

Dosen Pengampu: Adri Arisena



PRAKTIKUM DASAR

Dasar-dasar Pemrograman Python

Asisten Laboratorium:
Najlia Intani



"Modul ini diperuntukkan bagi Mata Kuliah Pemrograman Dasar dan Basis Data Program Studi Agribisnis"

A. PENGENALAN PYTHON DAN GOOGLE COLAB

Python adalah bahasa pemrograman *interpreter* tingkat tinggi, berorientasi objek, dengan memiliki semantik yang dinamis. Bahasa tingkat tinggi yang dibangun dalam struktur data, dikombinasikan dengan pengetikan dinamis dan pengikatan dinamis, membuatnya sangat menarik untuk Pengembangan Aplikasi Cepat, serta untuk digunakan sebagai bahasa *scripting*. Sintaksis Python yang sederhana dan mudah dipelajari menekankan keterbacaan dan karenanya mengurangi biaya pemeliharaan program. Python mendukung modul dan paket, yang mendorong modularitas program dan penggunaan kode kembali. Interpreter Python dan pustaka standar yang luas tersedia dari berbagai sumber dan dapat didistribusikan secara bebas.

Python mendukung pemrograman dengan paradigma multiguna, 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. Sama halnya dengan bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa *scripting* namun penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi. Kode python dapat dijalankan di berbagai platform sistem operasi, antara lain:

- Linux/Unix
- Windows
- Mac OS X
- Java Virtual Machine
- OS/2
- Amiga
- dan sebagainya.

Beberapa fitur yang dimiliki Python adalah:

1. Memiliki library atau pusaka yang sangat lengkap yang siap dipakai untuk berbagai memiliki tata bahasa yang jernih dan mudah dipelajari.
2. Tata Bahasa pemrogramannya mudah dipahami
3. Memiliki aturan layout kode sumber yang memudahkan pengecekan, pembacaan kembali dan penulisan ulang kode sumber.

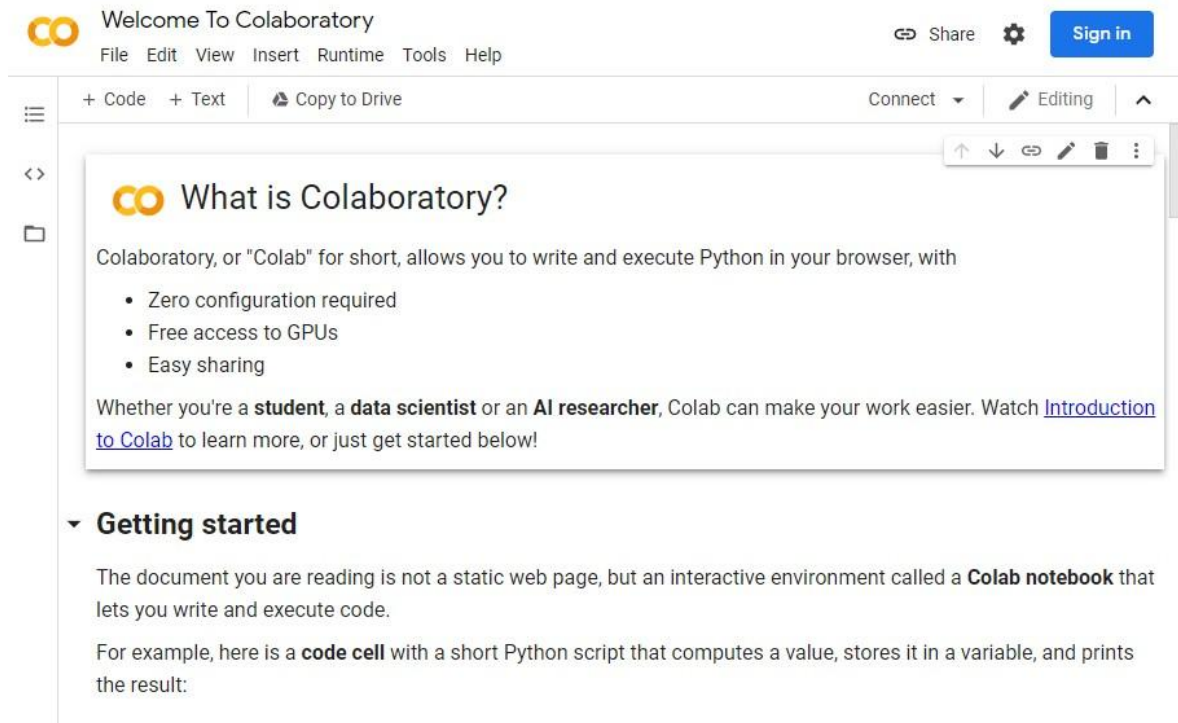
4. Berorientasi objek.
5. Memiliki sistem pengelolaan memori otomatis (garbage collection, seperti java)
6. Modular, mudah dikembangkan dengan menciptakan modul-modul baru; modulmodul tersebut dapat dibangun dengan bahasa Python maupun C/C++.
7. Garbage collection otomatis
8. Fasilitas pengaturan penggunaan ingatan komputer sehingga para pemrogram tidak perlu melakukan pengaturan ingatan komputer secara langsung.
9. Memiliki banyak fasilitas pendukung sehingga mudah dalam pengoperasiannya.

