugas01 x
E:\projek_pyhton\Dasar_Pemrograman\venv\Scripts\python.exe E:/proj
TOKO MAINAN ANAK
*****
Masukan Nama Pembeli : Ilham Kurniawan
Masukan Kode Mainan : MAIN-0909
Masukan Harga : 30000
Masukan Jumlah Beli : 10
=====
Nama Pembeli = Ilham Kurniawan
Kode Kue      = MAIN-0909
Harga         = 30000
Jumlah Beli   = 10
Total          = 300000
```

## Pertemuan 4

### Percabangan

Percabangan adalah cara yang digunakan untuk mengambil keputusan apabila di dalam program dihadapkan pada kondisi tertentu. Jumlah kondisinya bisa satu, dua atau lebih. Percabangan mengevaluasi kondisi atau ekspresi yang hasilnya benar atau salah. Kondisi atau ekspresi tersebut disebut ekspresi boolean.

Hasil dari pengecekan kondisi adalah True atau False. Bila benar (True), maka pernyataan yang ada di dalam blok kondisi tersebut akan dieksekusi. Bila salah (False), maka blok pernyataan lain yang dieksekusi.

Di Python ada 3 jenis pernyataan yang digunakan untuk percabangan, yaitu sebagai berikut:

Tabel 4.1 Tabel Percabangan

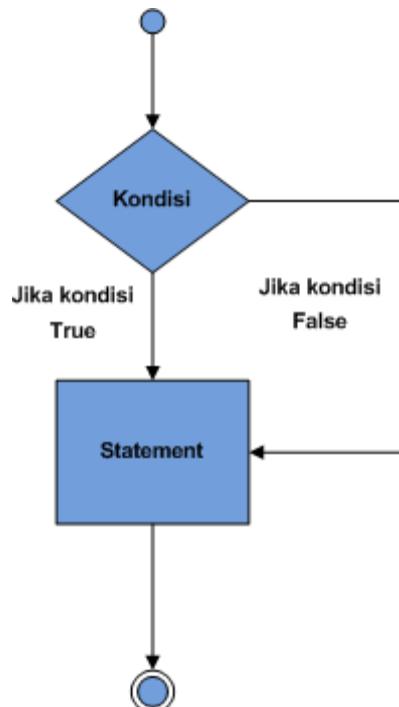
No	Pernyataan	Deskripsi
1	if	Pernyataan <b>if</b> terdiri dari ekspresi <i>boolean</i> diikuti oleh satu baris atau lebih pernyataan.
2	if...else	Bila pernyataan <b>if</b> benar, maka blok pernyataan <b>if</b> dieksekusi. Bila salah, maka blok pernyataan <b>else</b> yang dieksekusi.
3	if...elif...else	Disebut juga if bercabang. Bila ada kemungkinan beberapa kondisi bisa benar maka digunakan pernyataan <b>if...elif</b> atau <b>if...elif...else</b>

#### 4.1 Pernyataan if

Pernyataan if menguji satu buah kondisi. Bila hasilnya benar maka pernyataan di dalam blok if tersebut dieksekusi. Bila salah, maka pernyataan tidak dieksekusi. Sintaksnya adalah seperti berikut:

```
if tes kondisi:  
    blok pernyataan if
```

Gambar diagram alir untuk pernyataan if adalah seperti berikut:



Gambar 4.1 Diagram Alir Pernyataan if

```

app.py - Lati
app.py > ...
1 # Bila bilangan positif, tampilkan pesan
2
3 angka = 5
4 if angka > 0:
5     print(angka, "adalah bilangan positif.")
6
7 angka = -1
8 # yang berikut akan bernilai False sehingga tidak dieksekusi
9 if angka > 0:
10    print(angka, "adalah bilangan positif.")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python,
5 adalah bilangan positif.
PS D:\IMK\Latihan_Python>
  
```

Pada contoh di atas, awalnya angka berisi 5. Pada saat if yang pertama dieksekusi maka kondisinya adalah apakah  $5 > 0$ ? Karena hasilnya benar/True, maka statement di grup if ini dieksekusi dan menampilkan pesan 5 adalah bilangan positif.

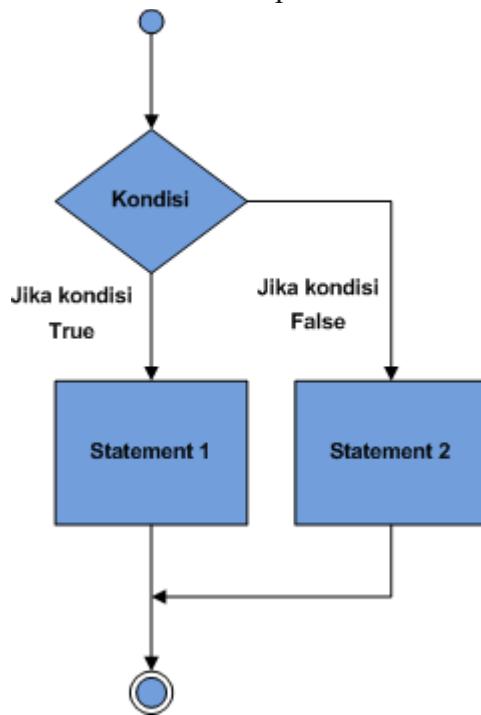
Selanjutnya angka sudah diubah jadi -1. Untuk if yang kedua, hasil pengujian kondisinya menjadi apakah  $-1 > 0$ ? Hasilnya salah/False. Oleh karena itu, pernyataan di dalam grupnya tidak dijalankan.

### 5.1. Pernyataan if...else

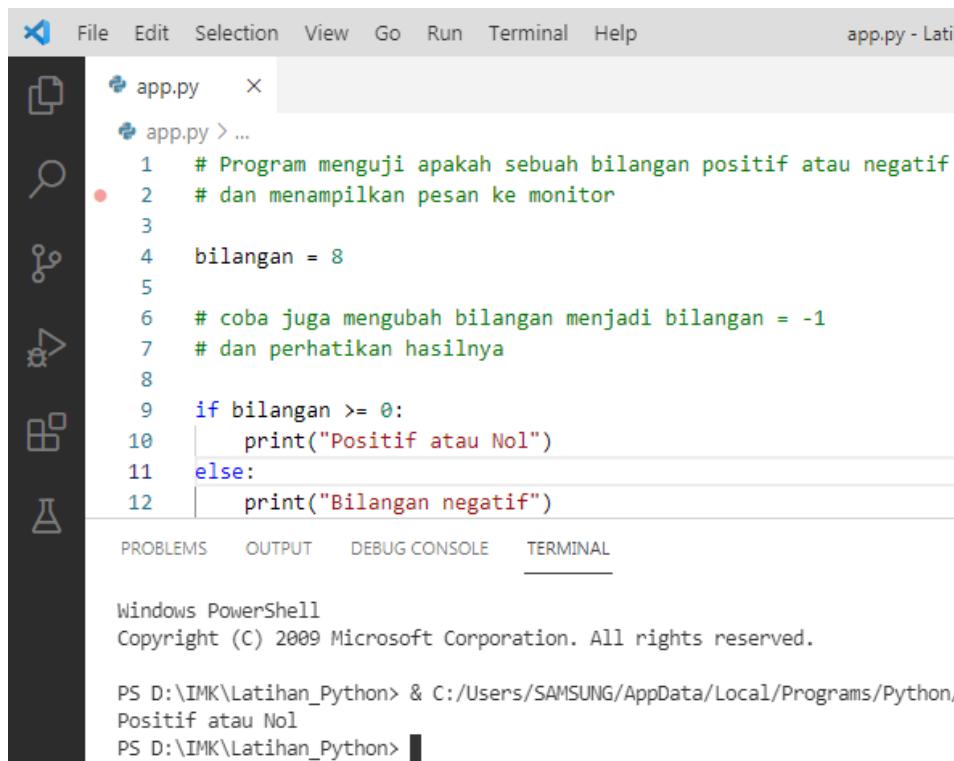
Pernyataan if...else menguji 2 kondisi. Kondisi pertama kalau benar, dan kondisi kedua kalau salah. Sintaksnya adalah seperti berikut:

```
if tes kondisi:  
    blok pernyataan if  
  
else:  
    blok pernyataan else
```

Diagram alir untuk pernyataan if...else adalah seperti berikut:



Gambar 4.2. Diagram Alir Pernyataan if...else



The screenshot shows a Python code editor interface. On the left is a dark sidebar with icons for file operations like open, save, search, and run. The main area has a tab bar with 'app.py' and a status bar indicating 'app.py - Lati'. The code editor displays the following Python script:

```
1 # Program menguji apakah sebuah bilangan positif atau negatif
2 # dan menampilkan pesan ke monitor
3
4 bilangan = 8
5
6 # coba juga mengubah bilangan menjadi bilangan = -1
7 # dan perhatikan hasilnya
8
9 if bilangan >= 0:
10     print("Positif atau Nol")
11 else:
12     print("Bilangan negatif")
```

Below the code editor are tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL'. The 'TERMINAL' tab is active, showing a Windows PowerShell window with the following output:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python,
Positif atau Nol
PS D:\IMK\Latihan_Python>
```

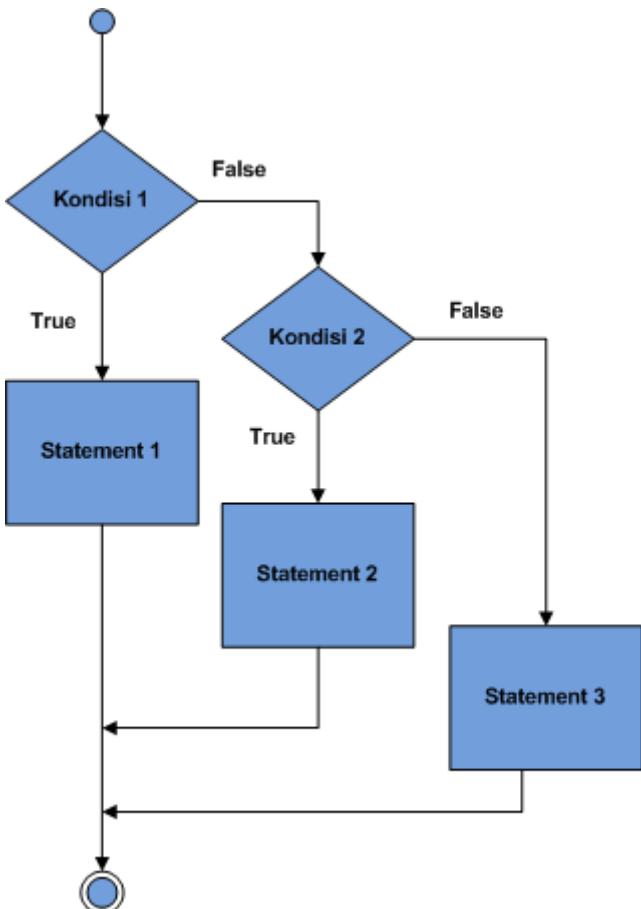
Pada contoh di atas, bilangan kita beri nilai 8. Kemudian pada pengujian if, kondisinya adalah apakah bilangan  $\geq 0$ ? Hasilnya adalah benar, maka hasil yang ditampilkan adalah Positif atau Nol. Seandainya kita ganti bilangan jadi -1, maka hasil pengujian if nya akan salah/False dan blok else yang akan dijalankan, yaitu menampilkan pesan Bilangan negatif.

## 4.2 Pernyataan if...elif...else...

Pernyataan if...elif...else digunakan untuk menguji lebih dari 2 kondisi. Bila kondisi pada if benar, maka pernyataan di dalamnya yang dieksekusi. Bila salah, maka masuk ke pengujian kondisi elif. Terakhir bila tidak ada if atau elif yang benar, maka yang dijalankan adalah yang di blok else. Sintaksnya adalah seperti berikut:

```
if tes kondisi:
    blok pernyataan if
elif tes kondisi:
    blok pernyataan elif
else:
    blok pernyataan else
```

Diagram alir if...else...if adalah sebagai berikut:



Gambar 4.3 Diagram Alir Pernyataan if...elif...else

```

File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python - Visual Studio Code [Administrator]

app.py > ...
1 '''Di sini kita menguji apakah sebuah bilangan adalah bilangan positif, nol, atau negatif -
2 dan menampilkan hasilnya ke layar '''
3
4 bilangan = 5.5
5
6 '''Coba juga mengganti bilangan jadi
7 | bilangan = 0
8 | bilangan = -5.5 '''
9
10 if bilangan > 0:
11     print("Bilangan positif")
12 elif bilangan == 0:
13     print("Nol")
14 else:
15     print("Bilangan negatif")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe d:/IMK/Latihan_Python/
Bilangan positif
PS D:\IMK\Latihan_Python>

```

Pada contoh di atas, bilangan kita beri nilai 5.5. Pada pengujian if, kondisinya adalah apakah bilangan  $> 0$ ? Hasilnya benar, maka yang ditampilkan adalah pesan Bilangan positif.

Bila nilai bilangan kita ganti menjadi 0, maka yang akan bernilai benar adalah pernyataan elif. Bila kita mengganti bilangan jadi minus, maka kondisi if dan elif salah, dan yang dijalankan adalah blok else.

Catatan: Python mengasumsikan bahwa nilai selain nol dan selain tipe None sebagai nilai True, dan yang nilai nol dan None sebagai False.

The screenshot shows the Visual Studio Code interface. The top window displays the Python code for calculating shirt prices based on brand and size. The bottom window shows the terminal output where the user inputs 'SP' and 'S' respectively, and the program outputs the brand 'SuperDry' and a price of 450000.

```
File Edit Selection View Go Run Terminal Help
app.py - L

app.py > ...
1  kode_baju = input("Masukan Kode Baju [SP/AD] : ")
2  ukuran = input("Masukan Ukuran Baju [S/M] : ")
3
4  if kode_baju == "SP" or kode_baju == "sp" :
5      merk = "SuperDry"
6      if ukuran=="S" or ukuran=="s":
7          harga = 450000
8      elif ukuran=="M" or ukuran=="m":
9          harga = 500000
10     else:
11         harga = 0
12 elif kode_baju == "AD" or kode_baju == "ad" :
13     merk = "Adidas"
14     if ukuran=="s" or ukuran=="S":
15         harga = 650000
16     elif ukuran=="M" or ukuran=="m":
17         harga = 700000
18     else:
19         harga = 0
20
21 else:
22     merk = "Anda Salah Input Kode Merk"
23     harga = 0
24
25 print("-----")
26 print("Merk Baju : "+str(merk))
27 print("Harga Baju : Rp.",harga)

File Edit Selection View Go Run Terminal Help
app.py - L

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python
Masukan Kode Baju [SP/AD] : SP
Masukan Ukuran Baju [S/M] : s
-----
Merk Baju : SuperDry
Harga Baju : Rp. 450000
PS D:\IMK\Latihan_Python>
```

### 4.3 Studi Kasus : Penjualan Tiket

Perusahaan XYZ bergerak dibidang penjualan tiket bus dengan detail tiket sebagai berikut :

1. setiap transaksi perlu menginput data pembeli seperti Nama Pembeli, jumlah tiket yang akan dibeli, no\_Hp, dan memilih jurusan sesuai kategori yang diinginkan.
2. Potongan 10% didapat jika jumlah beli  $\geq 3$ .

3. Totalharga=(jumlahbeli\*harga)-potongan.

Data harga tiket sebagai berikut :

Kodejurusan	Nama Kota	Harga
SBY	surabaya	300,000
BL	Bali	350,000
LMP	lampung	500,000

Buatlah program input dan hasil output sesuai perintah diatas menggunakan Bahasa pemrograman Python.

The screenshot shows a terminal window with two panes: 'Masukan' (Input) and 'Keluaran' (Output).

**Masukan (Input):**

```
"E:\OneDrive - Bina Sarana Informasi\penjualantiket.py"
Input Nama Pembeli : sopandi
Input No. Handphone : 082213445567
Input Jurusan [SBY/BL/LMP] : SBY
Masukkan Jumlah Beli : 3
```

**Keluaran (Output):**

```
PENJUALAN TIKET BUS
XYZ
-----
Nama Pembeli : sopandi
No. Handphone : 082213445567
Kode Jurusan yang dipilih : SBY
Nama Kota Tujuan : Surabaya
Harga : 300000
Jumlah Beli : 3
-----
potongan yang didapat : 90000.0
Total Bayar : 810000.0
Masukkan Uang Bayar : 1000000
Uang Kembali : 190000.0
-----
Process finished with exit code 0
```

```

#Input
pembeli=input("Input Nama Pembeli : ")
no_hp=input("Input No. Handphone : ")
jurusan=input("Input Jurusan [SBY/BL/LMP] : ")
#Proses
if jurusan=="SBY":
    namajurusan="Surabaya"
    harga=300000
elif jurusan=="BL":
    namajurusan="Bali"
    harga=350000
else :
    namajurusan="Lampung"
    harga=500000

#Input Jumlah Beli
jumlah=int(input("Masukkan Jumlah Beli : "))

#proses potongan
if jumlah>=3 :
    potongan=(jumlah*harga)*0.1
else:
    potongan=0

total=(jumlah*harga)-potongan

#Cetak Hasil
print("-----")
print("          PENJUALAN TIKET BUS")
print("          XYZ")
print("-----")
print("Nama Pembeli : "+str(pembeli))
print("No. Handphone : "+str(no_hp))
print("Kode Jurusan yang dipilih : "+str(jurusan))
print("Nama Kota Tujuan : "+str(namajurusan))
print("Harga : ",+(harga))
print("Jumlah Beli : ",+(jumlah))
print("-----")
print("potongan yang didapat : ",+(potongan))
print("Total Bayar : ",+(total))
ubay=int(input("Masukkan Uang Bayar : "))
uangkembali=ubay-total
print("Uang Kembali : ",+uangkembali)

```

1. setiap transaksi perlu menginput data calon mahasiswa seperti NIS,nama,jurusan
2. Data biaya kuliah sebagai berikut :

Kodejurusan	Nama Prodi	Harga
SI	Sistem Informasi	2,400,000
SIA	Sistem Informasi AKuntansi	2,000,000

Buatlah program input dan hasil output sesuai perintah diatas menggunakan Bahasa pemrograman Python.

### Tugas 1

PT. DINGIN DAMAI, memberi gaji pokok kepada karyawan kontraknya sebesar Rp. 300,000 perbulan, dengan memperoleh tunjangan-tunjangan sebagai berikut :

- Tunjangan Jabatan

Golongan	Persentase
1	5%
2	10%
3	15%

Logikanya : Jika seorang karyawan tersebut dengan golongan 3, maka mendapatkan tunjangan sebesar  $15\% * \text{Rp. } 300,000$

- Tunjangan Pendidikan

Tingkat Pendidikan	Persentase
SMA	2.5%
D1	5%
D3	20%
S1	30%

Jika seorang karyawan tersebut dengan Tingkat Pendidikan S1, maka mendapatkan tunjangan pendidikan sebesar  $30\% * \text{Rp. } 300,000$

#### Honor Lembur

Jumlah jam kerja normal sebanyak 8 jam, Honor lembur diberikan jika jumlah jam kerja lebih dari 8 jam, maka kelebihan jam kerja tersebut dikalikan dengan Rp. 3500 untuk setiap kelebihan jam kerja karyawan tersebut.Tampilan yang diinginkan sebagai berikut :

## Layar Masukkan

### PROGRAM HITUNG GAJI KARYAWAN

Nama Karyawan: ...

Golongan Jabatan : ...

Pendidikan : ...

Jumlah jam kerja : ...

## Layar Keluaran

Karyawan yang bernama .....

Honor yang diterima

Tunjangan Jabatan Rp ...

Tunjangan Pendidikan Rp ...

Honor Lembur Rp .....  
\_\_\_\_\_ +

Rp ...

