



Laporan Praktikum Algoritma & Pemrograman

Semester Genap 2025/2026

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

NIM	71251213
Nama Lengkap	Maryo Aurel Nubatonis
Minggu ke / Materi	01 / Bahasa Pemrograman Python

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA
2026**

BAGIAN 1: MATERI MINGGU INI (40%)

Pada bagian ini, tuliskan kembali semua materi yang telah anda pelajari minggu ini. Sesuaikan penjelasan anda dengan urutan materi yang telah diberikan di saat praktikum. Penjelasan anda harus dilengkapi dengan contoh, gambar/ilustrasi, contoh program (source code) dan outputnya. Idealnya sekitar 5-6 halaman.

MATERI 1 Kenapa Python?

Kenapa Python?

Python merupakan bahasa pemrograman tingkat tinggi yang dijalankan menggunakan interpreter. Bahasa ini mendukung konsep *Object Oriented Programming (OOP)* dan memiliki sifat *dynamic semantics*, sehingga lebih fleksibel dalam penulisan kode. Berdasarkan survei Stack Overflow tahun 2019, Python termasuk salah satu bahasa pemrograman yang paling banyak digunakan di dunia, tepat setelah JavaScript.

Salah satu alasan utama Python banyak diminati adalah karena sintaksnya sederhana dan mudah dipahami, terutama bagi pemula. Kesederhanaan ini dapat dilihat dari contoh program *Hello World*. Pada Python, program tersebut cukup ditulis dalam satu baris kode, sedangkan pada bahasa Java dan C diperlukan struktur kode yang lebih panjang. Hal

ini membuat Python lebih efisien dan praktis untuk digunakan. Dan code dibawah ini yang membedakan bahasa Java,C, dan Python.

```
1 public class Main {  
2     public static void main(String[] args){  
3         System.out.println("Hello World!");  
4     }  
5 }
```

Pada bahasa pemrograman C:

```
1 #include <stdio.h>  
2  
3 int main(){  
4     printf("Hello World");  
5     return 0;  
6 }
```

Pada bahasa pemrograman Python:

```
1 print("Hello World!");
```

Kelebihan dan Kekurangan Python

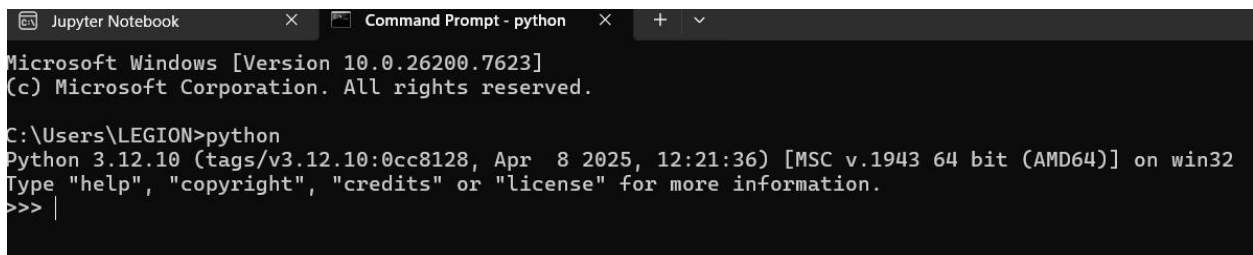
Python memiliki banyak keunggulan, salah satunya adalah dukungan pustaka yang sangat lengkap, baik pustaka bawaan maupun pustaka pihak ketiga. Python banyak dimanfaatkan di berbagai bidang, seperti *data science*, *machine learning*, dan *kecerdasan buatan*, karena tersedia library populer seperti NumPy, Pandas, TensorFlow, dan Keras. Selain itu, Python bersifat *open source*, sehingga dapat digunakan secara bebas, termasuk untuk keperluan komersial. Dokumentasi dan sumber belajar Python juga sangat melimpah, sehingga memudahkan proses pembelajaran.

Namun, Python juga memiliki beberapa kelemahan. Kecepatan eksekusinya relatif lebih lambat jika dibandingkan dengan bahasa

pemrograman seperti C. Selain itu, penggunaan memori Python cenderung lebih besar dan dukungannya untuk pengembangan aplikasi mobile masih terbatas. Meskipun demikian, kekurangan ini umumnya tidak menjadi masalah besar bagi pemula yang masih fokus mempelajari dasar-dasar pemrograman.

Instalasi dan Penggunaan Python

Saat ini terdapat dua versi Python yang umum digunakan, yaitu Python 2 dan Python 3. Dalam pembelajaran, digunakan Python 3 karena merupakan versi terbaru dan masih terus dikembangkan. Pada sistem operasi Linux dan macOS, Python 3 biasanya sudah terpasang secara otomatis. Sementara itu, pengguna Windows dapat menggunakan **Anaconda** karena proses instalasinya lebih mudah dan sudah dilengkapi berbagai package pendukung.



```
Jupyter Notebook x Command Prompt - python x + v
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LEGION>python
Python 3.12.10 (tags/v3.12.10:0cc8128, Apr 8 2025, 12:21:36) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```

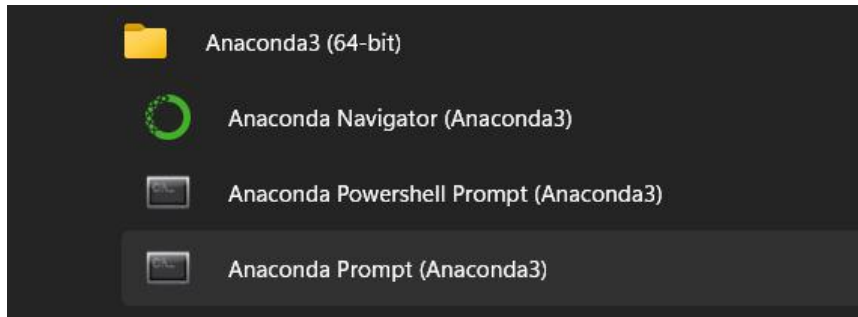
Python dapat dijalankan dalam **mode interaktif**, di mana pengguna dapat memasukkan perintah satu per satu dan langsung melihat hasilnya. Mode ini sangat cocok untuk mencoba perhitungan sederhana dan

memahami konsep dasar Python. Selain itu, Python juga dapat dijalankan dalam *mode script*, yaitu dengan menyimpan kode dalam file berekstensi *.py* dan menjalankannya melalui terminal atau editor.

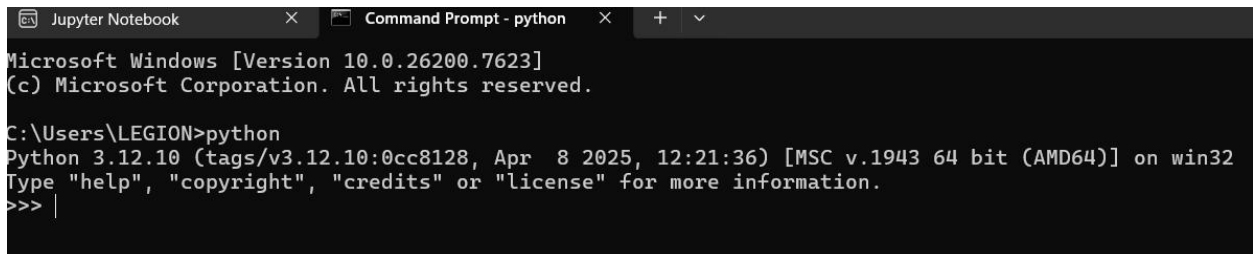


Menjalankan Python Mode Interaktif

Untuk masuk ke mode interaktif, kita harus jalankan perintah di `python3` pada terminal ubuntu jika ada Windows, jalankan di Anaconda prompt, kemudian ketik Perintah `python3` seperti gambar dibawah ini.



Gambar ini menggunakan menu Windows setelah Anaconda prompt terinstall.



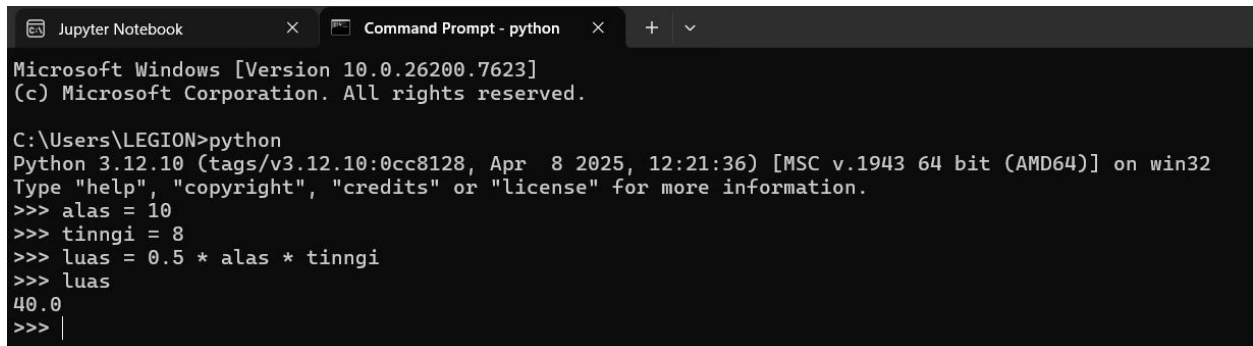
Gambar ini adalah versi Python yang akan di tampilkan dan siap menerima Perintah.

Penggunaan mode ini bisa anda masukan Perintah satu-persatu dan dapat langsung di proses oleh Python. Sebagai contoh anda bisa mulai dengan hitunglah luas segitiga yang alasnya memiliki 10cm dan tingginya 8cm. Mulai dengan ketikkan seperti dibawah ini dengan baris-berbaris, dan harus diakhiri dengan menekan tombol Enter.

- *alas = 10*
- *tinggi = 8*
- *luas = 0.5 * alas * tinggi*

- luas

Hasilnya akan seperti gambar dibawah ini.

A screenshot of a Windows Command Prompt window. The title bar shows 'Jupyter Notebook' and 'Command Prompt - python'. The window content displays the following text:

```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LEGION>python
Python 3.12.10 (tags/v3.12.10:0cc8128, Apr 8 2025, 12:21:36) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> alas = 10
>>> tinggi = 8
>>> luas = 0.5 * alas * tinggi
>>> luas
40.0
>>> |
```

Perintah pertama dan kedua adalah perintah assignment, yang mengisi nilai variabel alas dengan 10 dan variabel tinggi diisi dengan nilai 8. Sedangkan perintah ketiga mengisi variabel luas yaitu dengan perhitungan $0.5 * \text{alas} * \text{tinggi}$.

Editor untuk Python

Untuk menulis program Python diperlukan editor atau IDE agar penulisan source code menjadi lebih mudah dan terstruktur. Semakin lengkap fitur yang dimiliki editor, semakin terbantu proses pembuatan program.

Beberapa editor yang umum digunakan untuk Python antara lain ada.

-Visual Studio Code

-PyCharm

-Spyder

-ActivePython

-IDLE

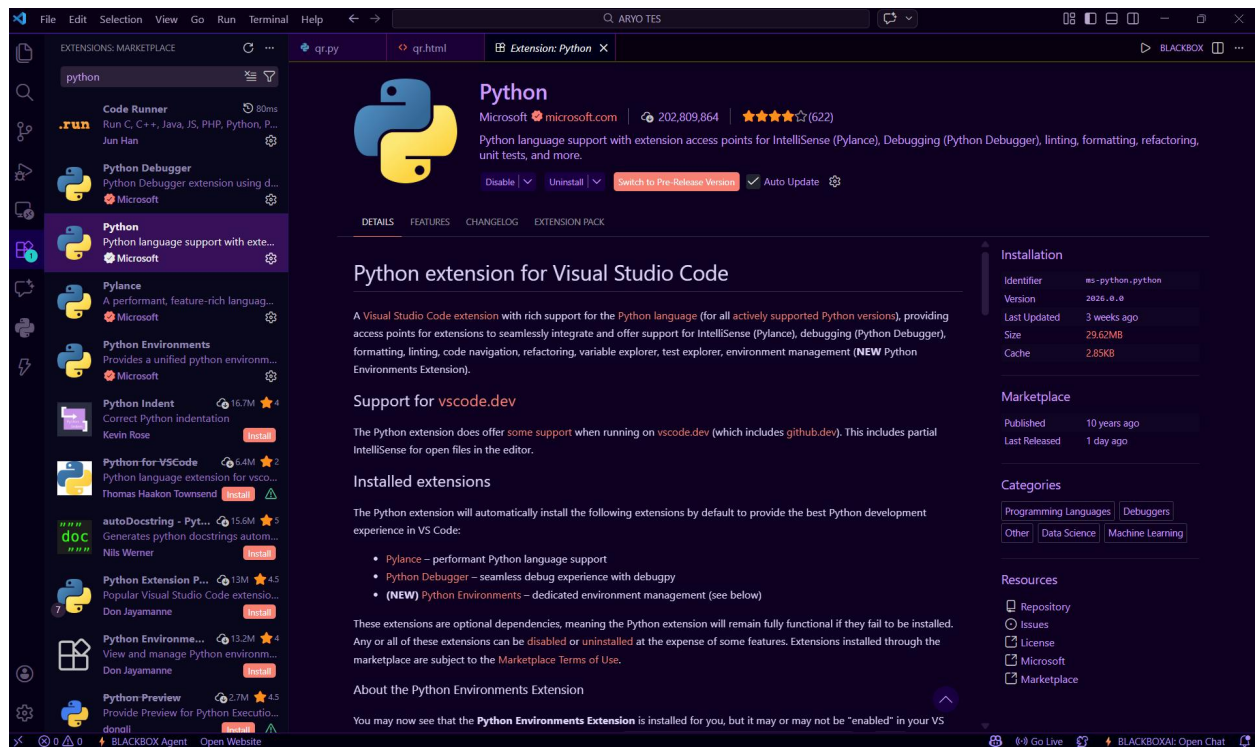
Di antara editor tersebut, PyCharm dan Spyder termasuk ke dalam *Integrated Development Environment (IDE)* karena menyediakan fitur yang lebih lengkap dan terintegrasi, sehingga cocok digunakan untuk project berskala menengah hingga besar.

Untuk keperluan praktikum, penggunaan editor sederhana seperti *Visual Studio Code* sudah cukup karena tampilannya ringan dan fitur yang disediakan sudah memenuhi kebutuhan dasar pemrograman. Pada modul ini, *Visual Studio Code* digunakan sebagai editor utama, sedangkan *PyCharm* dan *Spyder* digunakan sebagai editor pendukung. Disarankan untuk mencoba beberapa editor agar dapat memahami kelebihan dan kekurangan masing-masing.

Contoh tampilan *Visual Studio Code* yang sudah terpasang Python

Extension dapat dilihat

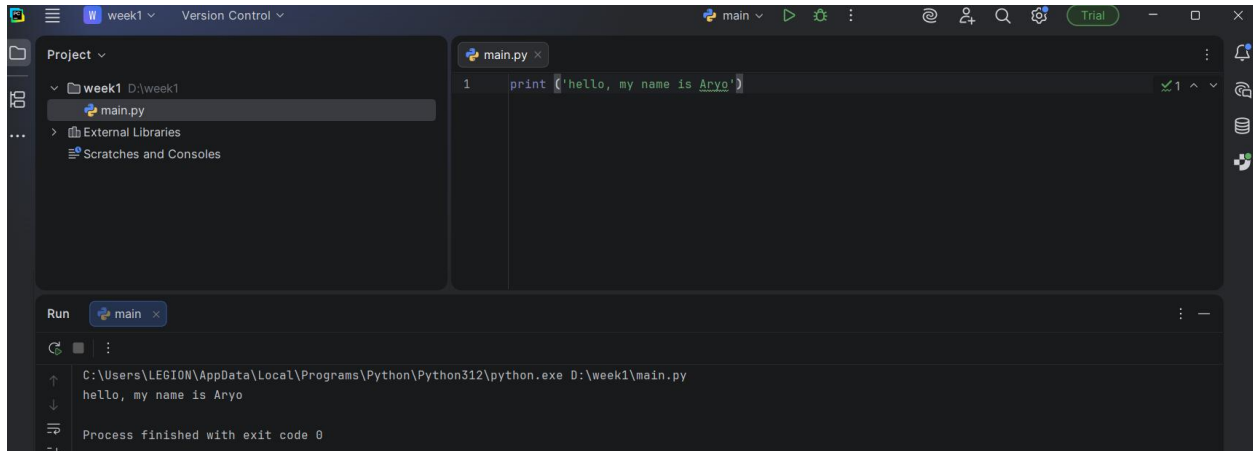
pada Gambar dibawah ini.



Contoh tampilan Visual Studio Code, dalam file Python dan saat menjalankan script Python seperti gambar dibawah ini.



Tampilan PyCharm saat mengedit file Python seperti gambar dibawah ini.



Menjalankan Script Python di Terminal/Console

Python dapat dijalankan menggunakan dua cara, yaitu mode interaktif dan mode script. Pada mode interaktif, pengguna mengetik perintah satu per satu dan hasilnya langsung ditampilkan oleh interpreter Python. Mode ini cocok digunakan untuk mencoba perintah atau fungsi baru secara cepat. Namun, kekurangannya adalah perintah harus diketik ulang jika keluar dari mode interaktif.

Selain itu, Python juga mendukung mode script, yaitu menjalankan sekumpulan perintah yang disimpan dalam file berekstensi `.py`. Dengan mode ini, program dapat dijalankan kembali tanpa perlu mengetik ulang perintah satu per satu. Script Python dijalankan melalui Terminal di Ubuntu atau Anaconda Prompt di Windows menggunakan perintah sesuai sistem operasi.

Mode script lebih efektif digunakan untuk program yang lebih panjang dan terstruktur, sedangkan mode interaktif lebih cocok untuk percobaan dan pembelajaran awal. Untuk mencoba Python, buatlah sebuah file bernama `nilaitukardollar.py`, dan ketiklah kode berikut ini di dalam file tersebut, seperti code dibawah ini dan simpan di suatu tempat.