Python dapat digunakan dengan berbagai macam antarmuka aplikasi seperti Anaconda, IntelliJ IDEA, Visual Studio Code, Sublime Text, Spyder, Google Colab, dan sebagainya. Google Colab (Google Collaboratory) merupakan tools yang berbasis cloud dan free untuk tujuan penelitian. Google colab dibuat dengan environment jupyter notebook dan mendukung hampir semua library yang dibutuhkan dalam berbagai lingkungan pengembangan, seperti Machine Learning, Artificial Intelligence (AI), Kriptografi, pengolahan data, dan sebagainya. Berikut adalah beberapa kelebihan dalam menggunakan google colab, antara lain:

1. Penggunaan google colab ditujukan bagi para peneliti yang sedang mengembangkan penelitian dan membutuhkan spesifikasi komputer yang tinggi. Hanya perlu diingat bahwa google colab membutuhkan koneksi internet.
2. Google memberikan akses cloud komputer dengan spesifikasi:
 - a. Intel(R) Xeon(R) CPU @ 2.30GHz
 - b. 12,6 GB VRAM
 - c. Tesla P100-PCIE-16GB, Cuda Cores: 2496
 - d. 33 GB
3. Dalam menggunakan google colab kita tidak memerlukan konfigurasi apapun, namun dapat menginstall Pustaka di Google Colab.
4. Dapat diintegrasikan dengan Google Drive dan berbagi dengan pengguna lainnya.
5. Dapat digunakan dimana saja karena bersifat cloud

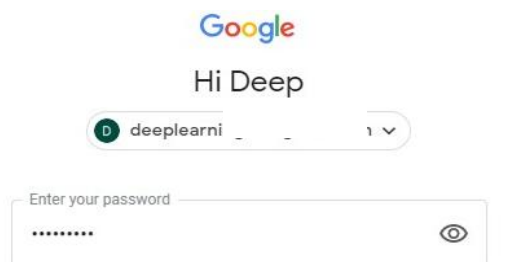
B. PENGGUNAAN GOOGLE COLAB

1. Akses Halaman Google Colab melalui laman ini : <https://colab.research.google.com/> , maka akan muncul halaman pembuka Google Colab



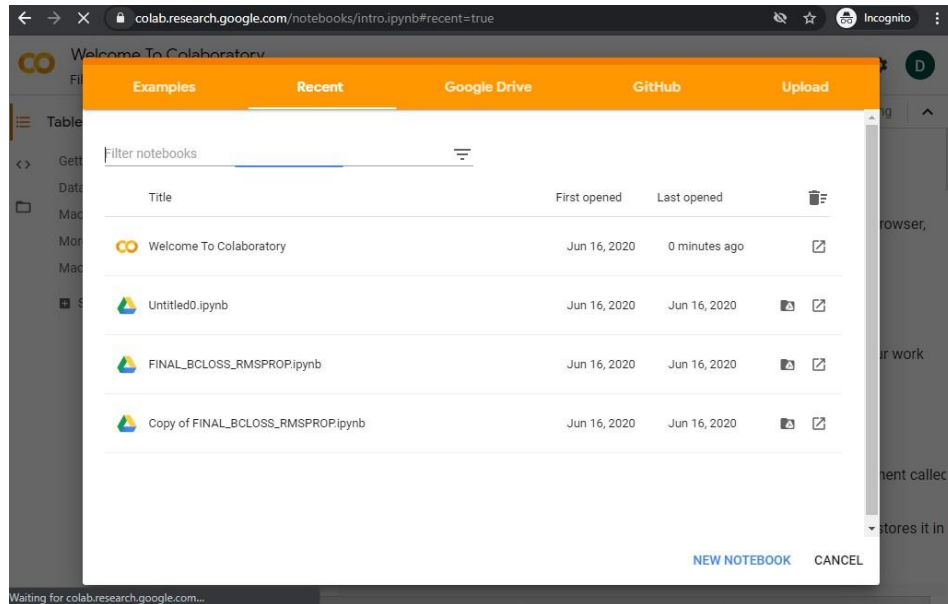
Gambar 1 Tampilan Awal Google Colab

2. Untuk menggunakan layanan Google Colab, kita diharuskan memiliki akun Google. Jika belum memiliki akun, silahkan daftar melalui link ini <https://accounts.google.com/>
3. Jika sudah memiliki akun Google, silahkan Sign In menggunakan akun Google.