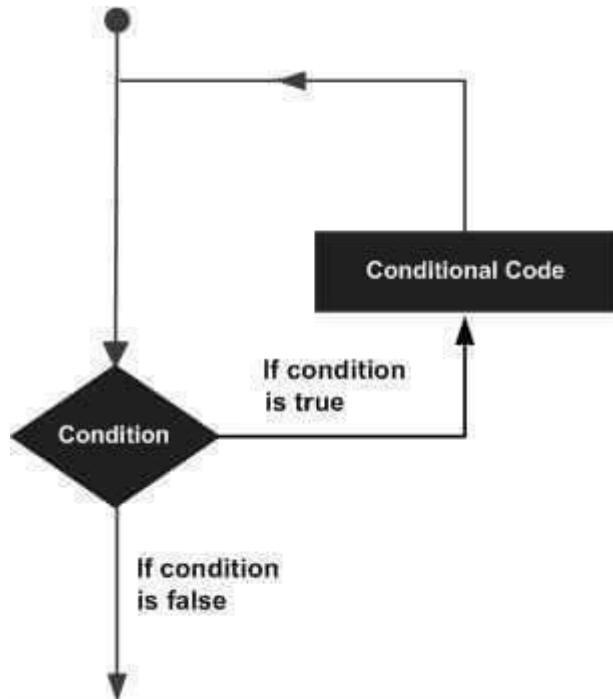
Total Gaji

(Gaji pokok + tunjangan + lembur)

## Pertemuan 5

### Perulangan

Secara umum, Python mengeksekusi program baris perbaris. Mulai dari baris satu, dua, dan seterusnya. Ada kalanya, kita perlu mengeksekusi satu baris atau satu blok kode program beberapa kali. Hal ini disebut dengan perulangan atau biasa disebut looping atau iterasi. Untuk lebih jelasnya perhatikan gambar berikut:



Gambar 5.1 Diagram Alir Perulangan (looping)

Pada gambar bisa dilihat bahwa perulangan juga memerlukan tes kondisi. Bila hasil tes kondisi True, maka blok kode kembali dieksekusi. Tapi jika False, maka keluar dari perulangan. Pada python, perulangan bisa dilakukan dengan dua cara atau metode, yaitu:

1. Menggunakan for
2. Menggunakan while

#### 5.1 Perulangan dengan Menggunakan For

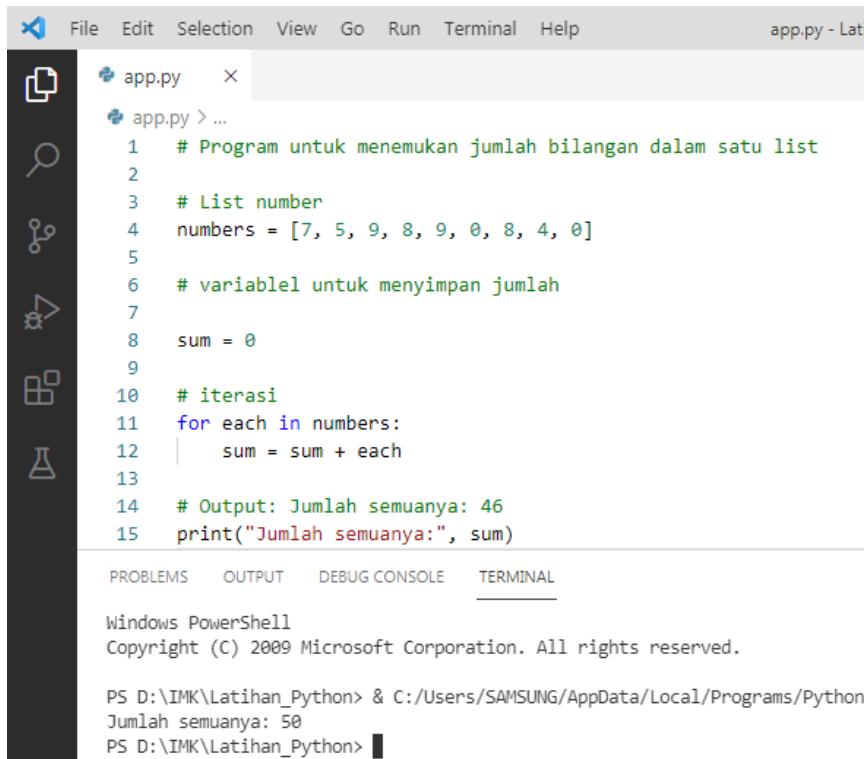
Perulangan dengan menggunakan for memiliki sintaks seperti berikut:

```
for var in sequence:
```

```
    body of for
```

var adalah variabel yang digunakan untuk penampung sementara nilai dari sequence pada saat terjadi perulangan. Sequence adalah tipe data berurut seperti string, list, dan tuple.

Perulangan terjadi sampai looping mencapai elemen atau anggota terakhir dari sequence. Bila loop sudah sampai ke elemen terakhir dari sequence, maka program akan keluar dari looping.



```
File Edit Selection View Go Run Terminal Help app.py - Latihan_Python
app.py    x
app.py > ...
1  # Program untuk menemukan jumlah bilangan dalam satu list
2
3  # List number
4  numbers = [7, 5, 9, 8, 9, 0, 8, 4, 0]
5
6  # variabel untuk menyimpan jumlah
7
8  sum = 0
9
10 # iterasi
11 for each in numbers:
12     sum = sum + each
13
14 # Output: Jumlah semuanya: 46
15 print("Jumlah semuanya:", sum)

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Jumlah semuanya: 50
PS D:\IMK\Latihan_Python>
```

### 5.1.1 Fungsi range

Fungsi range() dapat digunakan untuk menghasilkan deret bilangan. range(10) akan menghasilkan bilangan dari 0 sampai dengan 9 (10 bilangan). Kita juga bisa menentukan batas bawah, batas atas, dan interval dengan format range(batas bawah, batas atas, interval). Bila interval dikosongkan, maka nilai default 1 yang akan digunakan.

Fungsi range tidak menyimpan semua nilai dalam memori secara langsung. Ia hanya akan mengingat batas bawah, batas atas, dan interval dan membungkitkan hasilnya satu persatu hanya bila dipanggil. Untuk membuat fungsi ini langsung menampilkan semua item, kita bisa menggunakan fungsi list(). Untuk jelasnya perhatikan contoh berikut:

```
# Output: range(0,10)
print(range(10))
```

```
# Output: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
print(list(range(10)))
```

```
# Output: [2, 3, 4, 5, 6, 7]
print(list(range(2,8)))
```

```
# Output: [2, 5, 8, 11, 14, 17]
print(list(range(2, 20, 3)))
```

Kita bisa menggunakan fungsi range() dalam perulangan menggunakan for untuk iterasi bilangan berurut. Hal ini dengan cara mengkombinasikan fungsi range() dengan fungsi len(). Fungsi len() berfungsi untuk mendapatkan panjang atau jumlah elemen suatu data sekuensial atau berurut.

```
File Edit Selection View Go Run Terminal Help app.py - La
app.py > ...
1 # Program untuk iterasi list menggunakan pengindeksan
2
3 mapel = ['matematika', 'fisika', 'kimia']
4
5 # iterasi list menggunakan indeks
6 for i in range(len(mapel)):
7     print("Saya suka", mapel[i])
8

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

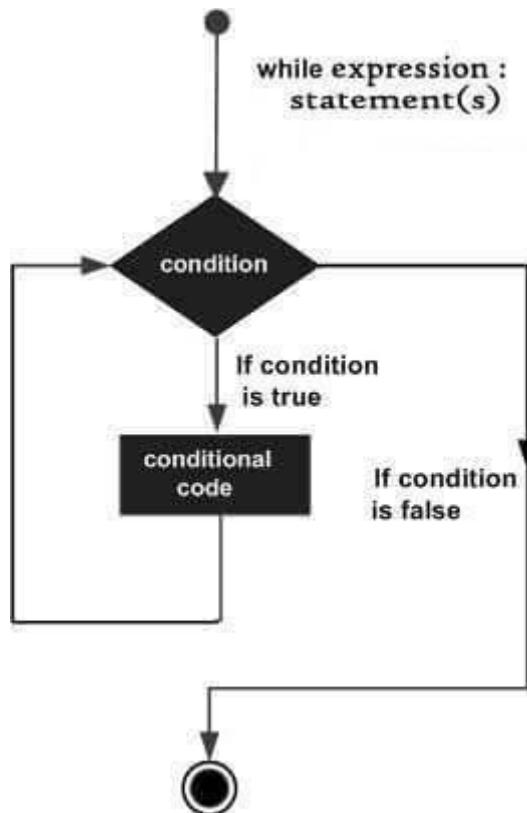
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Saya suka matematika
Saya suka fisika
Saya suka kimia
PS D:\IMK\Latihan_Python>
```

### 5.1.2 Perulangan Menggunakan While

Perulangan menggunakan while akan menjalankan blok pernyataan terus menerus selama kondisi bernilai benar. Adapun sintaks dari perulangan menggunakan while adalah:

```
while expression:
    statement (s)
```

Di sini, statement (s) bisa terdiri dari satu baris atau satu blok pernyataan. Expression merupakan ekspresi atau kondisi apa saja, dan untuk nilai selain nol dianggap True. Iterasi akan terus berlanjut selama kondisi benar. Bila kondisi salah, maka program akan keluar dari while dan lanjut ke baris pernyataan di luar while. Adapun diagram alir while adalah seperti gambar berikut:



Gambar 5.2. Diagram Alir Perulangan While

Perhatikan bahwa bila kondisi yang diuji bernilai salah, maka loop tidak akan pernah dieksekusi.

A screenshot of a Python IDE (Visual Studio Code) showing the execution of a while loop. The code in the editor is:

```

app.py > ...
1 count = 0
2 while (count < 5):
3     print('The count is:', count)
4     count = count + 1
5 print('Good bye!')
6

```

The terminal window shows the output of the script:

```

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
The count is: 0
The count is: 1
The count is: 2
The count is: 3
The count is: 4
Good bye!
PS D:\IMK\Latihan_Python>

```

Di sini, blok pernyataan `print('The count is:', count)`, dijalankan terus selama `count` masih lebih kecil dari 5. `Count` ditambah 1 setiap kali iterasi. Pada saat nilai `count` mencapai 5, maka

kondisi menjadi False dan program keluar dari looping while dan melanjutkan baris selanjutnya yaitu print("Good bye").

### 5.1.3 Infinite Loop

Sebuah kondisi dimana loop selalu benar dan tidak pernah salah disebut loop tidak terbatas (infinite loop). Terkadang hal ini menjadi masalah. Tapi sering juga infinite loop berguna, misalnya untuk program client/server dimana server perlu menjaga komunikasi tetap hidup dan tidak terputus. Pada contoh program while di atas, bila kita lupa menuliskan kode `count = count + 1`, maka akan jadi infinite loop. Hasilnya akan jadi seperti berikut:

Kita perlu menekan CTRL+C untuk menghentikan program.

### 5.1.4 Kendali Looping

Looping umumnya akan berhenti bila kondisi sudah bernilai salah. Akan tetapi, seringkali kita perlu keluar dari looping di tengah jalan tergantung keperluan. Hal ini bisa kita lakukan dengan menggunakan kata kunci **break** dan **continue**.

Statement break memaksa program keluar dari blok looping di tengah jalan. Sedangkan statement continue menyebabkan program langsung melanjut ke step / interval berikutnya dan mengabaikan (skip) baris kode di bawahnya (yang satu blok). Jelasnya perhatikan contoh berikut:

The screenshot shows a code editor interface with a dark theme. On the left is a vertical toolbar with icons for file operations, search, and other development tools. The main area has a tab for 'app.py' which contains the following Python code:

```
1 # contoh penggunaan statement break
2 for letter in "PythonProgramming":
3     if letter == "g":
4         break
5     print("Huruf sekarang:", letter)
6 print("Good bye")
```

Below the code editor is a terminal window titled 'Windows PowerShell'. It displays the output of running the script, showing it prints every character except 'g':

```
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Huruf sekarang: P
Huruf sekarang: y
Huruf sekarang: t
Huruf sekarang: h
Huruf sekarang: o
Huruf sekarang: n
Huruf sekarang: P
Huruf sekarang: r
Huruf sekarang: o
Good bye
PS D:\IMK\Latihan_Python>
```

Bila pada program di atas kita ganti kode **break** menjadi **continue**, maka hasilnya akan jadi seperti berikut:

The screenshot shows a code editor interface with a dark theme, similar to the one above. The 'app.py' file now contains this Python code:

```
1 # contoh penggunaan statement break
2 for letter in "PythonProgramming":
3     if letter == "g":
4         continue
5     print("Huruf sekarang:", letter)
6 print("Good bye")
```

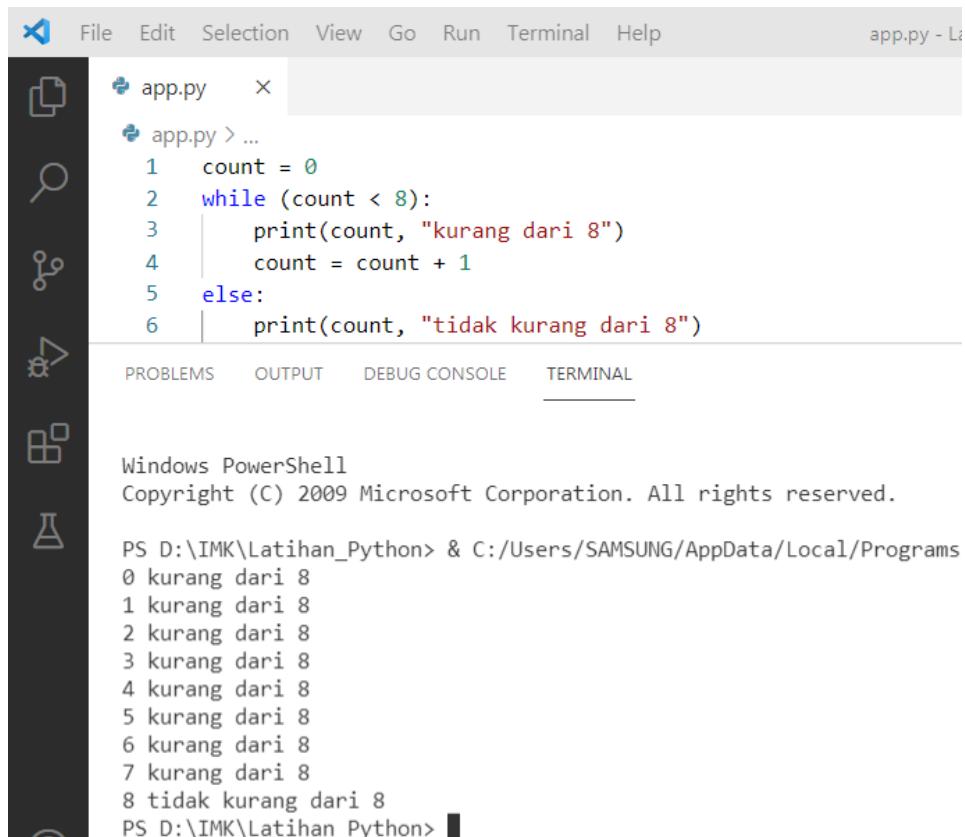
The terminal window below shows the output of running the script. Since the 'g' character is skipped due to the 'continue' statement, only the letters 'P', 'y', 't', 'h', 'o', 'n', 'P', 'r', 'o', 'g', 'r', 'a', 'm', 'm' are printed.

```
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Huruf sekarang: P
Huruf sekarang: y
Huruf sekarang: t
Huruf sekarang: h
Huruf sekarang: o
Huruf sekarang: n
Huruf sekarang: P
Huruf sekarang: r
Huruf sekarang: o
Huruf sekarang: g
Huruf sekarang: r
Huruf sekarang: a
Huruf sekarang: m
Huruf sekarang: m
Huruf sekarang: i
Huruf sekarang: n
Good bye
PS D:\IMK\Latihan_Python>
```

Perhatikan bahwa huruf **g** tidak pernah ditampilkan karena diabaikan karena kode **continue**.

## 5.2 While else

Python mendukung penggunaan else sebagai pasangan dari while. Blok pernyataan else hanya akan dieksekusi bila kondisi while bernilai salah.



The screenshot shows the Visual Studio Code interface. On the left is a dark sidebar with icons for file operations, search, and other development tools. The main area has a tab bar with 'app.py' and a preview of the code. Below the tabs are four buttons: PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, displaying the following text:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
0 kurang dari 8
1 kurang dari 8
2 kurang dari 8
3 kurang dari 8
4 kurang dari 8
5 kurang dari 8
6 kurang dari 8
7 kurang dari 8
8 tidak kurang dari 8
PS D:\IMK\Latihan_Python>
```

## Latihan

Lakukan pengulangan input data sebanyak 2 kali dengan data dibawah ini :

Data Ke- <berulang>

Masukkan NIM anda : <Input Data Ke 1>

Masukkan Nilai UTS : <Input Data Ke 1>

Masukkan Nilai UAS : <Input Data Ke 1>

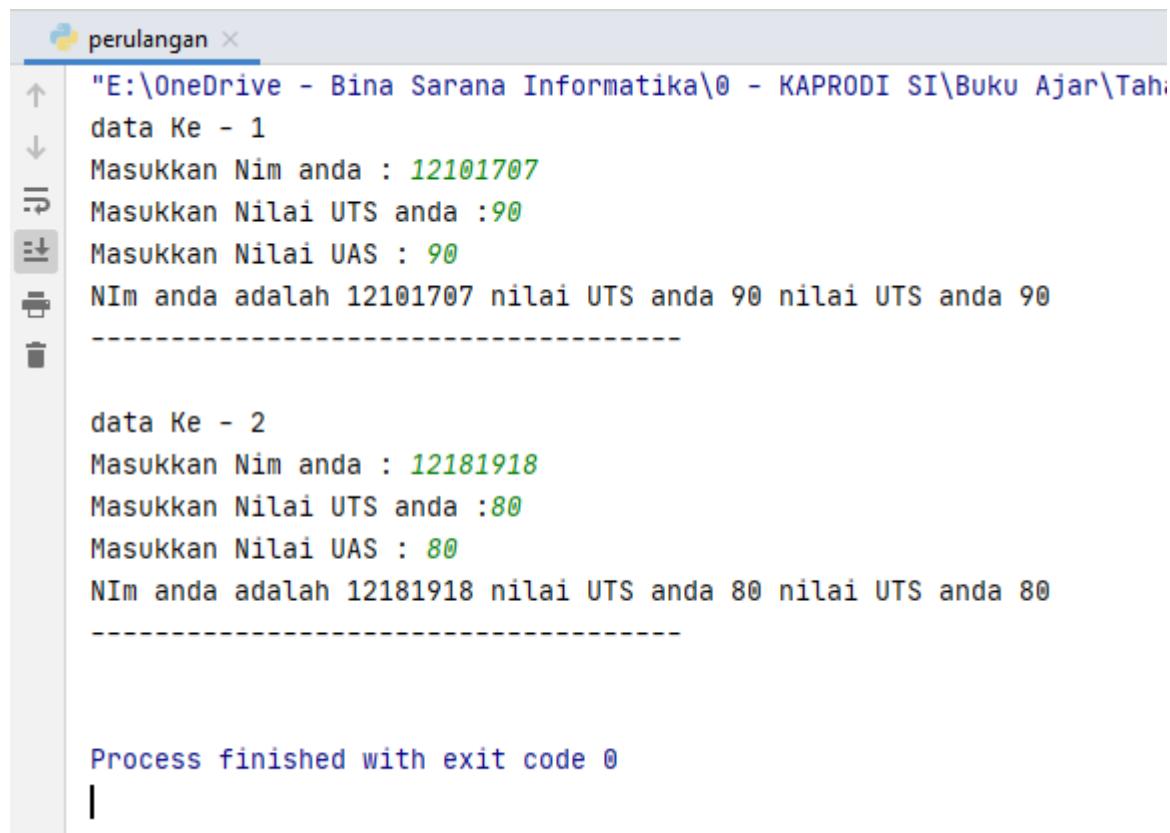
Nim anda adalah <outputnim1> nilai UTS anda <outpututs1> nilai UTS anda <outputuas1>

## Kode Program

```
ulang=2
for i in range(ulang):
    print ("data Ke - " + str(i+1))
    nama=input("Masukkan Nim anda : ")
    uts=int(input("Masukkan Nilai UTS anda :"))
    uas=int(input("Masukkan Nilai UAS :"))
    print("NIm anda adalah %s nilai UTS anda %i nilai UTS anda %i nilai UAS anda %i")
```

```
%i" % (nama,uts,uas))
print("-----\n")
```

## Output



```
E:\OneDrive - Bina Sarana Informatika\0 - KAPRODI SI\Buku Ajar\Tah  
data Ke - 1  
Masukkan Nim anda : 12101707  
Masukkan Nilai UTS anda :90  
Masukkan Nilai UAS : 90  
NIM anda adalah 12101707 nilai UTS anda 90 nilai UTS anda 90  
-----  
  
data Ke - 2  
Masukkan Nim anda : 12181918  
Masukkan Nilai UTS anda :80  
Masukkan Nilai UAS : 80  
NIM anda adalah 12181918 nilai UTS anda 80 nilai UTS anda 80  
-----  
  
Process finished with exit code 0
```

## Tugas 2

Sebuah perusahaan ayam goreng dengan nama “**GEROBAK FRIED CHICKEN**” yang telah lumayan banyak pelanggannya, ingin dibantu dibuatkan program untuk membantu kelancaran usahaannya.

“**GEROBAK FRIED CHICKEN**” mempunyai daftar harga ayam sebagai berikut :

Kode JenisPotong Harga

-----  
D Dada Rp. 2500  
P Paha Rp. 2000  
S Sayap Rp. 1500

Buatlah programnya dengan ketentuan:

- Setiap pembeli dikenakan pajak sebesar 10% dari pembayaran.
- Banyak Jenis, Jenis Potong dan Banyak Beli diinput.
- Tampilan yang diinginkan sebagai berikut:

**Layar Masukkan**

GEROBAK FRIED CHICKEN

-----  
Kode JenisPotong Harga  
-----D Dada Rp. 2500  
P Paha Rp. 2000  
S Sayap Rp. 1500  
-----

Banyak Jenis : ... <diinput>  
Jenis Ke - ... <proses counter>  
Kode Potong [D/P/S] : ... <diinput>  
Banyak Potong : ... <diinput>  
<<Terus berulang tergantung Banyak Jenis>>

**Layar Keluaran**

GEROBAK FIRED CHICHEN

-----  
No. Jenis Harga Bayak Jumlah  
Potong Satuan Beli Harga  
-----... ..... .... .... Rp ....  
... ..... .... .... Rp ....  
-----

Jumlah Bayar Rp ....  
Pajak 10% Rp ....  
Total Bayar Rp ....

# Pertemuan 6

## List & Tuple

## 6.1 List

Python menyediakan sejumlah tipe data yang dikenal dengan tipe data berurut (sequence). List adalah tipe data yang berisi satu atau beberapa nilai di dalamnya. Nilai – nilai ini sering juga disebut item, elemen, atau anggota list. List dibuat dengan menempatkan semua item di dalam tanda kurung [ ], dipisahkan oleh tanda koma. Anggota list bisa berisi satu tipe data, atau campuran.

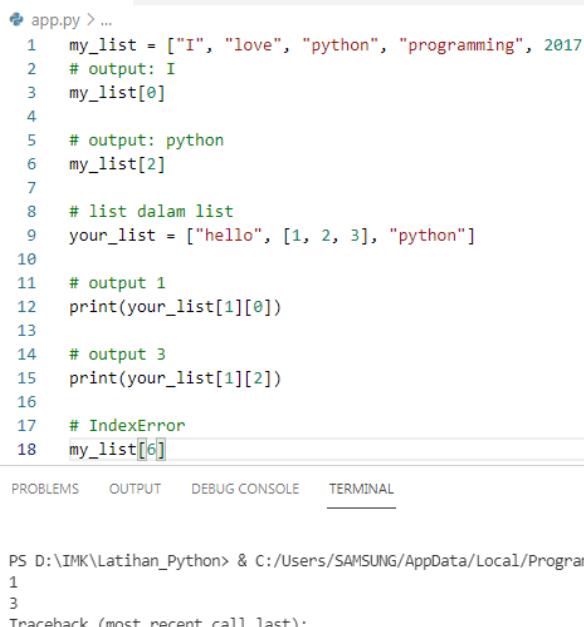
```
# list kosong
my_list = []
# list berisi integer my_list = [1,2,3,4,5]
# list berisi tipe campuran my_list = [1, 3.5, "Hello"]
```

List juga bisa berisi list lain. Ini disebut list bersarang