```
# nilai kurs 1 US$ ke IDR
```

```
kursusd = 13950
```

```
# informasi program
```

```
print('Program konversi US$ ke IDR')
```

```
print('Kurs saat ini 1 US$ = ',kursusd, 'Rupiah')
```

```
# input jumlah US$ yang mau ditukar
```

```
jumlahusd = float(input('Masukkan jumlah uang yang mau ditukar ke  
Rupiah: '))
```

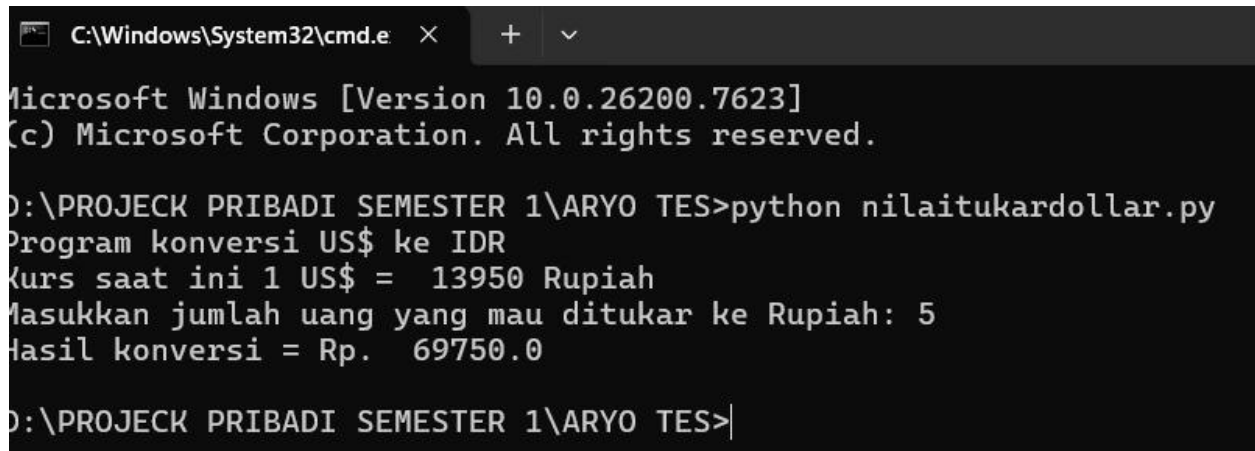
```
# hitung nilainya dalam Rupiah
```

```
dalamrupiah = jumlahusd * kursusd
```

```
# tampilkan hasilnya
```

```
print('Hasil konversi = Rp. ', dalamrupiah)
```

Untuk menjalankan script tersebut, gunakan perintah `python3` `nilaitukardollar.py` pada (Ubuntu) atau `python` `nilaitukardollar.py` (Windows).
Pada Ubuntu, tampilan hasilnya dapat dilihat seperti gambar dibawah ini.



```
C:\Windows\System32\cmd.e  X  +  v

Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

D:\PROJECK PRIBADI SEMESTER 1\ARYO TES>python nilaitukardollar.py
Program konversi US$ ke IDR
Kurs saat ini 1 US$ = 13950 Rupiah
Masukkan jumlah uang yang mau ditukar ke Rupiah: 5
Hasil konversi = Rp. 69750.0

D:\PROJECK PRIBADI SEMESTER 1\ARYO TES>|
```

Mencari Bug dan Memperbaikinya (debugging)

Bug adalah istilah dalam pemrograman yang berarti ada kesalahan dalam program. Kesalahan itu dapat membuat program secara umum dapat menjadi 2 yaitu :

- Kesalahan dalam menulis program. Biasa disebut sebagai syntax error atau compile error.
- Kesalahan jenis ini biasanya diakibatkan oleh pengguna karena typo.

```
nilaitukardollar.py X BLACKBOX ...
1 # nilai kurs 1 US$ ke IDR
2 kursud = 13950
3
4 # informasi program
5 print('Program konversi US$ ke IDR')
6 print('Kurs saat ini 1 US$ = ',kursud, 'Rupiah')
7 # input jumlah US$ yang mau ditukar
8 jumlahud = float(input('Masukkan jumlah uang yang mau ditukar ke Rupiah: '))
9 # hitung nilainya dalam Rupiah
10 dalamrupiah = jumlahud * kursud
11 # tampilkan hasilnya
12 print('Hasil konversi = Rp. ', dalamrupiah)
13
14

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\PROJEK PRIBADI SEMESTER 1\ARYO TES> python -u "d:\PROJEK PRIBADI SEMESTER 1\ARYO TES\tempCodeRunnerFile.py"
Program konversi US$ ke IDR
Kurs saat ini 1 US$ = 13950 Rupiah
Masukkan jumlah uang yang mau ditukar ke Rupiah: 5
Hasil konversi = Rp. 69750.0
PS D:\PROJEK PRIBADI SEMESTER 1\ARYO TES>
```

Jenis kesalahan ini mudah sekali untuk ditemukan dengan bantuan interpreter Python maupun editor yang dipakai.

- Kesalahan saat program berjalan. Biasa disebut sebagai runtime error.

Jenis kesalahan ini lebih sulit ditemukan dan diperbaiki dibanding jenis kesalahan yang pertama.

Python menggunakan sistem interpreter, yang di mana kode program dibaca dan dijalankan baris-perbaris. Setiap baris akan dicek apakah sudah sesuai dengan aturan penulisan (sintaks). Jika benar, baris tersebut langsung dieksekusi. Namun, jika terjadi kesalahan pada suatu baris saat dijalankan, maka program akan langsung berhenti dan tidak melanjutkan ke baris berikutnya.

#jumlah belanja (dalam rupiah)

belanja = 100000

#besarnya diskon (dalam persen)

diskon = 30

#besarnya diskon

*nominal_diskon = (diskon/100) * belanja*

#hitung jumlah yang harus dibayar

bayar = nominaldiskon - belanja

#tampilkan hasilnya

print('Anda harus membayar: Rp. ', bayar)

Program seharusnya menghitung total belanja Rp100.000 dengan diskon 30%, tetapi saat dijalankan tidak menghasilkan output karena terjadi **error di baris ke-11**. Error yang muncul adalah **NameError**, yaitu kesalahan karena variabel digunakan sebelum didefinisikan. Hal ini terjadi akibat **salah penulisan nama variabel (typo)**, di mana *nominal_diskon* ditulis menjadi *nominaldiskon*, sehingga program tidak bisa dijalankan.

```
D:\PROJECK PRIBADI SEMESTER 1\ARYO TES>python nilaitukardollar.py
Traceback (most recent call last):
  File "D:\PROJECK PRIBADI SEMESTER 1\ARYO TES\nilaitukardollar.py", line 11, in <module>
    bayar = nominaldiskon - belanja
            ^^^^^^^^^^^^^^^
NameError: name 'nominaldiskon' is not defined. Did you mean: 'nominal_diskon'?
D:\PROJECK PRIBADI SEMESTER 1\ARYO TES>|
```

Kesalahan saat script dijalankan.

Jika anda menjalankan di Visual Studio Code dia akan muncul seperti gambar dibawah ini sintaksnya eror.

```
1  #jumlah belanja (dalam rupiah)
2  belanja = 100000
3
4  #besarnya diskon (dalam persen)
5  diskon = 30
6
7  #besarnya diskon
8  nominal_diskon = (diskon/100) * belanja
9
10 #hitung jumlah yang harus dibayar
11 bayar = nominaldiskon - belanja
12
13 #tampilkan hasilnya
14 print('Anda harus membayar: Rp. ', bayar)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\PROJECK PRIBADI SEMESTER 1\ARYO TES> python -u "d:\PROJECK PRIBADI SEMESTER 1\ARYO TES\nilaitukardollar.py"

Traceback (most recent call last):

File "d:\PROJECK PRIBADI SEMESTER 1\ARYO TES\nilaitukardollar.py", line 11, in <module>

bayar = nominaldiskon - belanja

^^^^^^^^^^^^

NameError: name 'nominaldiskon' is not defined. Did you mean: 'nominal_diskon'?