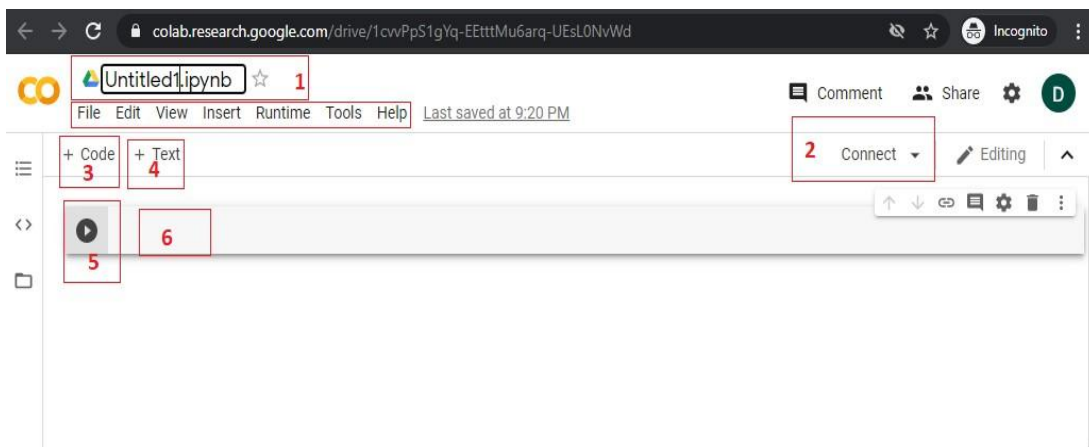
Gambar 2 Halaman Sign In Google

- Setelah login, maka akan muncul sebuah jendela yang memberitahukan apakah kita mau membuat File Notebook yang baru atau menggunakan File yang sudah ada sebelumnya



Gambar 3 Jendela Awal Setelah Login Google Colab

- Klik NEW NOTEBOOK untuk membuat sebuah file notebook baru untuk menjalankan kode python di Google Colab, maka akan dibuat sebuah file Notebook yang baru seperti gambar dibawah ini.



Gambar 4 File Notebook pada Google Colab

Keterangan Gambar 4:

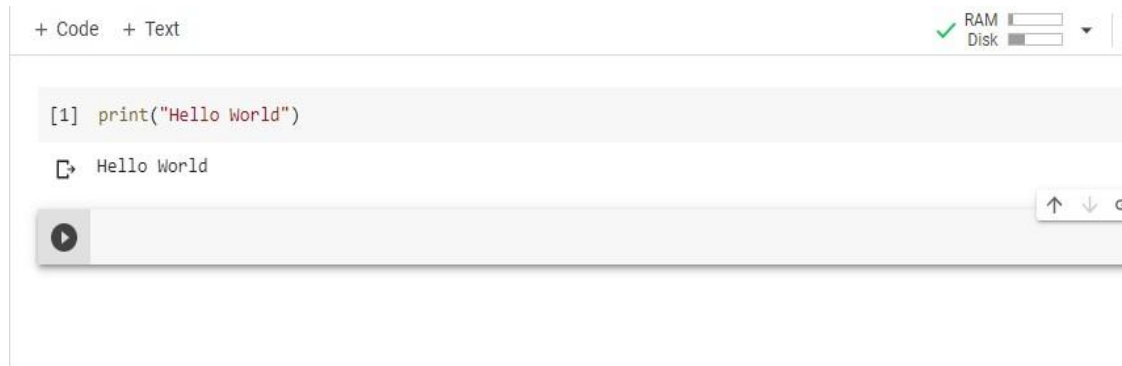
- a. Nama File
 - b. Untuk mengaktifkan (menghubungkan) Notebook kita di Google Colab
 - c. Menambah baris kode Notebook
 - d. Menambahkan keterangan Teks
 - e. Eksekusi kode python yang dituliskan kode baris
 - f. Cell Tempat baris kode dituliskan
-
6. Untuk mengaktifkan File Notebook yang telah dibuat, klik Connect (Seperti yang ditunjukkan pada Gambar 4, No 2)
 7. Setelah Connect, maka File Notebook sudah aktif dan bisa menjalankan kode python



Gambar 5 Notebook Sudah Aktif

C. MENJALANKAN KODE PROGRAM PYTHON DI COLAB

1. Ketik baris kode python di Cell kode python, kemudian klik tombol play di pojok kiri cell atau **Shift + Enter** bersamaan. Maka kode baris yang kita ketik, akan dieksekusi, dan memberikan Output dibawah baris kode yang kita buat.



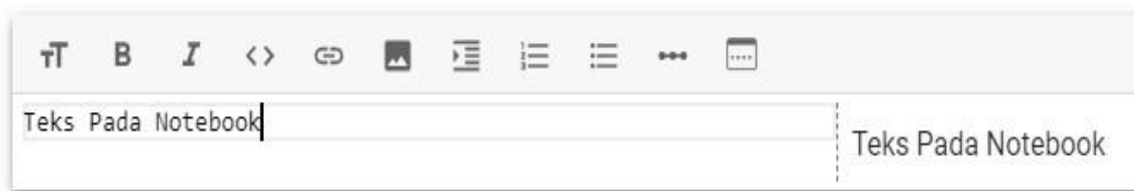
Gambar 6 Hasil Eksekusi Kode Python

2. Pada Cell kode baris tersebut, ada beberapa tools yang terdapat pada Cell, kita dapat memindahkan sebuah cell ke atas atau bawah, membuat *hyperlink*, menghapus cell, maupun memberikan komentar pada cell tersebut.