```
# list bersarang  
my list = ["hello", [2,4,6], ['a','b']]
```

### 6.1.1 Mengakses anggota List

Kita bisa mengakses anggota list dengan menggunakan indeksnya dengan format `namalist[indeks]`. Indeks list dimulai dari 0. List yang memiliki 5 anggota akan memiliki indeks mulai dari 0 s/d 4. Mencoba mengakses anggota list di luar itu akan menyebabkan error `IndexError`.



The screenshot shows a Python code editor with the following details:

- File Menu:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Toolbar:** Includes icons for file operations like Open, Save, Find, and Run.
- Code Area:** An open file named "app.py" containing the following code:

```
my_list = ["I", "love", "python", "programming", 2017]
# output: I
my_list[0]

# output: python
my_list[2]

# list dalam list
your_list = ["hello", [1, 2, 3], "python"]

# output 1
print(your_list[1][0])

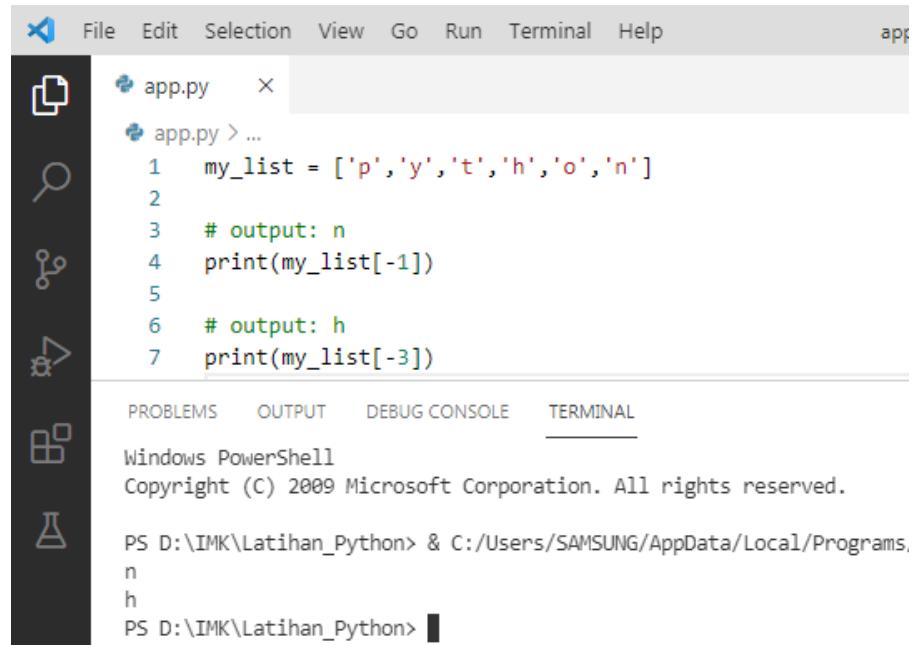
# output 3
print(your_list[1][2])

# IndexError
my_list[6]
```
- Bottom Navigation:** PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (underlined).
- Terminal Output:** Shows the execution of the script and an IndexError.

```
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python
1
3
Traceback (most recent call last):
  File "d:/IMK/Latihan_Python/app.py", line 18, in <module>
    my_list[6]
IndexError: list index out of range
PS D:\IMK\Latihan_Python>
```

### 6.1.2 List dengan Indeks Negatif

Python mendukung indeks negatif, yaitu urutan dimulai dari anggota terakhir. Indeks anggota paling belakang adalah -1, kemudian -2, dan seterusnya.



The screenshot shows the Microsoft Visual Studio Code interface. On the left is a dark sidebar with icons for file operations, search, and other tools. The main area has a tab for 'app.py' which contains the following Python code:

```
1 my_list = ['p', 'y', 't', 'h', 'o', 'n']
2
3 # output: n
4 print(my_list[-1])
5
6 # output: h
7 print(my_list[-3])
```

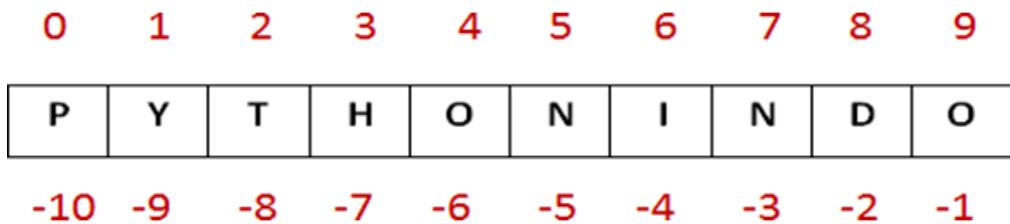
Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing the following terminal output:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
n
h
PS D:\IMK\Latihan_Python>
```

### 6.1.3 Memotong (Slicing) List

Kita bisa mengakses anggota list dari range tertentu dengan menggunakan operator slicing titik dua ( : ). Slicing akan lebih mudah bila kita memahami indeks dengan baik. Perhatikan gambar berikut:



The screenshot shows the Visual Studio Code interface. On the left is a dark sidebar with icons for file operations. The main area has a tab for 'app.py'. The code editor contains the following Python script:

```
1 my_list = ['p', 'y', 't', 'h', 'o', 'n', 'i', 'n', 'd', 'o']
2
3 # anggota list dari 3 s/d 5 (dari h s/d n)
4 print(my_list[3:6])
5
6 # anggota list dari 4 s/d yang terakhir
7 print(my_list[4:])
8
9 # anggota list dari 0 s/d 4
10 print(my_list[:5])
11
12 # indeks dari belakang dari -1 s/d -4
13 print(my_list[-1:-5])
```

Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing the output of the Python script running in a Windows PowerShell window:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs.
['h', 'o', 'n']
['o', 'n', 'i', 'n', 'd', 'o']
['p', 'y', 't', 'h', 'o']
[]
PS D:\IMK\Latihan_Python>
```

#### 6.1.4 Mengubah Anggota List

List adalah tipe data yang bersifat mutable, artinya anggotanya bisa diubah. Ini berbeda dengan string dan tuple yang bersifat immutable.

The screenshot shows the Visual Studio Code interface. On the left is a dark sidebar with icons for file operations. The main area has a tab for 'app.py'. The code editor contains the following Python script:

```
1 # misal ada nilai yang salah
2 ganjil = [1,3,4,7,9]
3
4 # ubah item ke 3 (indeks ke 2)
5 ganjil[2] = 5
6 print(ganjil)
7
8 # mengubah sekali banyak
9 ganjil[2:5] = [11,13,15]
10 print(ganjil)
```

Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing the output of the Python script running in a Windows PowerShell window:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs.
[1, 3, 5, 7, 9]
[1, 3, 11, 13, 15]
PS D:\IMK\Latihan_Python>
```

##### a. Metode List

List memiliki banyak metode untuk operasi seperti menambahkan anggota, menghapus, menyisipkan, menyortir, dan lain sebagainya. Mereka bisa diakses menggunakan format `list.metode()`.

##### b. Menambahkan Anggota List

Fungsi `append()` berguna untuk menambahkan anggota ke dalam list. Selain itu, ada metode `extend()` untuk menambahkan anggota list ke dalam list.

```
File Edit Selection View Go Run Terminal Help app.py - L
app.py  x
app.py > ...
1 ganjil = [1,3,5,7]
2 ganjil.append(9)
3 print(ganjil)
4 ganjil.extend([11,13,15])
5 print(ganjil)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
[1, 3, 5, 7, 9]
[1, 3, 5, 7, 9, 11, 13, 15]
PS D:\IMK\Latihan_Python>
```

Kita juga bisa menggunakan operator + untuk menggabungkan dua list, dan operator \* untuk melipatgandakan list.

```
File Edit Selection View Go Run Terminal Help app.py - L
app.py  x
app.py > ...
1 genap = [2, 4, 6]
2 print(genap + [8, 10, 12])
3 print(['p','y'] * 2)
4

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs.
[2, 4, 6, 8, 10, 12]
['p', 'y', 'p', 'y']
PS D:\IMK\Latihan_Python>
```

### c. Menyisipkan Anggota List

Fungsi insert() berfungsi untuk menyisipkan anggota list pada indeks tertentu.

```
File Edit Selection View Go Run Terminal Help app.py - L
app.py  x
app.py > ...
1 ganjil = [5,7,11,13,15]
2 # kita akan menyisipkan 9 setelah angka 7
3
4 ganjil.insert(2,9)
5 print(ganjil)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
[5, 7, 9, 11, 13, 15]
PS D:\IMK\Latihan_Python>
```

#### d. Menghapus Anggota List

Kita bisa menggunakan metode `remove()`, `pop()`, atau kata kunci `del` untuk menghapus anggota list. Selain itu kita bisa menggunakan `clear()` untuk mengosongkan list. Fungsi `pop()` selain menghapus anggota list, juga mengembalikan nilai indeks anggota tersebut. Hal ini berguna bila kita ingin memanfaatkan indeks dari anggota yang terhapus untuk digunakan kemudian.



```
File Edit Selection View Go Run Terminal Help app.py - Lati
app.py > ...
1 my_list = ['p', 'y', 't', 'h', 'o', 'n', 'i', 'n', 'd', 'o']
2 my_list.remove('p')
3
4 # output ['y', 't', 'h', 'o', 'n', 'i', 'n', 'd', 'o']
5 print(my_list)
6
7 my_list.remove('n')
8 # remove hanya menghapus elemen pertama yang dijumpai
9 # output: ['p', 'y', 't', 'h', 'o', 'i', 'n', 'd', 'o']
10
11 # Output 'y'
12 print(my_list.pop(1))
13
14 del my_list[2]
15 print(my_list)
16
17 my_list.clear()
18 # Output []
19 print(my_list)

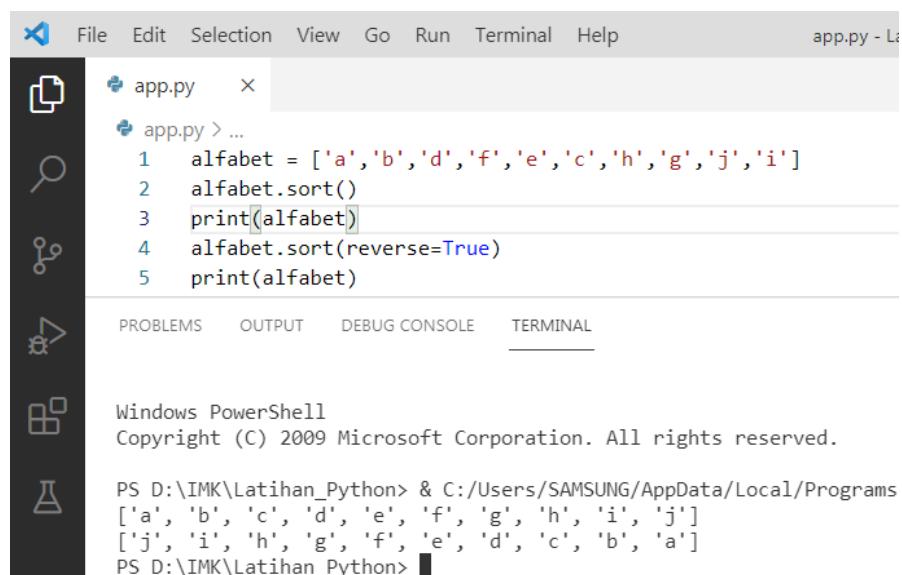
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python,
['y', 't', 'h', 'o', 'n', 'i', 'n', 'd', 'o']
t
['y', 'h', 'i', 'n', 'd', 'o']
[]
PS D:\IMK\Latihan_Python>
```

#### e. Mengurutkan anggota List

Pada saat kita perlu mengurutkan atau menyortir anggota list, kita bisa menggunakan metode `sort()`. Untuk membalik dengan urutan sebaliknya bisa dengan menggunakan argumen `reverse=True`.



```
File Edit Selection View Go Run Terminal Help app.py - L
app.py > ...
1 alfabet = ['a', 'b', 'd', 'f', 'e', 'c', 'h', 'g', 'j', 'i']
2 alfabet.sort()
3 print(alfabet)
4 alfabet.sort(reverse=True)
5 print(alfabet)

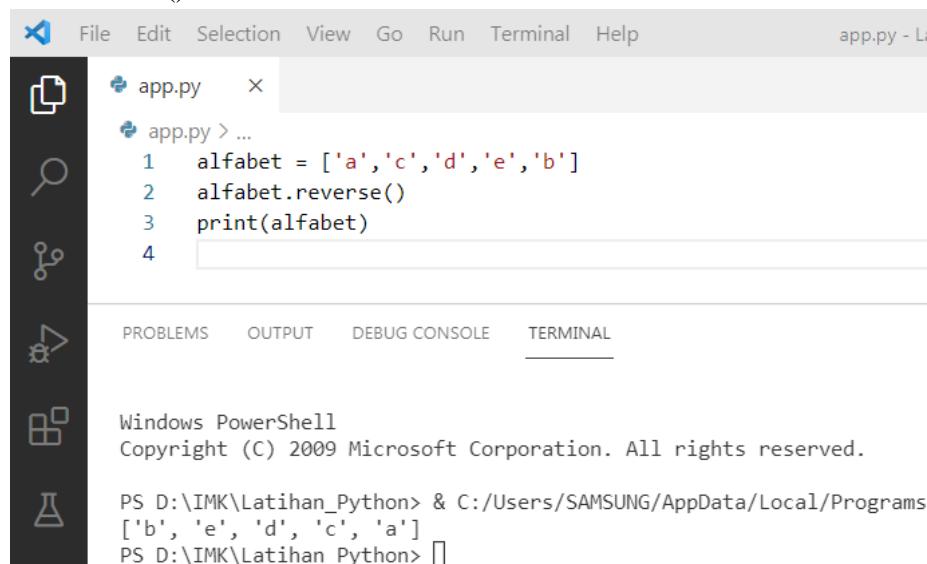
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
['j', 'i', 'h', 'g', 'f', 'e', 'd', 'c', 'b', 'a']
PS D:\IMK\Latihan_Python>
```

#### f. Membalik Urutan List

Selain mengurutkan, kita juga bisa membalikkan urutan list dengan menggunakan metode `reverse()`.



The screenshot shows a code editor interface with a dark theme. On the left is a sidebar with icons for file operations, search, and other tools. The main area has tabs for 'app.py' and 'app.py > ...'. The code in 'app.py' is:

```
1 alfabet = ['a', 'c', 'd', 'e', 'b']
2 alfabet.reverse()
3 print(alfabet)
4
```

Below the code editor are tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL'. The 'TERMINAL' tab is active, showing a Windows PowerShell window with the following output:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
['b', 'e', 'd', 'c', 'a']
PS D:\IMK\Latihan_Python>
```

## 6.2 Tuple

Tuple mirip dengan list. Bedanya, tuple bersifat immutable, sehingga anggotanya tidak bisa diubah. Kalau mirip, mengapa harus menggunakan tuple?

Kita menggunakan tuple tergantung kebutuhan. Untuk beberapa hal, tuple memiliki kelebihan sebagai berikut:

- Karena tuple adalah immutable, maka iterasi pada tuple lebih cepat dibandingkan list.
- Tuple bisa berisi anggota yang immutable yang dapat digunakan sebagai key untuk dictionary. List tidak bisa dipakai untuk itu.
- Kalau kita memerlukan data yang memang tidak untuk diubah, maka menggunakan tuple bisa menjamin bahwa data tersebut akan write-protected.

### 6.2.1 Membuat Tuple

Tuple dibuat dengan meletakkan semua anggota di dalam tanda kurung ( ), masing-masing dipisahkan oleh tanda koma. Menggunakan tanda kurung sebenarnya hanya opsional, tapi kita sebaiknya tetap menggunakan untuk kemudahan pembacaan kode. Tuple dapat berisi tipe data yang sama maupun campuran.

The image shows a screenshot of a Python development environment. At the top, there is a menu bar with File, Edit, Selection, View, Go, Run, Terminal, and Help. The title bar indicates "app.py - Latihan". On the left is a dark sidebar with various icons for file operations like copy, search, and refresh. The main area contains the Python code for tuples:

```
1 # membuat tuple kosong
2 my_tuple = ()
3 print(my_tuple)
4
5 # tuple dengan 1 elemen
6 # Output: (1,)
7 my_tuple = (1,)
8 print (my_tuple)
9
10 # tuple berisi integer
11 # output = (1, 2, 3)
12 my_tuple = (1, 2, 3)
13 print(my_tuple)
14
15 # tuple bersarang
16 # Output: ("hello", [1, 2, 3], (4, 5, 6))
17 my_tuple = ("hello", [1, 2, 3], (4, 5, 6))
18 print(my_tuple)
19
20 # Tuple bisa tidak menggunakan tanda ()
21 # Output (1, 2, 3)
22 my_tuple = 1, 2, 3
23
24 # memasukkan anggota tuple ke variabel yang bersesuaian
25 # a akan berisi 1, b berisi 2, dan c berisi 3
26 # output 1 2 3
27 a, b, c = my_tuple
28 print(a, b, c)
```

Below the code editor is a terminal window with its own menu bar and title bar "app.py - Latihan". It has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, with TERMINAL selected. The terminal displays the execution of the Python script:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Py
()
(1,)
(1, 2, 3)
('hello', [1, 2, 3], (4, 5, 6))
1 2 3
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Py
()
(1,)
(1, 2, 3)
('hello', [1, 2, 3], (4, 5, 6))
1 2 3
PS D:\IMK\Latihan_Python>
```

### 6.2.2 Mengakses anggota Tuple

Seperti halnya list, kita bisa mengakses anggota tuple lewat indeksnya menggunakan format `namatuple[indeks]`. Indeks dimulai dari 0 untuk anggota pertama. Selain itu, indeks negatif juga bisa dipakai mulai dari -1 untuk anggota terakhir tuple.

The screenshot shows a Visual Studio Code interface. On the left is a dark sidebar with icons for file operations, search, and other development tools. The main area has a tab for 'app.py' which contains the following Python code:

```
1 my_tuple = ('p','r','o','g','r','a','m','m','i','n','g')
2 # akses dari indeks 0 s/d 2
3
4 # output: ('p','r','o')
5 print(my_tuple[:3])
6
7 # Akses dari indeks 2 s/d 5
8 # output: ('r','o','g','r')
9 print(my_tuple[2:6])
10
11 # Akses dari indeks 3 sampai akhir
12 # output: ('r','o','g','r','a','m','m','i','n','g')
13 print(my_tuple[3:])
```

Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, showing a Windows PowerShell session with the following output:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python,
('p', 'r', 'o')
('o', 'g', 'r', 'a')
('g', 'r', 'a', 'm', 'm', 'i', 'n', 'g')
PS D:\IMK\Latihan_Python>
```

### 6.2.3 Mengubah Anggota Tuple

Setelah tuple dibuat, maka anggota tuple tidak bisa lagi diubah atau dihapus. Akan tetapi, bila anggota tuple-nya adalah tuple bersarang dengan anggota seperti list, maka item pada list tersebut dapat diubah. Jelasnya ada pada contoh berikut:

The screenshot shows a Visual Studio Code interface. On the left is a dark sidebar with icons for file operations, search, and other development tools. The main area has a tab for 'app.py' which contains the following Python code:

```
1 my_tuple = (2, 3, 4, [5, 6])
2 # kita tidak bisa mengubah anggota tuple
3 # bila kita hilangkan tanda komentar # pada baris ke 6
4 # akan muncul error: # TypeError: 'tuple' object does not support item assignment
5
6 # my_tuple[1] = 8
7
8 # tapi list di dalam tuple bisa diubah
9 # output: (2, 3, 4, [7, 6])
10 my_tuple[3][0] = 7
11 print(my_tuple)
12
13 # tuple bisa diganti secara keseluruhan dengan penugasan kembali
14 # output: ('p','y','t','h','o','n')
15 my_tuple = ('p','y','t','h','o','n')
16 print(my_tuple)
17
18 # anggota tuple juga tidak bisa dihapus menggunakan del
19 # perintah berikut akan menghasilkan error TypeError
20 # kalau Anda menghilangkan tanda komentar #
21
22 #del my_tuple[0]
23
24 # kita bisa menghapus tuple keseluruhan
25
26 del my_tuple
```

```
File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe d
(2, 3, 4, [7, 6])
('p', 'y', 't', 'h', 'o', 'n')
PS D:\IMK\Latihan_Python> []
```

## 1. Menguji Keanggotaan Tuple

Seperti halnya string dan list, kita bisa menguji apakah sebuah objek adalah anggota dari tuple atau tidak, yaitu dengan menggunakan operator `in` atau `not in` untuk kebalikannya.