PS D:\PROJECK PRIBADI SEMESTER 1\ARYO TES>

Dan untuk memperbaiki anda harus masukan code seperti dibawah ini

#jumlah belanja (dalam rupiah)

belanja = 100000

#besarnya diskon (dalam persen)

diskon = 30

#besarnya diskon

*nominaldiskon = (diskon/100) * belanja*

#hitung jumlah yang harus dibayar


```
bayar = belanja - nominaldiskon
```

```
#tampilkan hasilnya
```

```
print('Anda harus membayar: Rp. ', bayar)
```

Setelah diperbaiki sintaks yang eror tadi, sekarang anda sudah dapat jalankan perintahnya dengan baik seperti gambar dibawah ini

```
D:\PROJECK PRIBADI SEMESTER 1\ARYO TES>python nilaitukardollar.py
Anda harus membayar: Rp. 70000.0
D:\PROJECK PRIBADI SEMESTER 1\ARYO TES>|
```

Hasil ini sudah sesuai dengan harapan.

MATERI 2

Kegiatan Praktikum

Pada kegiatan praktikum ini, mahasiswa akan mempelajari cara menggunakan *Jupyter Notebook* serta mencoba *Python mode interaktif* untuk melakukan perhitungan sederhana. Praktikum ini bertujuan agar mahasiswa terbiasa dengan lingkungan kerja Python sebelum masuk ke materi pemrograman yang lebih lanjut.

Instalasi Jupyter Notebook

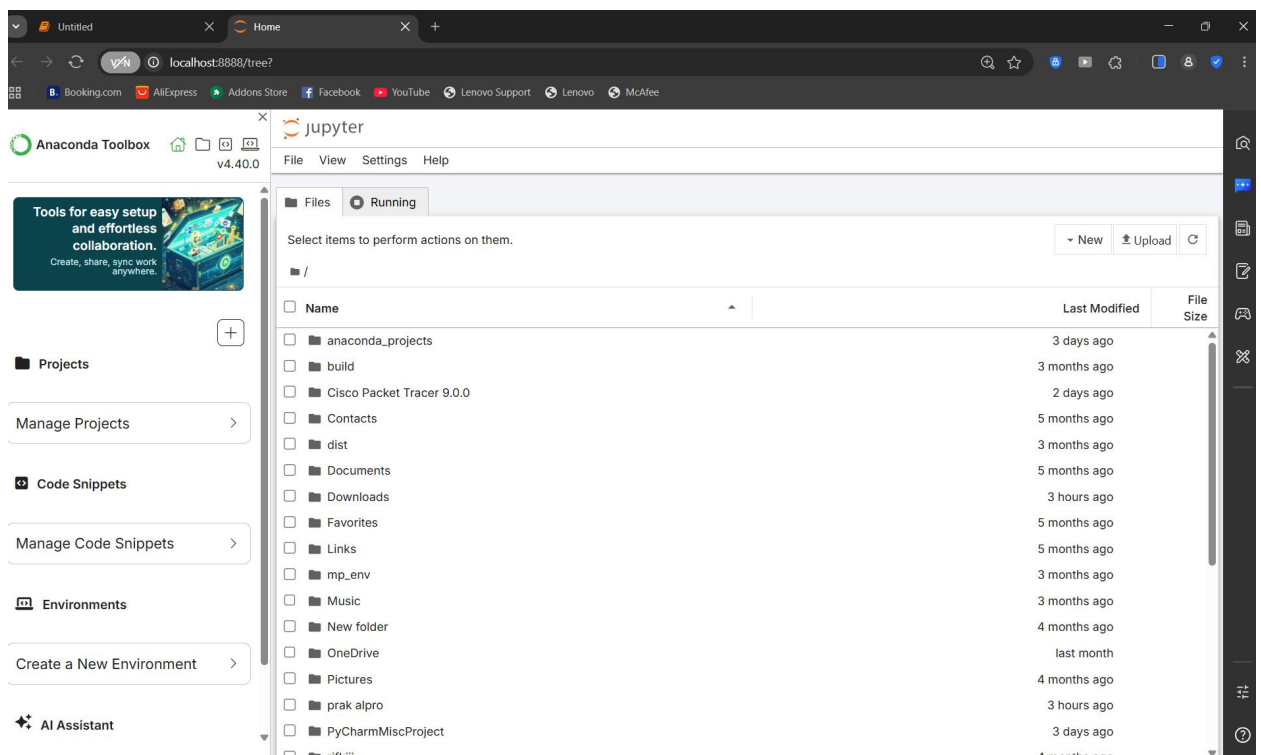
Jupyter Notebook merupakan aplikasi berbasis web yang digunakan untuk menulis dan menjalankan kode Python, menampilkan output, serta

visualisasi data. Jupyter Notebook banyak digunakan karena cukup dijalankan melalui browser sehingga mudah diakses.

Instalasi Jupyter Notebook dilakukan menggunakan *pip*, yaitu package installer untuk Python. Proses instalasi dilakukan melalui **Anaconda Prompt** pada Windows atau **Terminal** pada Linux.

Perintah yang digunakan untuk instalasi adalah:

- *pip install jupyter notebook* (Windows/Anaconda).
- *pip3 install jupyter notebook* (Ubuntu).



Ini tampilan Jupyter Notebook

```
C:\Windows\System32\cmd.exe x Anaconda Prompt (anaconda) x + v
(base) C:\Users\LEGION>(base) C:\Users\Maryo>
C:\Users\Maryo was unexpected at this time.

(base) C:\Users\LEGION>pip install jupyter notebook
Requirement already satisfied: jupyter in d:\anaconda\lib\site-packages (1.0.0)
Requirement already satisfied: notebook in d:\anaconda\lib\site-packages (7.2.2)
Requirement already satisfied: qtconsole in d:\anaconda\lib\site-packages (from jupyter) (5.5.1)
Requirement already satisfied: jupyter-console in d:\anaconda\lib\site-packages (from jupyter) (6.6.3)
Requirement already satisfied: nbconvert in d:\anaconda\lib\site-packages (from jupyter) (7.16.4)
Requirement already satisfied: ipykernel in d:\anaconda\lib\site-packages (from jupyter) (6.28.0)
Requirement already satisfied: ipywidgets in d:\anaconda\lib\site-packages (from jupyter) (7.8.1)
Requirement already satisfied: jupyter-server<3,>=2.4.0 in d:\anaconda\lib\site-packages (from notebook) (2.14.1)
Requirement already satisfied: jupyterlab-server<3,>=2.27.1 in d:\anaconda\lib\site-packages (from notebook) (2.27.3)
Requirement already satisfied: jupyterlab<4.3,>=4.2.0 in d:\anaconda\lib\site-packages (from notebook) (4.2.5)
Requirement already satisfied: notebook-shim<0.3,>=0.2 in d:\anaconda\lib\site-packages (from notebook) (0.2.3)
Requirement already satisfied: tornado<=6.2.0 in d:\anaconda\lib\site-packages (from notebook) (6.4.1)
Requirement already satisfied: anyio<=3.1.0 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (4.2.0)
Requirement already satisfied: argon2-cffi<=21.1 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (21.3.0)
Requirement already satisfied: Jinja2<=3.0.3 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (3.1.4)
Requirement already satisfied: jupyter-client<=7.4.0 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (8.6.0)
Requirement already satisfied: jupyter-core<=5.0.*,>=4.12 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (5.7.2)
Requirement already satisfied: jupyter-events<=0.9.0 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (0.10.0)
Requirement already satisfied: jupyter-server-terminals<=0.4.4 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (0.4.4)
Requirement already satisfied: nbformat<=5.3.0 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (5.10.4)
Requirement already satisfied: overrides<=5.0 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (7.4.0)
Requirement already satisfied: packaging<=22.0 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (24.1)
Requirement already satisfied: prometheus-client<=0.9 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (0.14.1)
Requirement already satisfied: pywinpty<=2.0.1 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (2.0.10)
Requirement already satisfied: pyzmq<=24 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (25.1.2)
Requirement already satisfied: send2trash<=1.8.2 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (1.8.2)
Requirement already satisfied: terminado<=0.8.3 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (0.17.1)
Requirement already satisfied: traitlets<=5.6.0 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (5.14.3)
Requirement already satisfied: websocket-client<=1.7 in d:\anaconda\lib\site-packages (from jupyter-server<3,>=2.4.0->notebook) (1.8.0)
Requirement already satisfied: async-lru<=1.0.0 in d:\anaconda\lib\site-packages (from jupyterlab<4.3,>=4.2.0->notebook) (2.0.4)
Requirement already satisfied: httpx<=0.25.0 in d:\anaconda\lib\site-packages (from jupyterlab<4.3,>=4.2.0->notebook) (0.27.0)
Requirement already satisfied: jupyter-lsp<=2.0.0 in d:\anaconda\lib\site-packages (from jupyterlab<4.3,>=4.2.0->notebook) (2.2.0)
Requirement already satisfied: setuptools<=40.1.0 in d:\anaconda\lib\site-packages (from jupyterlab<4.3,>=4.2.0->notebook) (75.1.0)
Requirement already satisfied: comm<=0.1.1 in d:\anaconda\lib\site-packages (from ipykernel->jupyter) (0.2.1)
Requirement already satisfied: debugpy<=1.6.5 in d:\anaconda\lib\site-packages (from ipykernel->jupyter) (1.6.7)
Requirement already satisfied: ipython<=7.23.1 in d:\anaconda\lib\site-packages (from ipykernel->jupyter) (8.27.0)
Requirement already satisfied: matplotlib-inline<=0.1 in d:\anaconda\lib\site-packages (from ipykernel->jupyter) (0.1.6)
Requirement already satisfied: nest-asyncio in d:\anaconda\lib\site-packages (from ipykernel->jupyter) (1.6.0)
Requirement already satisfied: psutil in d:\anaconda\lib\site-packages (from ipykernel->jupyter) (5.9.0)
Requirement already satisfied: babel<=2.10 in d:\anaconda\lib\site-packages (from jupyterlab-server<3,>=2.27.1->notebook) (2.11.0)
Requirement already satisfied: json5<=0.9.0 in d:\anaconda\lib\site-packages (from jupyterlab-server<3,>=2.27.1->notebook) (0.9.6)
Requirement already satisfied: jsonschema<=4.18.0 in d:\anaconda\lib\site-packages (from jupyterlab-server<3,>=2.27.1->notebook) (4.23.0)
```

