BAB 1

PYTHON

1.1 Sejarah Python

Python merupakan bahasa pemrograman yang sangat populer. Guido van Rossum merupakan yang membuat bahasan pemrograman Python dan dikenalkan pada tahun 1991. Bahasa python terinspirasi dari bahasa pemrograman ABC. Sampai saat ini, Guido masih menjadi penulis utama untuk bahasa pemrograman python, meskipun bersifat open source sehingga ribuan orang juga dapat berkontribusi dalam mengembangkannya. Saat ini pengembangan bahasa pemrograman Python terus dilakukan oleh sekumpulan pemrogram Python Software Foundation. Python Software Foundation merupakan organisasi non-profit yang dibentuk sebagai pemegang hak cipta intelektual Python sejak versi awal. Nama python bukan berasal dari nama ular yang sering kita kenal. Guido memilih nama Python sebagai nama bahasa ciptaannya dikarenakan Guido sangat menyukai acara televisi Monty Python's Flying Circus. Oleh sebab itu seringkali ungkapan dari acara tersebut sering muncul dalam korespondensi antar pengguna Python.

1.2 Pembahasan Python

Python merupakan bahasa pemrograman tinggi yang berguna untuk melakukan eksekusi jumlah instruksi multi guna secara interpretatif dengan metode OOP, serta menggunakan semantik dinamis untuk memberikan tingkat keterbacaan syntax. Sebagai bahasa pemrograman tinggi, python dapat dipelajari dengan mudah karena sudah dilengkapi dengan manajemen memori otomatis (*pointer*).

Gambar 1.1 Logo Python

Python bisa digunakan secara bebas, bahkan untuk kepentingan komersial sekalipun. Banyak perusahaan yang mengembangkan bahasa pemrograman python secara komersial untuk memberikan layanan. Misalnya Anaconda Navigator, adalah salah satu aplikasi untuk pemrograman python yang dilengkapi dengan tool-tool pengembangan aplikasi. Python dapat memberikan kualitas dan kecepatan untuk membangun sebuah aplikasi bertingkat atau *Rapid Application Development*. Hal tersebut didukung karena adanya *library* dengan modul-modul baik standar maupun tambahan contohnya NumPy, SciPy, dan sebagainya. Python mempunyai komunitas yang besar sebagai tempat tanya jawab. Syntax pada python dapat dijalankan dan ditulis untuk membangun sebuah aplikasi di berbagai sistem operasi, contohnya seperti:

1. Microsoft Windows : merupakan keluarga sistem operasi. yang dikembangkan oleh Microsoft, dengan menggunakan antarmuka pengguna grafis.

2. Linux atau Unix : adalah proyek perangkat lunak bebas dan sumber terbuka terbesar di dunia.

3. Java Virtual Machine : mesin virtual yang digunakan secara khusus mengeksekusi berkas bytecode java.

4. Android : sistem operasi berbasis Linux yang dirancang untuk perangkat bergerak layar sentuh seperti telepon pintar dan komputer tablet.

5. Mac OS : ntarmuka grafikal sistem operasi yang dikembangkan dan disebarkan oleh Apple Inc.

6. Amiga : merupakan komputer pribadi yang dikembangkan oleh Amiga Corporation

7. Palm : merupakan sistem operasi mobile yang memberikan kemudahan penggunaan dengan layar sentuh berbasis graphical user interface.

8. OS/2 : sistem operasi yang dibuat secara bersama-sama oleh International Business Machine Corporation dan Microsoft Corporation, untuk digunakan pada komputer IBM PS/2, sebagai pengganti sistem operasi DOS yang telah lama digunakan.

9. Symbian OS : merupakan sistem operasi yang dikembangkan oleh Symbian Ltd. yaitu yang dirancang untuk peralatan bergerak.

10. Win 9x/NT/2000 : merupakan sebuah sistem operasi Microsoft yang menjadi leluhur.

11. Unix (Solaris, Linux, FreeBSD, AIX, HP/UX, SunOS, IRIX, dan lain - lain)

12. Macintosh (Intel, PPC, 68K) : merupakan jenis komputer personal berbasis PowerPC yang diproduksi oleh Apple.

13. DOS : merupakan generik bagi setiap sistem operasi yang dimuat dari perangkat penyimpanan berupa disk saat sistem komputer dinyalakan.

14. Nokia mobile phones : merupakan eralatan telekomunikasi ponsel pintar dan ponsel genggam.

15. Windows CE : merupakan perangkat lunak keluaran Microsoft untuk perangkat keras organizer.

16. Acorn/RISC OS : merupakan perangkat lunak milik perorangan dan antarmuka pengguna grafis.

17. BeOS : merupakan sistem operasi untuk komputer pribadi yang dikembangkan pada tahun 1991 oleh Be Inc.

18. VMS/OpenVMS : merupakan server komputer sistem operasi yang berjalan pada VAX, Alfa dan berbasis Itanium keluarga komputer.

19. QNX : adalah sistem operasi komersial realtime mirip UNIX yang khusus ditujukan untuk perangkat embedded.

20. VxWorks : merupakan sistem operasi waktu nyata (real-time operating system RTOS) yang dapat digunakan dalam sistem tertanam.

21. Psion : merupakan komputer mungil atau kecil.

Python juga digunakan di berbagai pada bidang pengembangan. Berikut adalah beberapa contoh aplikasi penggunaan python yang paling populer:

1. Penelitian Ilmiah dan Numerik

   Bahasa pemrograman python dapat digunakan untuk mengerjakan riset ilmiah untuk dapat mempermudah perhitungan dalam numerik. Contohnya penerapan pada algoritma *Double Exponential Smoothing*, *Naive Bayes*, *Decision Tree*, *Least Square*, KNN dan sebagainya.

2. Data Science dan Big Data

   Python sangat memungkinkan untuk dapat melakukan analisis data dari database big data.

3. Media dalam Pembelajaran Program

   Bahasa pemrograman python bisa digunakan sebagai media pembelajaran di perguruan tinggi atau universitas. Bahsa pemrograman python hemat dan sangat mudah untuk dipelajari sebagai *Object Oriented Programming* dibandingkan bahasa pemrograman lainnya seperti, C++, MATLAB dan C#.

4. Pengembangan Software

   Python menyediakan dukungan struktur kode untuk mempermudah pengembangan software.

5. Graphical User Interface (GUI)

   Bahasa pemrograman python bisa digunakan untuk membuat interface dari sebuah aplikasi, dan tersedia *library* untuk membuat GUI menggunakan bahasa pemrograman python, contohnya win32extension, Qt dan GTK+.

6. Website dan Internet

   Bahasa pemrograman python bisa juga digunakan untuk *server side* yang diintegrasikan dengan berbagai macam internet protokol contohnya yaitu seperti HTML, JSON, FTP, IMAP dan Email Processing. Selain itu, juga python juga mempunyai *library* yang berguna untk pengembangan internet.

7. Aplikasi Bisnis

   Bahasa pemrograman python juga dapat digunakan untuk membuat sistem informasi baik itu untuk bisnis ataupun instansi.

1.3 Perbedaan Python2 dan Python3

Bahasa pemrograman python versi 2 dan python versi 3 tidak jauh berbeda. Tetapi, disini akan dijelaskan untuk memberikan sedikit perbedaan - perdeaan dari yang telah ditemukan dan diketahui sebelumnya. Berikut adalah contohnya:

1. Print

   Pada Python2, print diperlakukan seperti statemen ketimbang sebuah function.

   print ”Esi Vidia Rahcmadani”

   Sedangkan pada Python 3, print diperlakukan sebagai function.

   print (”Esi Vidia Rahcmadani”)

   Perubahan ini membuat sintaksis pada Python lebih konsisten. Penggunaan print() juga kompatibel dengan Python 2.7.

2. Pembagian Pada Integer Pada Python 2, semua tipe data angka yang tidak mengandung desimal akan diperlakukan sebagai integer. Terlihat mudah pada awalnya, ketika mencoba untuk membagi kedua integer akan didapatkan tipe data float.

   3 / 2 = 1.5

   Python 2 menggunakan floor division atau dibulatkan ke nilai paling rendah misalnya 1.5 jadi 1, 2.6 jadi 2 dan seterusnya. Pada Python 2.7 akan menjadi seperti ini

   ```
   x = 3 / 2
   print a
   #Output
   1
   ```

   Untuk desimal maka tambahkan jadi seperti ini 3.0 / 2.0 untuk mendapatkan hasil 1.5 pada Python 3, pembagian pada bilangan integer lebih intuitif :

   ```
   a = 3 / 2
   print(a)
   #Output
   1.5
   ```

   Kita juga masih bisa melakukan 3.0 / 2.0 untuk mendapatkan 1.5 namun untuk mendapatkan floor division maka pada Python 3 gunakan // :

   ```
   b = 3 // 2
   print(b)
   ```

```
#Output
1
```

Fitur Python 3 ini tidak kompatibel dengan Python 2.7

3. Dukungan Unicode

   Ketika bahasa pemrograman menangani tipe data string (yang mana merupakan sekumpulan dari beberapa karakter), mereka dapat melakukan beberapa cara berbeda sehingga komputer dapat mengubah angka ke huruf dan simbol lainnya. Python2 menggunakan alfabet ASCII (*American Standard Code for Information Interchange*) secara default, sehingga ketika kita mengetik Hallo! maka Python2 menangani string sebagai ASCII. Terbatas pada beberapa ratus karakter, ASCII mungkin bukan pilihan yang fleksibel untuk menangani proses encoding terutama yang non English. Untuk menggunakan unicode yang lebih luwes, mendukung lebih dari 128,000 karakter maka kita harus mengetik u Halo! , dengan tambahan u di depannya yang mana berarti Unicode. Python3 menggunakan Unicode secara default, yang mana menyelamatkan programmer dari tambahan kode lagi, lebih hemat waktu dan mudah untuk diisikan dan ditampilkan. Karena Unicode mendukung berbagai karakter linguistik yang beragam termasuk menampilkan emoji, penggunaan karakter secara default dengan encoding memastikan perangkat mobile didukung oleh program yang kita buat. Jika kita ingin kode Python3 kita mendukung Python2, tambahkan u di depan string.

4. Kelanjutan Pengembangan

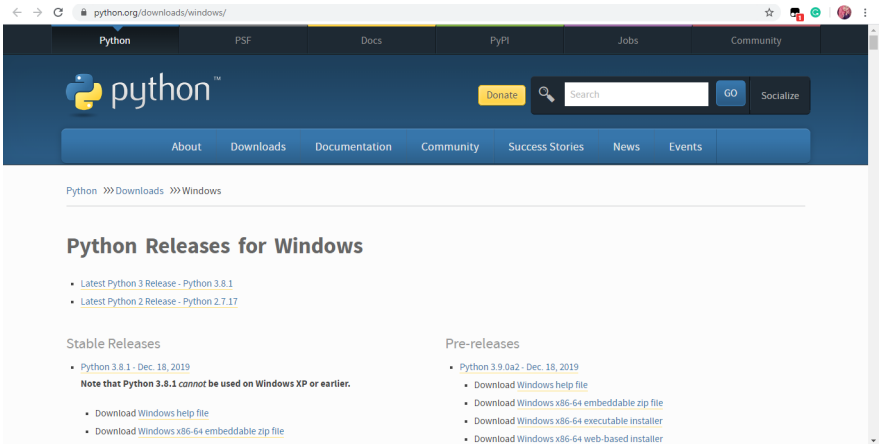
   Selain perbedaan mendasar antara Python2 dan Python3 secara sintatikal, pada fakta lainnya adalah Python 2.7 akan dihentikan dukungannya per tahun 2020 dan Python3 akan terus dikembangkan dengan fitur yang lebih banyak lagi kedepannya, dan tentu saja yang lebih penting adalah perbaikan dari bug dan performa. Pengembangan saat ini telah mendukung format *string* secara literal, kustomisasi pembuatan *class* secara sederhana dan sintaksis yang lebih bersih dan rapi untuk menangani perkalian matriks.