```
File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python
app.py > ...
1 my_tuple = (1, 2, 3, 'a', 'b', 'c')
2
3 # menggunakan in
4 # output: False
5 print('3' in my_tuple)
6
7 # output: False
8 print('e' in my_tuple)
9
10 # menggunakan not in
11 # output True
12 print('k' not in my_tuple)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe app.py
False
False
True
PS D:\IMK\Latihan_Python> []
```

## 2. Iterasi pada Tuple

Kita bisa menggunakan `for` untuk melakukan iterasi pada tiap anggota dalam tuple.

```
File Edit Selection View Go Run Terminal Help
app.py - Latihan_Python
app.py > ...
1 # output:
2 # Hi Galih
3 # Hi Ratna
4 nama = ('Galih', 'Ratna')
5 for name in nama:
6     print('Hi', name)

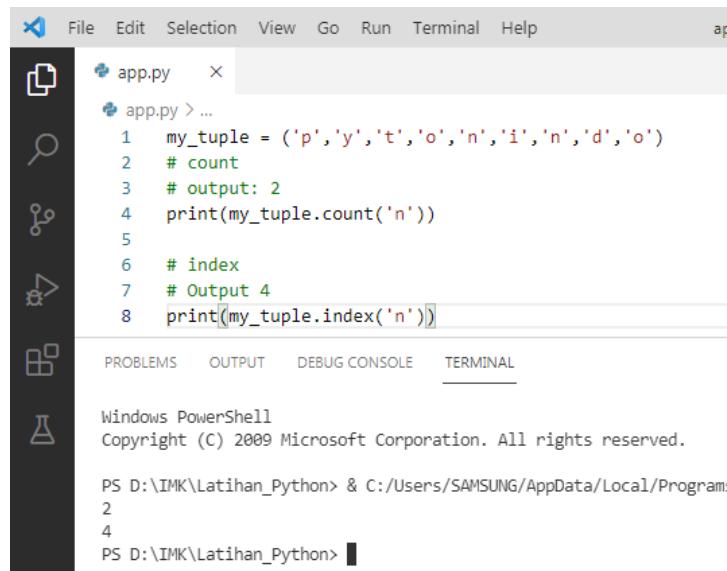
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe app.py
Hi Galih
Hi Ratna
PS D:\IMK\Latihan_Python> []
```

#### 6.2.4 Metode dan Fungsi Bawaan Tuple

Tuple hanya memiliki dua buah metode yaitu count() dan index().

- Metode count(x) berfungsi mengembalikan jumlah item yang sesuai dengan x pada tuple
- Metode index(x) berfungsi mengembalikan indeks dari item pertama yang sama dengan x.



```
File Edit Selection View Go Run Terminal Help
app.py > ...
1 my_tuple = ('p','y','t','o','n','i','n','d','o')
2 # count
3 # output: 2
4 print(my_tuple.count('n'))
5
6 # index
7 # Output 4
8 print(my_tuple.index('n'))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell  
Copyright (C) 2009 Microsoft Corporation. All rights reserved.  
PS D:\IMK\Latihan\_Python> & C:/Users/SAMSUNG/AppData/Local/Programs  
2  
4  
PS D:\IMK\Latihan\_Python>

Walaupun hanya memiliki dua metode, banyak fungsi bawaan python yang berfungsi untuk melakukan operasi pada tuple. Berikut adalah daftarnya:

Tabel 6.1 Operasi Pada Tuple

Fungsi	Deskripsi
all()	Mengembalikan <b>True</b> jika semua anggota tuple adalah benar ( tidak ada yang kosong )
any()	Mengembalikan <b>True</b> jika salah satu atau semua bernilai benar. Jika tuple kosong, maka akan mengembalikan <b>False</b> .
enumerate()	Mengembalikan objek enumerasi. Objek enumerasi adalah objek yang terdiri dari pasangan indeks dan nilai.
len()	Mengembalikan panjang (jumlah anggota) tuple

<b>max()</b>	Mengembalikan anggota terbesar di tuple
<b>min()</b>	Mengembalikan anggota terkecil di tuple
<b>sorted()</b>	Mengambil anggota tuple dan mengembalikan list baru yang sudah diurutkan
<b>sum()</b>	Mengembalikan jumlah dari semua anggota tuple
<b>tuple()</b>	Mengubah sequence (list, string, set, dictionary) menjadi tuple

### 3.1. Studi Kasus :

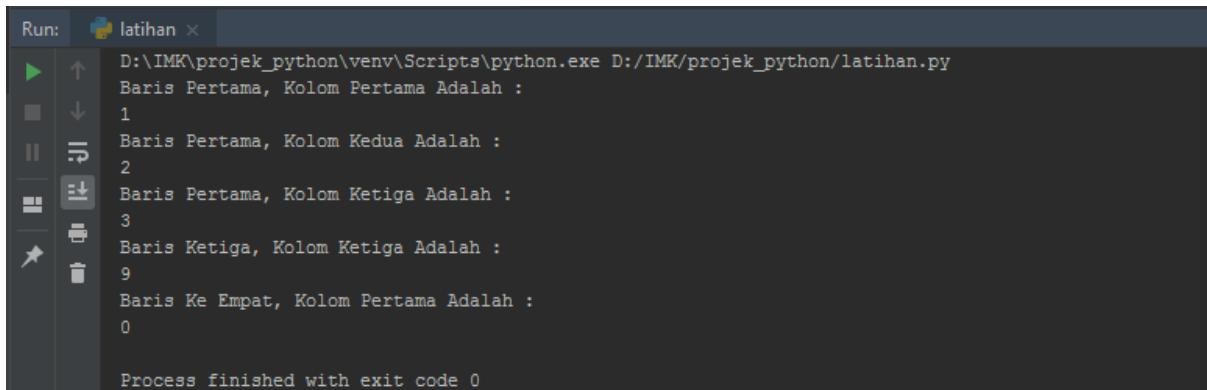
Buatlah sebuah list (array) 2 dimensi dimana terdapat baris dan kolom dengan nilai sebagai berikut :

<b>1</b>	<b>2</b>	<b>3</b>
4	5	6
7	8	9
0		

Tampilkan lah :

- Baris Pertama, Kolom Pertama
- Baris Pertama, Kolom Kedua
- Baris Pertama, Kolom Ketiga
- Baris Ketiga, Kolom Ketiga
- Baris Ke Empat, Kolom Pertama

Hasil :



```

Run:  latihan x
D:\IMK\projek_python\venv\Scripts\python.exe D:/IMK/projek_python/latihan.py
Baris Pertama, Kolom Pertama Adalah :
1
Baris Pertama, Kolom Kedua Adalah :
2
Baris Pertama, Kolom Ketiga Adalah :
3
Baris Ketiga, Kolom Ketiga Adalah :
9
Baris Ke Empat, Kolom Pertama Adalah :
0

Process finished with exit code 0
  
```

## Latihan

Buatlah input, proses dan output secara berulang dengan memanfaatkan fungsi matriks/list seperti pada koding dibawah ini :

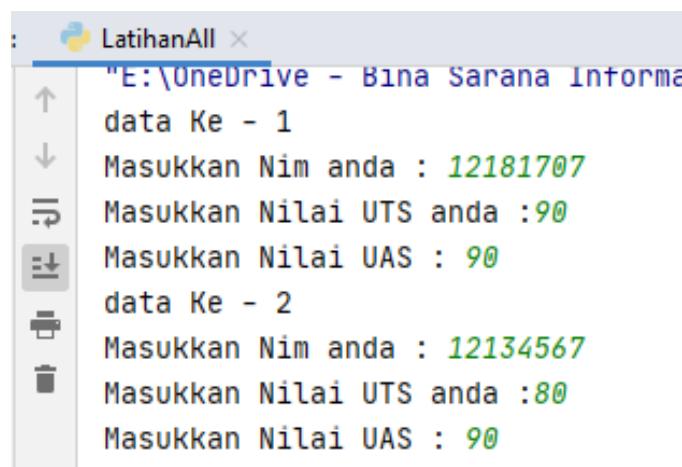
```
#variable yg berulang menggunakan List/matriks
list_nim=[]
list_uts=[]
list_uas=[]
list_total=[]

ulang=2
for i in range(ulang):
    print ("data Ke - " + str(i+1))
    list_nim.append(input("Masukkan Nim anda : "))
    list_uts.append(int(input("Masukkan Nilai UTS anda : ")))
    list_uas.append(int(input("Masukkan Nilai UAS : ")))

#proses
for i in range(ulang):
    list_total.append((list_uas[i] + list_uts[i]) / 2)
#Cetak
print("====")
print("Nim   Nilai Uts   Nilai UAS   Total")
print("====")
for i in range(ulang):
    print ("%s \t %i \t %i \t %i" % (list_nim[i],list_uts[i],list_uas[i],list_total[i]))

print("====")
```

### Hasil Tampilan Input



The screenshot shows a terminal window titled 'LatihanAll' with the following interaction:

```
"E:\OneDrive - Bina Sarana Informatika\PycharmProjects\LatihanAll\venv\Scripts\python.exe" "E:\OneDrive - Bina Sarana Informatika\PycharmProjects\LatihanAll\latihan.py"
data Ke - 1
Masukkan Nim anda : 12181707
Masukkan Nilai UTS anda :90
Masukkan Nilai UAS : 90
data Ke - 2
Masukkan Nim anda : 12134567
Masukkan Nilai UTS anda :80
Masukkan Nilai UAS : 90
```

### Hasil Tampilan Output

Nim	Nilai Uts	Nilai UAS	Total
12181707	90	90	90
12134567	80	90	85

S

## Pertemuan 7

### Matrix dan Library Pandas

Matriks dalam dunia metematika merupakan suatu bilangan, simbol, ataupun ekspresi yang disusun dalam baris dan kolom yang membentuk suatu bidang persegi/persegi panjang. Kumpulan data dalam matriks biasa disebut dengan elemen matriks. Elemen matriks dapat berisi bilangan, simbol, dan ekspresi matematika. Untuk membentuk matriks semua elemen ini diletakkan diantara kurung biasa ( . . . ) atau kurung siku [ . . . ]. Susunan elemen secara horizontal dalam matriks disebut baris (*row*) yang diwakilkan dengan huruf *m*. Sedangkan susunan elemen secara vertikal pada matriks disebut kolom (*column*) yang diwakilkan dengan huruf *n*. Untuk contoh matriks perhatikan gambar di bawah ini.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Gambar di atas merupakan matriks berordo  $3 \times 3$ . Ordo merupakan ukuran dimensi pada matriks yang dinotasikan dengan *m* x *n* yang dimana *m* melambangkan baris dan *n* melambangkan kolom [*row* x *column*]. Jadi matriks di atas memiliki 3 baris dan 3 kolom.

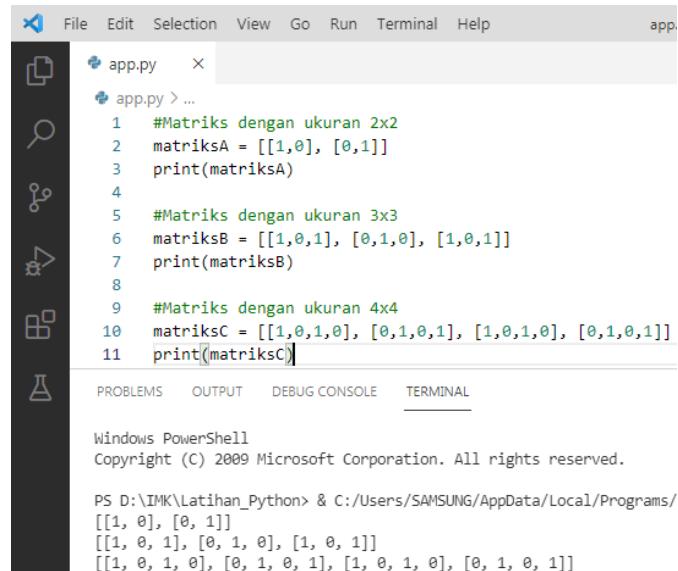
$$A_{m \times n} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix} \rightarrow \begin{array}{l} \text{baris ke-1} \\ \text{baris ke-2} \\ \text{baris ke-3} \\ \vdots \\ \text{baris ke-}m \end{array}$$

↓  
kolom ke-n  
↓  
kolom ke-3  
↓  
kolom ke-2  
↓  
kolom ke-1

## 7.1 Matrix

### 7.1.1 Membuat matrix di python.

Matriks dapat dikatakan sebagai *list* dua dimensi dimana suatu *list* berisi *list* lagi. Untuk merepresentasikan matriks, kita harus menyimpan *list* dengan panjang yang sama dalam suatu *list*. Bila *list* berbeda – beda panjangnya, maka *list* tersebut disebut sebagai *sparse matrix*. Sebagai contoh berikut adalah contoh representasi matriks di Python:

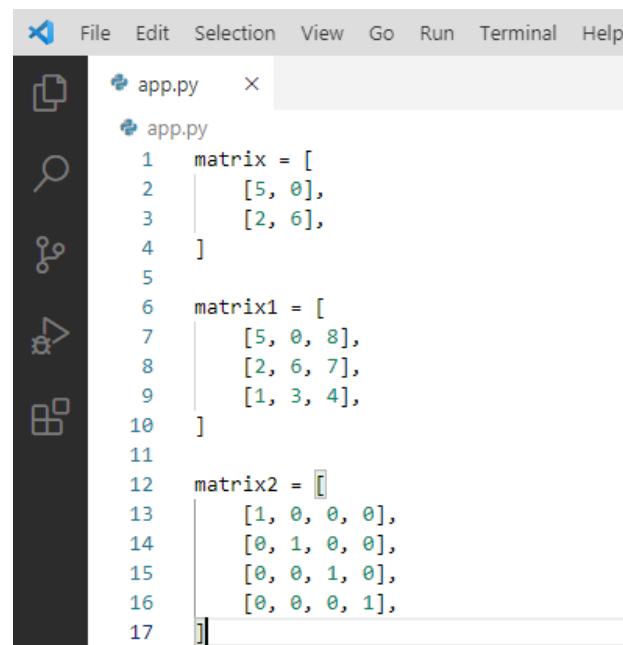


```
File Edit Selection View Go Run Terminal Help
app.py < ...
app.py > ...
1 #Matriks dengan ukuran 2x2
2 matriksA = [[1,0], [0,1]]
3 print(matriksA)
4
5 #Matriks dengan ukuran 3x3
6 matriksB = [[1,0,1], [0,1,0], [1,0,1]]
7 print(matriksB)
8
9 #Matriks dengan ukuran 4x4
10 matriksC = [[1,0,1,0], [0,1,0,1], [1,0,1,0], [0,1,0,1]]
11 print(matriksC)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/
[[1, 0], [0, 1]]
[[1, 0, 1], [0, 1, 0], [1, 0, 1]]
[[1, 0, 1, 0], [0, 1, 0, 1], [1, 0, 1, 0], [0, 1, 0, 1]]
```

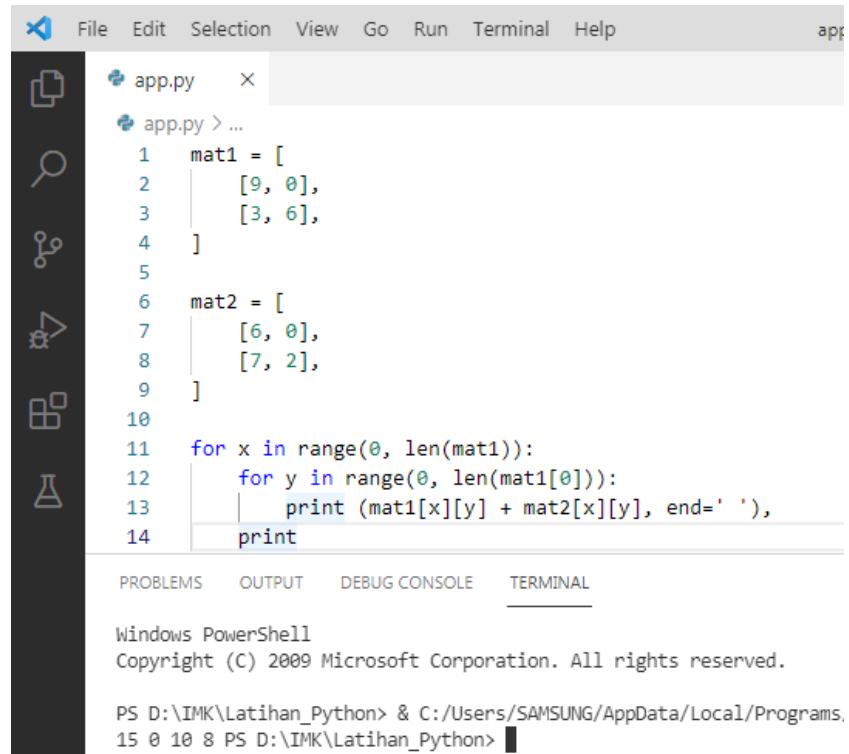


```
File Edit Selection View Go Run Terminal Help
app.py < ...
app.py
1 matrix = [
2     [5, 0],
3     [2, 6],
4 ]
5
6 matrix1 = [
7     [5, 0, 8],
8     [2, 6, 7],
9     [1, 3, 4],
10 ]
11
12 matrix2 = [
13     [1, 0, 0, 0],
14     [0, 1, 0, 0],
15     [0, 0, 1, 0],
16     [0, 0, 0, 1],
17 ]
```

Dalam mengolah matriks, ada berbagai operasi yang dapat dilakukan. Mulai dari translasi, rotasi, mencari determinan, operasi baris elementer, dan lainnya. Namun kita hanya akan membahas beberapa operasi dasar seperti penjumlahan, pengurangan dan perkalian dua matriks.

### 7.1.2 Melakukan Penjumlahan Matriks

Penjumlahan matriks dilakukan dengan menjumlahkan setiap elemen. Hasil penjumlahan tersebut akan menjadi elemen baru. Masing – masing matriks kita akses setiap elemennya pada koordinat yang sama kemudian kita jumlahkan untuk mendapatkan elemen baru.



The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar has icons for file operations like Open, Save, Find, and Run. The main editor window displays the following Python code:

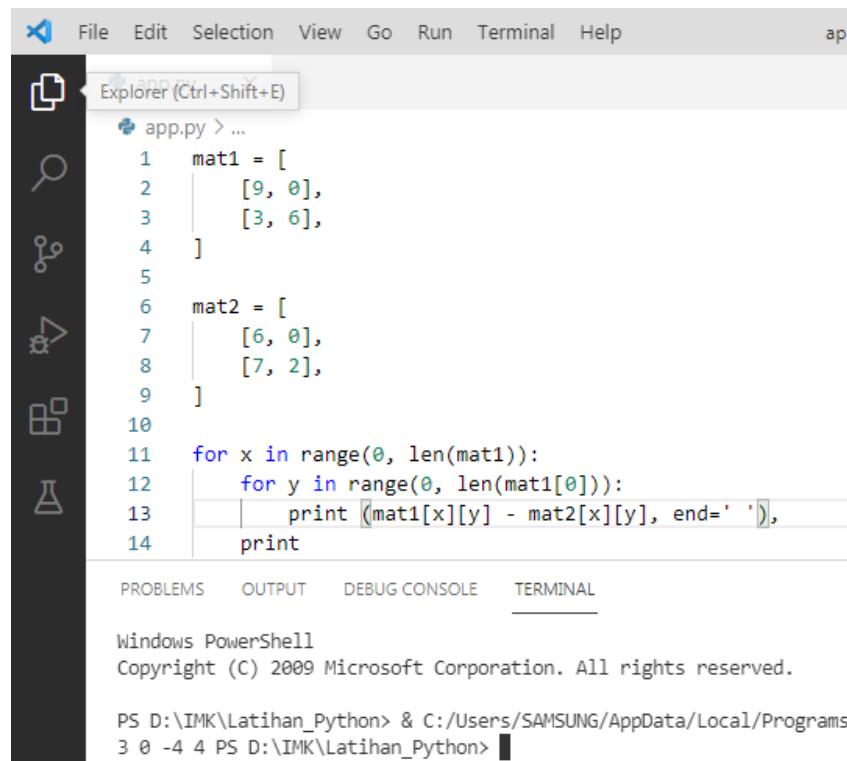
```
File Edit Selection View Go Run Terminal Help
app.py
app.py > ...
1 mat1 = [
2     [9, 0],
3     [3, 6],
4 ]
5
6 mat2 = [
7     [6, 0],
8     [7, 2],
9 ]
10
11 for x in range(0, len(mat1)):
12     for y in range(0, len(mat1[0])):
13         print (mat1[x][y] + mat2[x][y], end=' ')
14     print

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs,
15 0 10 8 PS D:\IMK\Latihan_Python>
```

### 7.1.3 Melakukan Pengurangan pada Matriks

Tidak berbeda jauh dengan penjumlahan matriks, pada pengurangan matriks kita hanya mengganti operatornya saja dengan tanda kurang (-). Maka matriks baru akan terbentuk sebagai hasil dari pengurangan setiap kedua elemen matriks. Sebagai contoh berikut adalah *source code* untuk melakukan pengurangan matriks:



```
File Edit Selection View Go Run Terminal Help
app.py > ...
1  mat1 = [
2      [9, 0],
3      [3, 6],
4  ]
5
6  mat2 = [
7      [6, 0],
8      [7, 2],
9  ]
10
11 for x in range(0, len(mat1)):
12     for y in range(0, len(mat1[0])):
13         print (mat1[x][y] - mat2[x][y], end=' ')
14     print

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
3 0 -4 4 PS D:\IMK\Latihan_Python>
```

#### 7.1.4 Melakukan Perkalian Matriks

Perkalian matriks merupakan salah satu operasi dasar yang *tricky*. Karena di dalamnya bukan hanya terdapat operasi perkalian, melainkan juga penjumlahan. Perkalian suatu matriks memang tidak sama dengan bilangan biasa, tidak juga langsung mengalikan setiap elemen. Perkalian matriks dilakukan dengan menjumlahkan hasil perkalian suatu baris matriks pertama ke kolom matriks kedua. Setiap baris di matriks pertama akan dikalikan ke setiap kolom di matriks kedua.