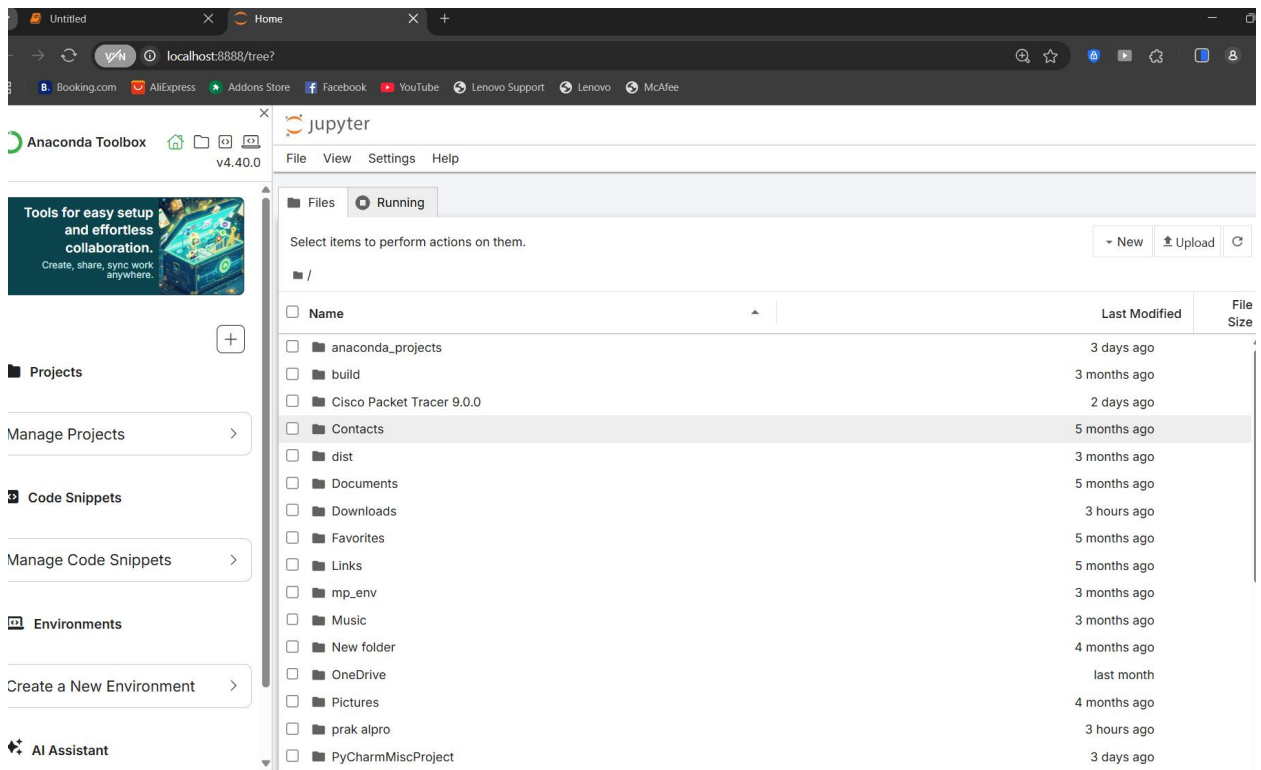
Ini contoh install Jupyter Notebook menggunakan PIP

Setelah proses instalasi selesai, Jupyter Notebook dijalankan dengan perintah jupyter notebook. Selanjutnya browser akan terbuka secara otomatis dan menampilkan halaman utama Jupyter Notebook.

File Python pada Jupyter Notebook disimpan dengan ekstensi .ipynb.

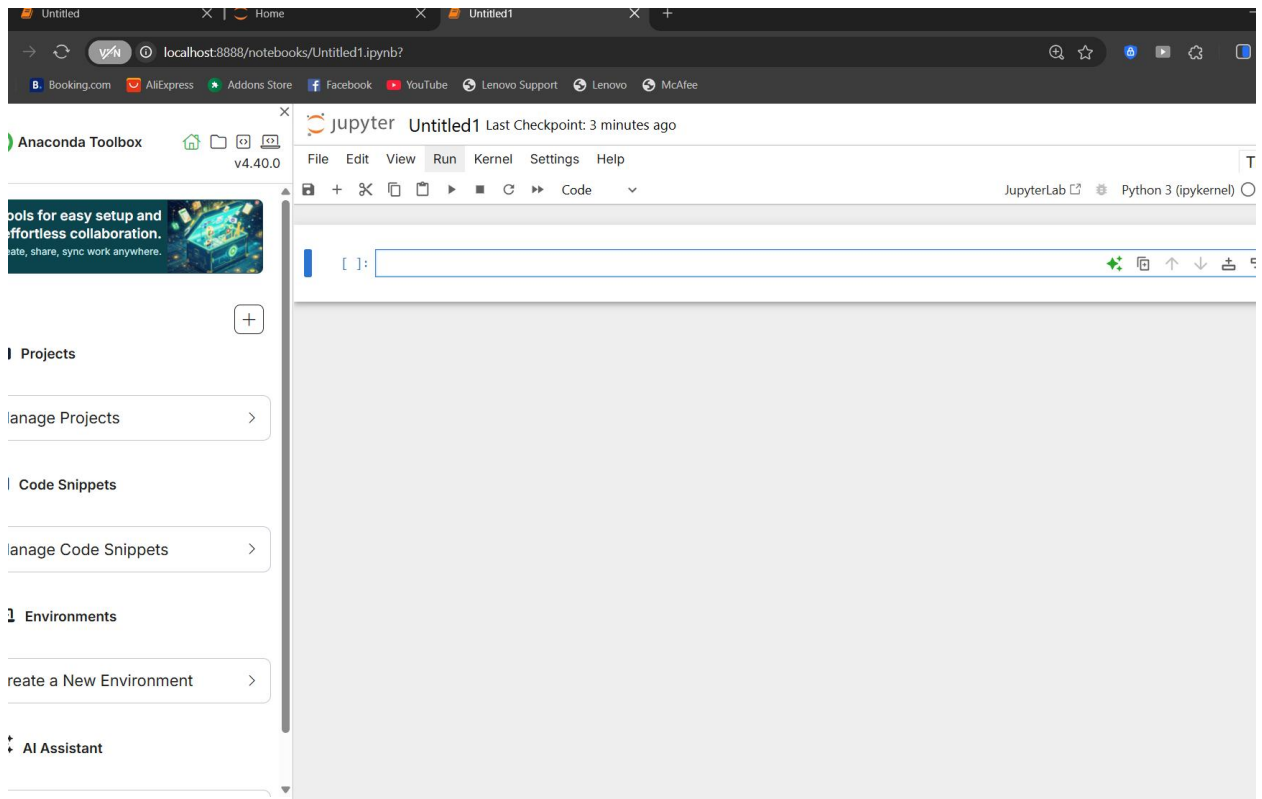
Untuk membuat notebook baru, digunakan menu New > Notebook Python 3.

Setiap notebook terdiri dari beberapa cell yang dapat dijalankan satu per satu atau sekaligus. Untuk menghentikan Jupyter Notebook, pengguna dapat menekan Ctrl + C pada Terminal atau Anaconda Prompt.