   Pengembangan Python3 akan terus berlangsung, hal ini berarti pengembang perangkat lunak akan dengan nyaman menggunakan Python3 karena terus didukung oleh komunitas dengan jangka waktu yang panjang. Tentu saja hal ini meningkatkan kinerja programmer dan kualitas program yang lebih baik lagi.

1.4 Instalasi Python pada Windows

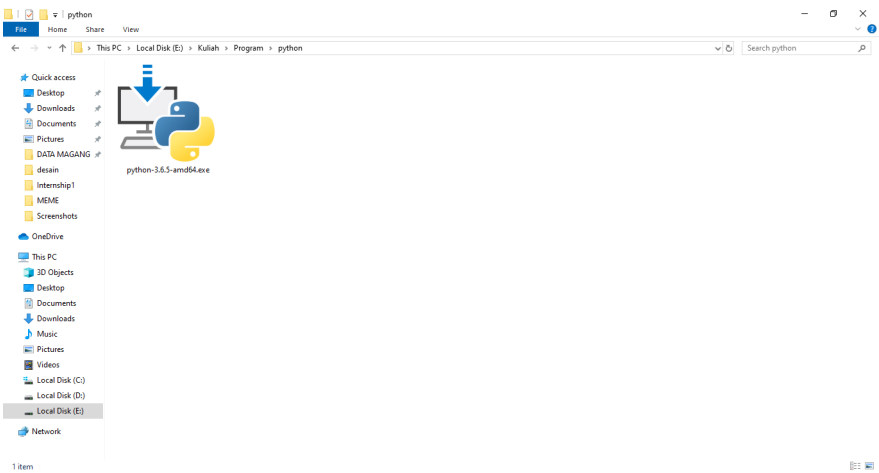
Instalasi python pada Windows sangatlah mudah. Langkah-langkanya yaitu sama seperti menginstall software Windows yang lainnya, next-next dan finish. Tetapi, terdapat konfigurasi yang harus dipilih terlebih dahulu ditengah-tengah proses instalasi, agar perintah pada Python dapat dikenali di CMD. Python yang akan di install

dalam adalah python versi 3. Sebelumnya download terlebih dahulu di situs resmi python yaitu di (python.org) dengan pilihan 64 bit atau 32 bit.



Gambar 1.2 Website python.org

1. Buka File Python, setelah mendownload aplikasi python selesai, maka sela kita akan mendapatkan file python. File ini akan melakukan instalasi ke sistem windows. Yang perlu diketahui tidak perlu lagi menginstal pip karena di python versi 3 ini sudah ada pipnya. Yang perlu menginstal pip yaitu python dengan versi 2. Klik dua kali untuk mengeksekusinya.

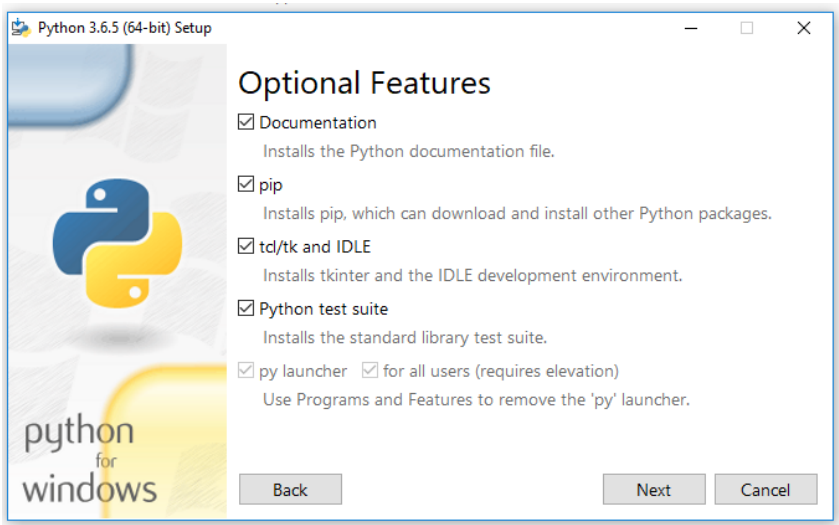


Gambar 1.3 File Python

2. Setelah membuka aplikasi python, ceklist Add Python 3.6 to PATH dan klik Customize installation.

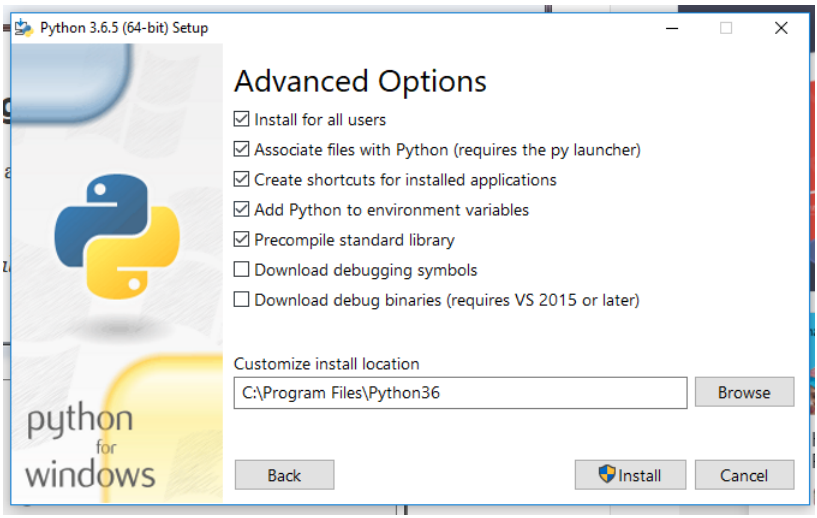
Gambar 1.4 Customize Installation

3. Ceklis atau pilih semua yaitu Documentation, pip, td/tk and IDLE, dan Python test suite. Setelah itu klik next.



Gambar 1.5 Optional Features

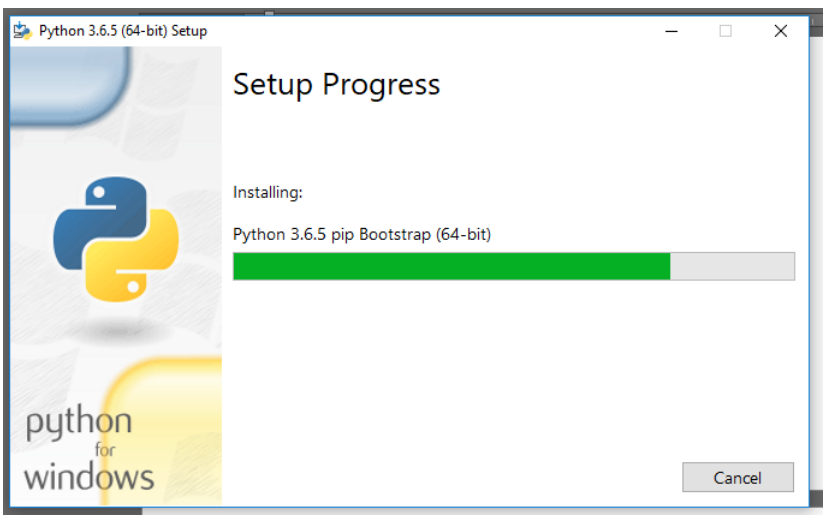
4. Ceklis atau pilih sesuai yang terdapat di gambar 1.6, lalu pilih lokasi penyimpanan dan klik install.



Gambar 1.6 Advanced Options

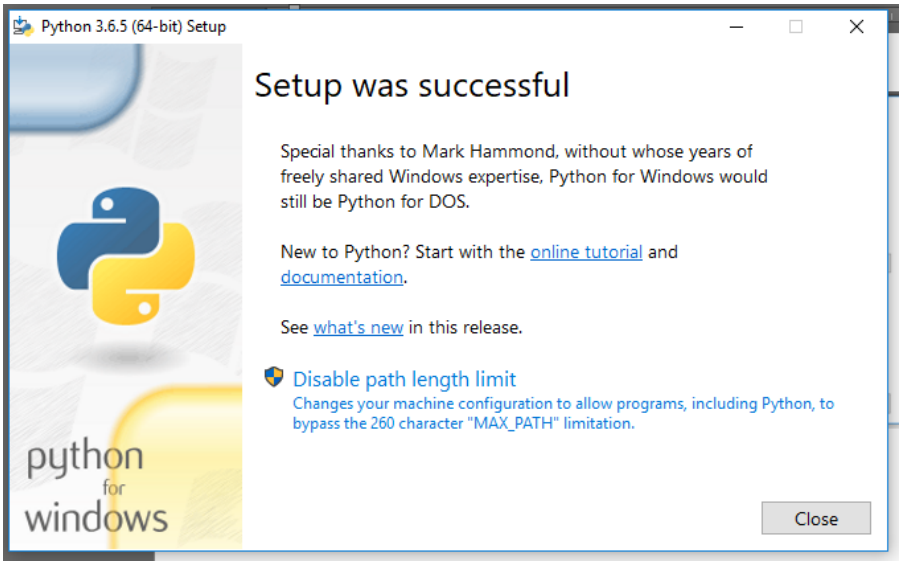
5. Setelah di klik install akan muncul pilihan apakah kita akan menginstal atau tidak, dan pilih yes.

6. Setelah itu kita tunggu sampai instalasi selesai



Gambar 1.7 Setup Progress

7. Instalasi selesai dan setelah itu dapat di close saja



Gambar 1.8 Setup Was Successful

1.5 Syntax Dasar

Selanjutnya akan diberikan contoh fungsi Python yang digunakan untuk mencetak atau *print*. Di Python untuk mencetak cukup gunakan fungsi print(), dimana sesuatu yang akan dicetak harus diletakkan diantara kurung buka dan kurung tutup, bahkan di Python2 tidak harus menggunakan tanda kurung kurawal, tetapi cukup pisahkan saja dengan spasi. Jika hendak mencetak tipe data *String* langsung, harus memasukanya ke dalam tanda kutip terlebih dahulu.

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

Saat menjalankan perintah script seperti yang terdapat diatas, akan melihat tampilan output berupa text Hello World.

1.6 Python Case Sensitive

Python sangat bersifat *case sensitive*, yang berarti huruf kecil dan huruf besar memiliki perbedaan. Contohnya jika menggunakan fungsi print dengan huruf kecil print() akan berhasil. Tetapi jika menggunakan huruf kapital Print() atau seperti PRINT(), akan muncul pesan *error*, ini berfungsi untuk nama variabel ataupun fungsi-fungsi lainya.

1.7 Komentar Python

Komentar merupakan kode di dalam *script* Python yang tidak akan dijalankan mesin atau tidak akan dieksekusi. Komentar hanya digunakan untuk memberikan keterangan tertulis pada script atau menandai. Komentar dapat digunakan untuk membiarkan orang lain memahami apa yang dilakukan pada script tersebut, dan atau untuk mengingatkan kepada programmer jika hendak mengedit script yang sudah ada. Untuk menggunakan komentar cukup mengetikkan tanda pagar #, dan diikuti dengan komentar yang sesuai dengan fungsi dari *script* tersebut. Komentar tidak akan dapat dieksekusi dan komentar hanya dapat digunakan untuk satu baris saja. Contoh penggunaan penggunaan komentar pada Python:

```
#Ini
#Adalah
#Sebuah
#Komentar
print("Hai Dunia") #ini juga komentar
#print("Hai")
#komentar
#bisa berisi
#spesial karakter #$&*(),!@./;'[%^]\
#mencetak
#nama
print("Vidia")
#mencetak
#angka/integer
print(1701)
```

Script diatas akan menampilkan *output* **Hai Dunia**, **Vidia**, dan **1701**.