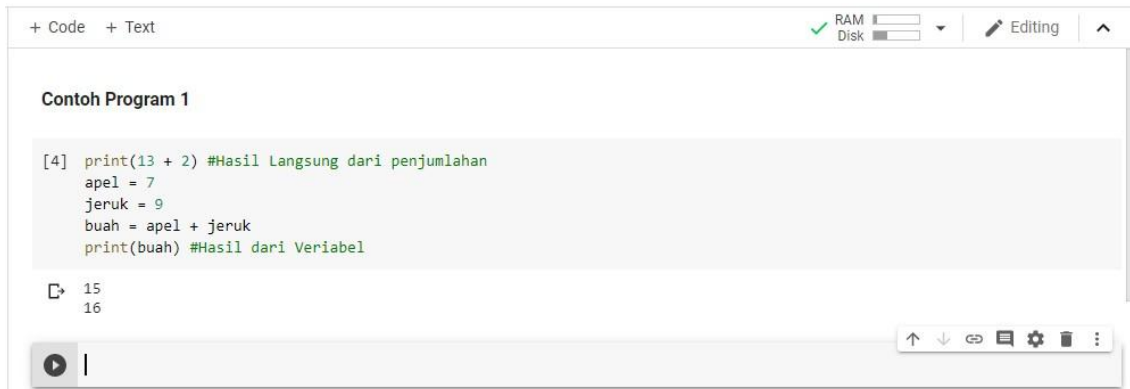
Gambar 7 Pengaturan pada Cell Notebook Goole Colab

3. Kita dapat menambahkan teks pada Notebook, dengan meng-klik tombol +Text. Ketikkan teks yang mau dibuat pada cell text editor, kemudian eksekusi dengan tombol play atau tekan tombol **Shift+Enter** bersamaan.



Gambar 8 Text Editor untuk menambahkan text pada Notebook

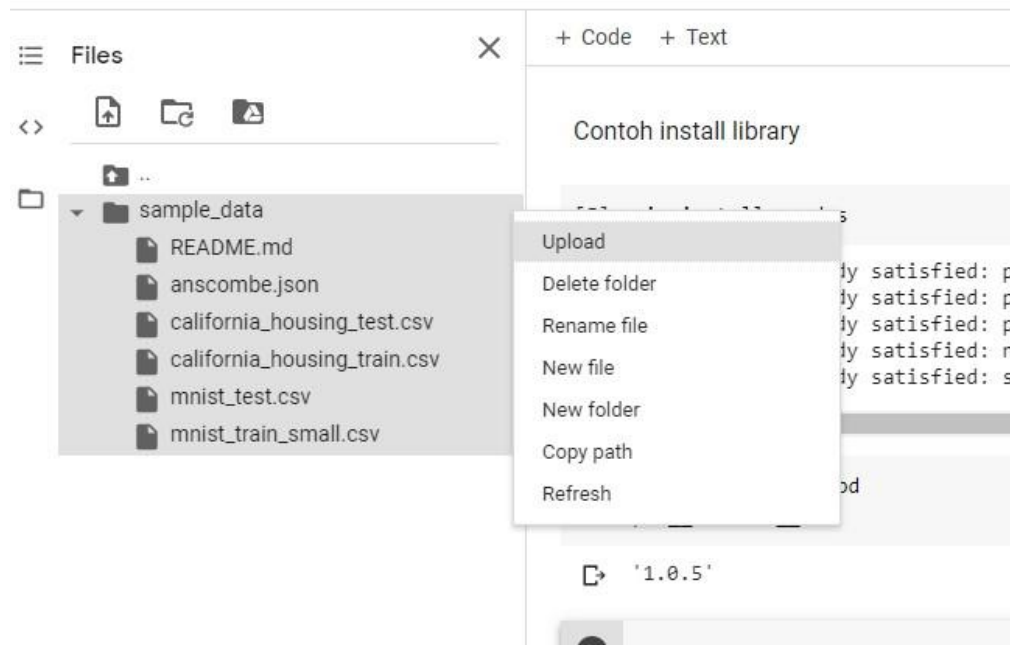
4. Contoh penggunaan Cell +Code dan +Text




Gambar 9 Contoh Kode Python dan Text

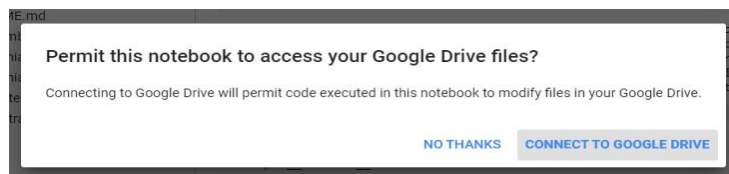
D. UPLOAD DAN AKSES GOOGLE DRIVE

1. Untuk upload file kedalam google colab, caranya klik pada **icon folder** pada sidebar sebelah kiri.
2. Kemudian klik folder “**content/sample_data**” kemudian pada sample_data klik kanan → upload. Jika upload sudah selesai klik refresh.