Tampilan awas Jupyter Notebook

Jika anda ingin membuat notebook baru ada di pojok kanan atas yang tulisannya New silakan di klik dan akan muncul beberapa pilihan dan anda cukup klik yang tulisannya Python3 dan akan muncul gambar seperti dibawah ini.



Tampilan awas Notebook di Jupyter Notebook.

Eksplorasi Python Mode Interaktif

Python mode interaktif digunakan untuk melakukan perhitungan secara langsung tanpa harus membuat file program terlebih dahulu. Mode ini sangat membantu mahasiswa untuk memahami konsep dasar Python.

Menghitung Usia

Jika diketahui tahun lahir seseorang, kita bisa menentukan usianya sekarang. Misalnya Andi lahir tahun 1990 dan sekarang adalah tahun

2020, maka usianya adalah 30 tahun. Gunakan Python untuk menjawab beberapa masalah berikut ini (diasumsikan sekarang adalah tahun 2020):

- Trump dilahirkan tahun 1965. Berapa usianya sekarang?
- Pada tahun berapa Trump akan berusia 80 tahun?

Untuk menjawab pertanyaan-pertanyaan tersebut, langkah-langkahnya adalah sebagai berikut:

1. Tahun lahir = 1965.

2. Usia sekarang (2020) = $2020 - 1965 = 55$ tahun.

3. Usia Trump 80 saat = $1965 + 80 = 2045$.

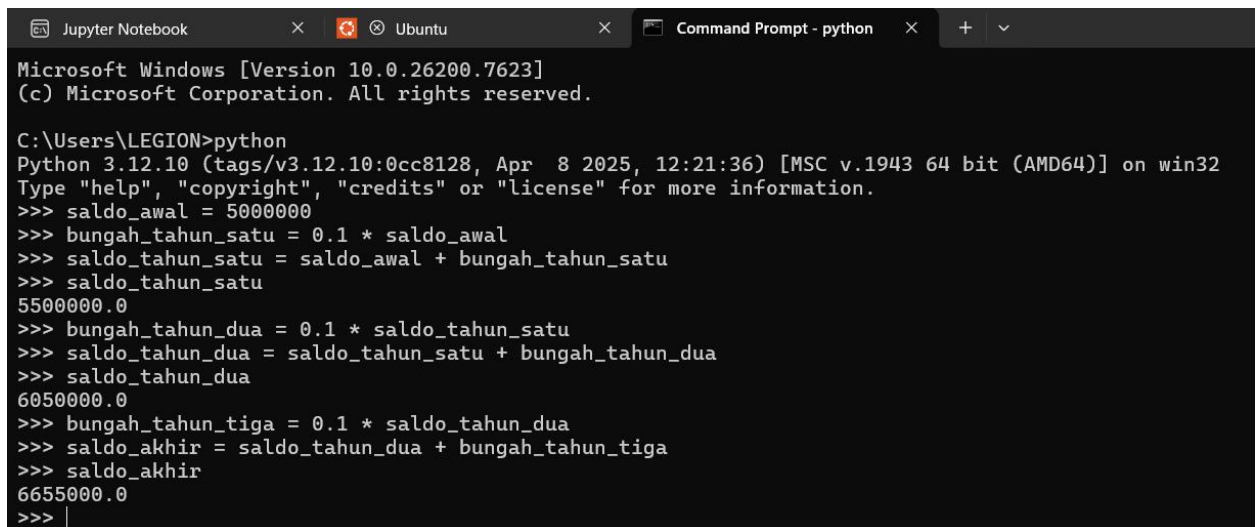
Untuk langkah-langkahnya ikuti seperti dibawah ini dalam menggunakan Python.

```
D:\PROJEK PRIBADI SEMESTER I\ARYO TES>python
Python 3.12.10 (tags/v3.12.10:0cc8128, Apr  8 2025, 12:21:36) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> tahun_lahir = 1965
>>> usia_sekarang = 2020 - 1965
>>> usia_sekarang
55
>>> usia_80 = 1965 + 80
>>> usia_80
2045
>>>
```

Penyelesaian untuk masalah usia Trump.

Compound Interest

Perhitungan compound interest pada contoh sebelumnya menghitung saldo di akhir tahun ketiga. Bagaimana jika dibutuhkan menghitung berapa saldo di akhir tahun ke-dua puluh? Kita akan mulai dari memasukkan nilai saldo awal, bunga, jumlah periode, dan lama waktu ke dalam rumus, Python dapat menghitung saldo akhir dengan lebih cepat dan sederhana seperti gambar dibawah ini.



```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LEGION>python
Python 3.12.10 (tags/v3.12.10:0cc8128, Apr  8 2025, 12:21:36) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> saldo_awal = 5000000
>>> bungah_tahun_satu = 0.1 * saldo_awal
>>> saldo_tahun_satu = saldo_awal + bungah_tahun_satu
>>> saldo_tahun_satu
5500000.0
>>> bungah_tahun_dua = 0.1 * saldo_tahun_satu
>>> saldo_tahun_dua = saldo_tahun_satu + bungah_tahun_dua
>>> saldo_tahun_dua
6050000.0
>>> bungah_tahun_tiga = 0.1 * saldo_tahun_dua
>>> saldo_akhir = saldo_tahun_dua + bungah_tahun_tiga
>>> saldo_akhir
6655000.0
>>> |
```

Hasil perhitungan saldo tiga tahun yaitu 6655000.0

Formula compound interest seperti dibawah ini

$A = P (1 + r/n)^{nt}$ pangkat nt.

A = saldo akhir.

P = saldo awal.

r = besarnya bunga.

n = jumlah perhitungan bunga yang dilakukan (dalam 1 tahun berapa kali).

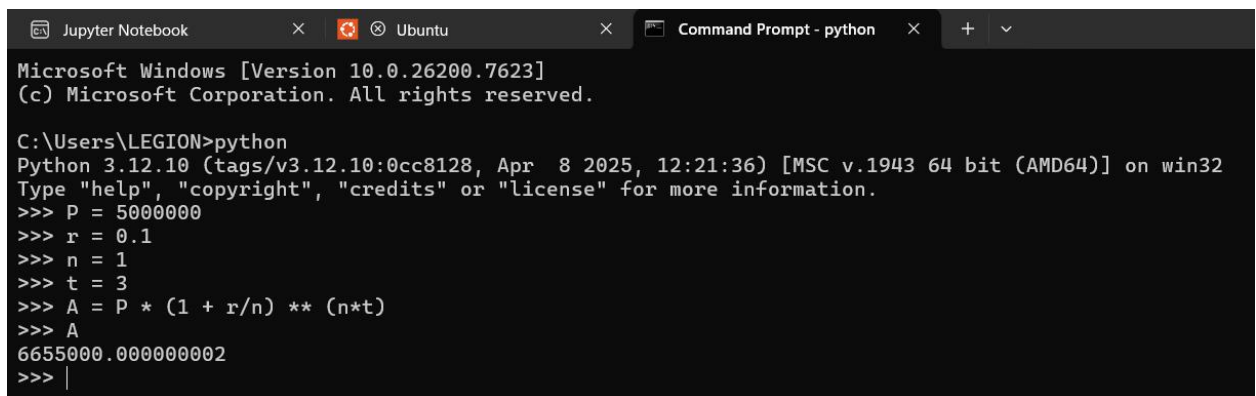
t = jumlah periode.

Dengan rumus ini kita bisa menyelesaikan langkah-langkah dengan singkat.

Secara umum dalam membuat program anda perlu terlebih dahulu

menentukan langkah-langkah yang diperlukan untuk memecahkan masalah.

Jangan terburu-buru untuk langsung membuatnya.



```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LEGION>python
Python 3.12.10 (tags/v3.12.10:0cc8128, Apr  8 2025, 12:21:36) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> P = 5000000
>>> r = 0.1
>>> n = 1
>>> t = 3
>>> A = P * (1 + r/n) ** (n*t)
>>> A
6655000.0000000002
>>> |
```

Hasil perhitungan compound interest menggunakan formula.