1.8 Tipe Data pada Python

Tipe data merupakan memori pada komputer atau media yang digunakan untuk menampung sebuah informasi. Python mempunyai tipe data yang cukup unik bila bandingkan dengan bahasa pemrograman yang lainnya. Berikut merupakan beberapa tipe data dari bahasa pemrograman Python:

Tabel 1.1 Tipe Data

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

Berikut adalah contoh pengimplementasian macam-macam tipe data dengan *script* Python:

```
#tipe data Float
print(4.17)
#tipe data Complex
print(9e)
#tipe data String
print("Semangat Kawan")
print('Kita Pasti Bisa')
#tipe data Tuple
print((2,4,7,8,9))
print(("dua", "empat", "tujuh"))
#tipe data Boolean
print(False)
#tipe data Integer
print(98)
#tipe data Hexadecimal
print(7c)
```

```
#tipe data List
print([2,4,7,8,9])
print(["dua", "empat", "tujuh"])
#tipe data Dictionary
print({"Nama":"EsiVR", 'Umur':22})
#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
type(biodata) #fungsi untuk mengecek jenis tipe data. akan tampil <
    class
'dict'> #yang berarti dict adalah tipe data dictionary
```

1.9 Variabel Python

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 natinya 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 didekralasikan tipe data tertentu dan variabel Python dapat diubah saat program dijalankan. Penulisan variabel Python sendiri juga memiliki aturan tertentu, yaitu :

1. Karakter pertama harus berupa huruf atau garis bawah/underscore _
2. Karakter selanjutnya dapat berupa huruf, garis bawah/underscore _ atau angka
3. 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, cukup menuliskan variabel lalu mengisinya dengan suatu nilai dengan cara menambahkan tanda sama dengan = diikuti dengan nilai yang ingin dimasukan.

Contoh penggunaan variabel dalam bahasa pemrograman Python.

```
#proses memasukan data ke dalam variabel
nama = "Dava Evar"
#proses mencetak variabel
print(nama)
#nilai dan tipe data dalam variabel dapat diubah
umur = 17 #nilai awal
print(umur) #mencetak nilai umur
type(umur) #mengecek tipe data umur
umur = "tujuh belas" #nilai setelah diubah
print(umur) #mencetak nilai umur
```

```
type(umur) #mengecek tipe data umur
namaDepan = "Arif"
namaBelakang = "Budiman"
nama = namaDepan + " " + namaBelakang
umur = 29
hobi = "Traveling"
print("Biodata\n", nama, "\n", umur, "\n", hobi)
#contoh variabel lainya
inivariabel = "Haaaii"
ini_juga_variabel = "Hooo"
_inivariabeljuga = "Hi"
inivariabel222 = "Haaa"
panjang = 7
lebar = 9
luas = panjang * lebar
print(luas)
```

1.10 Operator Aritmatika

Contoh penggunaan Operator Aritmatika dalam bahasa pemrograman Python:

```
#file /python_dasar/operator_aritmatika.py
#OPERATOR ARITMATIKA
#Penjumlahan
print(17 + 3)
apel = 6
jeruk = 8
buah = apel + jeruk #
print(buah)
#Pengurangan
hutang = 15000
bayar = 9000
sisaHutang = hutang - bayar
print("Sisa hutang Anda adalah ", sisaHutang)
#Perkalian
panjang = 4
lebar = 13
luas = panjang * lebar
print(luas)
#Pembagian
kue = 100
anak = 4
kuePerAnak = kue / anak
print("Setiap anak akan mendapatkan bagian kue sebanyak ", kuePerAnak
    )
#Sisa Bagi / Modulus
bilangan1 = 16
bilangan2 = 7
hasil = bilangan1 % bilangan2
print("Sisa bagi dari bilangan ", bilangan1, " dan ", bilangan2, "
    adalah ",
hasil)
#Pangkat
```

```
bilangan3 = 9
bilangan4 = 11
hasilPangkat = bilangan3 ** bilangan4
print(hasilPangkat)
#Pembagian Bulat
print(17//3)
#17 dibagi 3 adalah 5.6666. Karena dibulatkan maka akan menghasilkan nilai 6
```

1.11 Konfisi If

Pengambilan keputusan (kondisi if) digunakan untuk mengantisipasi kondisi yang terjadi saat jalanya program dan menentukan tindakan apa yang akan diambil sesuai dengan kondisi. Pada python ada beberapa statement/kondisi diantaranya adalah if, else dan elif Kondisi if digunakan untuk mengeksekusi kode jika kondisi bernilai benar. Jika kondisi bernilai salah maka statement/kondisi if tidak akan di-eksekusi. Dibawah ini adalah contoh penggunaan kondisi if pada Python

```
#Kondisi if adalah kondisi yang akan dieksekusi oleh program jika bernilai
benar atau TRUE
nilai = 9
#jika kondisi benar/TRUE maka program akan mengeksekusi perintah dibawahnya
if(nilai > 7):
 print("Selamat Anda Lulus")
#jika kondisi salah/FALSE maka program tidak akan mengeksekusi perintah
dibawahnya
if(nilai > 10):
 print("Selamat Anda Lulus")
```

Dari contoh diatas, jika program dijalankan maka akan mencetak string ”Selamat Anda Lulus Ujian” sebanyak 1 kali yaitu pada if pertama. Di if kedua statement bernilai salah, jadi perintah print(”Selamat Anda Lulus”) tidak akan dieksekusi.

1.12 If Else

Pengambilan keputusan (kondisi if else) tidak hanya digunakan untuk menentukan tindakan apa yang akan diambil sesuai dengan kondisi, tetapi juga digunakan untuk menentukan tindakan apa yang akan diambil/dijalankan jika kondisi tidak sesuai. Pada python ada beberapa statement/kondisi diantaranya adalah if, else dan elif Kondisi if digunakan untuk mengeksekusi kode jika kondisi bernilai benar. Kondisi if else adalah kondisi dimana jika pernyataan benar (true) maka kode dalam if akan dieksekusi, tetapi jika bernilai salah (false) maka akan mengeksekusi kode di dalam else. Contoh penggunaan kondisi if else pada Python:

```
#Kondisi if else adalah jika kondisi bernilai TRUE maka akan dieksekusi pada
```

```
if, tetapi jika bernilai FALSE maka akan dieksekusi kode pada else
nilai = 3
#Jika pernyataan pada if bernilai TRUE maka if akan dieksekusi,
    tetapi jika
FALSE kode pada else yang akan dieksekusi.
if(nilai > 7):
 print("Selamat Anda Lulus")
else:
 print("Maaf Anda Tidak Lulus")
```

Pada contoh diatas, jika program dijalankan maka akan mencetak string ”Maaf Anda Tidak Lulus” karena pernyataan pada if bernilai FALSE

1.13 Kondisi Elif

Pengambilan keputusan (kondisi if elif) merupakan lanjutan/percabangan logika dari ”kondisi if”. Dengan elif kita bisa membuat kode program yang akan menyeleksi beberapa kemungkinan yang bisa terjadi. Hampir sama dengan kondisi ”else”, bedanya kondisi ”elif” bisa banyak dan tidak hanya satu. Contoh penggunaan kondisi elif pada Python:

```
#Contoh penggunaan kondisi elif
hari_ini = "Minggu"
if(hari_ini == "Senin"):
 print("Saya akan kuliah")
elif(hari_ini == "Selasa"):
 print("Saya akan kuliah")
elif(hari_ini == "Rabu"):
 print("Saya akan kuliah")
elif(hari_ini == "Kamis"):
 print("Saya akan kuliah")
elif(hari_ini == "Jumat"):
 print("Saya akan kuliah")
elif(hari_ini == "Sabtu"):
 print("Saya akan kuliah")
elif(hari_ini == "Minggu"):
 print("Saya akan libur")
```

Pada contoh diatas, jika program dijalankan maka akan mencetak string ”Saya akan libur”.

1.14 Pengulangan ”Loop”

Secara umum, pernyataan pada bahasa pemrograman akan dieksekusi secara berurutan. Pernyataan pertama dalam sebuah fungsi dijalankan pertama, diikuti oleh yang kedua, dan seterusnya. Tetapi akan ada situasi dimana Anda harus menulis banyak kode, dimana kode tersebut sangat banyak. Jika dilakukan secara manual maka Anda hanya akan membuang-buang tenaga dengan menulis beratus-ratus bahkan beribu-ribu kode. Untuk itu Anda perlu menggunakan pengulangan di dalam bahasa pemrograman Python.

Di dalam bahasa pemrograman Python pengulangan dibagi menjadi 3 bagian, yaitu :

- While Loop
- For Loop
- Nested Loop

Pengulangan While Pengulangan While Loop di dalam bahasa pemrograman Python dieksesusi statement berkali-kali selama kondisi bernilai benar atau True. Contoh penggunaan pengulangan While Loop.

```
#Contoh penggunaan While Loop
count = 0
while (count < 9):
 print ('The count is:', count)
 count = count + 1
print ("Good bye!")
```

1.15 Pengulangan For

Pengulangan For pada Python memiliki kemampuan untuk mengulangi item dari urutan apapun, seperti list atau string. Contoh penggunaan pengulangan While Loop:

```
#Contoh pengulangan for sederhana
angka = [1,2,3,4,5]
for x in angka:
 print(x)
#Contoh pengulangan for
buah = ["nanas", "apel", "jeruk"]
for makanan in buah:
 print("Saya suka makan", makanan)
```

1.16 Pengulangan Bersarang

Bahasa pemrograman Python memungkinkan penggunaan satu lingkaran di dalam loop lain. Bagian berikut menunjukkan beberapa contoh untuk menggambarkan konsep tersebut. Contoh penggunaan Nested Loop.

```
#Contoh penggunaan Nested Loop
i = 2
while(i < 100):
 j = 2
 while(j <= (i/j)):
 if not(i%j): break
 j = j + 1
 if (j > i/j) : print i, " is prime"
 i = i + 1
print "Good bye!"
```

1.17 Dictionary Python

Dictionary Python berbeda dengan List ataupun Tuple. Karena setiap urutanya berisi key dan value. Setiap key dipisahkan dari value-nya oleh titik dua (:), item dipisahkan oleh koma, dan semuanya tertutup dalam kurung kurawal. Dictionary kosong tanpa barang ditulis hanya dengan dua kurung kurawal, seperti ini: . Nilai kamus bisa berupa tipe apa pun, namun key harus berupa tipe data yang tidak berubah seperti string, angka, atau tupel.

Akses Nilai Untuk mengakses elemen Dictionary, Anda dapat menggunakan tanda kurung siku yang sudah dikenal bersama dengan key untuk mendapatkan nilainya. Berikut adalah contoh sederhananya :

```
#Contoh cara membuat Dictionary pada Python
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
print ("dict['Name']: ", dict['Name'])
print ("dict['Age']: ", dict['Age'])
```

Update Nilai Anda dapat memperbarui Dictionary dengan menambahkan entri baru atau pasangan nilai kunci, memodifikasi entri yang ada, atau menghapus entri yang ada seperti ditunjukkan pada contoh sederhana yang diberikan di bawah ini.

```
#Update dictionary python
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
dict['Age'] = 8; # Mengubah entri yang sudah ada
dict['School'] = "DPS School" # Menambah entri baru
print ("dict['Age']: ", dict['Age'])
print ("dict['School']: ", dict['School'])
```

Hapus Nilai Anda dapat menghapus elemen Dictionary individual atau menghapus keseluruhan isi Dictionary. Anda juga dapat menghapus seluruh Dictionary dalam satu operasi. Untuk menghapus seluruh Dictionary secara eksplisit, cukup gunakan del statement. Berikut adalah contoh sederhana :

```
#Contoh cara menghapus pada Dictionary Python
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
del dict['Name'] # hapus entri dengan key 'Name'
dict.clear() # hapus semua entri di dict
del dict # hapus dictionary yang sudah ada
print ("dict['Age']: ", dict['Age'])
print ("dict['School']: ", dict['School'])
```

1.18 Fungsi

Fungsi adalah blok kode terorganisir dan dapat digunakan kembali yang digunakan untuk melakukan sebuah tindakan/action. Fungsi memberikan modularitas yang lebih baik untuk aplikasi Anda dan tingkat penggunaan kode yang tinggi.

Mendefinisikan Fungsi Python Anda dapat menentukan fungsi untuk menyediakan fungsionalitas yang dibutuhkan. Berikut adalah aturan sederhana untuk mendefinisikan fungsi dengan Python.