Di Python, kita akan menggunakan *nested loop for* di dalam **nested loop** yang kedua. *Looping* ketiga tersebut kita gunakan untuk melakukan proses penjumlahan hasil perkalian baris dan kolom. Hasilnya elemen matriks baru akan ditempatkan pada koordinat tersebut. Sebagai contoh berikut adalah *source code* yang melakukan proses perkalian matriks:

```
File Edit Selection View Go Run Terminal Help
app.py > ...
1  mat1 = [
2      [9, 0],
3      [3, 6],
4  ]
5  mat2 = [
6      [6, 0],
7      [7, 2],
8  ]
9  mat3 = []
10 for x in range(0, len(mat1)):
11     row = []
12     for y in range(0, len(mat1[0])):
13         total = 0
14         for z in range(0, len(mat1)):
15             total = total + (mat1[x][z] * mat2[z][y])
16         row.append(total)
17     mat3.append(row)
18
19 for x in range(0, len(mat3)):
20     for y in range(0, len(mat3[0])):
21         print (mat3[x][y], end=' ')
22 print ()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

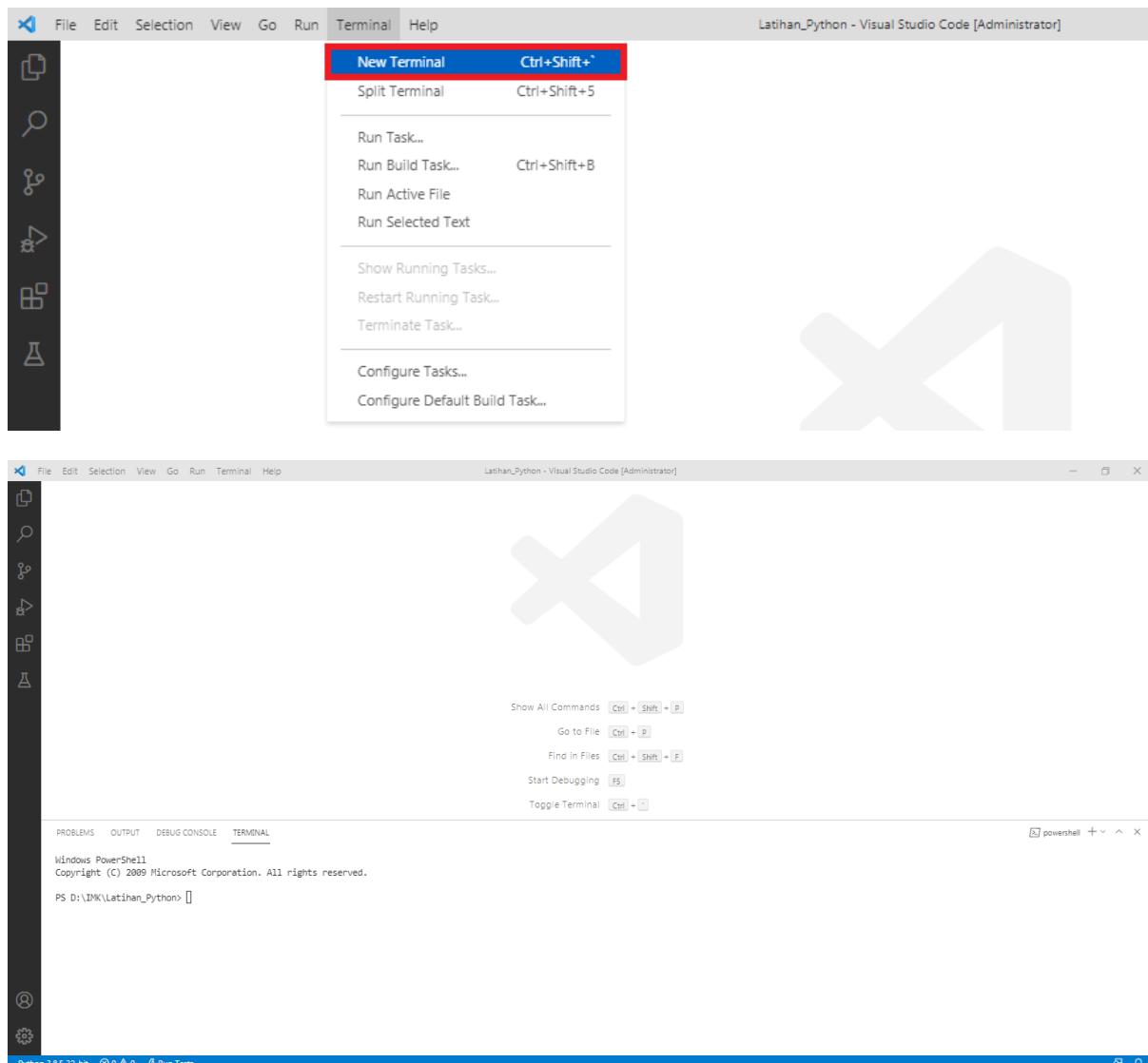
PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Progr
54 0
60 12
PS D:\IMK\Latihan_Python> []
```

## 7.2 Pandas Pada Python

Pandas kependekan dari **Python Data Analysis Library**. Nama Pandas tersebut adalah turunan dari kata Panel Data. Pandas adalah sebuah paket library pada python yang digunakan untuk mempermudah dalam mengolah dan menganalisa data-data terstruktur. Pandas merupakan paket penting yang wajib diketahui untuk seorang data engineer, data analyst dan data scientist jika ingin mengolah dan manganalisa data menggunakan python. Jika kamu telah terbiasa menggunakan SQL, maka tidak akan sulit untuk membiasakan diri menggunakan fungsi-fungsi pada Pandas. Panda memiliki format data yang sering digunakan, disebut DataFrame. Pandas DataFrame adalah struktur data 2 Dimensi. Data distrukturisasi seperti tabel yang berisi baris dan kolom, sehingga mudah untuk melakukan queri atau mengakses data tersebut. Baris merepresentasikan record dan kolom merepresentasikan field.

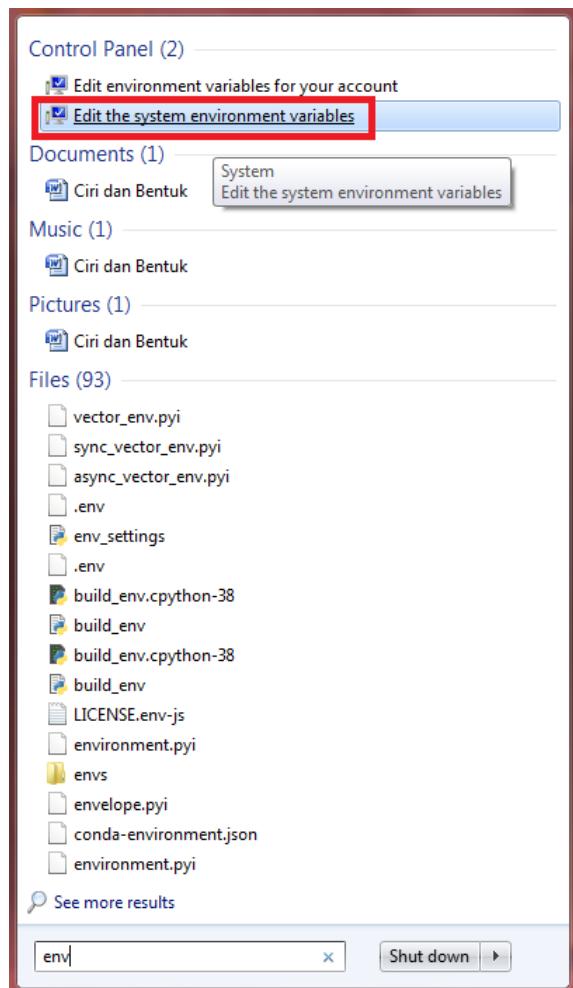
### 7.2.1 Instalasi Pandas

Untuk instalasi library pandas pada VS Code yaitu dengan menggunakan perintah **pip install pandas** pada terminal dalam VS Code seperti pada gambar berikut :

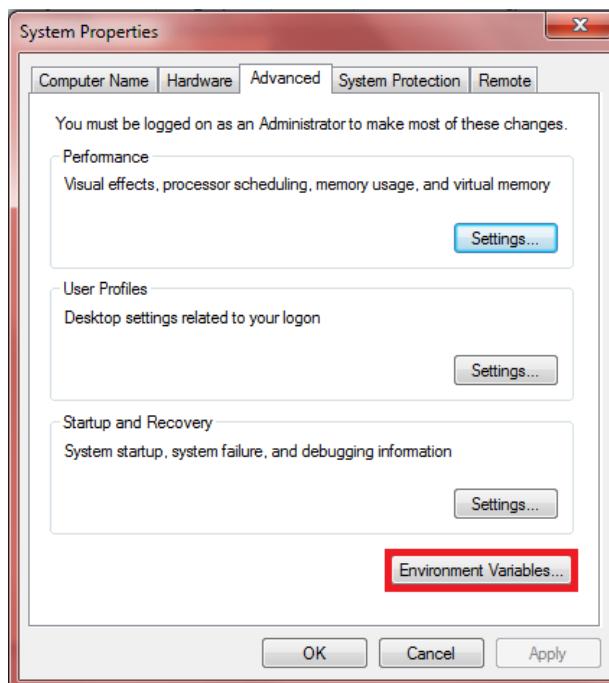


Untuk kelancaran proses instalasi library pandas, kita tetapkan terlebih dahulu PATH untuk instalasinya ikuti langkah pada gambar berikut :

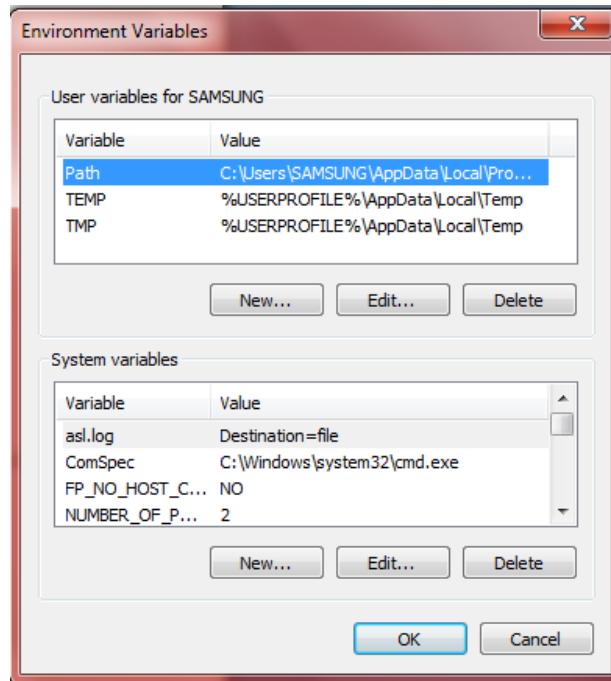
1. Buka “Edit the system environment variables” pada menu pencarian windows.



2. Kemudian akan muncul pop up windows seperti gambar berikut, kemudian pilih “Environment Variables”



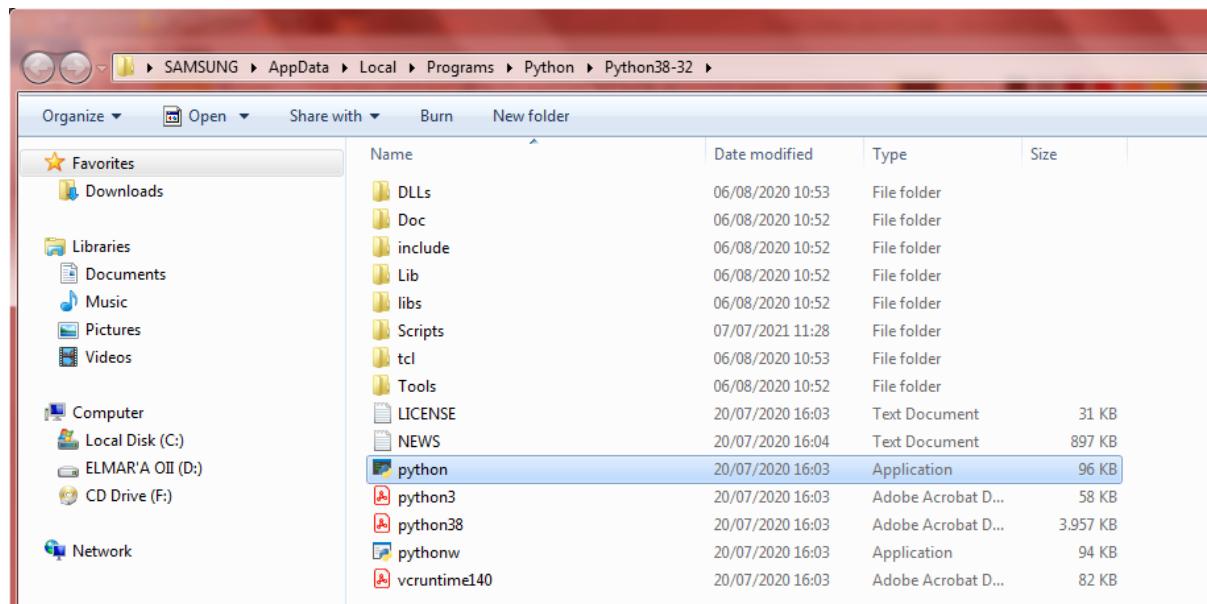
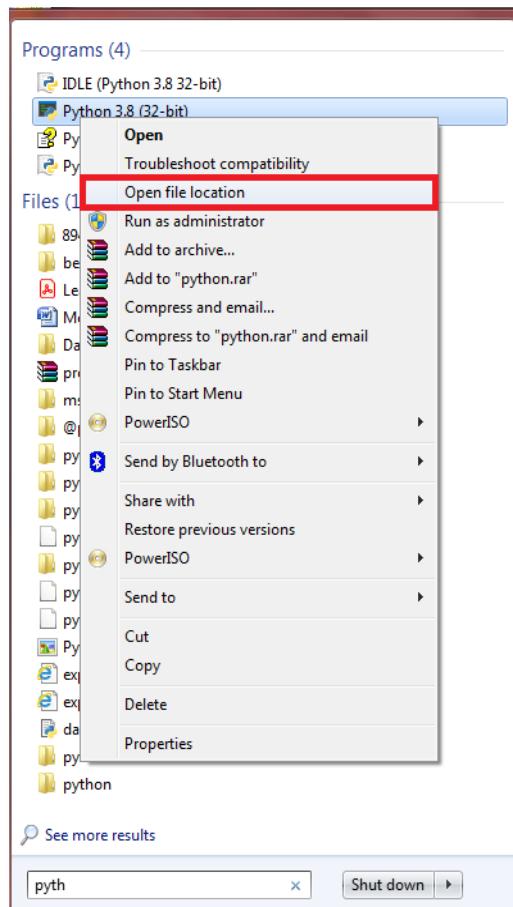
3. Setelah kita klik Environment variables maka akan muncul pop up windows kembali seperti pada gambar berikut :



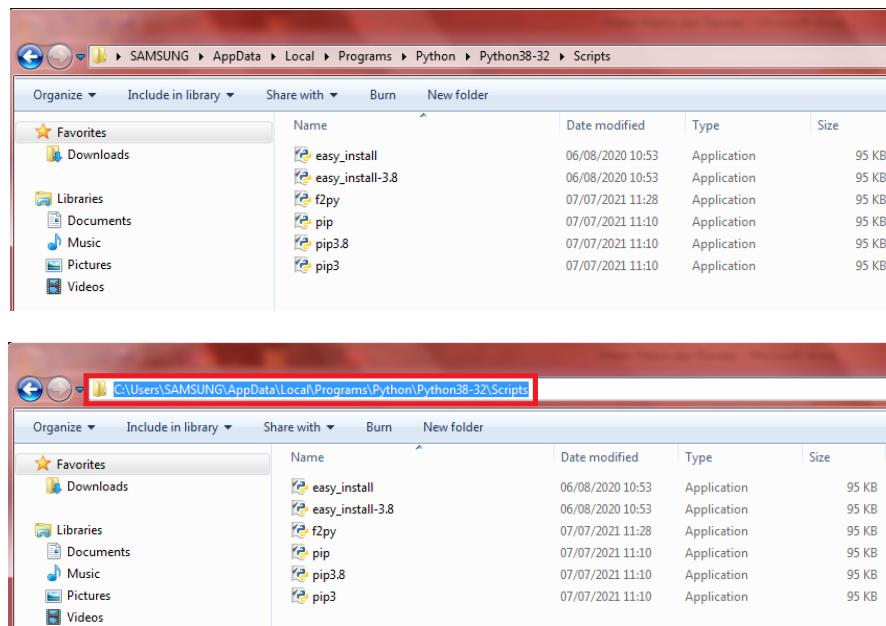
4. Langkah berikutnya adalah kita buka lokasi instalasi python untuk mencopykan PATH pada Environment Variables.



5. Klik kanan pada aplikasi python lalu pilih open file location, maka kita akan diarahkan pada folder tempat instalasi aplikasi python.

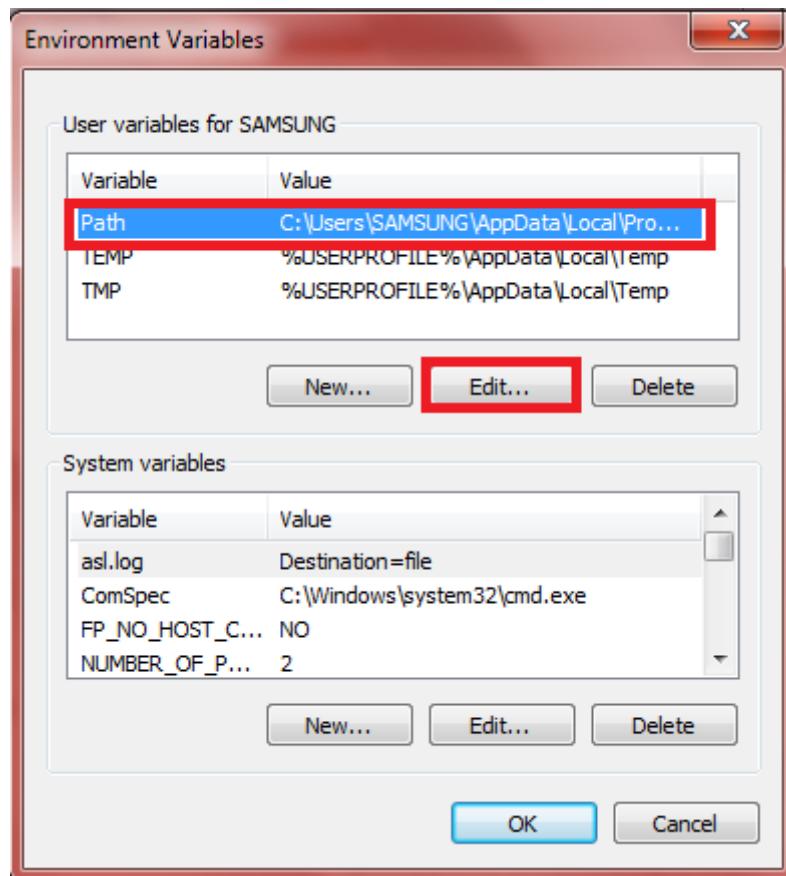


6. Langkah berikutnya adalah kita buka folder Scripts dan klik pada bagian navbar folder Scripts untuk mendapatkan detail PATH nya.

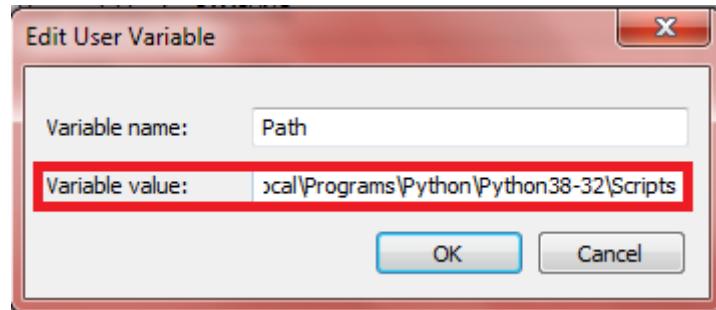


Contoh detail Path untuk folder scripts adalah  
 “C:\Users\SAMSUNG\AppData\Local\Programs\Python\Python38-32\Scripts”

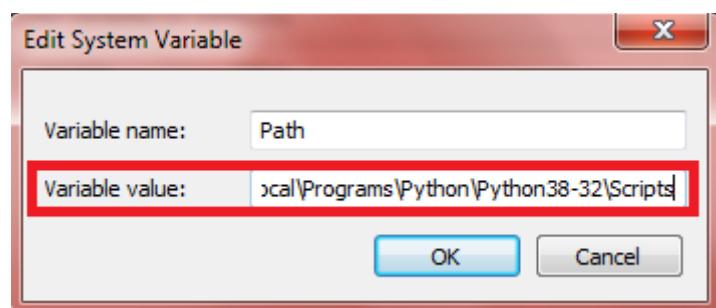
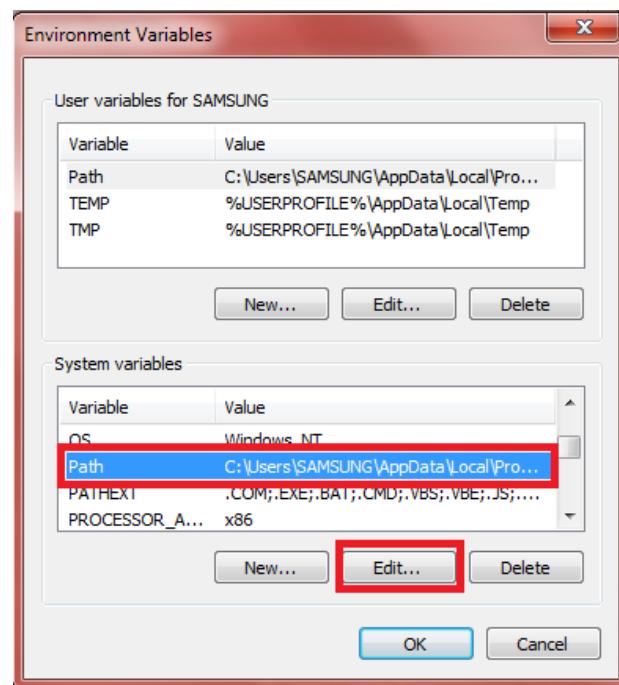
7. Kita akan mengcopykan PATH folder scripts kedalam Environment Variables agar kita dapat menggunakan perintah pip untuk instalasi library pandas.
8. Setelah kita mendapatkan path nya lalu kita buka kembali menu environment variables lalu kita pilih tombol edit pada menu path seperti pada gambar berikut.