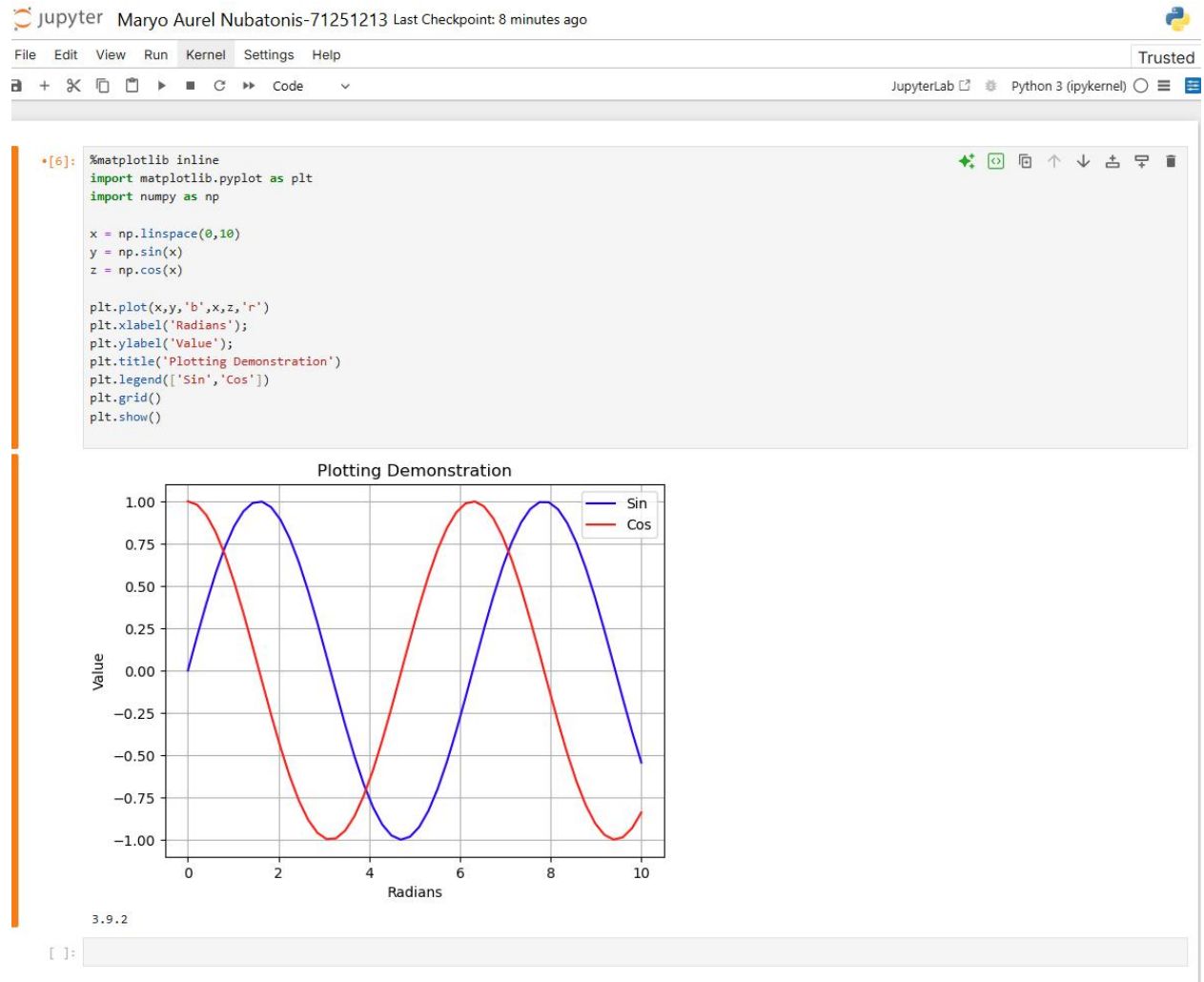
BAGIAN 2: LATIHAN MANDIRI (60%)

Pada bagian ini anda menuliskan jawaban dari soal-soal Latihan Mandiri yang ada di modul praktikum. Jawaban anda harus disertai dengan source code, penjelasan dan screenshot output.

Link Github: <https://github.com/maryonubatonis1-web/71251213-Maryo-Aurel-Nubatonis/upload/main>

SOAL

1



Penjelasan

1. Tujuan Program

Program ini bertujuan untuk menampilkan grafik fungsi sinus (sin) dan cosinus (cos) menggunakan bahasa pemrograman Python pada Jupyter Notebook.

2. Penggunaan Library

- NumPy digunakan untuk membuat dan mengolah data numerik, khususnya dalam perhitungan fungsi matematika.
- Matplotlib digunakan untuk memvisualisasikan data dalam bentuk grafik.

3. Menampilkan Grafik di Notebook

Mode inline diaktifkan agar grafik dapat langsung ditampilkan di dalam Jupyter Notebook tanpa membuka jendela baru.

4. Pembuatan Data Sumbu X

Data pada sumbu X dibuat dalam rentang nilai 0 sampai 10 dengan jarak yang sama. Data ini berfungsi sebagai input untuk perhitungan fungsi sin dan cos.

5. Perhitungan Fungsi Sin dan Cos

Nilai sinus dan cosinus dihitung berdasarkan setiap nilai pada sumbu X sehingga diperoleh dua himpunan data, yaitu data sin dan data cos.

6. Proses Plotting Grafik

Grafik fungsi sin dan cos ditampilkan dalam satu bidang koordinat. Grafik sin ditampilkan dengan warna biru, sedangkan grafik cos ditampilkan dengan warna merah agar mudah dibedakan.

7. Pemberian Keterangan Grafik

- Sumbu X diberi label Radians
- Sumbu Y diberi label Value
- Judul grafik ditambahkan untuk menjelaskan isi grafik
- Legenda digunakan untuk membedakan grafik sin dan cos

8. Penambahan Grid

Grid diaktifkan untuk mempermudah pembacaan nilai pada grafik.

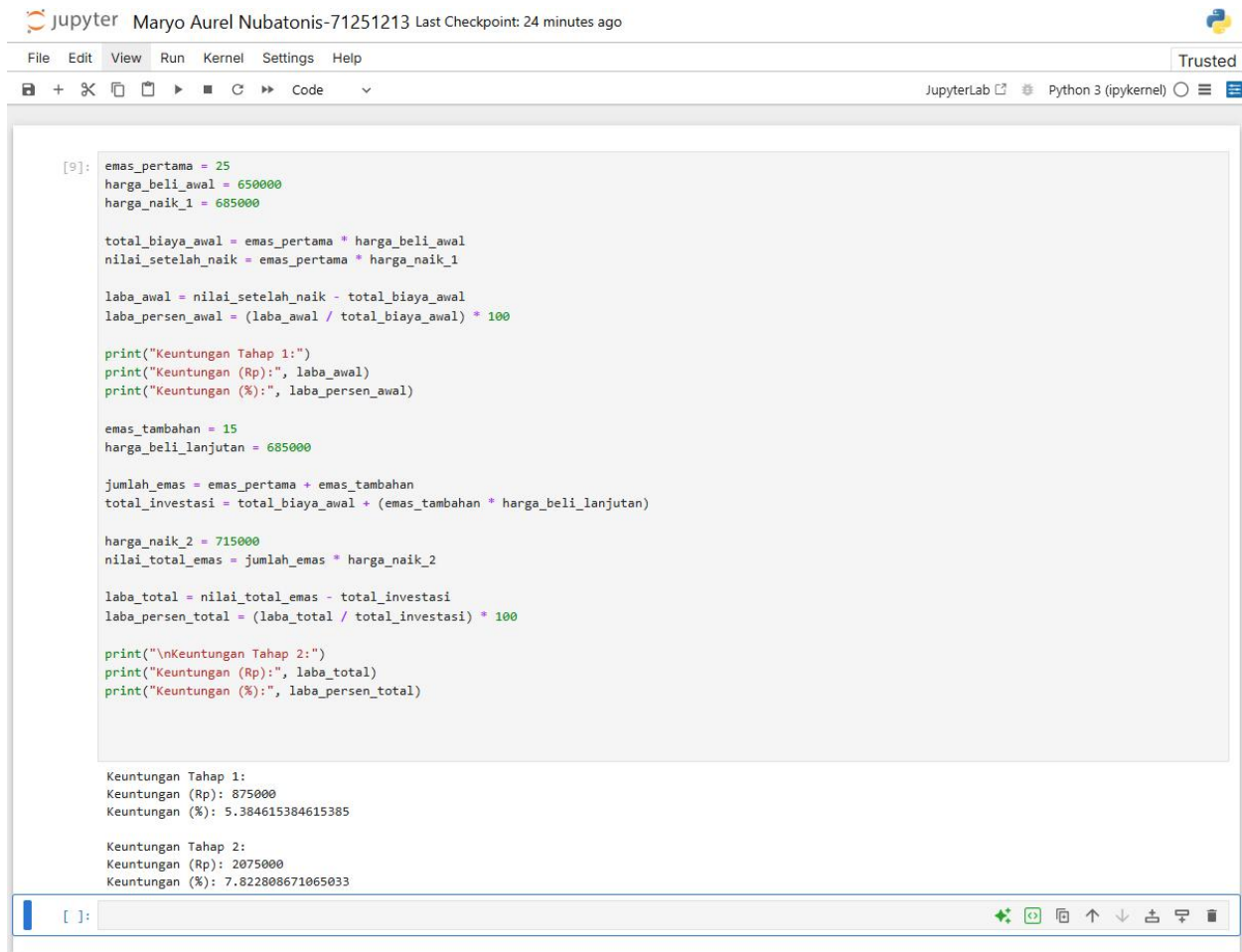
9. Hasil Program

Program berhasil menampilkan grafik fungsi sin dan cos dengan tampilan yang jelas, rapi, dan tanpa error.