- Fungsi blok dimulai dengan def kata kunci diikuti oleh nama fungsi dan tanda kurung (()).
- Setiap parameter masukan atau argumen harus ditempatkan di dalam tanda kurung ini. Anda juga dapat menentukan parameter di dalam tanda kurung ini.
- Pernyataan pertama dari sebuah fungsi dapat berupa pernyataan opsional string dokumentasi fungsi atau docstring.
- Blok kode dalam setiap fungsi dimulai dengan titik dua (:) dan indentasi.
- Pernyataan kembali [ekspresi] keluar dari sebuah fungsi, secara opsional menyampaikan kembali ekspresi ke pemanggil. Pernyataan pengembalian tanpa argumen sama dengan return None.

Contoh fungsi:

```
def printme( str ):
 "This prints a passed string into this function"
 print (str)
 return
```

BAB 2

GOOGLE COLAB

2.1 *Google Colab*

Google Colab adalah tools yang dikeluarkan oleh *Google Internal Research*. *Google Colab* merupakan salah satu produk Google berbasis *cloud* yang bisa digunakan secara gratis. *Google Colab* dibuat khusus untuk para researcher atau programmer yang kesulitan untuk mendapatkan akses komputer dengan spek tinggi. *Google Colab* merupakan coding environment bahasa pemrograman Python dengan format *notebook*. Tools pada *Google Colab* menyediakan layanan GPU gratis kepada pengguna sebagai *backend* komputasi dan dapat digunakan selama 12 jam pada suatu waktu. *Google Colab* berjalan diatas cloud milik Google dan menyimpan berkas kedalam Google Drive, serta dapat menjalankan command line langsung pada cell notebook dengan diawali tanda "!".

2.2 Manfaat Menggunakan *Google Colab*

1. Free GPU

   Google Colab memudahkan untuk menjalankan program pada komputer dengan spek tinggi (GPU Tesla, RAM 12GB, Disk 300GB yang masih bisa disambungkan dengan Google Drive, akses internet cepat untuk download file besar) dan running dalam waktu yang lama.

2. Colaborate

   Memudahkan untuk berkolaborasi dengan orang lain dengan cara membagikan kodingan secara online. Dapat lebih mudah bereksperimen secara bersamaan, atau sekadar menggunakan fitur ini untuk mempelajari codingan milik orang lain.

3. Mudah berintegrasi

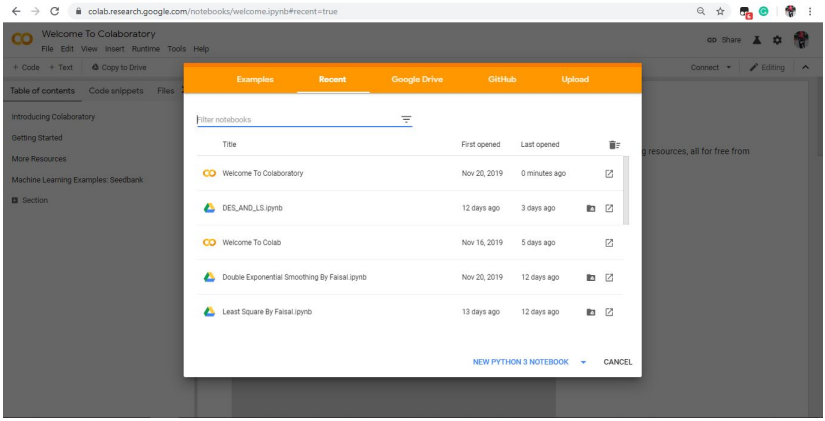
   Dapat dengan mudah menghubungkan *Google Colab* dengan jupyter notebook di komputer dengan *local runtime*, menghubungkan dengan *Google Drive*, atau dengan Github.

4. Fleksibel

   Bisa dengan mudah merunning *deep learning* program melalui handphone, karena pada *Google Colab* hanya perlu *running* di browser, dapat mengawasi via browser smartphone selama smartphone terhubung dengan *Google Drive* yang sama.

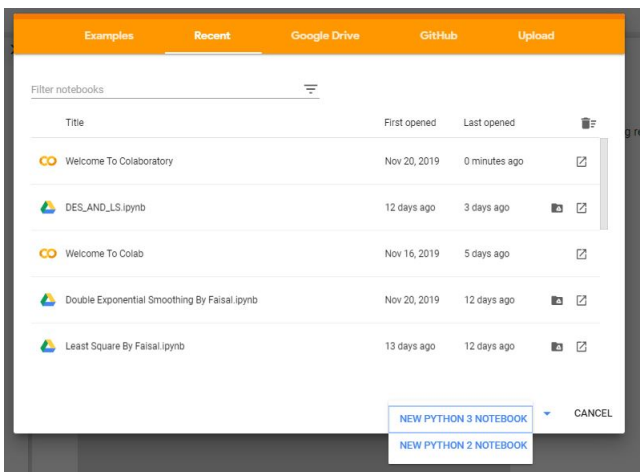
2.3 Cara Penggunaan *Google Colab*

1. Sebelumnya butuhkan terlebih dahulu akun Google dan selanjutnya kunjungi ke link https://colab.research.google.com/. Setelah itu akan menampilkan tampilan sebagai berikut:



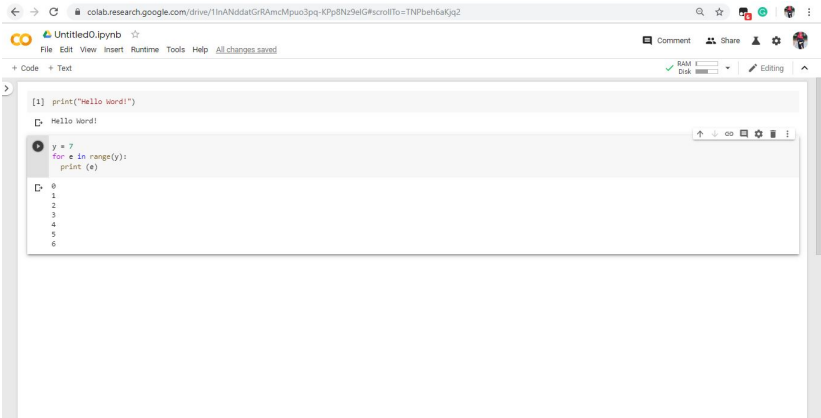
Gambar 2.1 Tampilan Awal *Google Colab*

2. Untuk membuat notebook baru, cukup klik New Python 3 Notebook atau Python 2 tergantung dari apa yang akan digunakan.



Gambar 2.2 Tampilan Membuat Notebook Baru

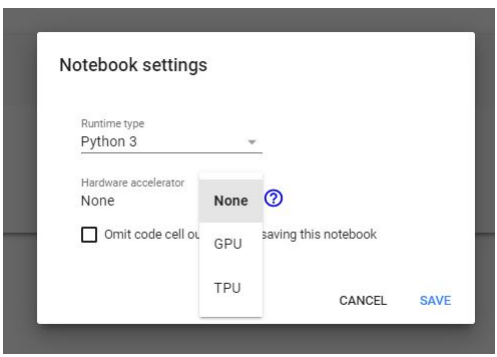
3. Setelah itu akan menampilkan tampilan ke halaman yang mirip dengan Jupyter Notebook. Nantinya, setiap notebook yang kita buat akan disimpan di *Google Drive*.



Gambar 2.3 Tampilan *Google Colab*

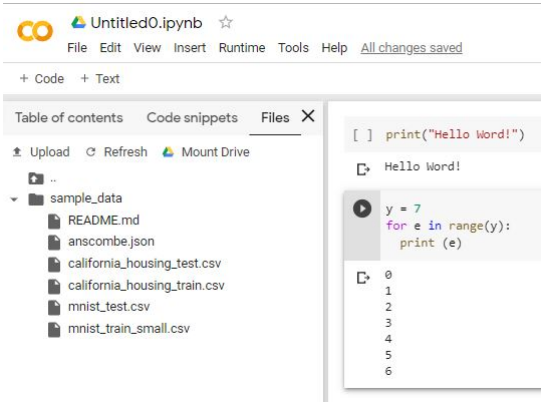
4. Pengaturan GPU

   Jika menjalankan program Python menggunakan GPU atau TPU, cukup pilih atau klik Edit > Notebook Settings. Setelah itu pada bagian *Hardware Accelerator* pilih GPU.

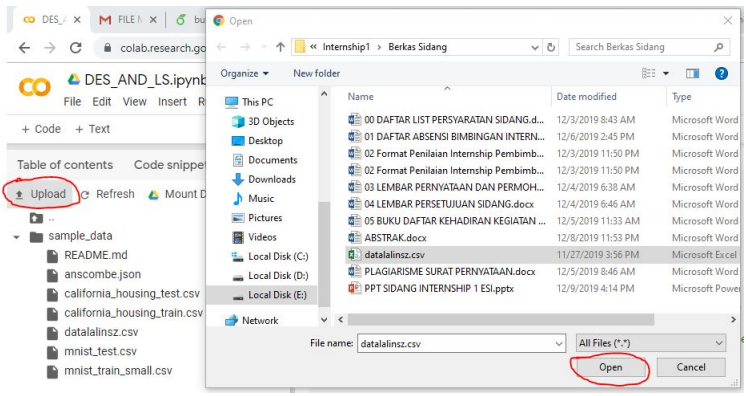


Gambar 2.4 Tampilan Pengaturan *Hardware Accelerator*

5. Selanjutnya mengupload data yang akan di olah di dalam *Google Colab* dengan mengupload data yang berformat csv. Caranya klik pada tulisan Upload, lalu pilih file mana yang akan di upload lalu klik Open.

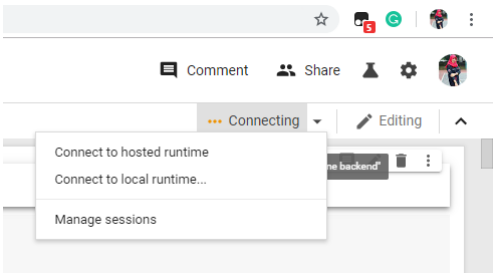


Gambar 2.5 Tampilan File Data csv di *Goole Colab*

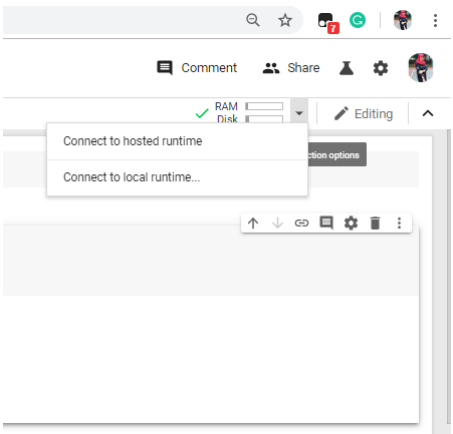


Gambar 2.6 Tampilan Upload Data csv

6. Ketika kita membuat file baru di *Google Colab* tentu belum langsung terkonek ke komputing di google seperti pada gambar 2.7. Lalu setelah itu pilih *Connect to hosted runtime*. Maka setelah itu akan terkonek seperti gambar 2.8.

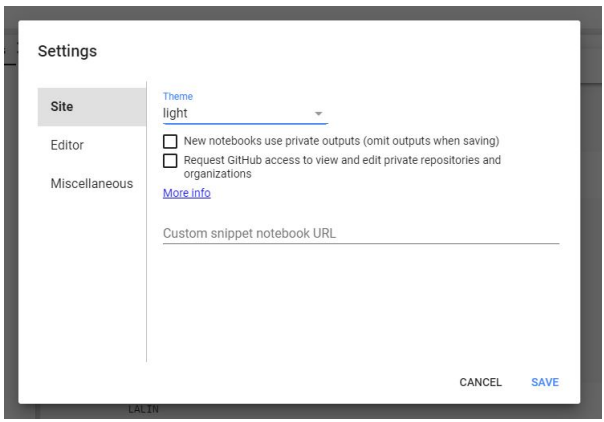


Gambar 2.7 Tampilan Sebelum Konek



Gambar 2.8 Tampilan Setelah Konek

7. Tampilan notebook dapat diubah sesuai keinginan. Terdapat pilihan *night mode* yang membuat tema notebook menjadi gelap. Langkah yang dilakukan dengan Tools > Settings > Site.