Gambar 10 Upload file pada Google Colab

3. Cara lain untuk membaca file adalah dengan mengintegrasikan Google Drive kita ke dalam google colab. Untuk langkah awal kita perlu upload file kedalam google drive. Kemudian klik tombol **Mount Drive** 
4. Kemudian akan muncul jendela pemberitahuan untuk mengizinkan Google Colab mengakses Google Drive Anda. Kemudian klik **Connect to Google Drive**

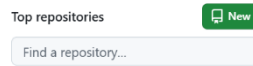


Gambar 11 Jendela permintaan akses Google Drive melalui Google Colab

5. Untuk membaca file yang diupload ke Google Drive, bisa kita akses dengan path “**/content/sample_data/**”

E. MENGHUBUNGKAN COLAB KE GITHUB

1. Buat repositories baru di github dengan klik “New”



Gambar 12 Membuat Repositories Baru

2. Beri nama dan klik “Create”

Create a new repository

A repository contains all project files, including the revision history. [Import a repository.](#)

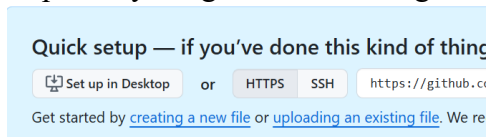
Required fields are marked with an asterisk (*).

Owner * / Repository name *

AlgoritmaAGB is available.

Gambar 13 Tampilan membuat nama repository

3. Buat file baru di dalam repository dengan klik “creating new file”



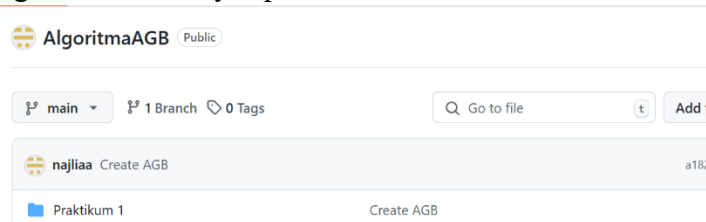
Gambar 14 Tampilan awal membuat file baru

4. Beri nama file



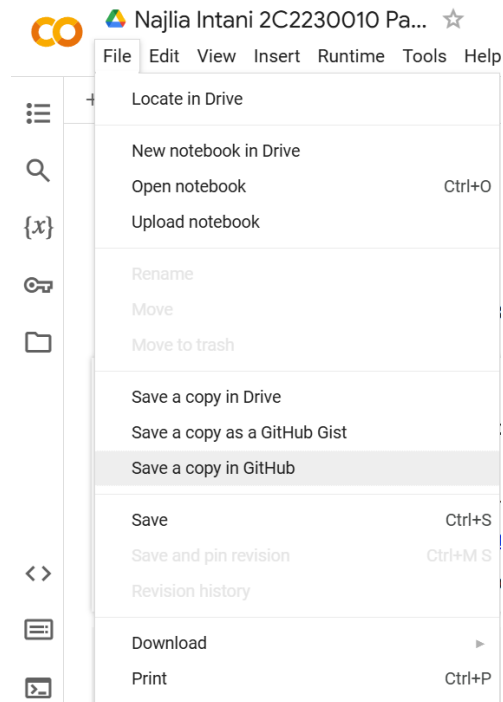
Gambar 15 Contoh Memberi nama file

5. “Commit Changes” untuk menyimpan



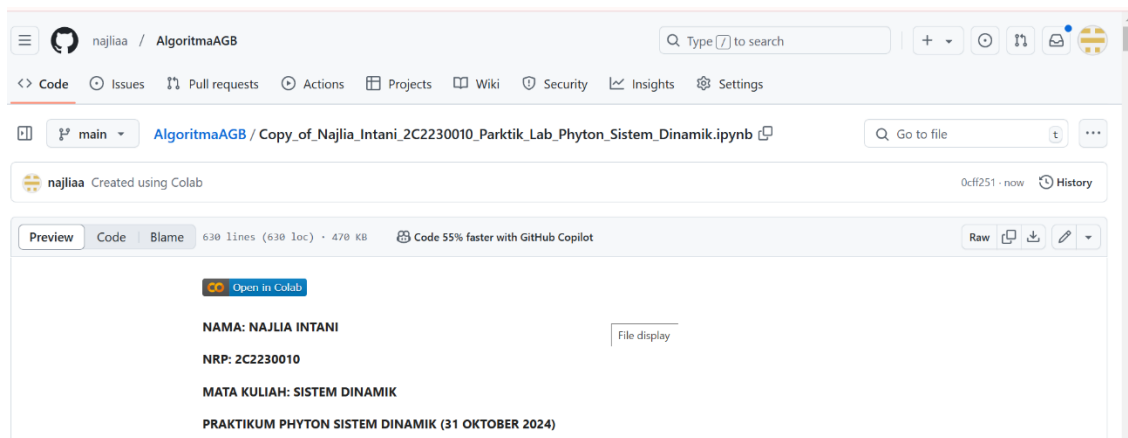
Gambar 16 Tampilan setelah disimpan

6. Buka file di Google Colab lalu klik “File” lalu klik “Save a copy in GitHub”



Gambar 17 Membuka pilihan File Colab

7. Menunggu proses beralih ke GitHub



Gambar 18 Tampilan Akhir pada GitHub

F. PENDAHULUAN PYTHON

a. Variabel dan Assignment

Variabel adalah lokasi memori yang dicadangkan untuk menyimpan nilai-nilai. Ini berarti bahwa ketika Anda membuat sebuah variabel Anda memesan beberapa ruang di memori. Variabel menyimpan data yang dilakukan selama program dieksekusi, yang nantinya isi dari variabel tersebut dapat diubah oleh operasi - operasi tertentu pada program yang menggunakan variabel.