10. Kesimpulan

Dengan menggunakan NumPy dan Matplotlib, grafik fungsi matematika dapat dibuat dan ditampilkan dengan mudah. Program ini menunjukkan hubungan antara nilai radian terhadap fungsi sin dan cos secara visual.

SOAL 2



```
[9]: emas_pertama = 25
harga_beli_awal = 650000
harga_naik_1 = 685000

total_biaya_awal = emas_pertama * harga_beli_awal
nilai_setelah_naik = emas_pertama * harga_naik_1

laba_awal = nilai_setelah_naik - total_biaya_awal
laba_persen_awal = (laba_awal / total_biaya_awal) * 100

print("Keuntungan Tahap 1:")
print("Keuntungan (Rp):", laba_awal)
print("Keuntungan (%)ate", laba_persen_awal)

emas_tambahan = 15
harga_beli_lanjutan = 685000

jumlah_emas = emas_pertama + emas_tambahan
total_investasi = total_biaya_awal + (emas_tambahan * harga_beli_lanjutan)

harga_naik_2 = 715000
nilai_total_emas = jumlah_emas * harga_naik_2

laba_total = nilai_total_emas - total_investasi
laba_persen_total = (laba_total / total_investasi) * 100

print("\nKeuntungan Tahap 2:")
print("Keuntungan (Rp):", laba_total)
print("Keuntungan (%)", laba_persen_total)

Keuntungan Tahap 1:
Keuntungan (Rp): 875000
Keuntungan (%): 5.384615384615385

Keuntungan Tahap 2:
Keuntungan (Rp): 2075000
Keuntungan (%): 7.822808671065033
```

Penjelasan

1. Pendefinisian Data Pembelian Pertama

- Variabel `emas_pertama` digunakan untuk menyimpan jumlah emas awal yang dibeli, yaitu 25 gram.
- Variabel `harga_beli_awal` menyimpan harga emas saat pembelian pertama sebesar Rp 650.000 per gram.

- Variabel `harga_naik_1` menyimpan harga emas setelah mengalami kenaikan menjadi Rp 685.000 per gram.

2. Perhitungan Modal Awal

- Variabel `total_biaya_awal` digunakan untuk menghitung total biaya pembelian pertama.
- Perhitungan dilakukan dengan mengalikan jumlah emas awal dengan harga beli awal.

3. Perhitungan Nilai Emas Setelah Kenaikan Harga

- Variabel `nilai_setelah_naik` menyimpan nilai emas setelah harga naik.
- Nilai ini diperoleh dari hasil perkalian jumlah emas awal dengan harga emas terbaru.

4. Perhitungan Keuntungan Tahap Pertama

- Variabel `laba_awal` menyimpan keuntungan dalam bentuk rupiah dari pembelian pertama.
- Variabel `laba_persen_awal` menyimpan keuntungan dalam bentuk persentase.
- Persentase keuntungan dihitung dengan membagi keuntungan rupiah dengan modal awal kemudian dikalikan 100.

5. Pendefinisian Data Pembelian Kedua

- Variabel `emas_tambahan` digunakan untuk menyimpan jumlah emas tambahan yang dibeli, yaitu 15 gram.
- Variabel `harga_beli_lanjutan` menyimpan harga emas pada pembelian kedua sebesar Rp 685.000 per gram.

6. Perhitungan Total Emas dan Total Investasi

- Variabel `jumlah_emas` digunakan untuk menghitung total emas yang dimiliki setelah pembelian kedua.
- Variabel `total_investasi` menyimpan total modal yang dikeluarkan Gerard dari seluruh pembelian emas.

7. Perubahan Harga Emas Tahap Kedua

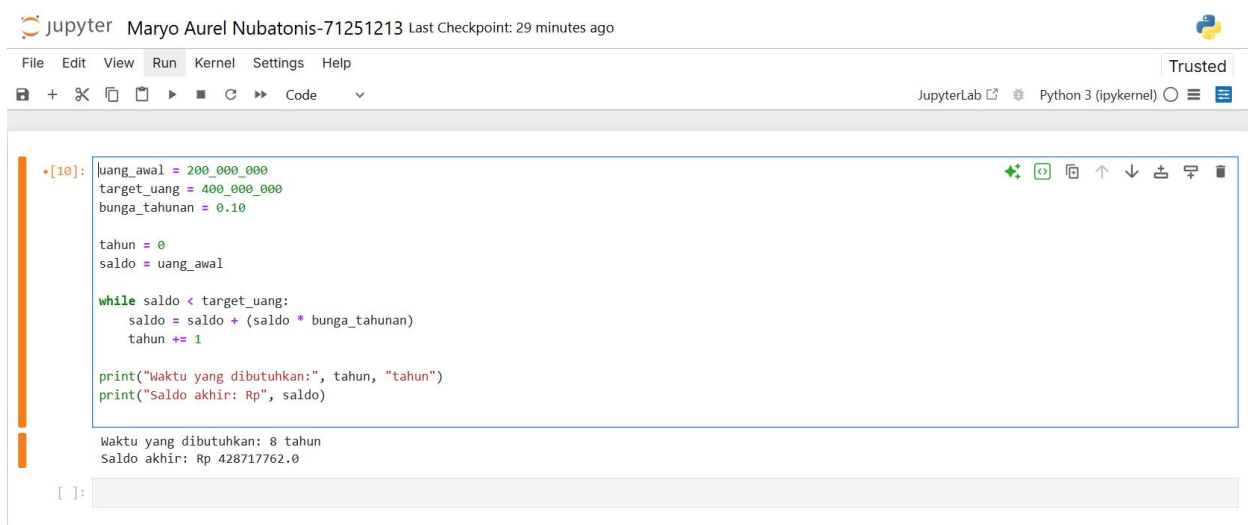
- Variabel `harga_naik_2` menyimpan harga emas setelah mengalami kenaikan kembali menjadi Rp 715.000 per gram.
- Variabel `nilai_total_emas` menyimpan nilai keseluruhan emas yang dimiliki berdasarkan harga terbaru.

8. Perhitungan Keuntungan Total

- Variabel `laba_total` digunakan untuk menghitung total keuntungan dalam rupiah.

- Variabel `laba_persen_total` digunakan untuk menghitung total keuntungan dalam persentase dari seluruh investasi.
- Perhitungan dilakukan dengan membandingkan nilai emas saat ini dengan total modal yang telah dikeluarkan.

SOAL 3



The screenshot shows a JupyterLab window with the following details:

- Header:** jupyter Maryo Aurel Nubatonis-71251213 Last Checkpoint: 29 minutes ago
- Menu Bar:** File, Edit, View, Run, Kernel, Settings, Help
- Toolbar:** Includes icons for file operations, running code, and code editor settings.
- Code Cell:**

```

•[10]: uang_awal = 200_000_000
        target_uang = 400_000_000
        bunga_tahunan = 0.10

        tahun = 0
        saldo = uang_awal

        while saldo < target_uang:
            saldo = saldo + (saldo * bunga_tahunan)
            tahun += 1

        print("Waktu yang dibutuhkan:", tahun, "tahun")
        print("Saldo akhir: Rp", saldo)

```
- Output:**

```

Waktu yang dibutuhkan: 8 tahun
Saldo akhir: Rp 428717762.0

```

Penjelasan

1. Menentukan Data Awal

- Erika memiliki uang awal sebesar Rp. 200.000.000
- Target saldo minimal adalah Rp. 400.000.000
- Bunga deposito sebesar 10% per tahun

2. Inisialisasi Variabel

- Variabel tahun digunakan untuk menghitung lama waktu penyimpanan deposito
- Variabel saldo digunakan untuk menyimpan nilai uang yang terus bertambah setiap tahun

3. Proses Perhitungan Bunga Majemuk

- Program menggunakan perulangan while
- Setiap satu tahun:
 - Saldo ditambah dengan bunga sebesar 10% dari saldo sebelumnya
 - Jumlah tahun bertambah satu
- Perulangan akan berhenti ketika saldo telah mencapai atau melebihi Rp 400.000.000

4. Menampilkan Hasil

- Program menampilkan:
 - Lama waktu yang dibutuhkan dalam satuan tahun
 - Jumlah saldo akhir setelah mencapai target

Hasil Perhitungan

- Erika membutuhkan waktu **8 tahun** agar uangnya mencapai minimal Rp 400.000.000
- Saldo akhir setelah 8 tahun melebihi target karena bunga majemuk terus bertambah setiap tahun