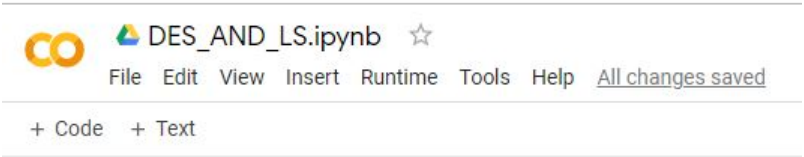


Gambar 2.9 Tampilan Pengaturan Notebook

Gambar 2.10 Tampilan Berhasil Merubah Warna

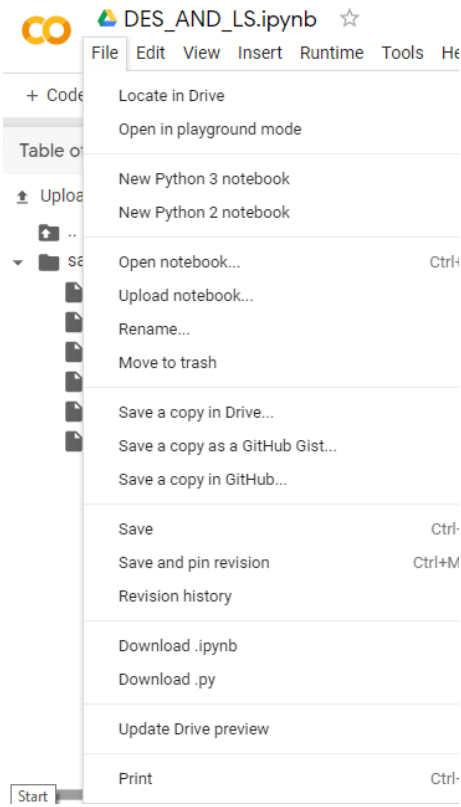
2.4 Tools *Google Colab*

Pada *Google Colab* terdapat beberapa *tools* yang dapat digunakan, berikut di bawah ini akan di jelaskan fungsi-fungsi dari setiap *tools* yang terdapat di *Google Colab*.



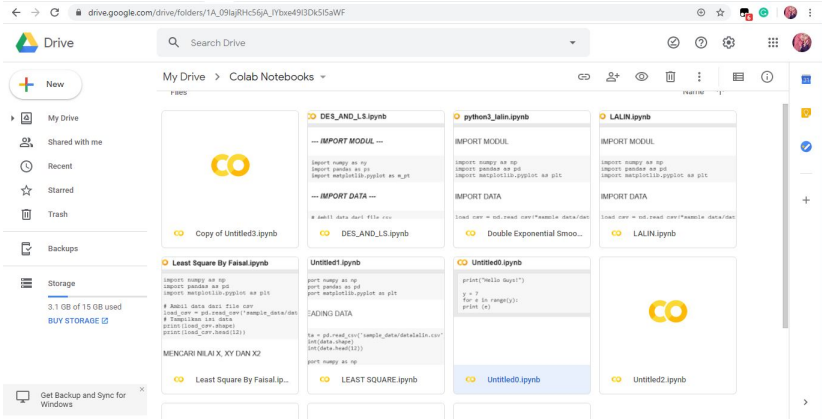
Gambar 2.11 Tools *Google Colab*

1. File :



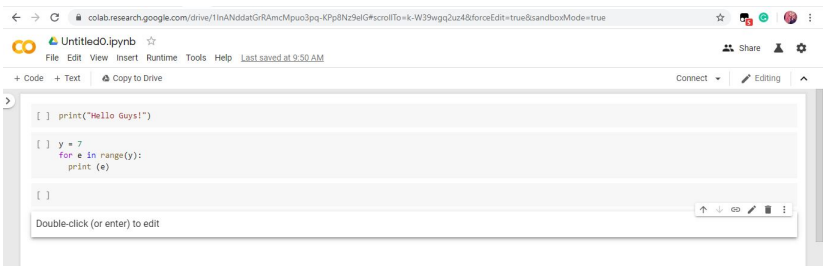
Gambar 2.12 *Tools File*

- *Locate in Drive* : Untuk menemukan file yang sudah tersimpan di dalam *Google Drive*, selanjutnya akan di arahkan menuju tampilan di google drive masing-masing.



Gambar 2.13 *Locate in Drive*

- *Open in playground mode* : Tidak terdapat *output* yang disimpan dalam mode tersebut, tetapi tidak dapat menggunakan fungsi *tools rename* dan *save and pin revision*.

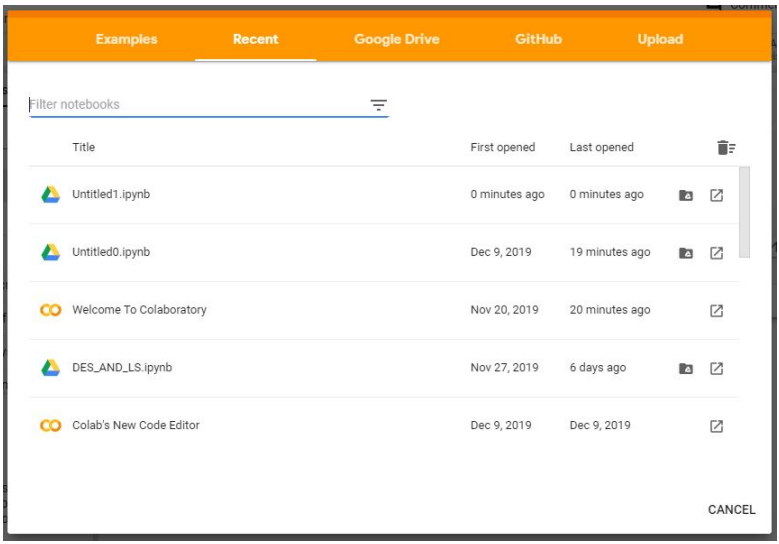


Gambar 2.14 *Open in playground mode*

- *New Python 3 notebook* : Membuat file baru di tab baru dengan format Python 3 *notebook*.

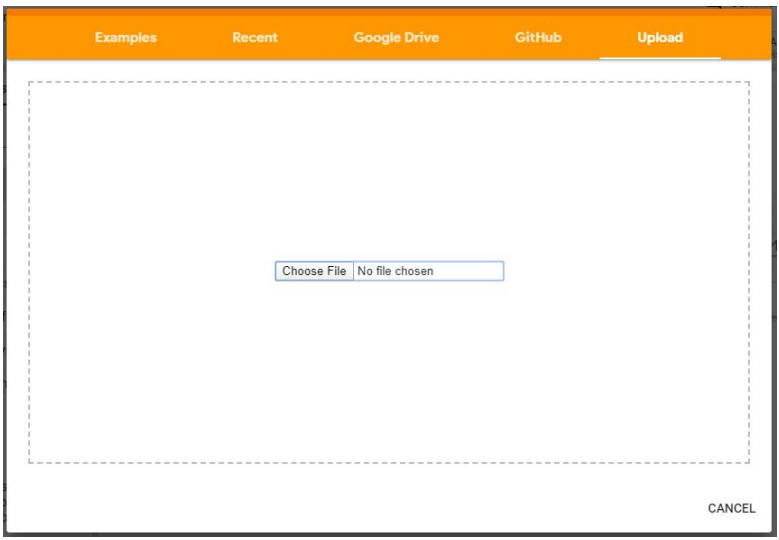
- *New Python 2 notebook* : Membuat file baru dengan di tab baru dengan format Python 2 *notebook*.

- *Open notebook* : Untuk membuka file lain yang sudah tersimpan sebelumnya.



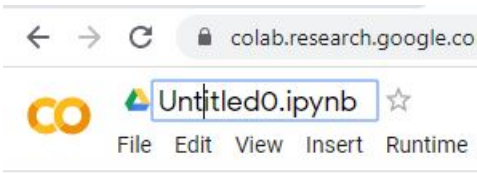
Gambar 2.15 *Open notebook*

- *Upload notebook* : Untuk mengupload file Python yang ada di PC untuk di tampilkan di dalam *Google Colab*.



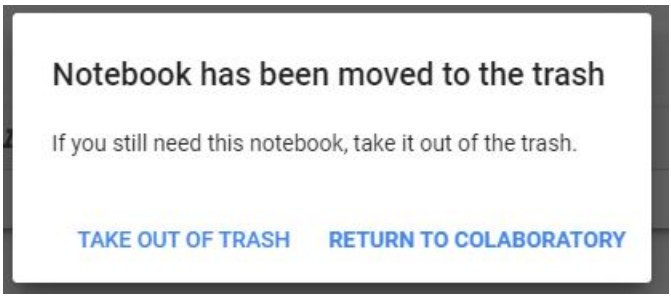
Gambar 2.16 *Upload notebook*

- *Rename* : Untuk mengubah nama dari file Python sesuai yang di inginkan, setelah itu akan diarahkan seperti gambar 2.17.



Gambar 2.17 *Rename*

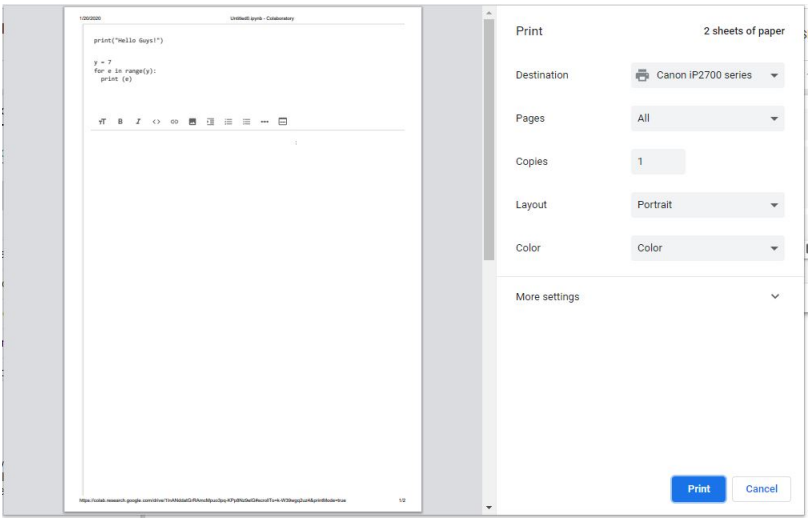
- *Move to trash* : Untuk menghapus file yang sedang di buka dan terdapat dua pilhan yaitu keluarkan dari tempat sampah atau kembali ke kolaboratori.



Gambar 2.18 *Move to trash*

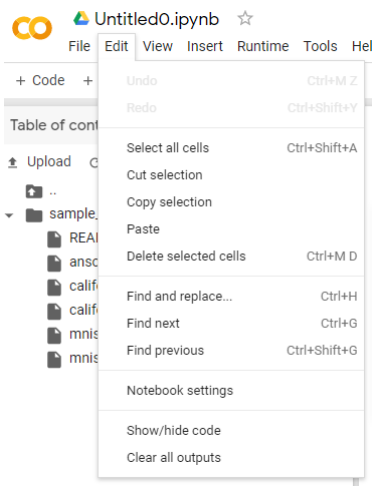
- *Save a copy in Drive* : Untuk menyimpan file *notebook* yang sudah dibuat kedalam *Google Drive*.
- *Save a copy as a GitHub Gist* : Untuk menyimpan file *notebook* yang sudah dibuat kedalam bentuk *Github Gist*.
- *Save a copy in GitHub* : Untuk menyimpan file *notebook* yang sudah dibuat kedalam *GitHub*.
- *Save* : Berfungsi sebagai menyimpan file *notebook* yang sudah dibuat ke dalam *Google Colab*.
- *Save and pin revision* : Menyimpan file yang telah di revisi atau yang telah diubah dengan cara di di pin.
- *Revision history* : Melihat data yang sudah di revisi sebelumnya di dalam *Google Colab*.
- *Download .ipynb* : Untuk menyimpan file *notebook* ke dalam PC yang sudah dibuat dalam format .ipynb.
- *Download .py* : Untuk menyimpan file *notebook* ke dalam PC yang sudah dibuat dalam format .py.