Variabel dapat menyimpan berbagai macam tipe data. Di dalam pemrograman Python, variabel mempunyai sifat yang dinamis, artinya variabel Python tidak perlu dideklarasikan tipe data tertentu dan variabel Python dapat diubah saat program dijalankan. Penulisan variabel Python sendiri juga memiliki aturan tertentu, yaitu :

- Karakter pertama harus berupa huruf atau garis bawah/underscore _
- Karakter selanjutnya dapat berupa huruf, garis bawah/underscore _ atau angka
- Karakter pada nama variabel bersifat sensitif (case-sensitif). Artinya huruf kecil dan huruf besar dibedakan. Sebagai contoh, variabel namaDepan dan namadepan adalah variabel yang berbeda.

Untuk mulai membuat variabel di Python caranya sangat mudah, Anda cukup menuliskan variabel lalu mengisinya dengan suatu nilai dengan cara menambahkan tanda sama dengan = diikuti dengan nilai yang ingin dimasukkan. Dibawah ini adalah contoh penggunaan variabel dalam bahasa pemrograman Python

```
#proses memasukan data ke dalam variabel
nama = "John Doe"
#proses mencetak variabel
print(nama)

#nilai dan tipe data dalam variabel dapat diubah
umur = 20 #nilai awal
print(umur) #mencetak nilai umur
type(umur) #mengecek tipe data umur
umur = "dua puluh satu" #nilai setelah diubah
print(umur) #mencetak nilai umur
type(umur) #mengecek tipe data umur

namaDepan = "Budi"
namaBelakang = "Susanto"
nama = namaDepan + " " + namaBelakang
umur = 22
hobi = "Berenang"
print("Biodata\n", nama, "\n", umur, "\n", hobi)
```

```
#contoh variabel lainnya
inivariabel = "Halo"
ini_juga_variabel = "Hai"
_inivariabeljuga = "Hi"
inivariabel222 = "Bye"

panjang = 10
lebar = 5
luas = panjang * lebar
print(luas)
```

b. Input dan Output

Dalam python, program untuk menulis **"Hello, World!"** ke layar adalah seperti berikut:

```
print("Hello, World!") # (1)

print("Ini program", end="") # (2)

print(" pertama saya") # (3)

print("Dengan", "Koma")
# Ini akan menghasilkan luaran "Dengan Koma"

print("Dengan" + " Plus")
# Ini akan menghasilkan luaran "DenganPlus"
# Perhatikan '+' pada print melakukan konkatenasi pada string saja.

variabel = " sebuah nilai"
print(f"Dengan format {variabel}")
# Ini akan menghasilkan luaran "Dengan format sebuah nilai"

# {variabel} akan diubah dengan nilai dari variabel
# bersesuaian.

# Ini adalah sebuah komentar.
# Semua yang ada setelah tanda pagar (#) akan diabaikan oleh interpreter.

"""
Selain itu, kita juga bisa membuat komentar multi-baris dengan tiga petik
(""") """
```

Bagian yang ditandai nomor 1 bertugas menuliskan **"Hello, World!"** ke layar. Sintaks **print** akan otomatis memindahkan baris setelah selesai menulis ke layar, kecuali disertai parameter `end=""`. Berarti, output program di atas adalah **"Hello, World!"** diikuti dengan **"Ini program pertama saya "** di baris selanjutnya.

Untuk melakukan input, kita membutuhkan penampung untuk menyimpan data yang diinputkan. Sebagai contoh, di bawah ini adalah program yang menerima input dan menuliskan ulang yang dimasukkan.

```
S = input("Masukkan kalimat: ") # (1)
print("Anda memasukkan kalimat: " + S) # (2)

N = int(input("Masukkan sebuah angka: ")) # (3)
print("Jika angka Anda ditambah 5, hasilnya: " + str(N + 5)) # (4)
```

c. Tipe Data

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 :

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

```

#tipe data Boolean
print(True)

#tipe data String
print("Ayo belajar Python")
print('Belajar Python Sangat Mudah')

#tipe data Integer
print(20)

#tipe data Float
print(3.14)

#tipe data Hexadecimal
print(0x9a) # Perbaikan: menambahkan awalan 0x

#tipe data Complex
print(5j)

#tipe data List
print([1,2,3,4,5])
print(["satu", "dua", "tiga"])

#tipe data Tuple
print((1,2,3,4,5))
print(("satu", "dua", "tiga"))

#tipe data Dictionary
print({"nama":"Budi", 'umur':20})
#tipe data Dictionary dimasukan ke dalam variabel biodata
biodata = {"nama":"Andi", 'umur':21} #proses inisialisasi variabel biodata
print(biodata) #proses pencetakan variabel biodata yang berisi tipe data
Dictionary
print(type(biodata)) #fungsi untuk mengecek jenis tipe data. akan tampil
<class 'dict'> yang berarti dict adalah tipe data dictionary

```

d. Konversi Tipe Data Python

Pada Python Anda bisa mengkonversi tipe data dengan menggunakan fungsi. Dibawah ini adalah beberapa fungsi untuk mengkonversi tipe data number Python.