9. Lalu kita copykan PATH folder Scripts tadi pada Variable Value seperti pada gambar berikut, lalu klik OK.



10. Copykan kembali PATH folder Scripts tadi pada System variable seperti pada gambar berikut, lalu klik OK.



11. Kita kembali lagi pada terminal VSCode untuk instalasi library nya, tapi sebelum itu kita rubah dulu direktori nya dengan perintah “**cd C:\Users\NAMAUSER\**”, cd adalah perintah untuk merubah direktori, C:\Users\NAMAUSER\ adalah tujuan direktori nya, seperti gambar betikut. Lalu tekan tombol ENTER.

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> cd C:\Users\SAMSUNG
PS C:\Users\SAMSUNG>
```

12. Berikutnya adalah kita akan instalasi library pandas, tapi sebelum itu pastikan laptop sudah terkoneksi dengan internet karena kita akan mendownload library python. Ketikan perintah pip install pandas pada terminal dan tunggu proses download + instalasi library selesai.

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> cd C:\Users\SAMSUNG
PS C:\Users\SAMSUNG> pip install pandas
Collecting pandas
  Downloading pandas-1.3.0-cp38-cp38-win32.whl (9.1 MB)
     |██████████| 2.9 MB 344 kB/s eta 0:00:18
```

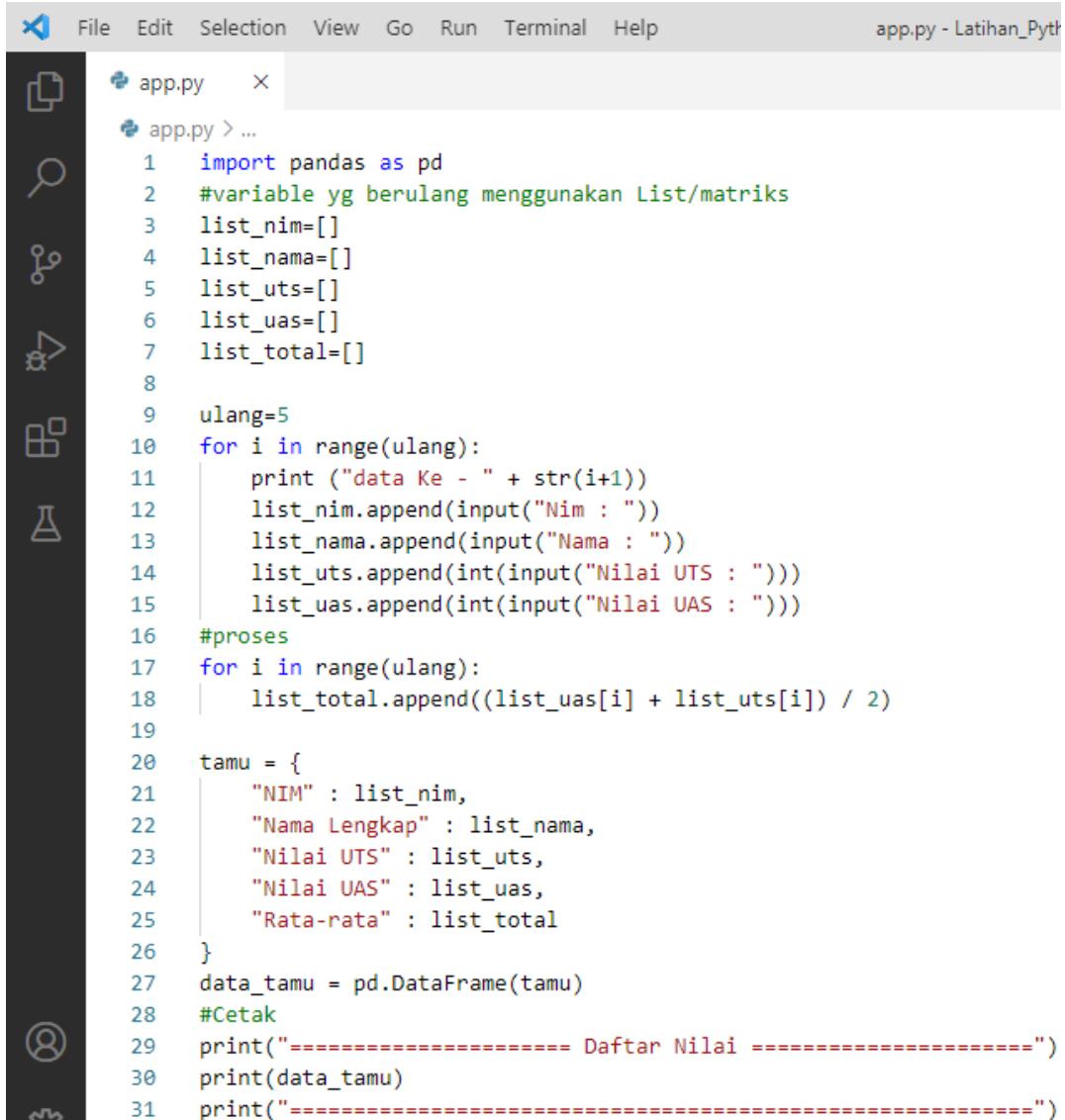
The screenshot shows a Visual Studio Code interface with a terminal window open. The terminal is running in PowerShell mode, indicated by the 'powershell' tab in the top right. The command entered is 'pip install pandas'. The output shows the download and installation process for the pandas library, which is already satisfied as it is included in the Python standard library.

```
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> cd C:\Users\SAMSUNG
PS C:\Users\SAMSUNG> pip install pandas
Collecting pandas
  Downloading pandas-1.3.0-cp38-cp38-win32.whl (9.1 MB)
    [██████████] | 9.1 MB 1.1 MB/s
Requirement already satisfied: numpy>=1.17.3 in c:\users\samsung\appdata\local\programs\python\python38-32\lib\site-packages (from pandas) (1.21.0)
Requirement already satisfied: pytz>=2017.3 in c:\users\samsung\appdata\local\programs\python\python38-32\lib\site-packages (from pandas) (2021.1)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\samsung\appdata\local\programs\python\python38-32\lib\site-packages (from pandas) (2.8.1)
Requirement already satisfied: six>=1.5 in c:\users\samsung\appdata\local\programs\python\python38-32\lib\site-packages (from python-dateutil>=2.7.3->pandas) (1.15.0)
Installing collected packages: pandas
  Successfully installed pandas-1.3.0
PS C:\Users\SAMSUNG>
```

Setelah semua tahapan selesai maka kita dapat menggunakan library pandas.

## 7.2.2 Contoh program dengan penggunaan modul Pandas.



```
File Edit Selection View Go Run Terminal Help
app.py - Latihan_Pyth
app.py > ...
1 import pandas as pd
2 #variable yg berulang menggunakan List/matriks
3 list_nim=[]
4 list_nama=[]
5 list_uts=[]
6 list_uas=[]
7 list_total=[]
8
9 ulang=5
10 for i in range(ulang):
11     print ("data Ke - " + str(i+1))
12     list_nim.append(input("Nim : "))
13     list_nama.append(input("Nama : "))
14     list_uts.append(int(input("Nilai UTS : ")))
15     list_uas.append(int(input("Nilai UAS : ")))
16 #proses
17 for i in range(ulang):
18     list_total.append((list_uas[i] + list_uts[i]) / 2)
19
20 tamu = {
21     "NIM" : list_nim,
22     "Nama Lengkap" : list_nama,
23     "Nilai UTS" : list_uts,
24     "Nilai UAS" : list_uas,
25     "Rata-rata" : list_total
26 }
27 data_tamu = pd.DataFrame(tamu)
28 #Cetak
29 print("===== Daftar Nilai =====")
30 print(data_tamu)
31 print("===== ===== =====")
```

The screenshot shows a terminal window in Visual Studio Code (VS Code) displaying the output of a Python script. The terminal tab is selected at the top. The output shows five student records (data Ke - 1 to 5) and a summary table.

```
data Ke - 1
Nim : 12216695
Nama : Ilham Kurniawan
Nilai UTS : 80
Nilai UAS : 70
data Ke - 2
Nim : 12218567
Nama : Dwi Cahyo
Nilai UTS : 60
Nilai UAS : 40
data Ke - 3
Nim : 12217745
Nama : Khaerul Anam
Nilai UTS : 90
Nilai UAS : 60
data Ke - 4
Nim : 12213398
Nama : Milah Jamilah
Nilai UTS : 90
Nilai UAS : 95
data Ke - 5
Nim : 12219942
Nama : Siti Romlah
Nilai UTS : 70
Nilai UAS : 90
===== Daftar Nilai =====
    NIM      Nama Lengkap  Nilai UTS  Nilai UAS  Rata-rata
0  12216695  Ilham Kurniawan      80       70      75.0
1  12218567      Dwi Cahyo      60       40      50.0
2  12217745      Khaerul Anam      90       60      75.0
3  12213398      Milah Jamilah      90       95      92.5
4  12219942      Siti Romlah      70       90      80.0
=====
```

PS D:\IMK\Latihan Python>

**Pertemuan 8**  
**UTS**

## Pertemuan 9

### Fungsi

Fungsi adalah grup/blok program untuk melakukan tugas tertentu yang berulang. Fungsi membuat kode program menjadi reusable, artinya hanya di definisikan sekali saja, dan kemudian bisa digunakan berulang kali dari tempat lain di dalam program.

Fungsi memecah keseluruhan program menjadi bagian – bagian yang lebih kecil . Dengan semakin besarnya program, maka fungsi akan membuatnya menjadi lebih mudah diorganisir dan dimanage.

Sejauh ini, kita sudah menggunakan beberapa fungsi, misalnya fungsi print(), type(), dan sebagainya. Fungsi tersebut adalah fungsi bawaan dari Python. Kita bisa membuat fungsi kita sendiri sesuai kebutuhan.

#### 9.1 Mendefinisikan Fungsi

Berikut adalah sintaks yang digunakan untuk membuat fungsi:

```
def function_name(parameters):
    """function_docstring"""
    statement(s)

    return [expression]
```

Penjelasannya dari sintaks fungsi di atas:

1. Kata kunci def diikuti oleh function\_name (nama fungsi), tanda kurung dan tanda titik dua (:) menandai header (kepala) fungsi.
2. Parameter / argumen adalah input dari luar yang akan diproses di dalam tubuh fungsi.
3. "function\_docstring" bersifat opsional, yaitu sebagai string yang digunakan untuk dokumentasi atau penjelasan fungsi. "function\_docstring" diletakkan paling atas setelah baris def.
4. Setelah itu diletakkan baris – baris pernyataan (statements). Jangan lupa indentasi untuk menandai blok fungsi.
5. return bersifat opsional. Gunanya adalah untuk mengembalikan suatu nilai expression dari fungsi.

Berikut adalah contoh fungsi untuk menyapa seseorang.

```

app.py > ...
1 def sapa(nama):
2     """Fungsi ini untuk menyapa seseorang sesuai nama yang dimasukkan sebagai parameter"""
3     print("Hi, " + nama + ". Apa kabar?")
4
5 # pemanggilan fungsi
6 # output: Hi, Umar. Apa kabar?
7 sapa('Umar')

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Windows PowerShell  
Copyright (C) 2009 Microsoft Corporation. All rights reserved.  
PS D:\IMK\Latihan\_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38-32/python.exe d:/IMK/sapa.py  
Hi, Umar. Apa kabar?  
PS D:\IMK\Latihan\_Python>

## 9.2 Memanggil Fungsi

Bila fungsi sudah didefinisikan, maka ia sudah bisa dipanggil dari tempat lain di dalam program. Untuk memanggil fungsi caranya adalah dengan mengetikkan nama fungsi berikut paramaternya. Untuk fungsi di atas, kita bisa melakukannya seperti contoh berikut:

```

>>> sapa('Galih')
Hi, Galih. Apa kabar?

>>> sapa('Ratna')
Hi, Ratna. Apa kabar?

```

### a. Docstring

Docstring adalah singkatan dari documentation string. Ini berfungsi sebagai dokumentasi atau keterangan singkat tentang fungsi yang kita buat. Meskipun bersifat opsional, menuliskan docstring adalah kebiasaan yang baik. Untuk contoh di atas kita menuliskan docstring. Cara mengaksesnya adalah dengan menggunakan format namafungsi.\_\_doc\_\_

```

>>> print(sapa.__doc__)
"""Fungsi ini untuk menyapa seseorang sesuai nama yang dimasukkan sebagai parameter"""

```

### 3. Pernyataan Return

Pernyataan return digunakan untuk keluar dari fungsi dan kembali ke baris selanjutnya dimana fungsi dipanggil.

Adapun sintaks dari return adalah:

```
return [expression_list]
```

return bisa berisi satu atau beberapa ekspresi atau nilai yang dievaluasi dan nilai tersebut akan dikembalikan. Bila tidak ada pernyataan return yang dibuat atau ekspresi dikosongkan, maka fungsi akan mengembalikan objek None. Perhatikan bila hasil keluaran dari fungsi sapa kita simpan dalam variabel.

```
>>> keluaran = sapa('Gani')
>>> print(keluaran) None
```

#### 4. Argumen Fungsi

Kita bisa memanggil fungsi dengan menggunakan salah satu dari empat jenis argumen berikut:

- Argumen wajib (required argument)

Argumen wajib adalah argumen yang dilewatkan ke dalam fungsi dengan urutan posisi yang benar. Di sini, jumlah argumen pada saat pemanggilan fungsi harus sama persis dengan jumlah argumen pada pendefinisian fungsi. Pada contoh fungsi sapa() di atas, kita perlu melewatkannya satu argumen ke dalam fungsi sapa(). Bila tidak, maka akan muncul error.

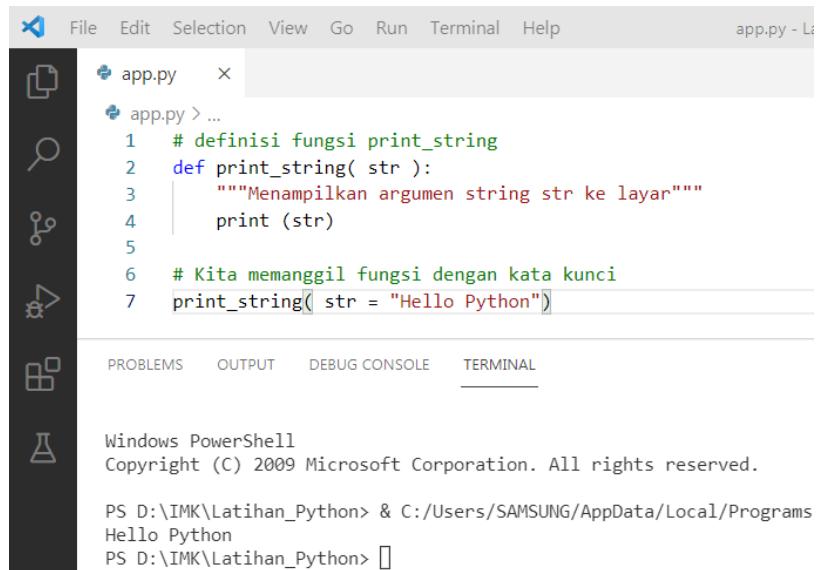
```
>>> sapa('Umar')
Hi Umar. Apa kabar?
```

```
>>> # akan muncul error
>>> sapa()
```

```
Traceback (most recent call last):
  File "<pyshell#5>", line 1, in <module>
    sapa()
TypeError: sapa() missing 1 required positional argument: 'nama'
```

- Argumen kata kunci (keyword argument)

Argumen dengan kata kunci berkaitan dengan cara pemanggilan fungsi. Ketika menggunakan argumen dengan kata kunci, fungsi pemanggil menentukan argumen dari nama parameternya. Hal ini membuat kita bisa mengabaikan argumen atau menempatkannya dengan sembarang urutan. Python dapat menggunakan kata kunci yang disediakan untuk mencocokkan nilai sesuai dengan parameternya. Jelasnya ada pada contoh berikut:



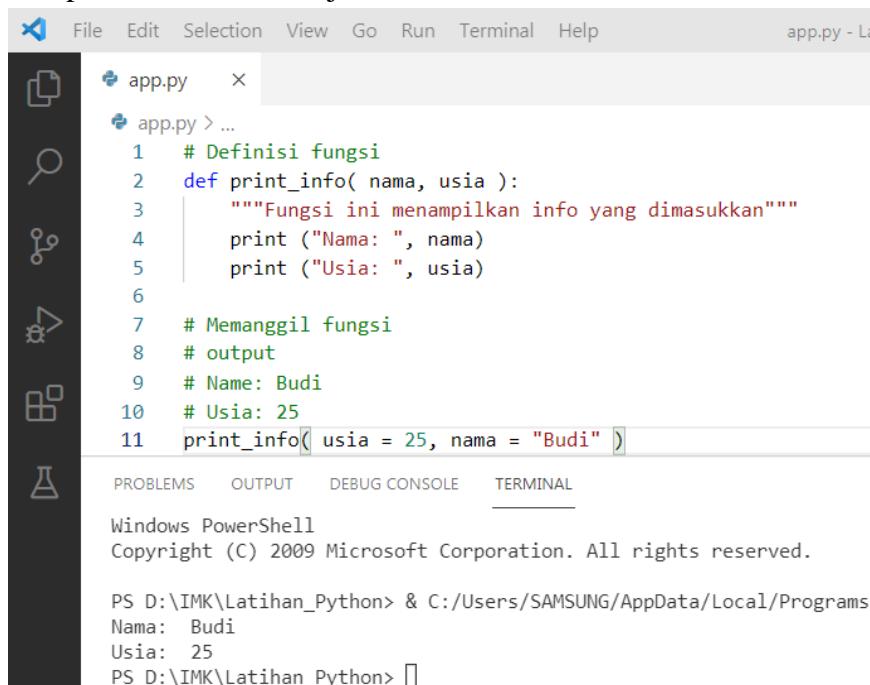
```
File Edit Selection View Go Run Terminal Help
app.py - L: 1 C: 1 I: 1 S: 1 E: 1
app.py    x
app.py > ...
1 # definisi fungsi print_string
2 def print_string( str ):
3     """Menampilkan argumen string str ke layar"""
4     print (str)
5
6 # Kita memanggil fungsi dengan kata kunci
7 print_string( str = "Hello Python" )

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Hello Python
PS D:\IMK\Latihan_Python>
```

Urutan parameter tidak menjadi masalah. Perhatikan contoh berikut:



```
File Edit Selection View Go Run Terminal Help
app.py - L: 1 C: 1 I: 1 S: 1 E: 1
app.py    x
app.py > ...
1 # Definisi fungsi
2 def print_info( nama, usia ):
3     """Fungsi ini menampilkan info yang dimasukkan"""
4     print ("Nama: ", nama)
5     print ("Usia: ", usia)
6
7 # Memanggil fungsi
8 # output
9 # Name: Budi
10 # Usia: 25
11 print_info( usia = 25, nama = "Budi" )

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Nama: Budi
Usia: 25
PS D:\IMK\Latihan_Python>
```

- Argumen default

Fungsi dengan argumen default menggunakan nilai default untuk argumen yang tidak diberikan nilainya pada saat pemanggilan fungsi. Pada contoh berikut, fungsi akan menampilkan usia default bila argumen usia tidak diberikan:

The screenshot shows the Visual Studio Code interface. On the left is a dark sidebar with icons for file operations, search, and other development tools. The main area has a tab for 'app.py' which contains the following Python code:

```
1 # Definisi fungsi
2 def print_info( nama, usia= 17 ):
3     """Fungsi ini menampilkan info yang dimasukkan"""
4     print ("Nama: ", nama)
5     print ("Usia ", usia)
6
7 # Memanggil fungsi print_info
8 print_info( usia = 29, nama = "Galih" )
9
10 # Pemanggilan fungsi tidak menyediakan argumen usia
11 print_info( nama = "Galih" )
```

Below the code editor are tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL'. The 'TERMINAL' tab is active, showing the output of running the script in a Windows PowerShell:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Nama: Galih
Usia 29
Nama: Galih
Usia 17
PS D:\IMK\Latihan_Python>
```

Pada contoh di atas, pemanggilan fungsi kedua tidak menyediakan nilai untuk parameter usia, sehingga yang digunakan adalah nilai default yaitu 17.