- *Update Drive preview* : Untuk menyimpan file *notebook* yang di simpan di dalam *Google Drive*.
- *Print* : Untuk mencetak hasil keseluruhan yang terdapat di *file notebook Google Colab*.



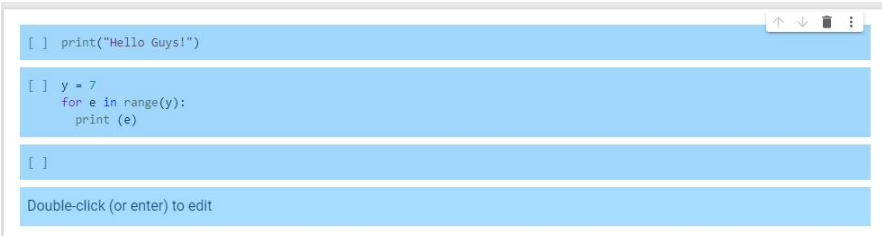
Gambar 2.19 *Print*

2. Edit :



Gambar 2.20 *Tools Edit*

- *Undo* : Perintah untuk membatalkan suatu perintah yang sudah dilakukan sebelumnya.
- *Redo* : Cara untuk mengulang sesuatu yang telah dibatalkan sebelumnya.
- *Select all cells* : Berfungsi untuk memblok seluruh sel yang berisikan source code di dalam *notebook Google Colab*.



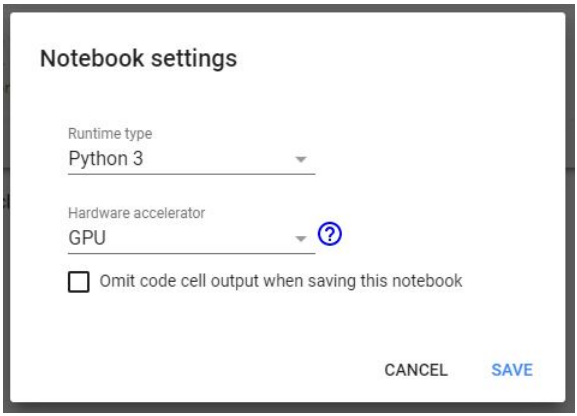
Gambar 2.21 *Select all cells*

- *Cut selection* : Befungsi untuk mengcut atau memotong *source code* yang sudah di pilih yang ada di dalam *notebook Google Colab*.
- *Copy selection* : Berguna untuk mencopy kata-kata atau *source code* yang sudah terseleksi atau terpilih.
- *Paste* : Yang digunakan untuk menempelkan kata, paragraf, kelompok kata, tabel, gambar, dan object apapun yang sebelumnya telah di Copy atau di Cut.
- *Delete selected cells* : Berfungsi untuk menghapus sel yang telah dipilih.
- *Find and replace* : Untuk mencari kata pada yang di inginkan dan menggganti kata tersebut sesuai dengan keinginan. Akan tampil pada bagian pojok kanan bawah pada google colab.

Gambar 2.22 *Find and replace*

- *Find text* : Untuk mencari kata pada lembar kerja yang di inginkan dan mengganti kata tersebut sesuai dengan kemauan. Tampilannya sama dengan *Find and replace*.
- *Find previous* : Untuk menemukan kata-kata sebelumnya. Tampilannya sama dengan *Find and replace*.

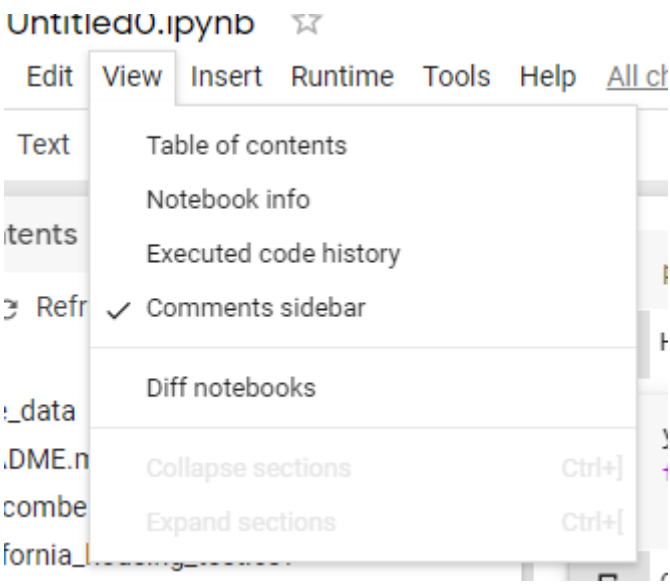
- *Notebook settings* : Untuk mengatur pengaturan yang ada di dalam *notebook* seperti mengatur runtime type, hardware accelerator.



Gambar 2.23 *Notebook settings*

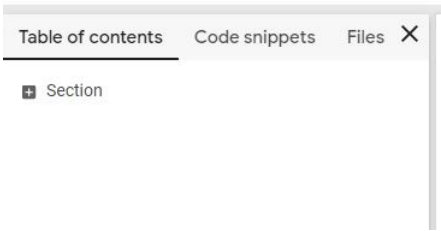
- *Show/hide code* : Berfungsi untuk menampilkan dan menyembunyikan *code*.
- *Clear all outputs* : Berfungsi untuk menghapus semua *output* yang ada.

3. View :



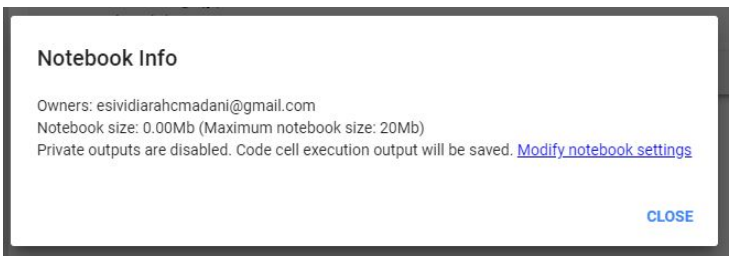
Gambar 2.24 *Tools View*

- *Table of contents* : Salah satu fasilitas yang digunakan untuk pembuatan daftar isi secara otomatis.



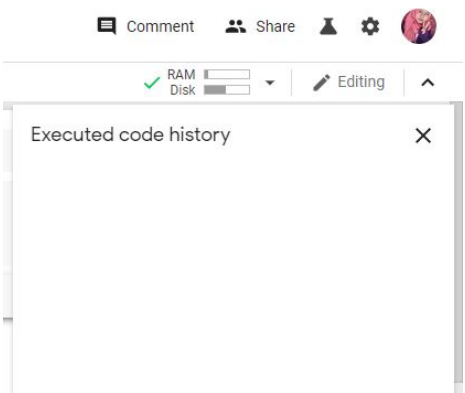
Gambar 2.25 *Table of contents*

- *Notebook info* : Menu yang berisi menampilkan informasi terkait owener, notebook size dan private outputs.



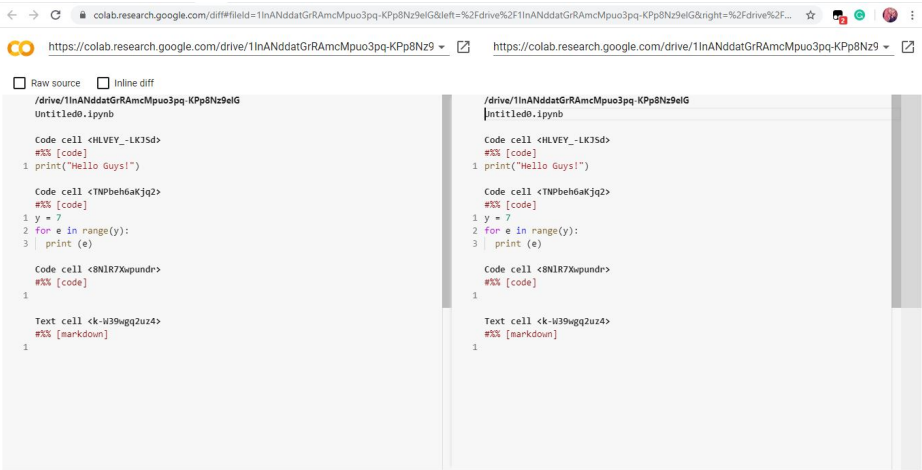
Gambar 2.26 *Notebook info*

- *Executed code history* : Riwayat kode yang dieksekusi.



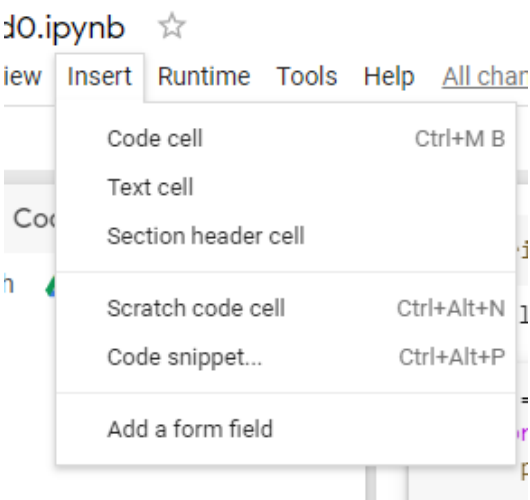
Gambar 2.27 *Executed code history*

- *Comments sidebar* : Untuk menambahkan komentar di dalam *source code* di sampingnya.
- *Diff notebooks* : Perintah atau menu yang berfungsi untuk menampilkan *notebook* yang baru atau yang berbeda.



Gambar 2.28 *Diff notebooks*

4. Insert :

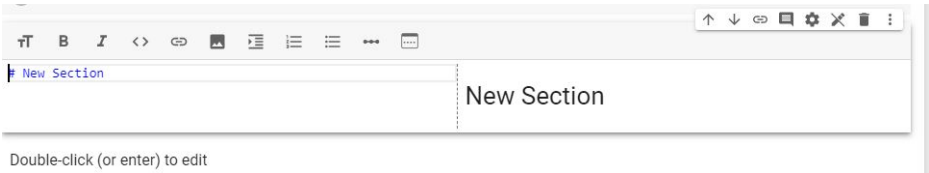


Gambar 2.29 *Tools Insert*

- *Code cell* : Untuk menambah sel baru untuk membuat baris *source code* baru.

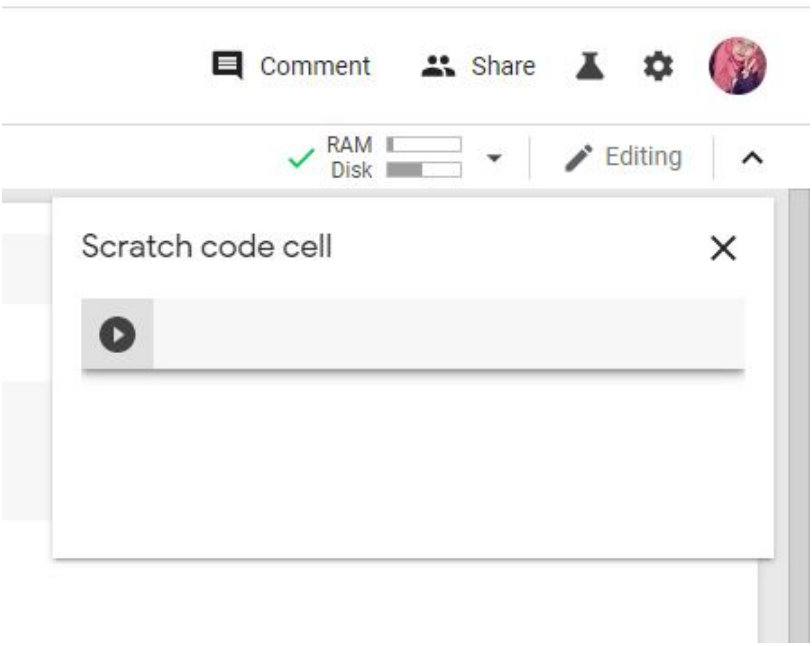
- *Text cell* : Untuk menambahkan sel baru yang berisi hanya *text* biasanya untuk menambahkan keterangan atau judul sebelum *source code*.