- Konversi Integer ke Float

```
x = float(5)

print(x)
```

- Konversi Float ke Integer

```
x = int(5.6)

print(x)
```

- Konversi nilai ke String

```
x = str(9)

print(x)
```

- Konversi kumpulan data ke Set, Tuple, dan List

```
x = set([1, 2, 3])
y = tuple({5, 6, 7})
z = list(('hello'))

print(x)
print(y)
print(z)
```

- `complex(x)` untuk meng-konversi x menjadi complex number dengna real part x dan imaginary part zero.
- `complex(x, y)` untuk meng-konversi x dan y menjadi complex number dengan real part x dan imaginary part y. x dan numeric expressions y.

e. Ekspresi

Ekspresi dalam Python adalah sebuah kombinasi dari nilai, variabel, operator, dan fungsi yang dievaluasi untuk menghasilkan sebuah nilai dalam suatu tipe tertentu.

Struktur umum dari ekspresi adalah

<Operan1> <Operator> <Operan2>

Berikut merupakan penjelasannya

1. Operan dapat berupa nilai, variabel, konstanta, atau ekspresi lain.
2. Operator merupakan suatu fungsi standar yang disediakan dalam bahasa pemrograman untuk melakukan beberapa hal dasar seperti perhitungan.

Contoh sederhana dari ekspresi adalah sebagai berikut:

Ekspresi

$$5x + 3 = 8$$

Suku Bilangan

Contoh penerapan ekspresi pada Python:

```
x = 8
y = 2

result = x + y

print(result)
```

f. Operator

Operator dalam Python adalah simbol atau kata kunci yang digunakan untuk melakukan operasi pada nilai (operand). Operator berfungsi untuk melakukan perhitungan matematika, manipulasi data, atau pengambilan keputusan dalam program. Berikut beberapa jenis operator dalam Python:

1. Operator Aritmatika

Operator	Contoh	Penjelasan
Penjumlahan <code>+</code>	<code>1 + 3 = 4</code>	Menjumlahkan nilai dari masing-masing operan atau bilangan
Pengurangan <code>-</code>	<code>4 - 1 = 3</code>	Mengurangi nilai operan di sebelah kiri menggunakan operan di sebelah kanan
Perkalian <code>*</code>	<code>2 * 4 = 8</code>	Mengalikan operan/bilangan
Pembagian <code>/</code>	<code>10 / 5 = 2</code>	Untuk membagi operan di sebelah kiri menggunakan operan di sebelah kanan
Sisa Bagi <code>%</code>	<code>11 % 2 = 1</code>	Mendapatkan sisa pembagian dari operan di sebelah kiri operator ketika dibagi oleh operan di sebelah kanan
Pangkat <code>**</code>	<code>8 ** 2 = 64</code>	Memangkatkan operan disebelah kiri operator dengan operan di sebelah kanan operator
Pembagian Bulat <code>//</code>	<code>10 // 3 = 3</code>	Sama seperti pembagian. Hanya saja angka dibelakang koma dihilangkan

```

# Penjumlahan
print(13 + 2)
apel = 7
jeruk = 9
buah = apel + jeruk # Menjumlahkan jumlah apel dan jeruk
print(f"Jumlah buah: {buah}")

# Pengurangan
hutang = 10000
bayar = 5000
sisahutang = hutang - bayar # Menghitung sisa hutang
print(f"Sisa hutang Anda adalah {sisahutang}")

# Perkalian
panjang = 15
lebar = 8
luas = panjang * lebar # Menghitung luas persegi panjang
print(f"Luas persegi panjang: {luas}")

# Pembagian
kue = 16
anak = 4
kue_per_anak = kue / anak # Menghitung bagian kue per anak
print(f"Setiap anak akan mendapatkan bagian kue sebanyak {kue_per_anak}")

# Sisa Bagi / Modulus
bilangan1 = 14
bilangan2 = 5
hasil = bilangan1 % bilangan2 # Menghitung sisa bagi
print(f"Sisa bagi dari bilangan {bilangan1} dan {bilangan2} adalah {hasil}")

# Pangkat
bilangan3 = 8
bilangan4 = 2
hasil_pangkat = bilangan3 ** bilangan4 # Menghitung hasil pangkat
print(f"Hasil pangkat: {hasil_pangkat}")

# Pembagian Bulat
print(10 // 3)
# 10 dibagi 3 adalah 3.3333. Karena dibulatkan maka akan menghasilkan
nilai 3