- Argumen dengan panjang sembarang

Terkadang kita butuh untuk memproses fungsi yang memiliki banyak argumen. Nama – nama argumennya tidak disebutkan saat pendefinisian fungsi, beda halnya dengan fungsi dengan argumen wajib dan argumen default. Sintaksnya fungsi dengan argumen panjang sembarang adalah seperti berikut:

```
def function_name([formal_args,] *var_args_tuple):
    """function_docstring"""
    statement(s)
    return [expression]
```

Tanda asterisk (\*) ditempatkan sebelum nama variabel yang menyimpan nilai dari semua argumen yang tidak didefinisikan. Tuple ini akan kosong bila tidak ada argumen tambahan pada saat pemanggilan fungsi. Berikut adalah contohnya:

The screenshot shows the Visual Studio Code interface. On the left is a dark sidebar with various icons for file operations like copy, search, and refresh. The main area has a light background. At the top, there's a menu bar with File, Edit, Selection, View, Go, Run, Terminal, Help, and a tab labeled 'app.py - Latihan\_Python - Visual'. Below the menu is a code editor window titled 'app.py' containing the following Python code:

```
1 # Definisi fungsi
2 def print_info( arg1, *vartuple ):
3     """Fungsi untuk menampilkan nilai argumen sembarang yang dilewatkan"""
4     print ("Outputnya adalah: ")
5     print (arg1)
6     for var in vartuple:
7         print (var)
8
9 # Pemanggilan fungsi
10 # Satu argumen
11 print_info( 10 )
12
13 # Empat argumen
14 print_info( 10, 30, 50, 70 )
```

Below the code editor are four tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing the command prompt PS D:\IMK\Latihan\_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38. It then displays the output of the program:

```
Outputnya adalah:
10
Outputnya adalah:
10
30
50
70
PS D:\IMK\Latihan_Python>
```

## 5. Ruang Lingkup (Scope) Variabel

Di Python, tidak semua variabel bisa diakses dari semua tempat. Ini tergantung dari tempat dimana kita mendefinisikan variabel. Ruang lingkup variabel ada dua, yaitu:

- Global
- Local

Variabel yang didefinisikan di dalam fungsi memiliki scope lokal, sedangkan variabel yang didefinisikan di luar fungsi memiliki scope global. Ini berarti, variabel lokal hanya bisa diakses dari dalam fungsi di mana ia di definisikan, sedangkan variabel global bisa diakses dari seluruh tempat dimanapun di dalam program. Berikut adalah contohnya:

The screenshot shows the Visual Studio Code interface. On the left is a dark sidebar with icons for file operations, search, and other development tools. The main area has a tab bar with 'app.py' and a preview of the code. The code itself is:

```
2 # Variabel global
3 # Definisi fungsi
4 def sum( arg1, arg2 ):
    """Menambahkan variabel dan mengembalikan hasilnya."""
    total = arg1 + arg2;
    # total di sini adalah variabel lokal
    print ("Di dalam fungsi nilai total : ", total)
    return total
10
11 # Pemanggilan fungsi sum
12 sum( 10, 20 )
13 print ("Di luar fungsi, nilai total : ", total )
```

Below the code editor is a navigation bar with tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, with TERMINAL being the active tab. The terminal window displays the following output:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Di dalam fungsi nilai total : 30
Di luar fungsi, nilai total : 0
PS D:\IMK\Latihan_Python>
```

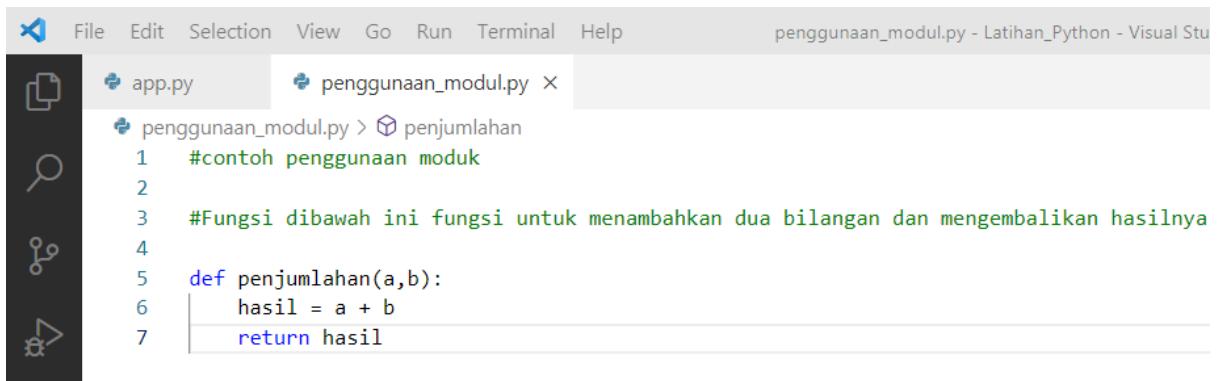
Perhatikan bagaimana variabel total di dalam dan di luar fungsi adalah dua variabel yang berbeda.

## Pertemuan 10

### Modul dan Eksepsi

Modul adalah sebuah file dengan ekstensi .py, yang berisikan kode program dalam bahasa pemrograman python. Nama modul sesuai dengan nama file yang digunakan. misalnya: contoh\_penggunaan\_modul.py, disebut modul dan nama modulnya adalah contoh\_penggunaan\_modul. Ada dua syntax untuk menggunakan sebuah modul, pertama menggunakan *import* dan kedua menggunakan *from, bisa juga dengan menggunakan keduanya*.

Python memiliki banyak modul bawaan, misalnya modul **math**, **os**, **sys** dan lain sebagainya. Modul – modul tersebut berada di dalam direktori Lib ditempat Python terinstall. Python juga memiliki ribuan modul siap pakai yang tersedia luas di internet, salah satunya di [pypi.python.org](https://pypi.python.org). Modul digunakan untuk memecah sebuah program besar menjadi file – file yang lebih kecil agar lebih mudah *dikelola*. Modul membuat kode bersifat *reusable*, artinya satu modul bisa dipakai berulang dimana saja jika diperlukan. Berikut ini adalah contoh penggunaan modul.



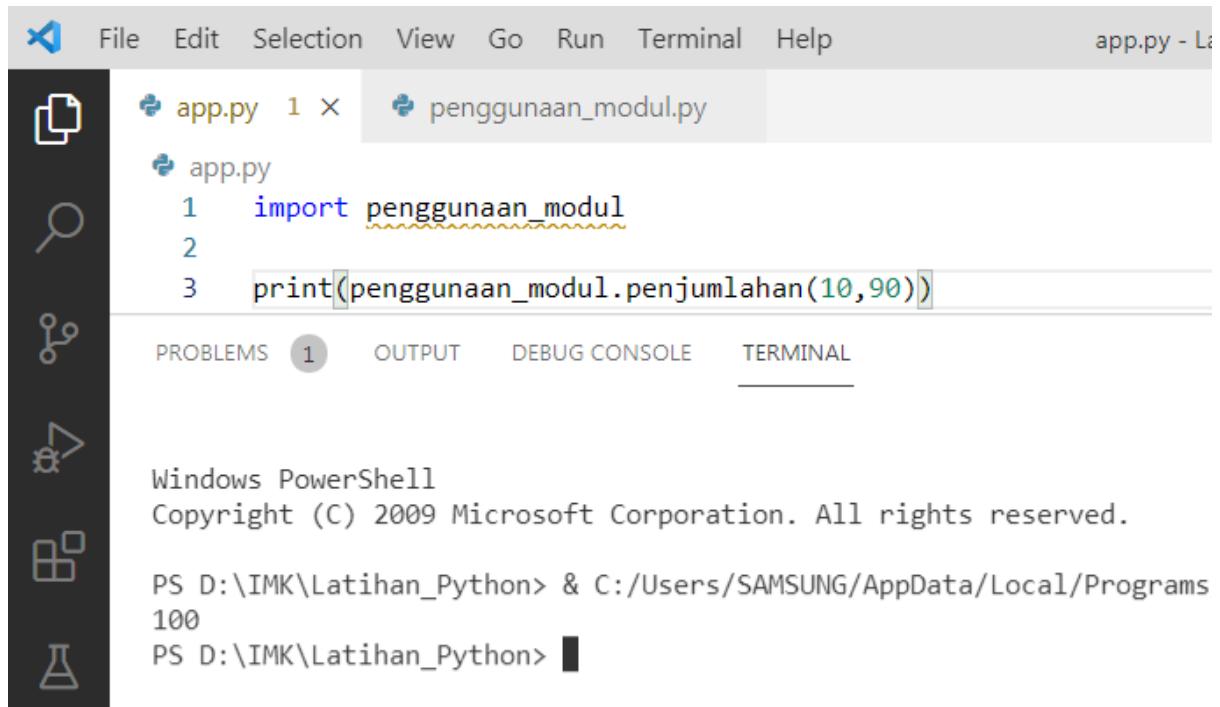
```
File Edit Selection View Go Run Terminal Help penggunaan_modul.py - Latihan_Python - Visual Studio Code
app.py penggunaan_modul.py
penggunaan_modul.py > penjumlahan
1 #contoh penggunaan moduk
2
3 #Fungsi dibawah ini fungsi untuk menambahkan dua bilangan dan mengembalikan hasilnya
4
5 def penjumlahan(a,b):
6     hasil = a + b
7     return hasil
```

#### 10.1 Menggunakan Perintah Import

Kita bisa mengimpor modul python ke dalam program yang kita buat. Dengan mengimpor modul, maka definisi, variabel, fungsi dan yang lainnya yang ada di dalam modul itu bisa kita pergunakan kapan saja jika diperlukan. Kita mengimpor modul dengan menggunakan kata kunci **import**. Misalnya, kita akan mengimpor modul **contoh\_penggunaan\_modul** yang sudah kita buat di atas, maka kita bisa mengetikkan perintah berikut :

```
import contoh_penggunaan_modul
```

Setelah kita import, maka kita bisa mengakses isi dari modul **contoh\_penggunaan\_modul**. Kita bisa mengakses fungsi maupun variabel global di dalam modul tersebut dengan menggunakan operasi titik (.). Misalnya sebagai berikut:



The screenshot shows the Visual Studio Code interface. In the top navigation bar, the tabs 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', 'Terminal', and 'Help' are visible. The active tab is 'Terminal'. The status bar at the bottom right shows 'app.py - La'. The left sidebar has icons for file explorer, search, and other development tools. The main area shows two files: 'app.py' and 'penggunaan\_modul.py'. The code in 'app.py' is:

```
import penggunaan_modul
print(penggunaan_modul.penjumlahan(10,90))
```

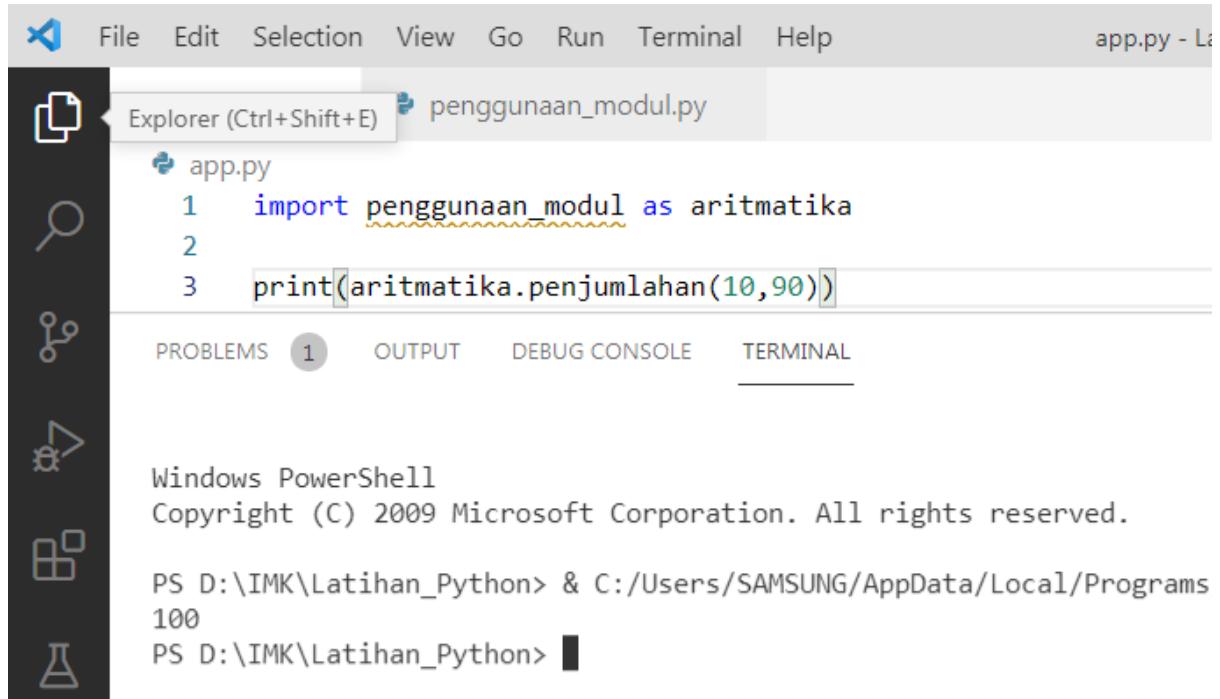
The terminal window below shows the output of running the script:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
100
PS D:\IMK\Latihan_Python>
```

## 10.2 Menggunakan Perintah Alias pada Modul

Untuk memanggil suatu modul pada python, kita juga bisa memanggil modul tersebut dengan merubah namanya dengan perintah **as**, misalkan pada contoh dibawah ini :



The screenshot shows the Visual Studio Code interface. The top navigation bar and status bar are identical to the previous screenshot. The terminal window shows the command to run the script:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
```

The code in 'app.py' uses an alias:

```
import penggunaan_modul as aritmatika
print(aritmatika.penjumlahan(10,90))
```

The terminal output shows the result of the calculation:

```
100
PS D:\IMK\Latihan_Python>
```

## Cara Lain untuk Mengimpor Modul

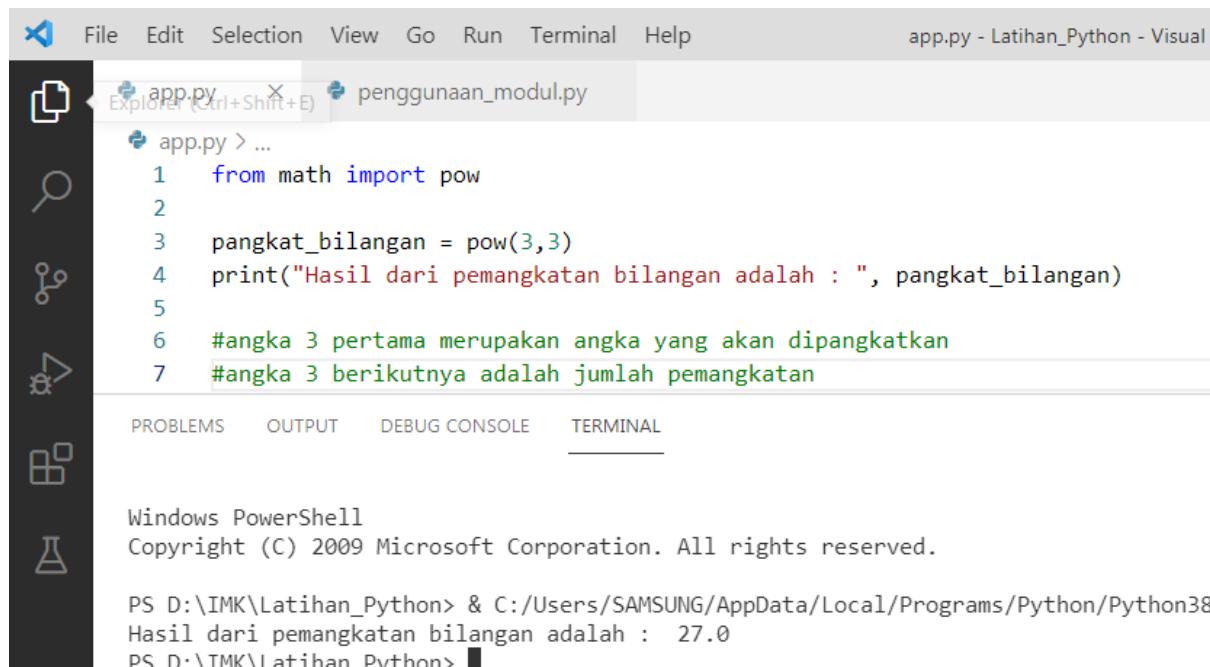
Ada beberapa sintaks yang bisa digunakan untuk mengimpor modul, yaitu sebagai berikut:

- Cara import standar, formatnya **import nama\_modul**
- Cara import dengan rename (alias), formatnya **import nama\_modul as alias**

- Cara mengimport sebagian, formatnya `from...import` something
- Cara mengimport semua isi modul, formatnya `import *`

### 10.3 Mengimport sebagian fungsi modul pada Python

Pada python kita juga bisa menggunakan sebagian fungsi dari suatu modul dengan perintah `from... import...` seperti pada contoh dibawah ini :



The screenshot shows the Visual Studio Code interface. The left sidebar has icons for file explorer, search, symbols, and terminal. The top bar shows "File Edit Selection View Go Run Terminal Help" and the file "app.py - Latihan\_Python - Visual Studio Code". The Explorer tab is selected, showing files "app.py" and "penggunaan\_modul.py". The code editor contains the following Python code:

```

app.py > ...
1  from math import pow
2
3  pangkat_bilangan = pow(3,3)
4  print("Hasil dari pemangkatan bilangan adalah : ", pangkat_bilangan)
5
6  #angka 3 pertama merupakan angka yang akan dipangkatkan
7  #angka 3 berikutnya adalah jumlah pemangkatan

```

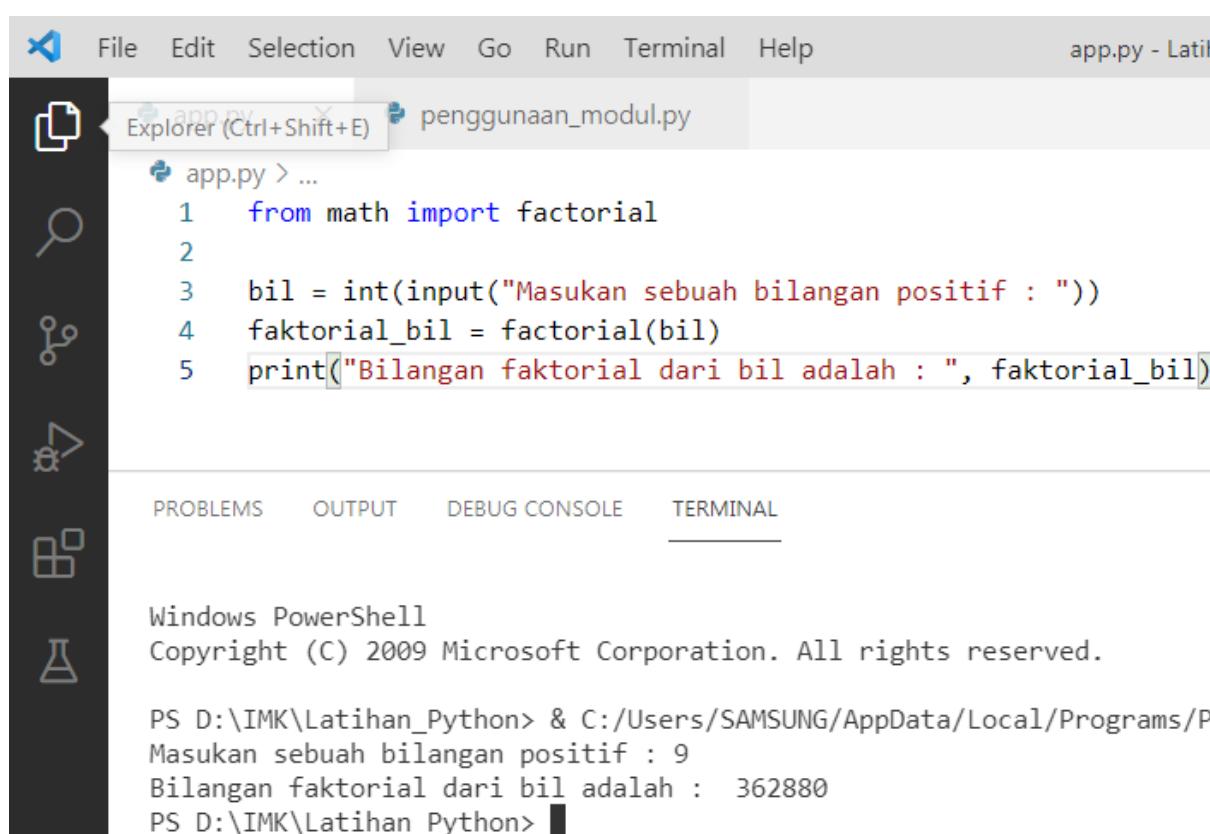
Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing Windows PowerShell output:

```

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38
Hasil dari pemangkatan bilangan adalah :  27.0
PS D:\IMK\Latihan_Python>

```



The screenshot shows the Visual Studio Code interface. The left sidebar has icons for file explorer, search, symbols, and terminal. The top bar shows "File Edit Selection View Go Run Terminal Help" and the file "app.py - Latihan\_Python - Visual Studio Code". The Explorer tab is selected, showing files "app.py" and "penggunaan\_modul.py". The code editor contains the following Python code:

```

app.py > ...
1  from math import factorial
2
3  bil = int(input("Masukan sebuah bilangan positif : "))
4  faktorial_bil = factorial(bil)
5  print("Bilangan faktorial dari bil adalah : ", faktorial_bil)

```

Below the code editor are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is selected, showing Windows PowerShell output:

```

Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/P
Masukan sebuah bilangan positif : 9
Bilangan faktorial dari bil adalah :  362880
PS D:\IMK\Latihan_Python>

```

#### 10.4 Mengimpor semua fungsi modul pada Python

Jika pada contoh sebelumnya menggunakan sebagian fungsi dari suatu modul pada python, maka contoh kali ini kita dapat menggunakan semua fungsi pada suatu modul, seperti pada contoh dibawah ini :

The screenshot shows the Visual Studio Code interface. In the top right, it says "app.py - Latihan\_Python - Visual Studio Code". The left sidebar has icons for file operations. The main area shows two tabs: "app.py" and "penggunaan\_modul.py". The "app.py" tab contains the following code:

```
1  from math import*
2  pangkat_bilangan = pow(3, 3)
3  print("Hasil dari pemangkatan bilangan adalah : ", pangkat_bilangan)
4
5  bil= int(input("Masukan sebuah bilangan positif : "))
6  faktorial_bil = factorial(bil)
7  print("Bilangan faktorial dari bil adalah : ", faktorial_bil)
```

Below the code editor are tabs for "PROBLEMS", "OUTPUT", "DEBUG CONSOLE", and "TERMINAL". The "TERMINAL" tab shows the output of running the script:

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs/Python/Python38
Hasil dari pemangkatan bilangan adalah :  27.0
Masukan sebuah bilangan positif : 9
Bilangan faktorial dari bil adalah :  362880
PS D:\IMK\Latihan_Python>
```

Berikut ini merupakan modul-modul yang tersedia dalam python dan funginya :

Tabel Modul

Nama Modul	Fungsinya
__main__	Tempat skrip level teratas dijalankan.
array	Array dengan nilai numerik.
binascii	Alat untuk mengonversi antara representasi biner berenkode ASCII dan biner.
bisect	Algoritma pembagian dua untuk pencarian biner.
calendar	Fungsi untuk kalender, termasuk beberapa emulasi program Unix cal.
cmath	Fungsi matematika untuk bilangan kompleks.
csv	Menulis dan membaca data tabel dan dari file yang dibatasi.
datetime	Fungsi tanggal dan waktu.
decimal	Penerapan Spesifikasi Aritmatika Desimal.

email	Paket yang mendukung parsing, manipulasi, dan pembuatan pesan email.
enum	Implementasi kelas pencacahan.
getpass	Pembacaan password portabel dan pengambilan userid.
html	Fungsi untuk memanipulasi HTML.
http	Kode dan pesan status HTTP
json	Encode dan decode format JSON.
math	Fungsi matematika (sin () dll.).
numbers	Kelas dasar abstrak numerik (Kompleks, Real, Integral, dll.).
random	Hasilkan bilangan pseudo-random dengan berbagai distribusi umum.
statistics	Fungsi statistik matematika
sys	Akses parameter dan fungsi khusus sistem.
time	Akses dan konversi waktu.
warnings	Keluarkan pesan peringatan dan kendalikan disposisi mereka.
zipfile	Membaca dan menulis file arsip berformat ZIP.
zipimport	Dukungan untuk mengimpor modul Python dari arsip ZIP.