- *Section header cell* : Untuk menambahkan *section* baru di dalam *notebook* yang sedang di kerjakan.



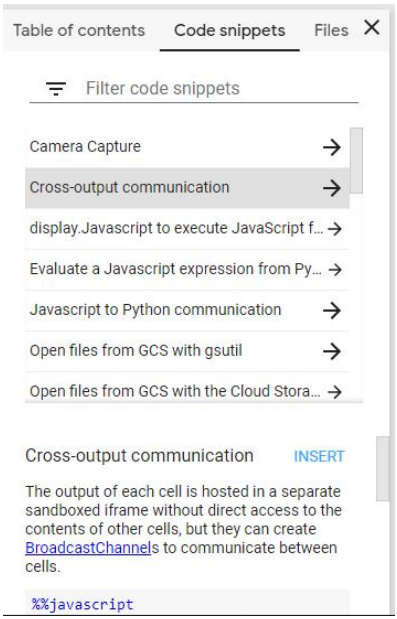
Gambar 2.30 *Section header cell*

- *Scratch code cell* : untuk menggoreskan kode sel atau membuat dan merunging *script* di dalam *Scratch code cell* seperti pada 2.31.



Gambar 2.31 *Scratch code cell*

- *Code snippet* : untuk cuplikan kode atau menampilkan kode-kode sesuai dengan yang dicari.

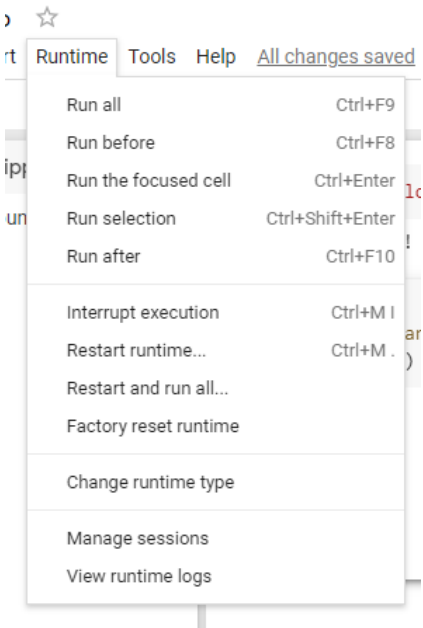


Gambar 2.32 *Code snippet*

- *Add a from field* : untuk menambahkan field baru dengan harus menginputkan form field type, variable name dan variable type.

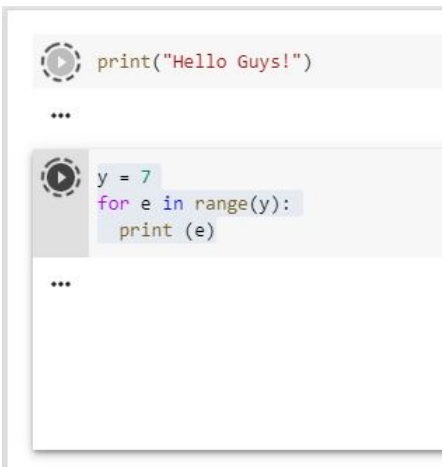
Gambar 2.33 *Add a from field*

5. Runtime :



Gambar 2.34 *Tools Runtime*

- *Run all* : Perintah untuk menjalankan semua yang ada di dalam *notebook*.



Gambar 2.35 *Run all*

- *Run before* : Perintah untuk menjalankan *source code* sebelumnya atau yang dimulai dari awal.

Gambar 2.36 *Run before*

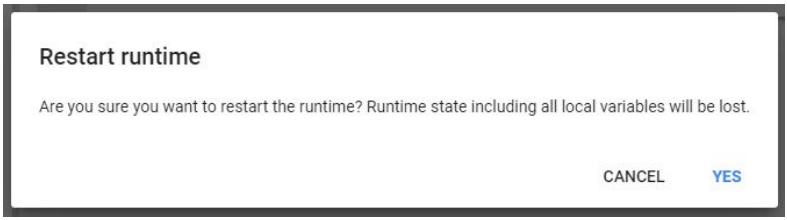
- *Run the focused cell* : Perintah yang hanya akan meruning jika satu cell yang di fokuskan.

Gambar 2.37 *Run the focused cell*

- *Run selection* : Perintah untuk menjalankan *source code* yang di seleksi saja.
- *Run after* : Perintah untuk menjalankan pada *source code* sebelumnya.

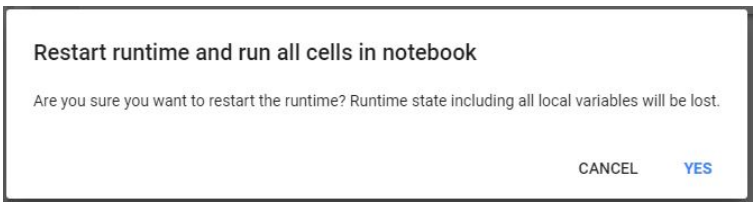
- *Interrupt execution* : Perintah untuk menjalankan eksekusi interupsi yang sudah di perintah.

- *Restart runtime* : Perintah yang berfungsi atau berguna untuk memulai kembali menjalankan *source code*.



Gambar 2.38 *Popup Restart runtime*

- *Restart and run all* : Perintah untuk memulai ulang dan setelah itu menjalankan semua atau meruning *source code* yang ada di dalam notebook.

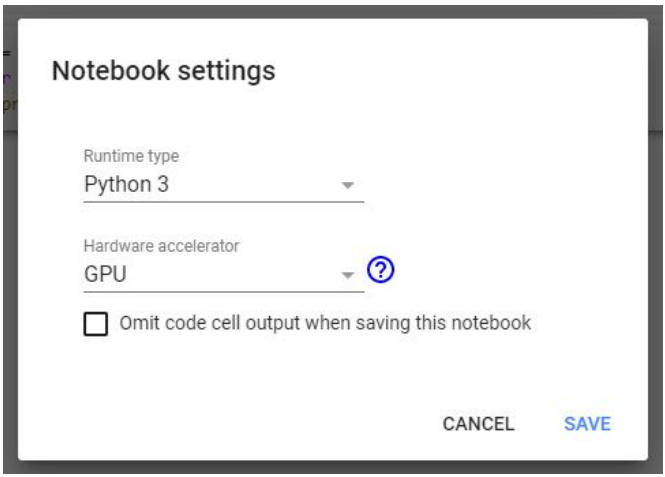


Gambar 2.39 *Popup Restart and run all*

- *Factory reset runtime* : Perintah untuk mengatur ulang runtime termasuk semua variabel dan file lokal, akan hilang.

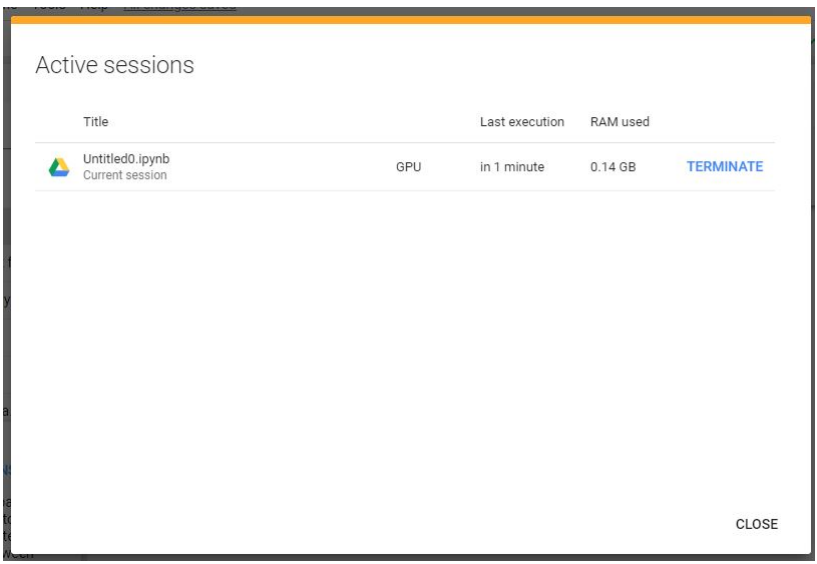
Gambar 2.40 *Popup Factory reset runtime*

- *Change runtime type* : Perintah yang berguna untuk mengubah jenis runtime type dan hardware accelerator.

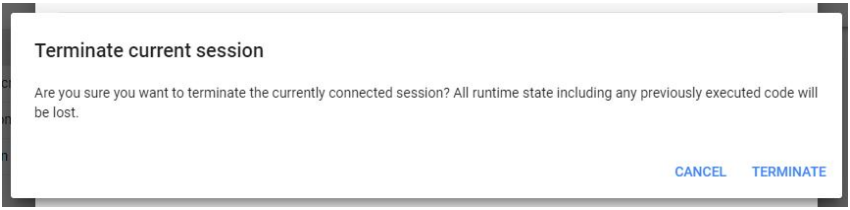


Gambar 2.41 *Change runtime type*

- *Manage sessions* : Perintah untuk mengakhiri sesi yang sudah terhubung dan semua status runtime termasuk kode yang dieksekusi akan hilang.



Gambar 2.42 *Manage sessions*

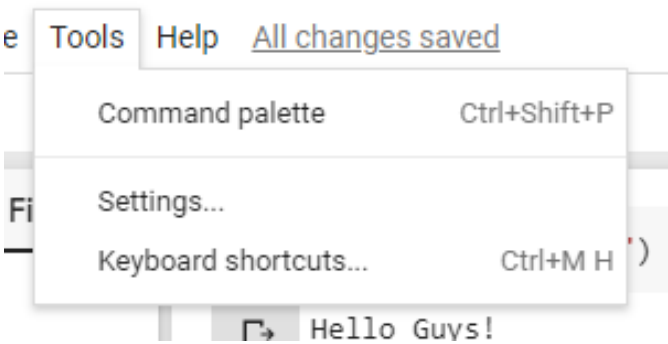


Gambar 2.43 *Setelah Terminate*

- *View runtime logs* : Perintah untuk meruning atau menjalankan *source code* yang ada di notebook melalui colab-jupyter.log.

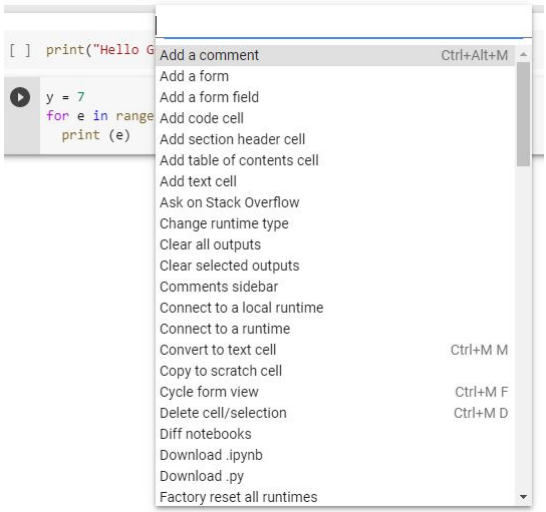
Gambar 2.44 *View runtime logs*

6. Tools :



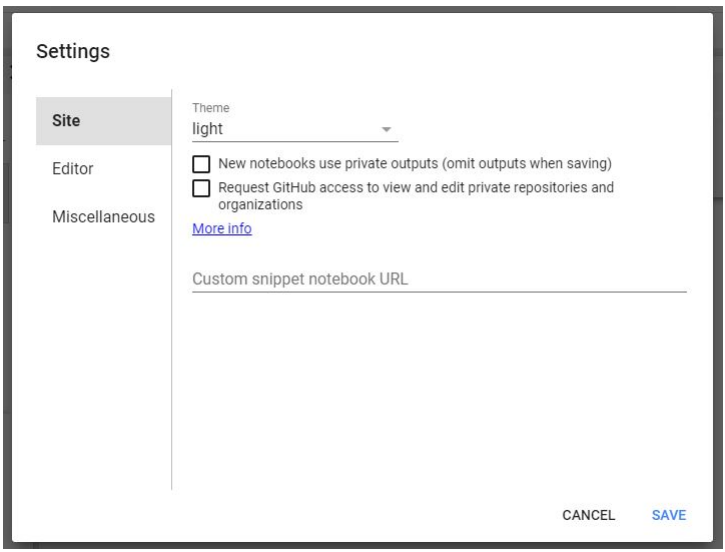
Gambar 2.45 *Tools*

- *Command palette* : Palet perintah mencakup daftar item atau perintah yang sering digunakan.