```

2. Operator Perbandingan

Operator	Contoh	Penjelasan
Sama dengan <code>==</code>	<code>1 == 1</code>	bernilai True Jika masing-masing operan memiliki nilai yang sama, maka kondisi bernilai benar atau True.
Tidak sama dengan <code>!=</code>	<code>2 != 2</code>	bernilai False Akan menghasilkan nilai kebalikan dari kondisi sebenarnya.
Tidak sama dengan <code><></code>	<code>2 <> 2</code>	bernilai False Akan menghasilkan nilai kebalikan dari kondisi sebenarnya.
Lebih besar dari <code>></code>	<code>5 > 3</code>	bernilai True Jika nilai operan kiri lebih besar dari nilai operan kanan, maka kondisi menjadi benar.
Lebih kecil dari <code><</code>	<code>5 < 3</code>	bernilai True Jika nilai operan kiri lebih kecil dari nilai operan kanan, maka kondisi menjadi benar.
Lebih besar atau sama dengan <code>>=</code>	<code>5 >= 3</code>	bernilai True Jika nilai operan kiri lebih besar dari nilai operan kanan, atau sama, maka kondisi menjadi benar.
Lebih kecil atau sama dengan <code><=</code>	<code>5 <= 3</code>	bernilai True Jika nilai operan kiri lebih kecil dari nilai operan kanan, atau sama, maka kondisi menjadi benar.

```
# SAMA DENGAN
print(1 == 1) # Hasilnya akan bernilai True karena satu sama dengan satu
print(1 == 2) # Hasilnya akan bernilai False karena satu tidak sama dengan dua

# TIDAK SAMA DENGAN
print(2 != 2) # Hasilnya akan bernilai False karena dua seharusnya sama dengan dua
print(2 != 3) # Hasilnya akan bernilai True karena dua tidak sama dengan tiga
```

```
# LEBIH BESAR DARI
print(5 > 3) # Hasilnya akan bernilai True karena lima lebih besar dari
tiga

# LEBIH KECIL DARI
print(5 < 3) # Hasilnya akan bernilai False karena lima tidak lebih besar
dari tiga

# LEBIH BESAR DARI SAMA DENGAN
print(5 >= 3) # Hasilnya akan bernilai True karena lima lebih besar dari
sama dengan tiga

# LEBIH KECIL DARI SAMA DENGAN
print(5 <= 3) # Hasilnya akan bernilai False karena lima tidak lebih besar
dari sama dengan tiga
```

3. Operator Penugasan

Operator	Contoh	Penjelasan
Sama dengan <code>=</code>	<code>a = 1</code>	Memberikan nilai di kanan ke dalam variabel yang berada di sebelah kiri.
Tambah sama dengan <code>+=</code>	<code>a += 2</code>	Memberikan nilai variabel dengan nilai variabel itu sendiri ditambah dengan nilai di sebelah kanan.
Kurang sama dengan <code>-=</code>	<code>a -= 2</code>	Memberikan nilai variabel dengan nilai variabel itu sendiri dikurangi dengan nilai di sebelah kanan.
Kali sama dengan <code>*=</code>	<code>a *= 2</code>	Memberikan nilai variabel dengan nilai variabel itu sendiri dikali dengan nilai di sebelah kanan.
Bagi sama dengan <code>/=</code>	<code>a /= 4</code>	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan.
Sisa bagi sama dengan <code>%=</code>	<code>a %= 3</code>	Memberikan nilai variabel dengan nilai variabel itu sendiri dibagi dengan nilai di sebelah kanan. Yang diambil nantinya adalah sisa baginya.
Pangkat sama dengan <code>**=</code>	<code>a **= 3</code>	Memberikan nilai variabel dengan nilai variabel itu sendiri dipangkatkan dengan nilai di sebelah kanan.
Pembagian bulat sama dengan <code>//=</code>	<code>a //= 3</code>	Membagi bulat operan sebelah kiri operator dengan operan sebelah kanan operator kemudian hasilnya diisikan ke operan sebelah kiri.

g. Prioritas Eksekusi Operator di Python

Operator	Keterangan
<code>**</code>	Aritmatika
<code>~, +, -</code>	Bitwise
<code>*, /, %, //</code>	Aritmatika
<code>+, -</code>	Aritmatika
<code>>>, <<</code>	Bitwise
<code>&</code>	Bitwise
<code>^</code>	Bitwise
<code><=, <, >, >=</code>	Perbandingan
<code><>, ==, !=</code>	Perbandingan
<code>=, %=, /=, //=, -=, +=, *=, **=</code>	Penugasan
<code>is, is not</code>	Identitas
<code>in, not in</code>	Membership (Keanggotaan)
<code>not, or, and</code>	Logika

LATIHAN PRAKTIKUM DASAR

1. Buatlah kode program dalam bahasa Python untuk menghitung luas persegi. Kode program butuh 1 inputan berupa panjang sisi persegi, kemudian tampilkan output luas persegi.
2. Buatlah program yang meminta data inputan berupa Nama, NIM, Fakultas, Jurusan, Kota Asal dan Alamat mahasiswa. Program kemudian menyimpan data ini di dalam variabel lalu menampilkan semua hasilnya.