## 10.5 Eksepsi

Pada saat menulis dan menjalankan program, kita sering dihadapkan pada munculnya kesalahan atau error. Seringkali error menyebabkan program berhenti sendiri.

Error dapat terjadi akibat kesalahan struktur (sintaks) program. Hal ini disebut syntax error. Contohnya adalah seperti berikut:

```
>>> if x < 5
      File "<stdin>", line 1
          if x < 5
SyntaxError: invalid syntax
```

Kita bisa melihat bahwa penyebabnya adalah lupa titik dua pada pernyataan if.

Error juga dapat terjadi pada saat runtime (saat program berjalan). Error seperti ini disebut eksepsi. Misalnya, bila kita membuka file yang tidak ada, maka akan muncul pesan kesalahan FileNotFoundError. Bila kita membagi bilangan dengan nol akan muncul ZeroDivisionError, dan lain sebagainya.

Pada saat terjadi eksepsi, Python akan menampilkan traceback dan detail dimana kesalahan terjadi.

```
>>> 1/0
```

```

Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero

```

Tabel 10.1 Daftar Eksepsi Built-in Python

<b>Eksepsi</b>	<b>Penyebab Error</b>
AssertionError	Muncul pada saat pernyataan <code>assert</code> gagal
AttributeError	Muncul pada saat penugasan terhadap attribute atau referensi gagal
EOFError	Muncul saat fungsi <code>input()</code> mendapatkan kondisi akhir file (end-of-file)
FloatingPointError	Muncul saat operasi terhadap bilangan float gagal
GeneratorExit	Muncul saat metode <code>close()</code> generator dipanggil
ImportError	Muncul saat modul yang hendak diimpor tidak ditemukan
IndexError	Muncul saat indeks dari sequence berada di luar range
KeyError	Muncul saat suatu key tidak ditemukan di dalam dictionary
KeyboardInterrupt	Muncul saat user menekan tombol interupsi (Ctrl + C)

MemoryError	Muncul saat operasi kehabisan memori
NameError	Muncul saat variabel tidak ditemukan
NotImplementedError	Muncul oleh metode abstrak
OSError	Muncul saat sistem operasi bersangkutan mengalami error
OverflowError	Muncul saat hasil operasi perhitungan terlalu besar untuk direpresentasikan
ReferenceError	Muncul saat <i>weak reference</i> digunakan untuk mengakses referensi sampah program
RuntimeError	Muncul saat error yang terjadi di luar semua kategori eksepsi lain
StopIteration	Muncul oleh fungsi <code>next()</code> untuk menunjukkan bahwa tidak ada lagi item yang tersisa pada iterator
SyntaxError	Muncul oleh parser saat terjadi kesalahan sintaks
IndentationError	Muncul saat ada indentasi yang salah
TabError	Muncul saat indentasi memiliki jumlah spasi atau tab yang tidak konsisten
SystemError	Muncul saat interpreter mendeteksi kesalahan internal

SystemExit	Muncul oleh fungsi <code>sys.exit()</code>
TypeError	Muncul saat melakukan operasi pada tipe data yang tidak sesuai
UnboundLocalError	Muncul saat referensi dibuat untuk variabel lokal dari fungsi, tapi tidak ada nilainya.
UnicodeError	Muncul saat terjadi kesalahan berkenaan dengan encoding dan decoding unicode
UnicodeEncodeError	Muncul saat terjadi kesalahan pada proses encoding
UnicodeDecodeError	Muncul saat terjadi kesalahan pada proses decoding
UnicodeTranslateError	Muncul saat terjadi kesalahan berkenaan dengan penerjemahan unicode
ValueError	Muncul saat fungsi menerima argumen yang tipe datanya salah
ZeroDivisionError	Muncul saat terjadi operasi pembagian bilangan dengan nol

### 3.1. Menangani Eksepsi Dengan Try, Except, dan Finally

Terjadinya eksepsi pada program dapat menyebabkan program terhenti. Untuk mencegah hal tersebut, kita harus mengantisipasi hal tersebut. Python menyediakan metode penanganan eksepsi dengan menggunakan pernyataan try dan except.

Di dalam blok try kita meletakkan baris program yang kemungkinan akan terjadi error. Bila terjadi error, maka penanganannya diserahkan kepada blok except. Berikut adalah contoh penanganan eksepsi pada operasi pembagian bilangan.

```

File Edit Selection View Go Run Terminal Help
app.py x
app.py > ...
2 import sys
3
4 lists = ['a', 0, 4]
5 for each in lists:
6     try:
7         print("Masukan:", each)
8         r = 1/int(each)
9         break
10    except:
11        print("Upps!", sys.exc_info()[0], " terjadi.")
12        print("Masukan berikutnya.")
13        print()
14    print("Kebalikan dari ", each, " =", r)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS D:\IMK\Latihan_Python> & C:/Users/SAMSUNG/AppData/Local/Programs
Masukan: a
Upps! <class 'ValueError'> terjadi.
Masukan berikutnya.

Masukan: 0
Upps! <class 'ZeroDivisionError'> terjadi.
Masukan berikutnya.

Masukan: 4
Kebalikan dari 4 = 0.25
PS D:\IMK\Latihan_Python>

```

Pada program di atas kita mencari kebalikan dari bilangan, misalnya 4, maka kebalikannya adalah  $1/4 = 0.25$ .

Pembagian dengan huruf ‘a’, dan juga dengan 0 tidak bisa dilakukan, sehingga muncul error. Bila tidak dilakukan penanganan eksepsi, maka program akan langsung terhenti pada saat terjadi error.

### 3.2. Menangani Eksepsi Tertentu

Pada contoh di atas kita hanya menangani error secara umum. Tidak dikelompokkan, apakah dia adalah `TypeError`, `ValueError`, `SyntaxError`, dan lain sebagainya. Sebuah pernyataan `try`, bisa memiliki sejumlah pernyataan `except` untuk menangani jenis – jenis eksepsi secara lebih spesifik. Kita juga bisa mendefinisikan beberapa error sekaligus menggunakan tuple. Contohnya adalah seperti berikut:

```

try:
    # lakukan sesuatu
    pass

except ValueError:
    # tangani eksepsi ValueError
    pass

```

```

except (TypeError, ZeroDivisionError):
    # menangani multi eksepsi
    # TypeError dan ZeroDivisionError
    pass

except:
    # menangani eksepsi lainnya
pass

```

Pernyataan `pass` adalah pernyataan yang tidak melakukan apa-apa. Istilahnya adalah statemen kosong. `pass` sering digunakan untuk mengisi blok fungsi atau kelas yang masih kosong.

### 3.3. Memunculkan Eksepsi

Eksepsi muncul bila terjadi error pada saat runtime atau saat program berjalan. Akan tetapi, kita juga bisa memunculkan eksepsi dengan sengaja untuk maksud tertentu dengan menggunakan kata kunci `raise`. Contohnya adalah seperti berikut:

```

>>> raise KeyboardInterrupt
Traceback (most recent call last):
...
KeyboardInterrupt

>>> try:
        a = int(input("Masukkan sebuah bilangan positif: "))
        if a <= 0:
            raise ValueError("Itu bukan bilangan positif!")
    except ValueError as ve:
        print(ve)

```

```

Masukkan sebuah bilangan positif: -3
Itu bukan bilangan positif!

```

## Pertemuan 11

### Object-Oriented Programming (OOP)

**OOP** (Object Oriented Programming) adalah suatu metode pemrograman yang berorientasi kepada objek. **Tujuan dari OOP diciptakan** adalah untuk mempermudah pengembangan program dengan cara mengikuti model yang telah ada di kehidupan sehari-hari. Jadi setiap bagian dari suatu permasalahan adalah objek, nah objek itu sendiri merupakan gabungan dari beberapa objek yang lebih kecil lagi.

Sebagai salah satu contoh Pesawat, Pesawat adalah sebuah objek. Pesawat itu sendiri terbentuk dari beberapa objek yang lebih kecil lagi seperti mesin, roda, baling-baling, kursi, dll. Pesawat sebagai objek yang terbentuk dari objek-objek yang lebih kecil saling berhubungan, berinteraksi, berkomunikasi dan saling mengirim pesan kepada objek-objek yang lainnya. Begitu juga dengan program, sebuah objek yang besar dibentuk dari beberapa objek yang lebih kecil, objek-objek itu saling berkomunikasi, dan saling berkirim pesan kepada objek yang lain.

Tabel 11.1 Istilah Dalam OOP

Istilah	Penjelasan
Class	Prototipe yang ditentukan pengguna untuk objek yang mendefinisikan seperangkat atribut yang menjadi ciri objek kelas apa pun. Atribut adalah data anggota (variabel kelas dan variabel contoh) dan metode, diakses melalui notasi titik.
Object	Contoh unik dari struktur data yang didefinisikan oleh kelasnya. Objek terdiri dari kedua anggota data (variabel kelas dan variabel contoh) dan metode.
Method	Jenis fungsi khusus yang didefinisikan dalam definisi kelas.
Instance	Objek individu dari kelas tertentu. Objek yang termasuk dalam Lingkaran kelas, misalnya, adalah turunan dari Lingkaran kelas.
Instantiation	Penciptaan sebuah instance dari sebuah kelas.
Class variable	Sebuah variabel yang dibagi oleh semua contoh kelas. Variabel kelas didefinisikan dalam kelas tapi di luar metode kelas manapun. Variabel kelas tidak digunakan sesering variabel contoh.
Data member	Variabel kelas atau variabel contoh yang menyimpan data yang terkait dengan kelas dan objeknya.
Function overloading	Penugasan lebih dari satu perilaku ke fungsi tertentu. Operasi yang dilakukan bervariasi menurut jenis objek atau argumen yang terlibat.
Operator overloading	Penugasan lebih dari satu fungsi ke operator tertentu.
Inheritance	Pengalihan karakteristik kelas ke kelas lain yang berasal darinya.
Instantiation	Penciptaan sebuah instance dari sebuah kelas.

#### 11.1 Konsep OOP

- **Kelas** merupakan deskripsi abstrak informasi dan tingkah laku dari sekumpulan data.
- **Kelas** dapat diilustrasikan sebagai suatu cetak biru(blueprint) atau prototipe yang digunakan untuk menciptakan objek.

- **Kelas** merupakan tipe data bagi objek yang mengenkapsulasi data dan operasi pada data dalam suatu unit tunggal.
- **Kelas** mendefinisikan suatu struktur yang terdiri atas data kelas (data field), prosedur atau fungsi (method), dan sifat kelas (property).

## 11.2 Class

Pada konsep pemrograman berbasis object, anda tidak akan asing lagi mendengar istilah class, object, attribute, behaviour, inheritance, dll. Semua itu pasti akan anda temui disemua bahasa pemrograman yang support OOP. Jika dianalogikan, class merupakan suatu tubuh dari OOP. Class merupakan abstraksi atau *blueprint* yang mendefinisikan suatu object tertentu. Class akan menampung semua attribute dan perilaku dari object itu. Berikut contoh implementasi class pada Python:

```
class Car:

    color = 'black'
    transmission = 'manual'

    def __init__(self, transmission):
        self.transmission = transmission
        print('Engine is ready!')

    def drive(self):
        print('Drive')

    def reverse(self):
        print('Reverse. Please check your behind.')

Jika diperhatikan, dalam class Car terdapat 2 attribute yaitu color = 'black', transmission = 'manual' dan method yaitu drive(), reverse(). Method dalam konsep OOP mewakili suatu 'behaviour' dari class atau object itu sendiri. Kita akan bahas lebih detail mengenai method.
```

## 11.3 Function/Method

Fungsi method dalam konsep OOP adalah untuk merepresentasikan suatu behaviour. Dalam contoh di atas suatu object 'mobil' memiliki behaviour antara lain adalah bergerak dan mundur. Suatu method bisa juga memiliki satu atau beberapa parameter, sebagai contoh:

```
gear_position = 'N'

def change_gear(self, gear):
    self.gear_position = gear
    print('Gear position on: ' + self.gear_position)
```

Pada method **change\_gear()** terdapat 1 parameter yaitu **gear**. Ketika method tersebut dipanggil dan anda tidak memberikan value pada parameter tersebut, maka program akan melempar error. Bagaimanapun juga parameter yang sudah didefinisikan pada suatu method harus memiliki value meskipun value tersebut **None**. Cara lainnya adalah dengan mendefinisikan default value pada parameter tersebut sejak awal method tersebut dibuat:

```

gear_position = 'N'

def change_gear(self, gear='N'):
    self.gear_position = gear
    print('Gear position on: ' + self.gear_position)

self.change_gear()
>>> 'Gear position on: N'
self.change_gear('R')
>>> 'Gear position on: R'

```

Jika diperhatikan, terdapat keyword **self** pada salah satu parameter method di atas. Keyword **self** mengacu pada Class Instance untuk mengakses attribute atau method dari class itu sendiri. Dalam bahasa pemrograman Java, terdapat keyword **this** yang memiliki fungsi yang mirip dengan keyword **self** pada Python. Pemberian keyword **self** pada parameter awal suatu method menjadi wajib jika anda mendefinisikan method tersebut di dalam block suatu class.

Suatu method juga bisa mengembalikan suatu value ketika method tersebut dipanggil. Berikut contoh implementasinya:

```

def get_gear_position(self):
    return self.gear_position

gear_position = self.get_gear_position()

```

## 11.4 Constructor

Pada contoh awal tentang penjelasan class, terdapat sebuah method bernama **\_\_init\_\_**. Method itulah yang disebut dengan constructor. Suatu constructor berbeda dengan method lainnya, karena constructor akan otomatis dieksekusi ketika membuat object dari class itu sendiri.

```

class Car:
    color = 'black'
    transmission = 'manual'

    def __init__(self, transmission):
        self.transmission = transmission
        print('Engine is ready!')

    ...

honda = Car('automatic')
>>> 'Engine is ready!'

```

Ketika object **honda** dibuat dari class **Car**, constructor langsung dieksekusi. Hal ini berguna jika anda membutuhkan proses inisialisasi ketika suatu object dibuat. Suatu constructor juga bisa memiliki satu atau beberapa parameter, sama seperti method pada umumnya namun constructor tidak bisa mengembalikan value.

## 11.5 Object

Object merupakan produk hasil dari suatu class. Jika class merupakan blueprint dari suatu rancangan bangunan, maka object adalah bangunan itu sendiri. Begitulah contoh analogi yang bisa saya gambarkan mengenai relasi antara class dan object. Berikut contoh implementasi dalam bentuk code program:

```
class Car:

    color = 'black'
    transmission = 'manual'
    gear_position = 'N'

    def __init__(self, transmission):
        self.transmission = transmission
        print('Engine is ready!')

    def drive(self):
        self.gear_position = 'D'
        print('Drive')

    def reverse(self):
        self.gear_position = 'R'
        print('Reverse. Please check your behind.')

    def change_gear(self, gear='N'):
        self.gear_position = gear
        print('Gear position on: ' + self.gear_position)

    def get_gear_position(self):
        return self.gear_position

car1 = Car('manual')
car1.change_gear('D-1')

car2 = Car('automatic')
gear_position = car2.get_gear_position()
print(gear_position)
>>> 'N'
```

Dari contoh di atas, terdapat 2 buah object `car1` dan `car2` yang dibuat dari class yang sama. Masing-masing dari object tersebut berdiri sendiri, artinya jika terjadi perubahan attribute dari object `car1` tidak akan mempengaruhi object `car2` meskipun dari class yang sama.

## 11.6 Inheritance

Salah satu keuntungan dari konsep OOP ialah *reusable codes* yang bisa mengoptimalkan penggunaan code program agar lebih efisien dan meminimalisir redudansi.

- **Kita dapat mendefinisikan** suatu kelas baru dengan mewarisi sifat dari kelas lain yang sudah ada.
- **Penurunan sifat** ini bisa dilakukan secara bertingkat tingkat, sehingga semakin ke bawah kelas tersebut menjadi semakin spesifik.
- **Sub kelas** memungkinkan kita untuk melakukan spesifikasi detail dan perilaku khusus dari kelas supernya.

- **Dengan konsep pewarisan**, seorang programmer dapat menggunakan kode yang telah ditulisnya pada kelas super berulang kali pada kelas-kelas turunannya tanpa harus menulis ulang semua kode-kode itu.

Semua itu berkat adanya fitur inheritance yang memungkinkan suatu class (parent) menurunkan semua attribute dan behaviour nya ke class (child) lain. Berikut contoh penerapannya:

```
class Tesla(Car):
    pass    # use 'pass' keyword to define class only

tesla = Tesla()
tesla.drive()
>>> 'Drive'
```

Pada potongan code di atas, class **Tesla** merupakan turunan dari class **Car**. Jika diperhatikan pada class **Tesla** tidak didefinisikan method **drive()** namun class tersebut bisa memanggil method **drive()**. Method tersebut berasal dari class parentnya yaitu class **Car**, sehingga tidak perlu lagi didefinisikan ulang pada class childnya. Dengan cara seperti ini anda bisa melakukan reusable codes sehingga source code menjadi lebih *clean*.

## 11.7 Overriding

Ada suatu kondisi dimana suatu method yang berasal dari parent ingin anda modifikasi atau ditambahkan beberapa fitur sesuai kebutuhan pada class child, disinilah peran dari 'overriding method'. Dengan menggunakan fungsi `super()`, anda bisa memanggil instance dari class parent di dalam suatu method untuk memanggil fungsi dari parent tersebut. Perhatikan contoh di bawah ini:

```
class Tesla(Car):

    def drive(self):
        super().drive()
        print('LOL Gas')
```

## 11.8 Private Attribute/Function

Tidak semua attribute maupun method bisa diturunkan pada class child. Anda bisa menentukan mana attribute atau method yang ingin diproteksi agar tidak bisa digunakan pada class turunannya. Berikut caranya:

```
__factory_number = '0123456789'

def __get_factory_number(self):
    return self.__factory_number
```

## 11.9 Polymorphism

polimorfisme yang memungkinkan anda untuk membuat banyak bentuk dari satu object. Berikut contoh implementasinya:

```
class Car:
```

```
def fuel(self):
    return 'gas'

class Honda(Car):
    pass

class Tesla(Car):
    def fuel(self):
        return 'electricity'

def get_fuel(car):
    print(car.fuel())

get_fuel(Tesla())
get_fuel(Honda())
>>> 'electricity'
>>> 'gas'
```

- **Polimorfisme** merupakan kemampuan objek objek yang berbeda kelas namun terkait dalam pewarisan untuk merespon secara berbeda terhadap suatu pesan yang sama.
- **Polimorfisme** juga dapat dikatakan kemampuan sebuah objek untuk memutuskan method mana yang akan diterapkan padanya, tergantung letak objek tersebut pada jenjang pewarisan.