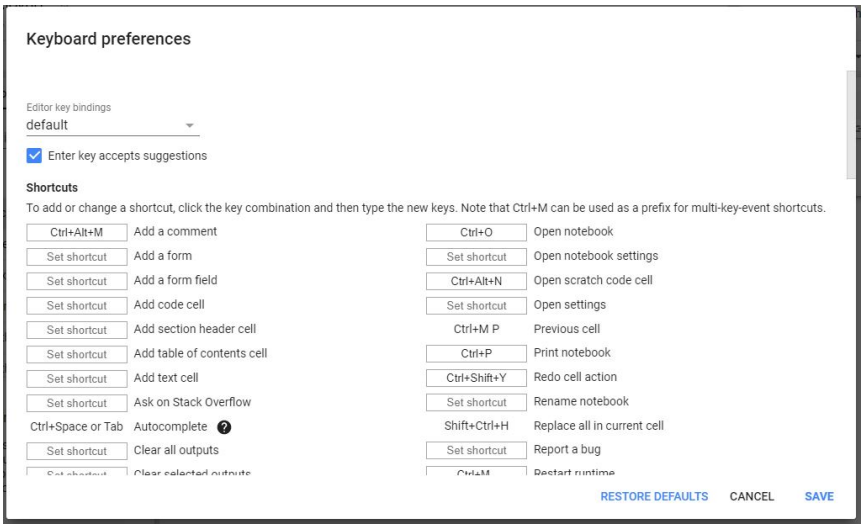
Gambar 2.46 *Command palette*

- *Settings* : Perintah untuk mengatur pengaturan pada google colab seperti mengatur site, editor, dan Miscellaneous.



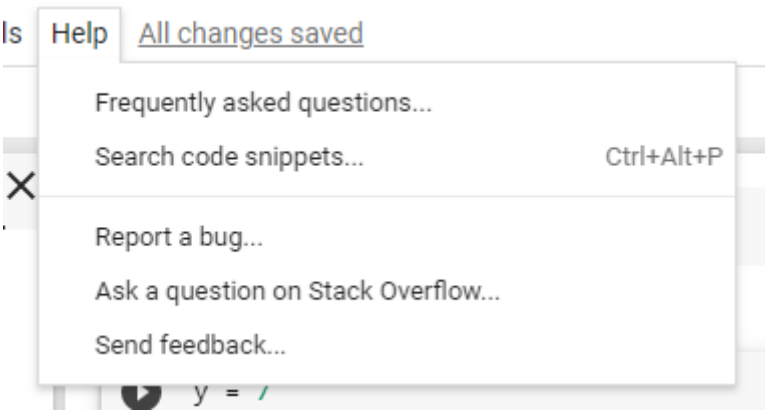
Gambar 2.47 *Settings*

- *Keyboard shortcuts* : Yaitu perintah atau petunjuk *Keyboard preferences* yang dapat di gunakan di dalam google colab agar mempermudah pengguna dalam memakai google colab.



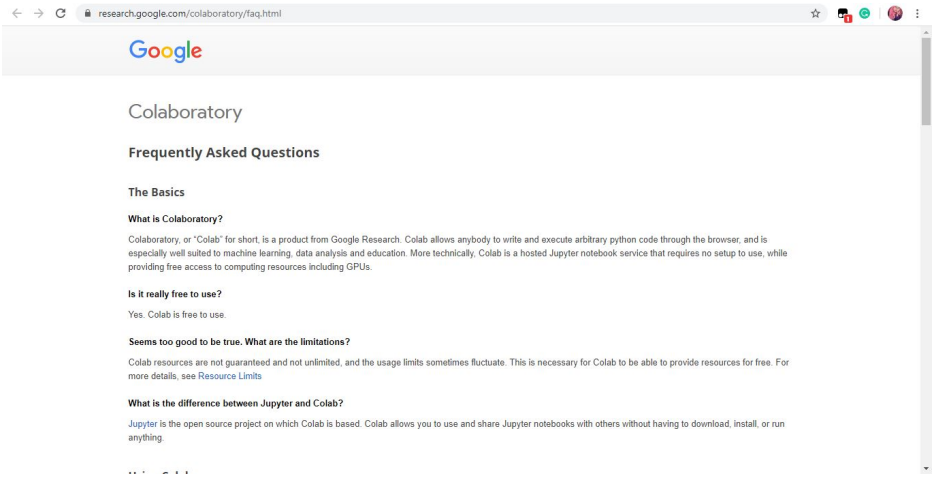
Gambar 2.48 *Keyboard shortcuts*

7. Help :



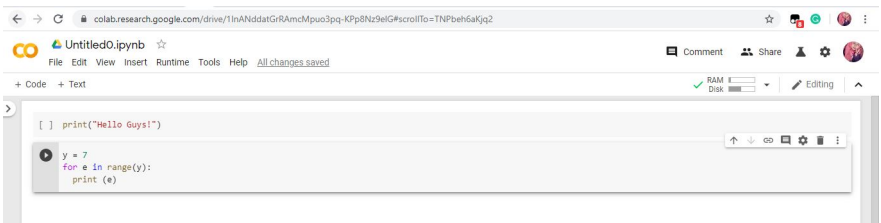
Gambar 2.49 *Tools Help*

- *Frequently asked questions* : Perintah yang sering digunakan untuk pertanyaan yang sering diajukan sehingga akan terjadi saling tanya jawab.



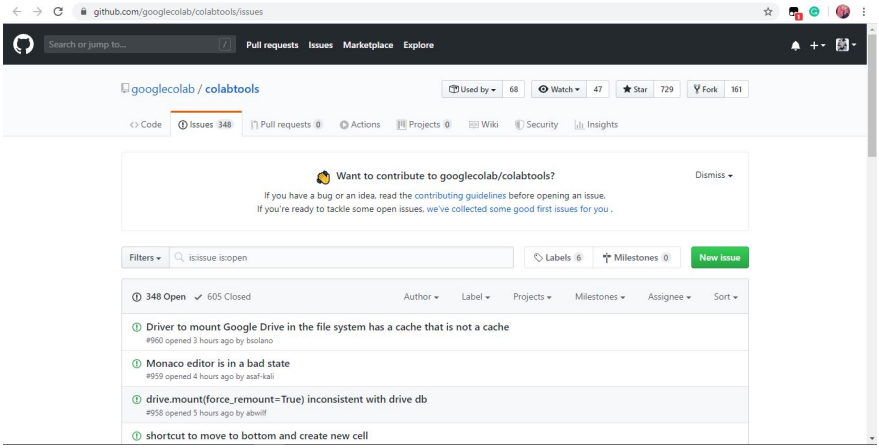
Gambar 2.50 *Frequently Asked Questions*

- *Search code snippets* : Perintah untuk membantu pengguna google colab dalam mencari cuplikan dari *source code* yang nantinya akan menampilkan full hanya bagian dari *source code* saja seperti pada gambar 2.51.



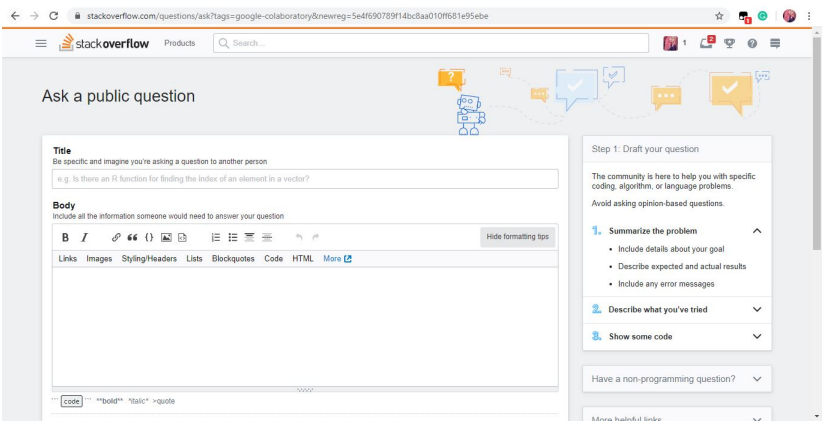
Gambar 2.51 *Search code snippets*

- *Report a bug* : Perintah untuk melaporkan bug yang ada pada google colab yang setalah itu akan muncul pada tampilan github seperti pada gambar 2.52.



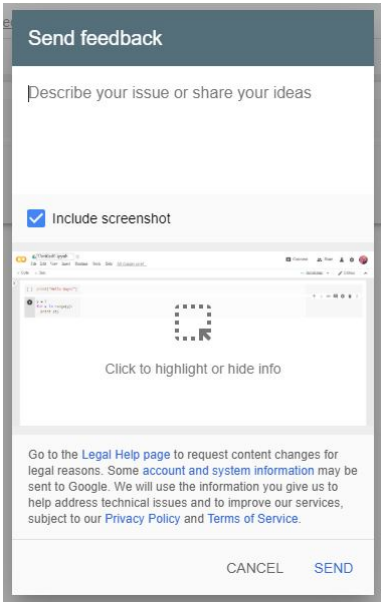
Gambar 2.52 *Report a bug*

- *Ask a question on Stack Overflow* : Perintah untuk mencari pertanyaan yang di tanyakan pada *platform Stack Overflow*, jika belum memiliki akun pada *Stack Overflow* dapat membuat dulu akun terlebih dahulu atau dapat juga langsung login dengan menggunakan akun google yang sudah ada.



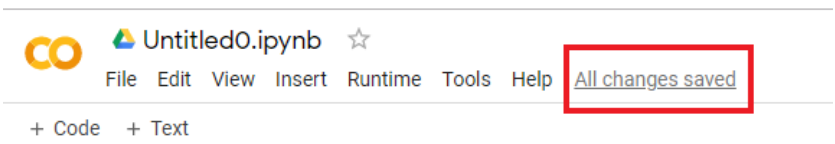
Gambar 2.53 *Ask a question on Stack Overflow*

- *Send feedback* : Perintah untuk melakukan meminta umpan balik dari apa yang telah di buat.



Gambar 2.54 *Send feedback*

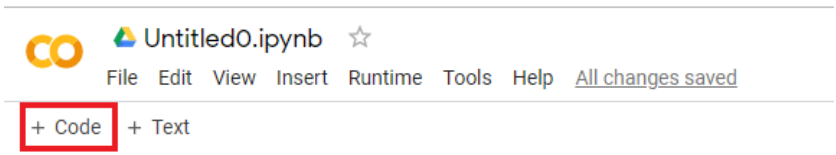
8. All changes saved : Perintah untuk melihat semua history yang dikerjakan.



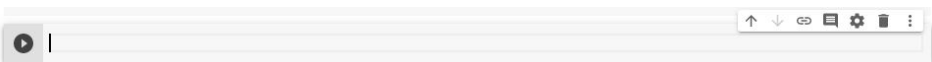
Gambar 2.55 *All changes saved*

Gambar 2.56 *Tampilan All changes saved*

9. Code : Perintah untuk menambahkan code baru dalam mengerjakan di dalam notebook google colab.

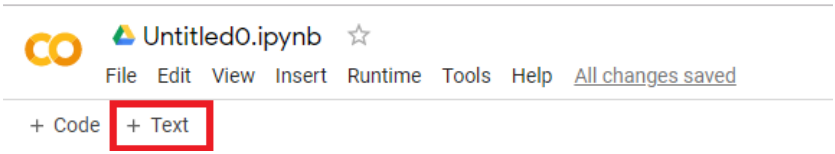


Gambar 2.57 *Code*



Gambar 2.58 *Tampilan Code*

10. Text : Perintah untuk menambahkan text baru dalam mengerjakan di dalam notebook google colab.



Gambar 2.59 *Text*

Gambar 2.60 *Tampilan Text*